МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра Прикладная математика и информатика
(наименование)
09.04.03 Прикладная информатика
(код и наименование направления подготовки)
Информационные системы и технологии корпоративного управления
(направленность (профиль))

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ)

на тему Проектирование корпоративных информационных систем класса ERP для управления сетью территориально распределенных филиалов

Студент А.А. Павлов
(И.О. Фамилия) (личная подпись)

Научный руководитель кандидат педагогических наук, доцент, О.Ю. Копша
(ученая степень, звание, И.О. Фамилия)

Тольятти 2021

Оглавление

Введение

Актуальность исследования. Одним из определяющих факторов эффективной деятельности для любой компании является умение управлять ресурсами предприятия. Такое управление включает в себя решение задачи оптимального использования материальных ресурсов, денежных средств, человеческих ресурсов (кадров предприятия). Нерациональное использование ресурсов, задержки в поставках материалов способны вызвать остановку производства, срыв поставок продукции и, как следствие, проблемы с поступлением денежных средств на предприятие. Следовательно, данные компоненты предприятия взаимосвязаны. Управление предприятием сводится к задаче планирования ресурсов.

Информационные технологии расширяют возможности для эффективного управления, предоставляя руководителям различных служб предприятия новейшие методы обработки и анализа информации, необходимой для принятия решений. Принятие решений должно быть основано на подсистемах планирования, именно концепция ERP-систем позволяет оптимизировать деятельность предприятия.

В настоящее время ERP-системы играют важную роль в управлении множеством бизнес-процессов и способствуют доступу к важной бизнес-информации, то есть помогают улучшить управление компанией. Основное назначение данных систем предоставлять быстрый доступ к информации, оптимизировать бизнес-процессы, удалить ненужные операции, предоставить информационный обмен между всеми подразделениями компании.

ERP-системы позволяют автоматизировать администрирование, производство, общее управление, финансы, бухгалтерский учет и другие функциональные области бизнеса.

Благодаря ERP-системе компания будет принимать более обоснованные решения за счет централизации информации из различных

областей. Вы будете получать эту информацию в режиме реального времени, что позволит вам планировать будущие действия, которые могут принести пользу бизнесу. Программное обеспечение ERP помогает повысить производительность труда сотрудников. Оптимизация процессов позволяет улучшить контроль над покупками, продажами и заказами, тем самым сокращая время операций и финансовые циклы.

Прослеживаемость и контроль, предлагаемые ERP-системой, позволяют отслеживать все производственные процессы, от получения сырья до изготовления изделия и последующей продажи клиенту.

Решение ERP улучшает взаимодействие между отделами, поскольку все они имеют доступ к одной и той же базе данных, которая обновляется в режиме реального времени. Это позволяет более эффективно управлять экономическими ресурсами компании, обладая всей интегрированной информацией и историей выполненных операций мы можем корректировать состояние материальных и товарных запасов.

Внедрение ERP-системы улучшает информационную безопасность благодаря ролям, которые приобретает каждый пользователь, подключающийся к системе. Пользователь работает только с той информацией, которую администратор предварительно определил в его профиле, ограничивая то, что ему не нужно. Как результат предотвращение кражи или потери данных.

ERP-систему можно адаптировать и масштабировать, поскольку она состоит из модулей, которые адаптируются к потребностям любой компании в зависимости от ее развития.

ERP-система помогает снизить затраты и повысить конкурентоспособность. Уменьшение количества ошибок и дублирования задач снижает затраты. Это помогает увеличить прибыль и снизить цену на каждый продукт. С ERP-системой могут быть интегрированы различные инструменты или программы, такие как интернет-магазин или мобильное приложение.

Актуальность исследования заключается в том, что необходимо внедрение ERP-системы, которая лучше всего подходит для процессов и характеристик конкретного предприятия. В данном исследовании проанализирована организационная структура, бизнес-процессы конкретного предприятия, выявлены процессы требующие реорганизации. Решаются вопросы проектирования функциональной структуры информационной системы, распределенной базы данных и расчет параметров ее функционирования Произведен выбор наиболее подходящей по функциональности ERP-системы для конкретного предприятия с территориально-распределенной структурой.

Целью исследования является проектирование корпоративной информационной системы ERP класса для управления сетью территориально распределенных филиалов производственно-торгового предприятия легкой промышленности, разработка проекта внедрения ERP-системы и внедрение на практике.

Объектом исследования являются вопросы проектирования корпоративной информационной системы класса ERP для управления сетью территориально распределенных филиалов производственно-торгового предприятия.

Предметом исследования является специализированный интегрированный пакет прикладного программного обеспечения - ERP-система.

Гипотеза исследования: Поддержка принятия решений в управлении ресурсами территориально-распределенного предприятия будет более эффективной, если:

– для управления ресурсами используется концепция ERP-систем;

– определено понятие, структура, элементы ERP-систем и область их применения;

– определены критерии выбора конкретной ERP-системы;

– проведено моделирование предметной области автоматизации;

– выявлены бизнес-процессы требующие перепроектирования;

– разработана модель распределенной базы данных предприятия;

– выбрана и спроектирована ERP-система под бизнес-процессы предприятия.

Для достижения поставленной цели необходимо решить следующие задачи:

– исследование предметной области автоматизации предприятия;

– построить функциональную модель структуры информационной системы;

– проанализировать известные на рынке ERP-системы;

– выявить критерии выбора программного продукта;

– сформировать модель распределенной базы данных предприятия, состоящего из головного офиса и территориально-распределенных филиалов;

– спроектировать и внедрить корпоративную информационную систему ERP класса для управления сетью территориально распределенных филиалов производственно-торгового предприятия легкой промышленности.

Теоретической и методологической основой исследования являются теории баз данных и вычислительных сетей, методов системного анализа, объектно-ориентированного подхода и математического программирования.

Научная новизна исследования заключается в том, что на основе анализа и обобщения подходов к проектированию автоматизированных информационных систем разработана модель для корпоративной информационной системы предприятия для управления сетью территориально распределенных филиалов производственно-торгового предприятия легкой промышленности; определены оптимальные параметры функционирования распределенной базы данных; выявлены особенности производственно-торгового предприятия как объекта управления.

Теоретическая значимость заключается в определении роли информационных технологий в поддержке принятия решений в управлении ресурсами предприятия.

Практическая значимость заключается во внедрении корпоративной информационной системы ERP класса и оптимизации бизнес-процессов конкретного предприятия.

На защиту выносятся:

– концептуальные положения создания корпоративной информационной системы ERP класса;

– модель ERP-системы для предприятия с территориально-распределенной структурой;

– спроектированная и внедренная ERP-система для конкретного предприятия.

Апробация результатов исследования осуществлялась в Обществе с ограниченной ответственностью Торговый Дом «Купеческий»

В результате апробации практическая реализация ERP-системы подтвердила востребованность и эффективность данной системы на конкретном предприятии. Работа включает результаты теоретической и практической деятельности в области исследования и практического применения ERP-систем.

Объем и структура диссертации: состоит из введения, четырех глав, заключения, списка литературы и приложений. Работа изложена на 202 страницах, содержит 37 рисунков и 11 таблиц.

В первой главе рассматриваются научно-методологические основы создания и использования корпоративной информационной системы ERP класса для предприятия с территориально-распределенной структурой.

Во второй главе представлены анализ организационной структуры предприятия, бизнес-процессов предприятия в нотации IDEF0, логическая модель ERP-системы, модель распределенной базы данных предприятия.

В третьей главе описан проект внедрения ERP-системы.

В четвертой главе проводится апробация результатов внедрения ERP-системы.

Определения, обозначения и сокращения

ERP (англ. Enterprise Resource Planning, планирование ресурсов предприятия) - организационная стратегия интеграции производства и операций, управления трудовыми ресурсами, финансового менеджмента и управления активами, ориентированная на непрерывную балансировку и оптимизацию ресурсов предприятия посредством специализированного интегрированного пакета прикладного программного обеспечения, обеспечивающего общую модель данных и процессов для всех сфер деятельности. ERP - системой называют конкретный программный пакет, который реализует ERP-стратегию.

Управление цепочками поставок (англ. supply chain management, SCM) - управленческая концепция и организационная стратегия, заключающаяся в интегрированном подходе к планированию и управлению всем потоком информации о сырье, материалах, продуктах, услугах, возникающих и преобразующихся в логистических и производственных процессах предприятия, нацеленном на измеримый совокупный экономический эффект (снижение издержек, удовлетворение спроса на конечную продукцию).

MRP (англ. Material Requirements Planning - планирование потребности в материалах) - система планирования потребностей в материалах, одна из наиболее популярных в мире логистических концепций, на основе которой разработано и функционирует большое число микрологистических систем. На смену ей пришла более развитая концепция MRP II и системы класса ERP.

Глава 1 Теоретико-методологические основы построения корпоративных информационных систем

1.1 Современные концепции о построении корпоративных информационных систем

1.1.1 Определение единого информационного пространства

Банки и базы данных, технологии их обслуживания, использующие информационные и телекоммуникационные системы и сети, функционирующие на основе общих принципов, которые обеспечивают информационное взаимодействие составляют единое информационное пространство предприятия (ЕИП) [25]. Построение такого пространства является актуальным и дает возможность упорядочить работу сотрудников и подразделений, принимать решения с более высокой скоростью. Данная задача может быть разрешена в едином аппаратно-программном решении.

Его компонентами являются организационные структуры, информационные ресурсы, средства информационного взаимодействия. Организационные структуры обеспечивают работу с информацией. Информационные ресурсы - данные на носителях информации. Доступ к информационным ресурсам обеспечивают средства информационного взаимодействия.

Структурированность - характерное свойство единого информационного пространства. Его элементы выделены, с установлением связей между ними, введены соответствующие обозначения, упорядочены элементы и связи.

Высокий уровень структурированности дает возможность представить информацию в виде документов, позволяет манипулировать данными. Выделяют следующие стадии структурированности информационного пространства:

– неструктурированное;

– слабо структурированное;

– структурированное;

– формализовано-структурированное;

– машинно-структурированное.

В роли информационных ресурсов могут выступать кроме данных и различные прикладные программы. Из других информационных систем ЕИП доступны приложения, в которых реализуется часть методов обработки данных. При взаимодействии информационных систем первая использует сервисы, предоставляемые второй, и получает обработанные данные, которые может подвергнуть дальнейшей обработке. Такой подход соответствует одноранговой, распределенной архитектуре взаимодействия. При данной архитектуре приложения из разных информационных систем могут выступать как в роли сервера, так и клиента по отношению друг к другу. Это позволяет избежать дублирования приложений. Распределение приложений по различным информационным системам приводит к оптимальному балансу загрузки аппаратной части и приложений. Следовательно, позволяет использовать информационные ресурсы более эффективно.

Только приложение, обрабатывающее данные из определенной базы данных, должно знать схему этой базы данных. Система-клиент использует сервисы системы-сервера, которая обеспечивает методы обработки данных. Поэтому система-клиент не должна знать схему базы данных системы-сервера. Это решает проблему изменения схемы удаленной базы данных. Учитывая, что в информационных системах сосредоточены не только данные, но и методы их обработки это позволяет уменьшить затраты на сопровождение и модификацию информационных систем ЕИП.

Единое информационное пространство характеризуется следующим:

– обеспечивает надежность, отказоустойчивость и безопасность;

– основано на международных и промышленных стандартах;

– дает возможность интегрировать старые (унаследованные) приложения в новые информационные системы;

– обеспечивает расширяемость системы (новые компоненты легко добавляются в существующие ИС);

– предоставляет возможность формализованные знания накапливать, развивать и тиражировать;

– позволяет снизить затраты на создание информационных систем.

Единая (электронная) форма представления данных является главным требованием для формирования ЕИП. Это требуется для хранения данных на машинных носителях. Развитие ЕИП предусматривает оперативный доступ к существующим информационным ресурсам и включению их в единое информационное пространство. Для этого необходим перевод данных на традиционных носителях в электронную форму, и предоставление доступа к имеющимся данным в электронной форме [27].

Построение ЕИП возможно при соблюдении стандартов на взаимодействие между информационными системами, и приложениями входящих в их состав. Необходимы работы по сертификации и стандартизации систем и средств информатизации. Требуется инструментальная и нормативно-методическая поддержка для сертификации средств информационных технологий. Нужно применение международных стандартов, регламентирующие форму представления информации, протоколов связи и коммуникаций для обеспечения возможности пользователям входить в корпоративную информационную систему со своих оконечных устройств.

Стратегия построения автоматизированных систем поддержки единого информационного пространства состоит из шагов:

– создание динамических средств описания информационных объектов (с возможностью расширения описаний по мере увеличения знаний об объектах, с соответствующим отображением в структуре базы данных);

– создание инструментального набора средств обработки информации по информационным объектам (с возможностью его конфигурирования);

– создание системы поддержки ЕИП с эффективными средствами расширения (целостность, репликация, управление);

– создание технологии миграции информации по информационным объектам при расширении ЕИП на новые узлы (узел - субъект осуществляющий обработку информации);

– разработка технологии создания приложений, обрабатывающих связанные данные по нескольким объектам.

Основное свойство программных средств поддержки ЕИП - простое и эффективное информационное и пространственное расширение (масштабирование - например, создание новых рабочих мест).

Процесс построения системы начинается с формирования ядра единого информационного пространства - разработка информационных объектов, их начального описания, разработка инструментальных средств. Затем производится развертывание единого информационного пространства по технологиям «сверху-вниз» или «снизу-вверх».

Технология «сверху-вниз» начинает построение ЕИП с формирования узлов интеграции. Распространение единого информационного пространства происходит по древовидной структуре - первый уровень узлов интеграции связывается ветвями с ядром единого информационного пространства, затем второй уровень интеграции соединяется ветвями с предыдущим узлом, и так далее до формирования автоматизированного рабочего места.

Для начала работы единого информационного пространства по технологии «снизу-вверх» можно использовать один (несколько) вычислительных узлов (серверов), которые формируют программно-аппаратное ядро. В данных узлах образуется центр обработки, система хранения данных, сервер приложений, репозиторий. Построение ядра единого информационного пространства аналогично мероприятиям при технологии «сверху-вниз», но с преимуществом - при ограниченном числе

узлов на начальном этапе наблюдается некоторое снижение времени реализации и общих затрат. Составляющие единого информационного пространства производственного предприятия представлены на рисунке 1.

Рисунок 1 - Единое информационное пространство

На рисунке 1 представлено единое информационное пространство предприятия в виде взаимодействующих частей. Показано как потребности и внешние связи влияют непосредственно на производство.

Инженерные данные на рисунке представляют собой - PDM (Product Data Management, система управления инженерными данными). Управление инженерными данными делится на управление конструкторскими спецификациями и технологическими спецификациями. Управление конструкторскими спецификациями - занимается определением изделий в системе; управление технологическими спецификациями - задачи определения процессов изготовления изделий.

Совокупность распределенных баз данных с едиными правилами хранения, поиска, обновления, передачи информации образует единое информационное пространство. Данная среда дает возможность осуществлять безбумажное взаимодействие между подразделениями компании.

Существенным компонентом единого информационного пространства является система «Планирование и управление ресурсами предприятия» (ERP- система).

В данном подразделе представлено содержание и понятие единого информационного пространства, показано место ERP системы в нем. Перейдем к описанию особенностей ERP-систем.

1.1.2 Определение и область применения ERP-систем

Набор интегрированных приложений создающих информационное пространство для автоматизации задач планирования, управления, учета, контроля, анализа основных бизнес-операций предприятия называется ERP-системой (системой планирования ресурсов предприятия).

Система планирования ресурсов предприятия реализует следующую функциональность:

- подсистема управления информацией о продукции, технологии;
- модуль управления затратами;
- управление финансами;
- подсистема управления кадрами;
- управление проектами и задачами;
- прогнозирование.

В основу ERP-системы положен принцип единого хранилища данных, содержащего всю бизнес-информацию компании: данные о сотрудниках и производстве, финансовую информацию.

Системы автоматизирующие учет на предприятиях сосредотачивают в себе значительные объемы оперативной информации. Руководство и аналитики компаний сталкиваются с вопросами своевременного получения

аналитической информации необходимой для принятия управленческих решений.

Интегрированные средства подготовки аналитической информации позволяют подготавливать необходимые виды отчетности и являются инструментальными средствами для принятия решений.

1.1.3 Структура и элементы ERP-систем

ERP-системы включают различные функциональные модули, которые осуществляют автоматизацию бизнес-процессов предприятия. Каждый такой модуль ориентирован на определенный бизнес-процесс или специфическую область деятельности организации.

ERP-системы появились как результат эволюционного развития систем класса MRP/MRPII, поэтому включают в себя элементы и этих систем.

Структуру ERP-системы по составу применяемых модулей можно разбить на базовые и расширенные элементы.

К базовым элементам системы относятся функции управления производством: управление запасами и закупками, спецификациями изделий, укрупненное и детальное планирование производственных операций и мощностей, маршрутизация производства, планирование потребностей в материальных ресурсах, построение основного плана производства.

Расширенные элементы - это функции, занимающиеся обеспечением производства. Как правило, они представлены в виде отдельных модулей.

К расширенным элементам относятся:

– модуль CRM, управление взаимоотношениями с заказчиками);

– модуль SCM, управление цепочками поставок - осуществляет прогнозирование спроса, планирование и управление логистикой (производственная и складская логистика, сбыт продукции и логистика внешних поставок), управление закупками и поставщиками);

– управление жизненным циклом изделия (отслеживание жизненного цикла изделия от разработки до утилизации - управление данными о

продукте, потребностями клиентов-заказчиков, жизненный цикл оборудования, контроль процесса проектирования);

– управление человеческими ресурсами (планирование состава сотрудников, кадровый учет, графики работы, учет рабочего времени, расчет заработной платы);

– управление финансами (управление денежными средствами, расчеты с дебиторами и кредиторами, бухгалтерский и налоговый учет, составление финансовой отчетности).

Состав элементов и модулей может отличаться в разных ERP-системах, например, ряд производителей дополняют свои системы модулями управления проектами и качеством. Внедрение системы может проводиться как комплексно, реализуя весь функционал, так и по отдельным модулям.

1.1.4 Функциональные модули ERP-систем

В соответствии со стандартом на информационные системы класса MRPII системы планирования реального времени имеют следующие группы функций:

– подсистема планирования продаж и производства;

– подсистема управления спросом;

– модуль составления плана производства;

– подсистема планирования потребностей в материалах;

– модуль учета спецификаций изделий;

– подсистема управления запасами;

– модуль учета планируемых поступлений по открытым заказам;

– подсистема оперативного управления производством;

– модуль планирования потребностей в мощностях;

– подсистема управления входным/выходным материальным потоком;

– материально-техническое обеспечение;

– планирование распределенных ресурсов;

– модуль планирования и управления инструментальными средствами;

– финансовое планирование;

– моделирование;

– оценка результатов деятельности.

В дополнение к базовым модулям обязательны следующие:

– модуль прогнозирования;

– модуль управления проектами и программами;

– модуль управления затратами;

– модуль управления финансами;

– модуль управления кадрами.

Следующие функциональные возможности необходимы в современных ERP-системах:

– CRM (Customer Relationship Management, управление взаимоотношениями с клиентами);

– SCM (Supply Chain Management, управление и оптимизация цепочек поставок);

– APS (Advanced Planning and Scheduling, синхронное планирование и оптимизация);

– PDM (Product Data Management, управление данными об изделии);

– ES (Electronic Commerce, электронная коммерция);

– Business Intelligence (решения на OLAP и DSS (Decision Support Systems);

– SCE (Stand Alone Configuration Engine, модуль конфигурирования системы);

– FRP (Finite Resource Planning, окончательное (детализированное) планирование ресурсов).

На рисунке 2 представлен типовой функционал, реализуемый ERP-системой.

Рисунок 2 - Типовой функционал ERP-системы

На рисунке 2 представлены функциональные модули, из которых состоит ERP-система.

ERP-системы построены на основе общей архитектура. Первый элемент общей архитектуры, это платформа - среда для работы для работы модулей и компонентов. Платформа предоставляет базовые возможности. Изменения в код платформы вносит разработчик (пользователи и специалисты по внедрению не имеют возможности редактировать код платформы).

Платформа включает:

– ядро (программная среда для работы (можно добавить какие-то надстройки и компоненты));

– базовый функционал, который встраивается в платформу и не может быть отключен (базовые справочники и функции (справочник пользователей с правами доступа, справочник товаров, справочник клиентов и т.д.));

Второй элемент общей архитектуры - управление данными. База данных, методы хранения и обработки данных: хранилище данных на сервере, программное обеспечение для работы с базами данных (например,

SQL), инструменты для обработки, интерпретации данных, отправки данных в программные модули.

Третий элемент общей архитектуры - модульная структура ERP-системы. Модуль – это компонент, который может подключается к платформе. Модули работают с единой базой данных и применяют базовый функционал (при необходимости), работают независимо и могут подключаться, и отключаться (если использование модуля стало ненужным). Важная отличительная черта ERP-систем - модульная структура.

Использование ERP-системы способствует эффективной реализации стратегии предприятия, исполнение которой приводит к правильному планированию и управлению ресурсами предприятия. Как следствие достигается оптимизация работы подразделений предприятия (максимальная согласованность между ними и сокращение административных издержек). ERP-система улучшает прозрачность бизнес-процессов.

Внедрение ERP-системы решает проблемы с упорядочиванием и поиском необходимой информации, приводит к сокращению времени на заполнение документации, избавляет от возможных ошибок, увеличивается скорость документооборота на предприятии. Достигается организация единого информационного пространства между головным отделением и филиалами предприятия. Повышается актуальность и достоверность данных. Все это приводит к увеличению скорости принятия в компании управленческих решений.

Внедрение ERP-системы повышает дисциплинированность и точность в исполнении бизнес-процессов. Повышение конкурентоспособности достигается не только за счет оптимизации бизнес-процессов, но и за счет использования инструментов ERP-систем. Передовые инструменты планирования, моделирования и анализа приводят к общему сокращению расходов предприятия. Данные инструменты позволяют осуществить оптимизацию ресурсов производственной, финансовой, логистической и транспортной сфер организации.

Современные предприятия территориально распределены. Важно обеспечить филиалы полноценным доступом к общему информационному хранилищу данных. Это реализуется в ERP-системах самыми передовыми сетевыми технологиями, задействованными при разработке, которые также предусматривают разграничение прав доступа пользователей к хранящимся в информационном хранилище сведениям.

Важно понимать, что внедрение ERP-системы - процесс, включающий не только доработку программного обеспечения, но и проведение мероприятий по оптимизации основной деятельности предприятия (реинжиниринг бизнес-процессов), направленные на соответствие бизнес-логике внедряемой системы.

В данном разделе представлено понятие ERP-систем, выделены основные элементы, рассмотрен перечень функциональных модулей и дано описание архитектуры данных систем. Перейдем к разделу 1.2 «Распределенные корпоративные информационные системы».

1.2 Распределенные корпоративные информационные системы

Корпоративная информационная система предприятия должна обеспечивать работу нескольких территориально-распределенных подразделений. Для обеспечения бесперебойного доступа к базам данных информационные ресурсы должны быть распределенными.

Распределенной базой данных называют совокупность логически взаимосвязанных баз данных, распределенных в компьютерной сети. Она обладает характеристиками:

– локальная автономность: под локальным владением и управлением должны находиться локальные данные;

– децентрализация функций: на какой-либо специально выделенный центральный узел не должен возлагаться никакой конкретный сервис;

– непрерывность функционирования: система должна функционировать при изменении параметров конкретных узлов распределенной базы данных (добавления новых узлов, изменения СУБД на каком-то узле и прочее);

– обеспечивает независимость от местоположения: не обязаны знать пользователи и приложения о том, где физически расположены данные;

– обеспечивает независимость от фрагментации: пользователи и приложения могут ничего не знать о том, как обрабатываются фрагменты базы данных;

– предоставляет независимость от тиражирования;

– обеспечивает распределенную обработку запросов;

– обеспечивает управление распределенными транзакциями;

– предоставляет независимость от оборудования;

– обеспечивает независимость от операционных систем;

– обеспечивает независимость от локальных СУБД;

– предоставляет независимость от сети.

На практике системы не удовлетворяют всем вышеперечисленным характеристикам, так как сложно обеспечить выполнение всех вышеперечисленных условий.

Разделяют распределенные базы данных на однородные и неоднородные.

Построение распределенных баз данных может осуществляться по принципу «сверху-вниз» и «снизу-вверх». Различают формы распределения данных: фрагментация и тиражирование. В случае фрагментации данные фрагментируются - разделяются на порции, распределяемые между множеством физических ресурсов. При тиражировании данные дублируются на нескольких узлах.

В подразделе дано понятие распределенных информационных систем, разъяснено, почему они применяются для управления сетью филиалов предприятия. Перейдем к обзору ERP-систем по функциональности.

1.3 Обзор ERP-систем в России

Представим обзор наиболее популярных ERP-систем отечественного производства. В последнее время в сегменте корпоративных систем российского производства наметились положительные тенденции - появились достойные системы ERP класса.

Наиболее популярные решения на рынке систем класса ERP в России представлены в таблице 1.

Таблица 1 - Популярные ERP-системы в России

Отечественные ERP-системы
Галактика ERP 1С: ERP управление предприятием 2 Парус Альфа

Перейдем к сравнению ERP-систем по функциональности.

Осуществим сравнение ERP-систем по критериям основных функциональных модулей. Это даст понимание насколько та или иная система соответствует современным требованиям к системам планирования ресурсов предприятия. Итак, сравнение систем выполним по следующим аспектам:

- планирование продаж и производства;
- управление кадрами;
- управление проектами и программами;
- управление складом;
- управление производством;
- контроль материальных потоков;
- финансы;
- отраслевые решения;

– управление потоком операций (Workflow);
– моделирование и прогнозирование;
– управление и оптимизация цепочек поставок;
– управление взаимоотношениями с клиентами (CRM);
– поддержка MS Office;
– управление взаимоотношениями с клиентами (CRM);
– «облачные» технологии и мобильное использование;
– бизнес-аналитика (BI) и OLAP;
– электронный документооборот;
– безопасность и администрирование;
– средняя продолжительность внедрения.

По результатам сравнения подготовлена таблица А.1 - Сравнение функциональных возможностей российских и импортных ERP-систем, которая представлена в Приложении А.

В данном подразделе выполнено сравнение функциональных возможностей российских ERP-систем. Система 1С:ERP является лидером отечественных ERP-решений.

В целом дано понятие ERP-систем и их особенностей, проведен сравнительный анализ популярных ERP-систем по критерию функциональности.

Выводы по главе 1

В главе 1 представлены понятия единого информационного пространства предприятия, распределенных корпоративных информационных систем, освещена концепция ERP-систем - архитектуры их построения, описаны их особенности. Дано сравнение известных на российском рынке ERP-систем по критерию функциональности.

Глава 2 Анализ деятельности предприятия ООО Торговый Дом «Купеческий» и моделирование бизнес-процессов

2.1 Технико-экономическая характеристика деятельности ООО ТД «Купеческий»

Подготовка магистерской диссертации по разработке проекта внедрения ERP-системы выполнена на основании данных о деятельности ООО Торговый Дом «Купеческий» (г. Киров).

Компания ООО ТД «Купеческий» - производитель одежды и аксессуаров.

Компании ООО ТД «Купеческий» удалось создать производство замкнутого технологического цикла, оснащенное современным оборудованием, позволяющим разрабатывать современную одежду.

Сейчас компания выпускает широкий ассортимент одежды для охранных структур, униформы для медицинских предприятий и корпоративных заказчиков; экипировки для охоты, рыбалки и активного отдыха, также реализует широкий ассортимент одежды и обуви массового назначения.

Рассмотрим организационную структуру предприятия. В рамках этой структуры протекает весь управленческий процесс: движение потоков информации, контроль достоверности и анализ, принятие управленческих решений, в котором участвует весь персонал. Структура необходима для того, чтобы все протекающие в организации процессы осуществлялись своевременно и качественно.

Ключевыми понятиями структуры управления являются элементы, связи (отношения), уровни и полномочия. Элементами структуры управления могут быть как отдельные работники, так и службы, в которых заняты специалисты, выполняющие свои функциональные обязанности. Отношения между элементами структуры управления поддерживаются связями, которые

подразделяют на вертикальные и горизонтальные (линейные и функциональные).

Организационная структура организации представлена на рисунке 3.

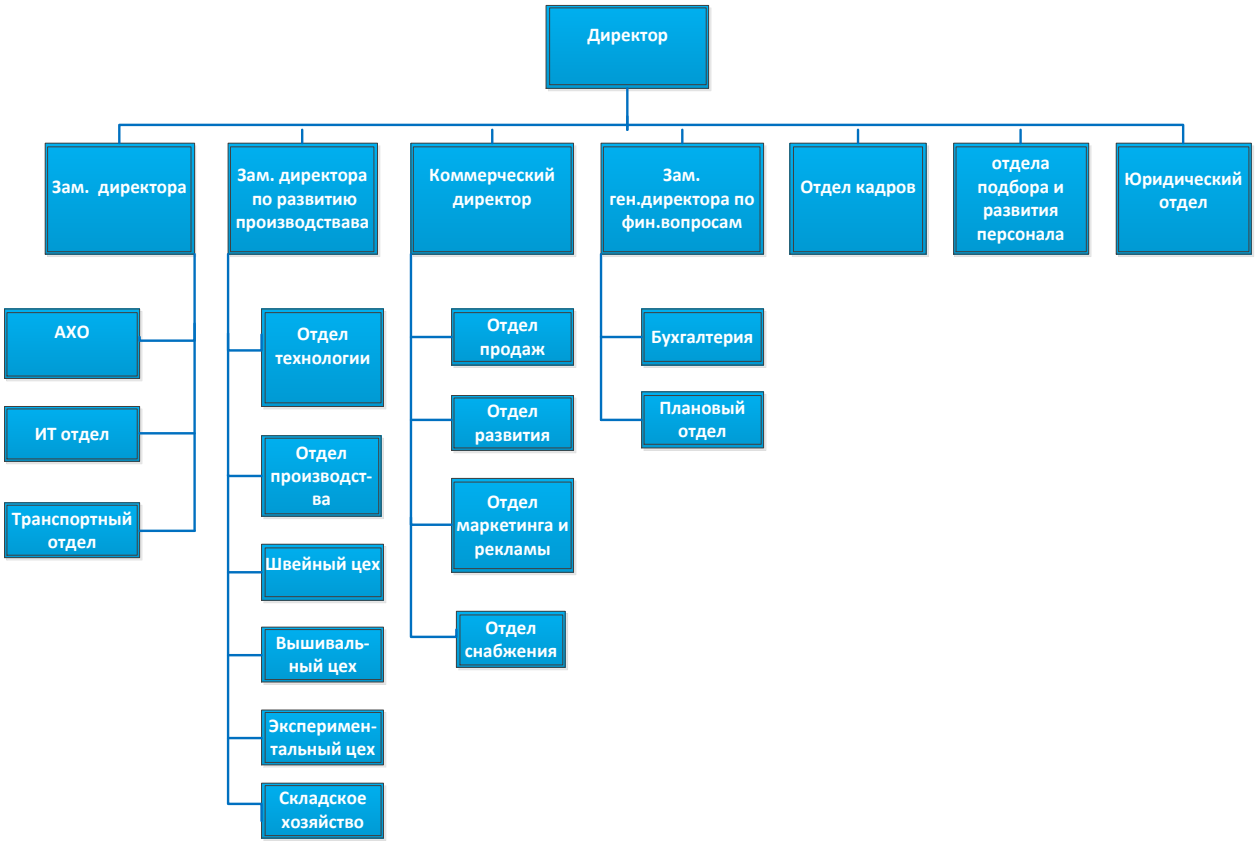


Рисунок 3 - Организационная структура предприятия ООО ТД «Купеческий»

Все бизнес-процессы можно разделить на три группы (см. рисунок 4).

Рисунок 4 - Виды бизнес-процессов

Основные процессы (создания стоимости) объединяют задания и работу для выполнения определенных требований клиента с применением ключевых производственных компетенций. Они являются стратегически важными и в то же время специфическими (уникальными, так как, например, вследствие применения фирменных знаний их сложно скопировать). К ним относятся:

- обработка и выполнение заказа;
- разработка, проектирование и дизайн продукта;
- производство и др.

Управляющие процессы содержат в себе задачи и деятельность, направленные на долгосрочное развитие компании и реализацию целей компании. К ним относятся:

- стратегическое развитие компании;
- долго- и среднесрочное планирование в компании;
- развитие персонала;
- инвестиционное планирование;
- мотивация персонала и др.

Обеспечивающие процессы содержат необходимые задания и работы для поддержания основных процессов, но не приводящие к непосредственной ценности для клиента, например:

- обработка данных;
- техническое обслуживание;
- логистика;
- административные процессы и др.

Производственный цикл предприятия относится к мелкосерийному производству. Это связано с тем, что зачастую выполняются индивидуальные заказы на пошив изделий, вышивку и т.д. Бизнес-процесс каждого отдельного заказа уникален, поэтому производственные цепочки разных заказов отличаются. Рассмотрим функционально-перекрёстную блок-схему

процесса изготовления индивидуального заказа клиента - юридического лица (рисунок 5).

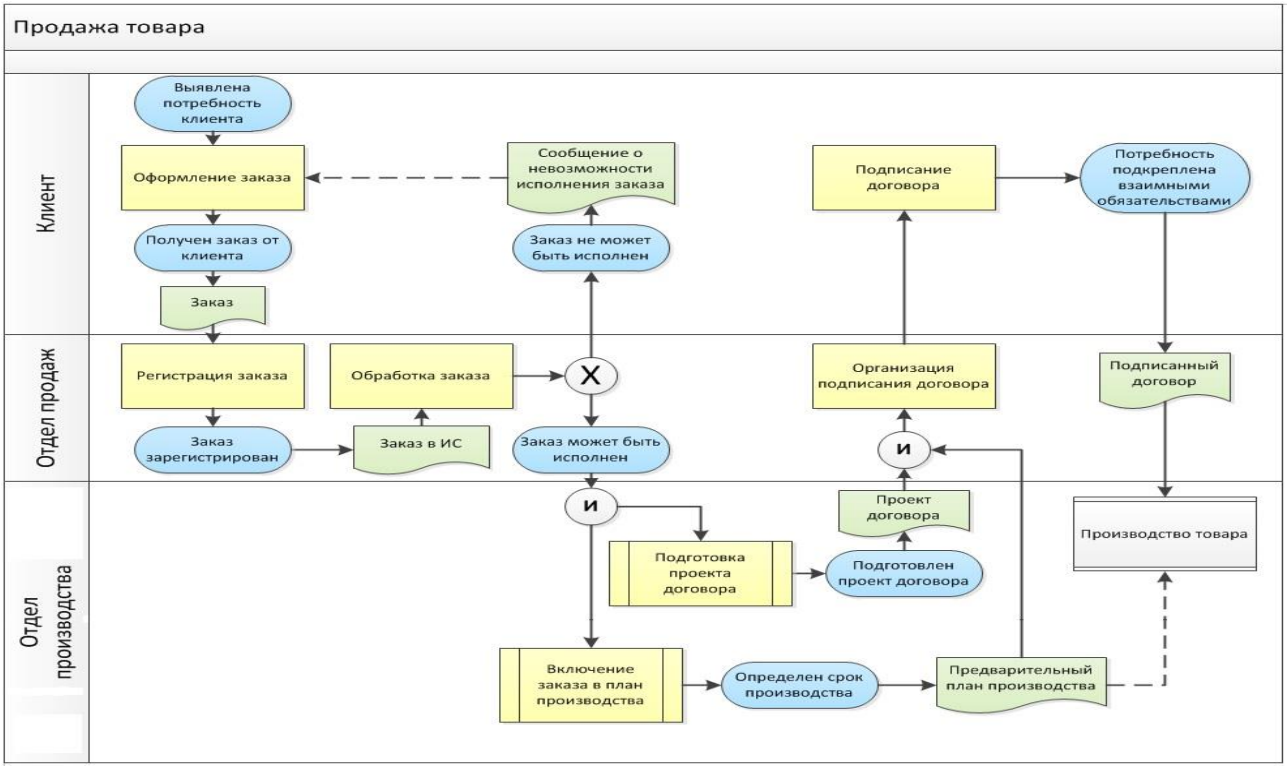


Рисунок 5 - Схема производства индивидуального заказа

Перейдем к концептуальному моделированию деятельности ООО ТД «Купеческий».

2.2 Концептуальное моделирование деятельности ООО ТД «Купеческий»

2.2.1 Разработка и анализ модели бизнес-процессов «Как есть»

Концептуальная функциональная модель представлена на рисунке 6.

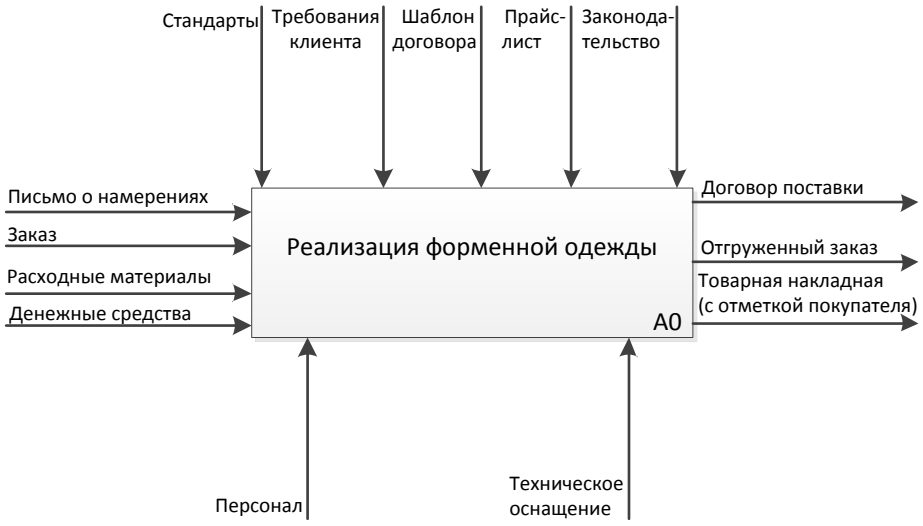


Рисунок 6 - Контекстная диаграмма («as is»)

Деятельность компании по реализации форменной одежды представлена функциональным блоком (прямоугольник на рисунке 1.5). Входными потоками являются:

– письмо о намерениях заключить сделку;

– заказ клиента;

– расходные материалы необходимые для производства заказа;

– денежные средства клиента - оплата произведенного заказа.

Выходными потоками являются:

– договор поставки, регламентирующий отношения компании с клиентом;

– товарная накладная, отражающий факт получения клиентом заказанной продукции.

Потоками управления являются:

– стандарты - ГОСТы и технические регламенты швейной промышленности, касающиеся производства верхней одежды;

– требования клиента по пошиву изделий;

– шаблон договора - основа для заключения рамочного договора между компанией и клиентом;

– прайс-лист - продуктовое предложение фирмы;

– законодательство - законы, регулирующие хозяйственную деятельность предприятий (в частности, в сфере производства одежды).

Потоками механизмов являются:

– персонал компании, вовлеченный в процессы производства и реализации продукции;

– техническое оснащение - оборудование необходимое для производства и реализации форменной одежды.

Осуществим декомпозицию концептуальной диаграммы (рисунок 6).

Диаграмма декомпозиции (уровень 1) состоит из пяти функциональных блоков: формирование договора, прием заказа, оплата заказа, изготовление заказа, отгрузка заказа.

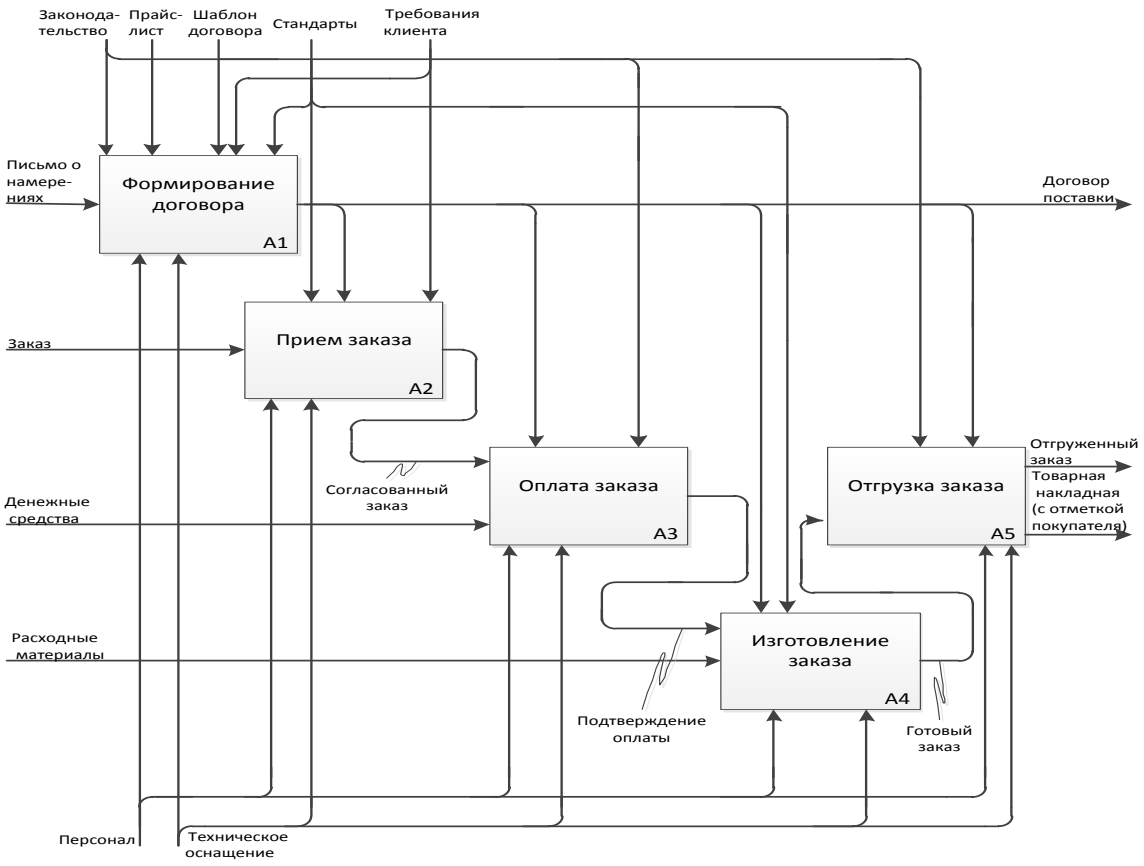


Рисунок 6 - Диаграмма декомпозиции A0 («as is»)

Функция «Формирование договора» регламентирует отношения компании и покупателя - в спецификации к договору отражается информация

о моделях, подлежащих реализации (изготовлению), требования клиента к моделям одежды.

Важнейшей с точки зрения формирования денежного потока для фирмы является функция «Прием заказа». Здесь учитываются требования клиента, возможности осуществления заказа с учетом производственной программы предприятия. Затем данные требования клиента учитываются на этапе производства заказа.

Затем следует функция «Оплата заказа» - денежные средства для осуществления заказа. Далее функция «Изготовление заказа», которая является самой основной с точки зрения деятельности фирмы. На этом этапе учитываются требования клиента к изделию, которые уже сформулированы в виде договора поставки и производственной спецификации на изготовление модели одежды. Последним этапом является функция «Отгрузка заказа». На этом этапе производятся складские операции по отгрузке заказа клиенту.

Произведем декомпозицию блоков «Прием заказа» и «Изготовление заказа» (рисунки 7, 8).

Функция «Прием заказа» состоит из трех функциональных блоков:

– определение требований клиента;
– рассмотрение вариантов исполнения заказа;
– выбор варианта исполнения заказа.

На этапе «Определение требований клиента» пожелания клиента формализуются в более точные описания, с которыми могут работать технологи и специалисты по производству.

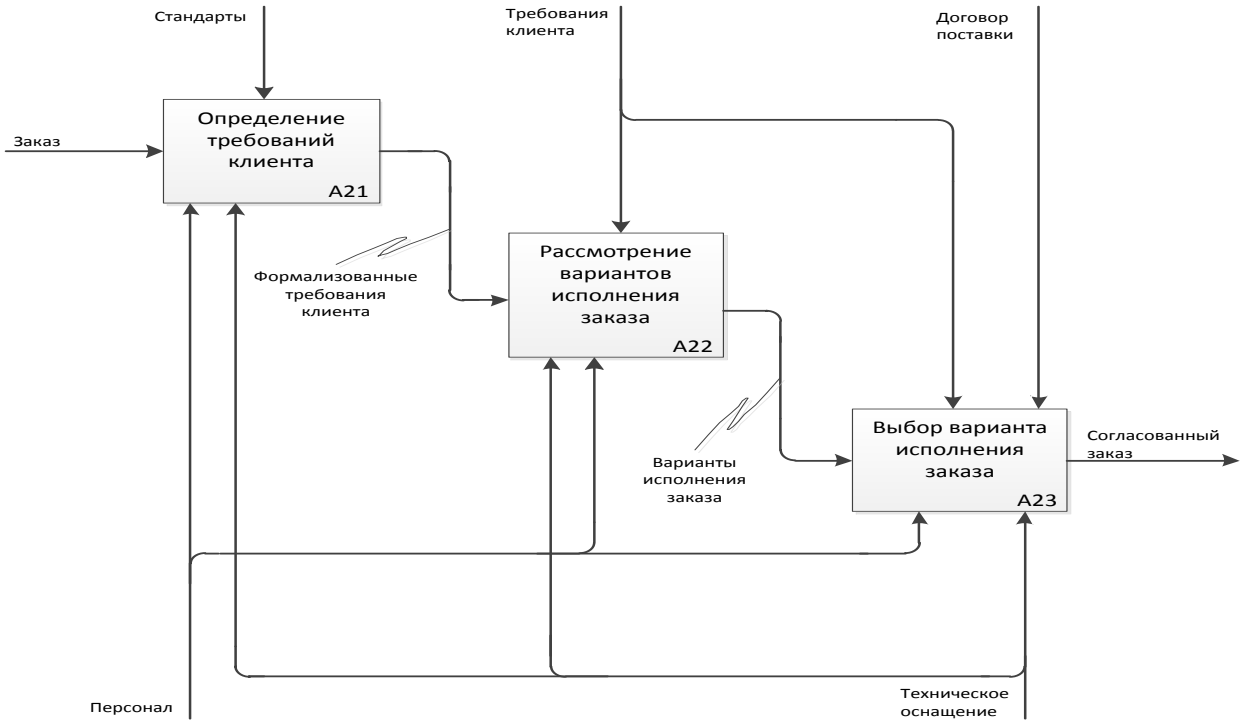


Рисунок 7 - Диаграмма декомпозиции A2 процесса приемки заказа («as is»)

Декомпозиция функции «Изготовление заказа» представлена на рисунке 8.

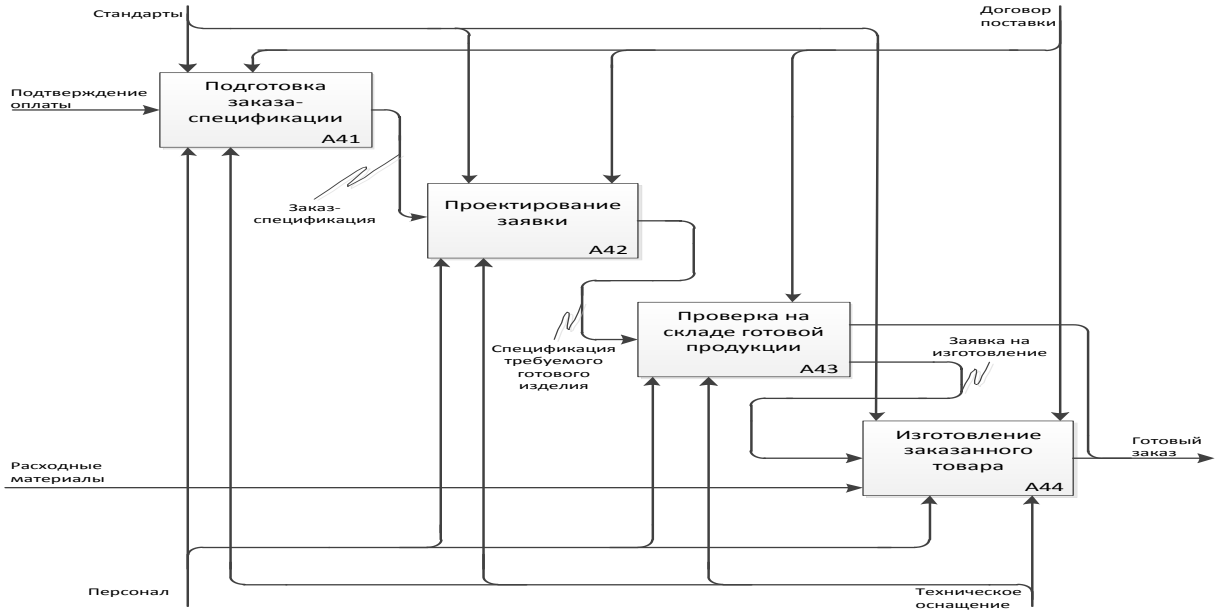


Рисунок 8 - Диаграмма декомпозиции процесса A4 («as is»)

Состоит из четырех блоков:

– подготовка заказа-спецификации;

– проектирование заказа;

– проверка на складе готовой продукции;

– изготовление заказанного товара.

Этап «Подготовка заказа-спецификации» состоит в составлении документа, в котором отражено как будет выглядеть готовое изделие. Данный документ содержит техническое описание и используется специалистами по производству для изготовления изделия.

Этап «Проектирование заказа» осуществляют менеджеры по производству с привлечением технологов швейного производства. На выходе имеем заявку на производство - документ, по которому будет изготавливаться изделие. Данный документ содержит исчерпывающий перечень характеристик, которым должно соответствовать готовое изделие.

Следующий этап «Проверка на складе готовой продукции» состоит в том, что заказанное изделие может быть уже изготовлено и находится на складе. В случае наличия изделия процесс «Изготовление заказанного товара» не запускается и осуществляется отгрузка имеющегося товара со склада готовой продукции.

Этап «Изготовление заказанного товара» заключается в производстве требуемого изделия. По его завершению готовое изделие перемещается на склад готовой продукции, с которого затем будет произведена отгрузка исполненного заказа.

Отразим на DFD-диаграмме потоки данных сопровождающие процесс обслуживания клиентов (рисунок 9).

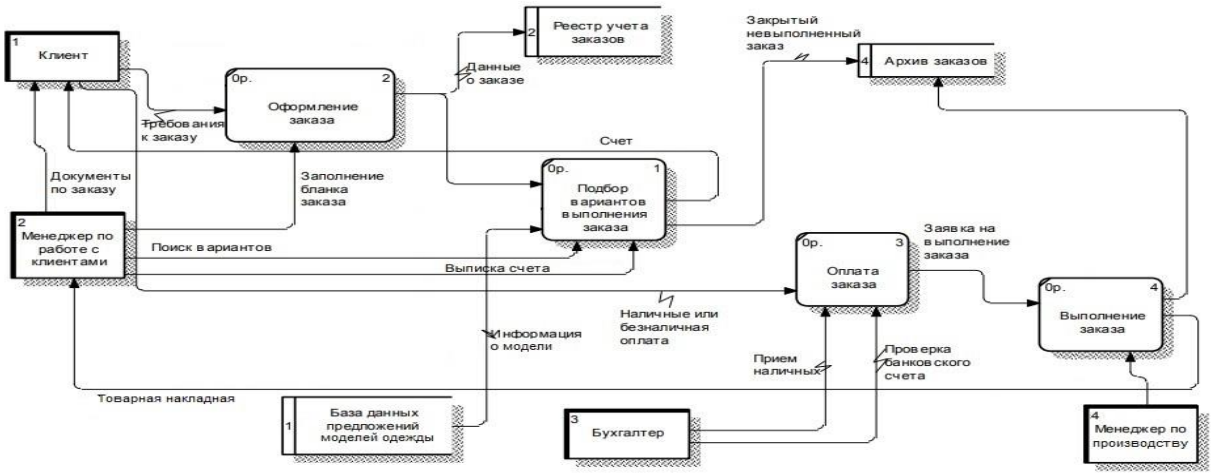


Рисунок 9 - DFD-диаграмма потоков данных («as is»)

В результате анализа существующих бизнес-процессов предприятия были выявлены следующие недостатки:

– история общения с клиентами разобщена и регистрируется только бухгалтерская составляющая процесса сделки, важно видеть какие требования имел клиент и что он выбрал для заказа, возможность осуществления и изготовления заказов по аналогии;

– разрозненная клиентская база - данные о клиентах хранятся в разных источниках, как следствие теряются при смене менеджеров, отсутствует возможность комплексного анализа данных;

– существуют риски потери информации при передаче её между подразделениями;

– регламентированные бизнес-процессы недостаточно автоматизированы - отсутствует возможность оперативного контроля хода выполнения бизнес-процессов;

– отсутствует возможность анализа клиентской базы и построения комплексных отчетов по истории взаимоотношений и продажам клиентам;

– отсутствие инструмента для прогнозирования продаж, как следствие затруднения с прогнозами закупок материалов и формирования потребностей в производственных мощностях;

– сотрудники теряют много времени на ответы по типовым вопросам клиентов;

– составление типового договора или коммерческого предложения занимает значительное время у менеджеров;

– для составления отчетности по продажам перед руководителем менеджерам приходится тратить много времени;

– жалобы клиенты теряются или не рассматриваются вовремя - отсутствие возможности получить отчетность по типам жалоб в разрезе отдельных менеджеров;

– руководитель отдела продаж половину своего рабочего времени вынужден тратить на контроль работы сотрудников;

– отсутствует база знаний по работе с клиентами - знания хранятся только в головах опытных сотрудников, теряются при их увольнении, передача знаний от опытного сотрудника новичку занимает много времени и приводит к увеличению издержек и снижению продаж;

– отсутствуют оптимальные процедуры планирования ресурсов предприятия.

Для устранения данных недостатков необходимо внедрение единой автоматизированной информационной системы - ERP-системы предприятия.

В данном параграфе описана модель бизнес-процессов «Как есть» и выявлены в них существующие недостатки, для преодоления которых требуется внедрение автоматизированной системы планирования ресурсов предприятия (ERP-системы). Перейдем к разработке модели процессов «Как должно быть».

2.2.2 Разработка модели бизнес-процессов «Как должно быть»

Составим модель автоматизированной информационной системы - контекстная диаграмма («as-to-be») представлена на рисунке 10.

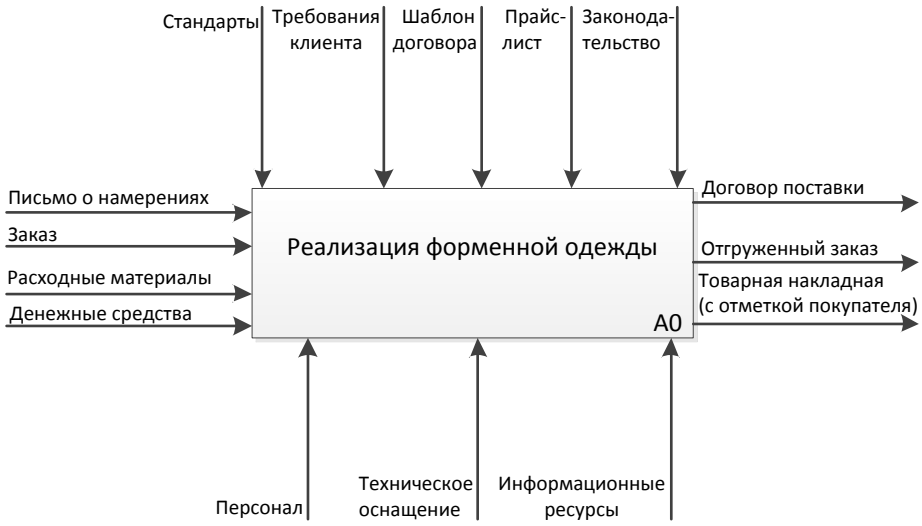


Рисунок 10 - Контекстная диаграмма («as to be»)

Диаграммы декомпозиции модели «КАК ДОЛЖНО БЫТЬ» приведены на рисунках 11, 12, 13.

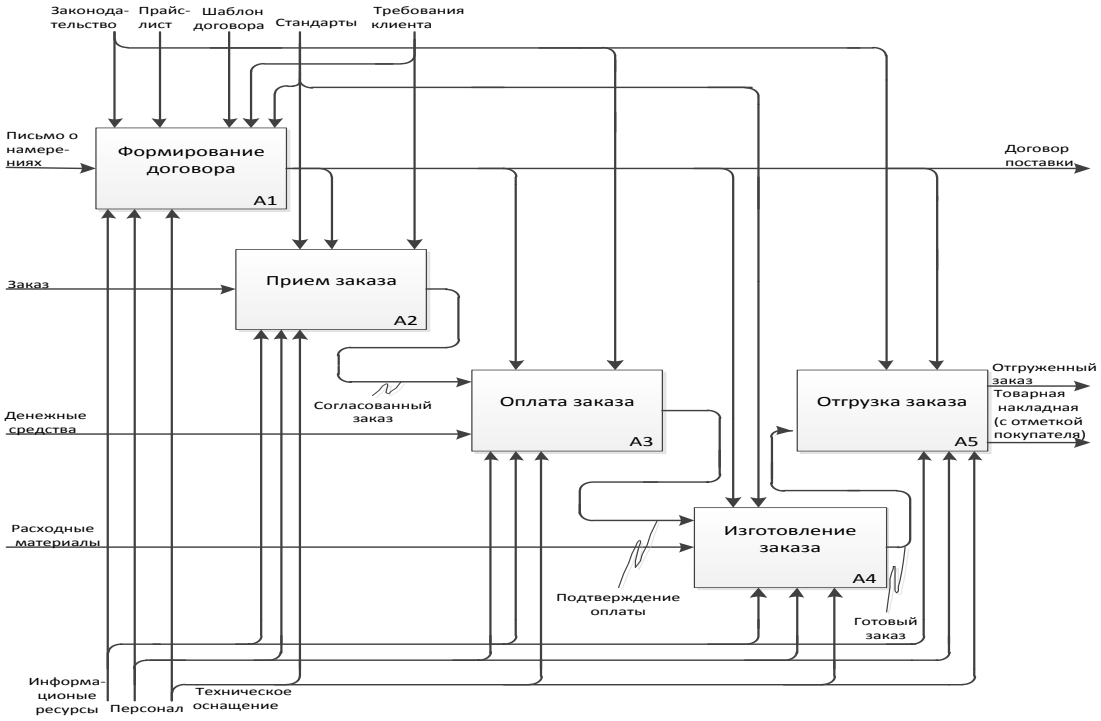


Рисунок 11 - Диаграмма декомпозиции A0 («as to be»)

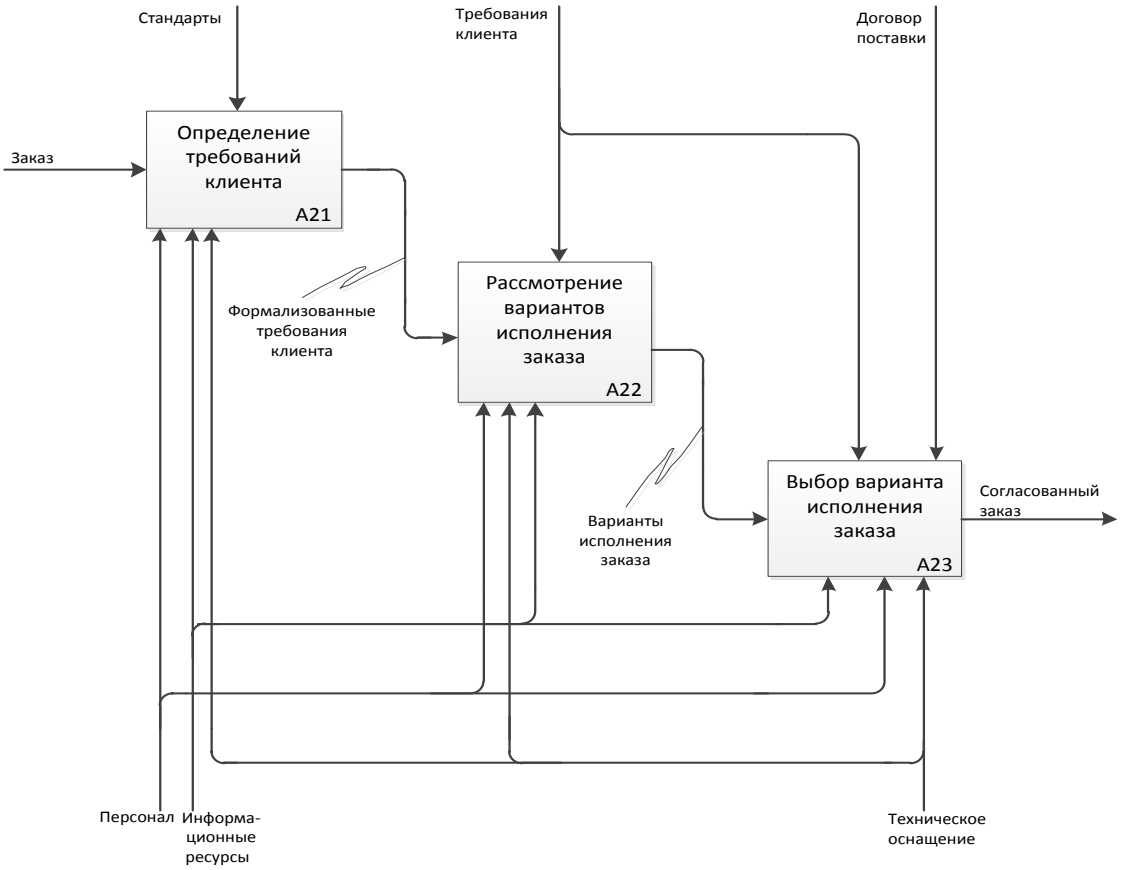


Рисунок 12 - Диаграмма декомпозиции A2 («as to be»)

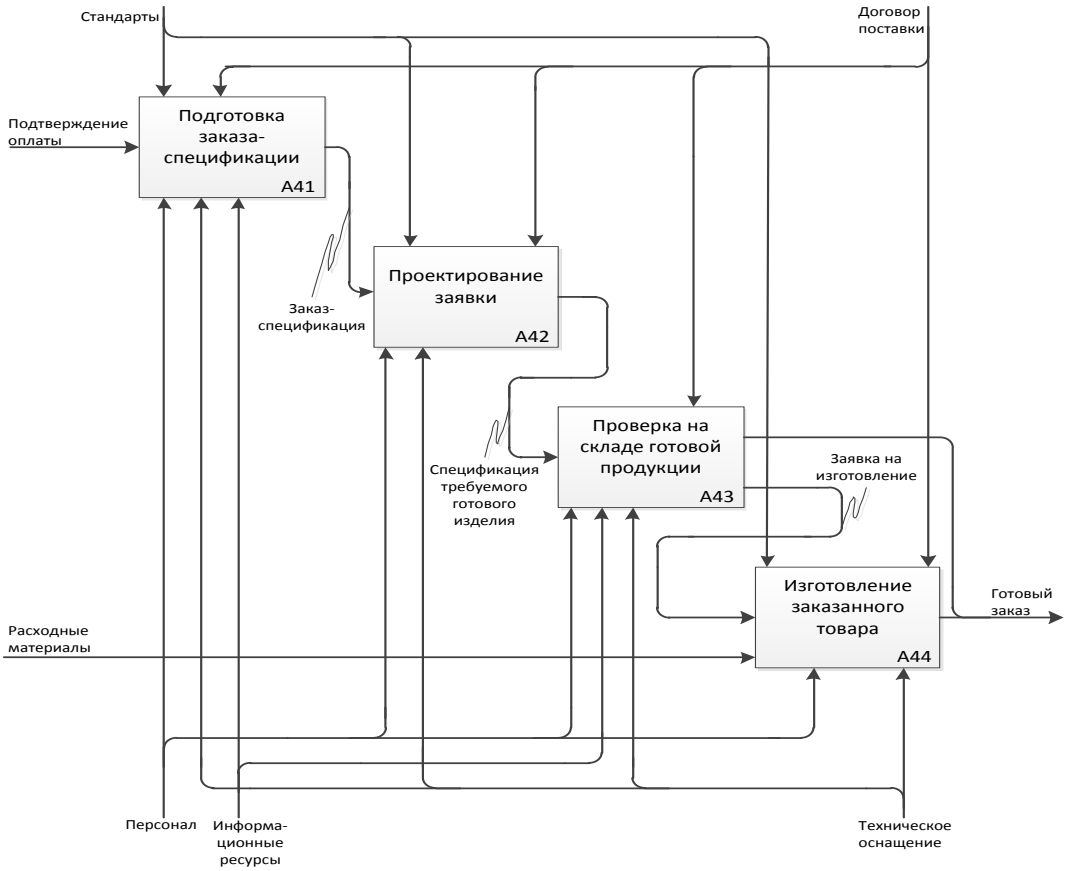


Рисунок 13 - Диаграмма декомпозиции A4 («as to be»)

В качестве механизма «Информационные ресурсы» на данных диаграммах обозначена автоматизированная информационная система планирования ресурсов предприятия ERP-система, решающая вышеописанные недостатки в бизнес-процессах предприятия.

Для занесения информации о клиенте, оформления заказа клиента, осуществления контроля за его выполнением, формирования автоматизированных отчетов, сопровождающую данную деятельность, требуется согласно модели «Как должно быть», создание автоматизированного рабочего места менеджера по продажам.

Процесс обслуживания клиента по новой технологии следующий:

– заполнение формы заказа;

– поиск вариантов исполнения заказа;

– фиксация требуемого ассортимента одежды в заказе;

– выставление счета;

– контроль оплаты;

– формирование заявки на изготовление заказа (если требуемый ассортимент отсутствует на складе готовой продукции);

– формирование товарной накладной на выполненный заказ;

– формирование отчетов и ответов на запросы.

Хранилищем информации в данной системе выступает база данных хранящая информацию о клиентах, заказах покупателей и данные о выполнении заказов клиентов. Согласно новой технологии работы поток данных представлен электронными документами. Поток данных включает следующие составляющие: данные о клиенте, информация об оплате, данные о заказе. Диаграмма потоков данных в модели «as to be» представлена на рисунке 14.

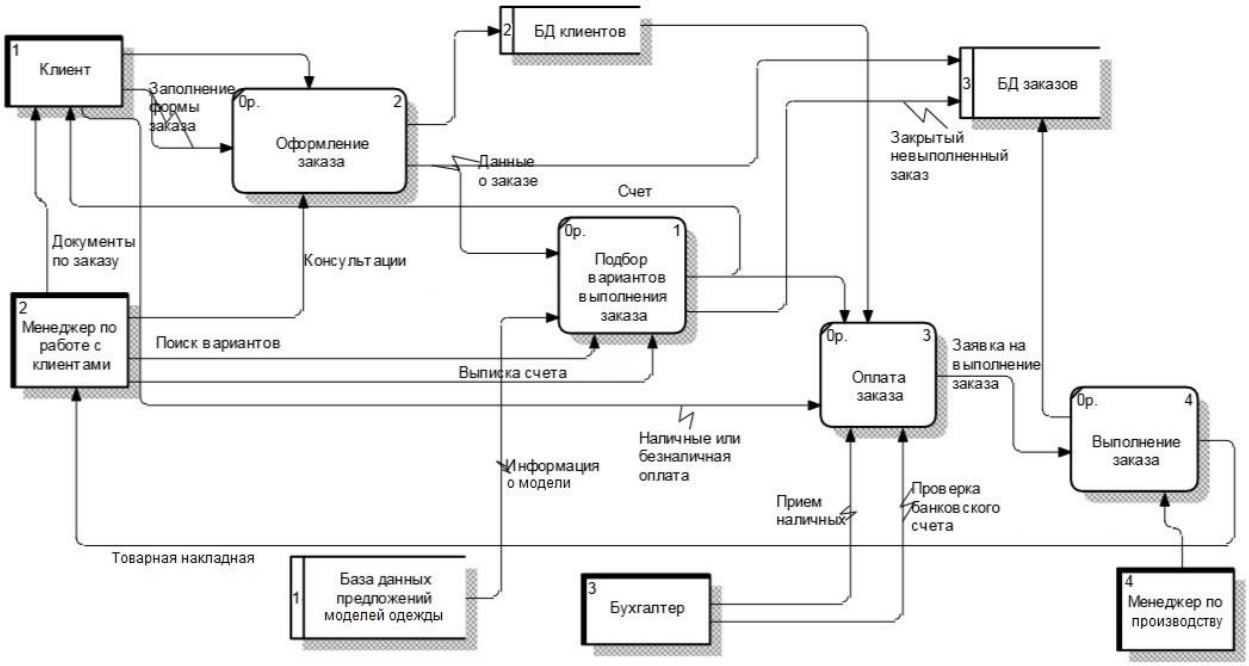


Рисунок 14 - DFD-диаграмма потоков данных «as to be»

Для устранения недостатков в существующих бизнес-процессах предприятия необходимо внедрение ERP-системы, которое обеспечит:

– сбор и структурирование в единую базу данных информации о клиентах, партнерах, поставщиках, конкурентах;

– ведение истории взаимоотношений с клиентами;

– организацию электронного обмена информацией между сотрудниками компании;

– аналитические инструменты и инструменты прогнозирования;

– инструменты для планирования, проведения и анализа эффективности работы предприятия;

– оптимальное планирования ресурсов предприятия исходя из планов продаж, закупок и производства (согласование данных планов);

– инструменты по управлению качеством продукции;

– структурированное хранение и использование накопленных знаний в области продаж одежды форменной и для активного отдыха.

Таким образом построена модель бизнес-процессов «Как должно быть» и осуществлена постановка задачи внедрения ERP-системы в ООО ТД «Купеческий».

2.3 Логическое моделирование ERP-системы

2.3.1 Диаграмма вариантов использования

Концептуальная модель системы описывается диаграммой вариантов использования. Она дает возможность выделить функциональную структуру системы, не рассматривая детали реализации. Производится предварительное выделение объектов системы и их классификация.

Диаграмма вариантов использования представлена на рисунке 15.

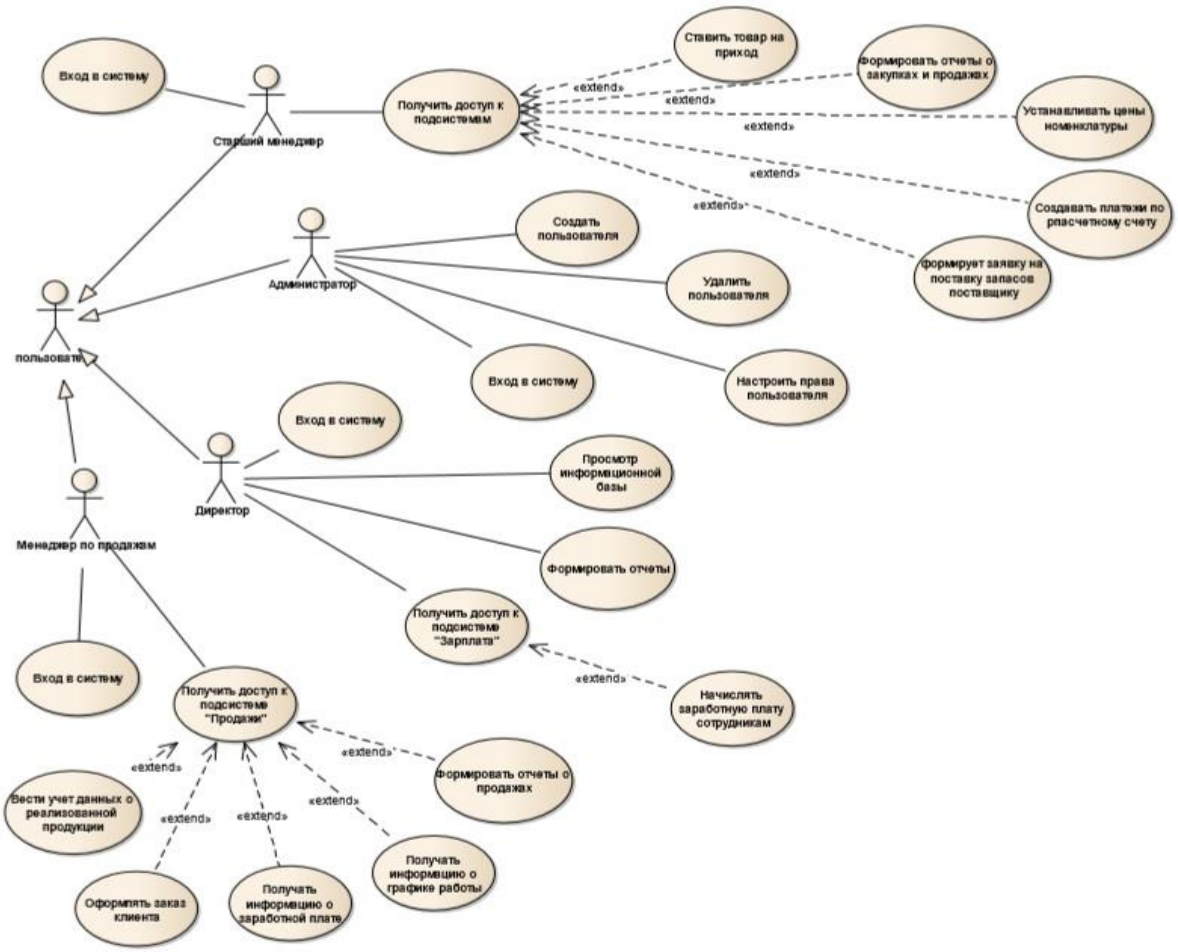


Рисунок 15 - Диаграмма вариантов использования программной системы

Диаграмма вариантов использования описывает последовательность действий, выполняемых системой, которая приводит к заметному результату, представляющему ценность для отдельного действующего лица.

Для данной предметной области выделим следующих актеров:

– администратор;

– директор;

– менеджер по продажам;

– старший менеджер.

Рассмотрим, какие возможности должна предоставлять проектируемая к внедрению система:

– администратор по умолчанию наделен правом входа в систему, может совершать операции по добавлению пользователя, удаления пользователя, настройки прав пользователя;

– директор наделен правом входа в систему, имеет права на просмотр информационной базы, формирование отчетов, обладает доступом к подсистеме «Зарплата» (начисление и выплаты заработной платы);

– менеджер по продажам наделен правом входа в систему, имеет доступ к подсистеме «Продажи»: оформляет заказ клиента, ведет учет данных о реализованной продукции, формирует отчеты о продажах, получает информацию о графике работы и заработной плате;

– старший менеджер наделен правом входа в систему, имеет доступ к функционалу: оприходование товаров, установке цен номенклатуры товаров, формирование заявок на поставку товаров, материалов поставщику, создавать платежи по расчетному счету, формирует отчеты о закупках и продажах.

Таким образом, описанная диаграмма показывает компоненты разрабатываемой системы необходимые для реализации необходимой функциональности.

Далее перейдем к диаграмме классов, проектируемой к внедрению ERP-системы.

2.3.2 Диаграмма классов

На рисунке 16 представлена диаграмма классов проектируемой ERP-системы.

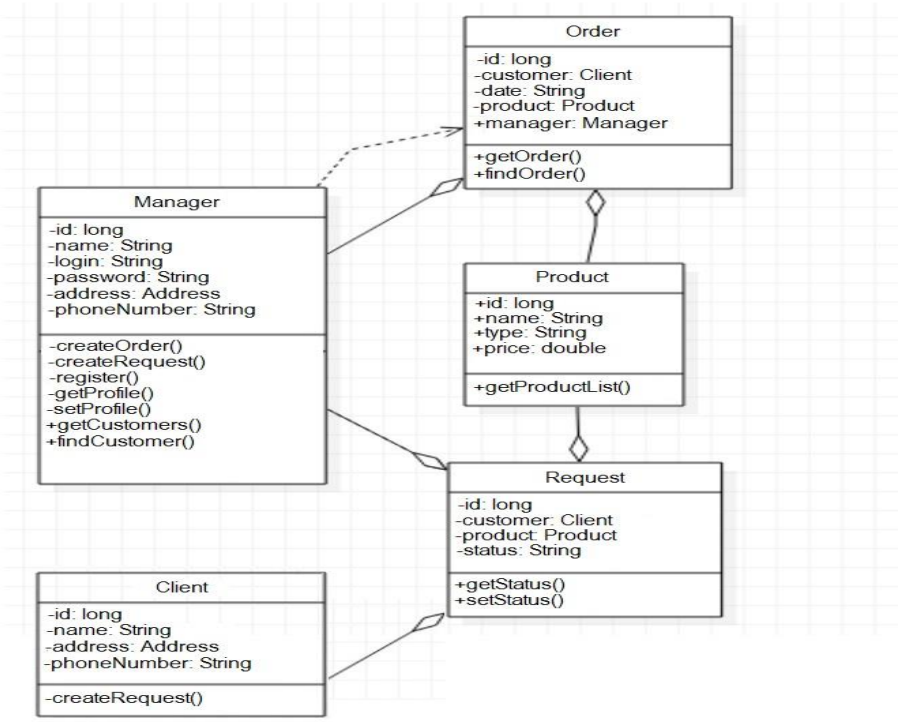


Рисунок 16 - Диаграмма классов

Данная диаграмма показывает атрибуты, операторы классов и их взаимосвязь. В данном разделе разработаны диаграмма вариантов использования и диаграмма классов - вместе они дают представление о функциональной структуре и взаимодействии объектов проектируемой системы. Далее перейдем к описанию концептуальной и логической модели данных системы.

2.3.3 Концептуальная и логическая модель данных ERP-системы

Концептуальная модель данных содержит описание главных сущностей и их отношений. Эта модель изображена на рисунке 17.

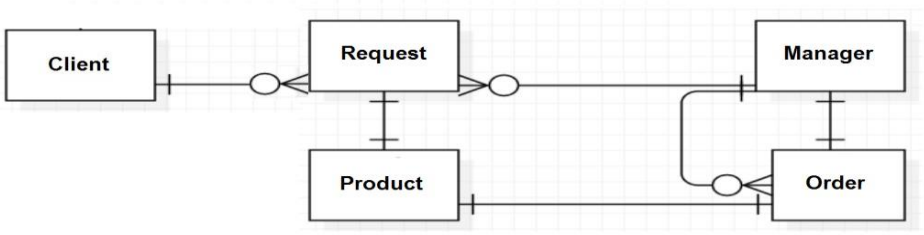


Рисунок 17 - Концептуальная модель данных

Концептуальная модель содержит пять сущностей:

– клиент (включает подсистемы управления клиентами и историей взаимодействия с ними);

– запрос (включает подсистемы управления клиентскими запросами);

– менеджер (включает подсистемы управления персоналом);

– продукт (включает подсистемы производства и создания продукта);

– заказ (включает подсистемы обработки заказа клиента).

Построим логическую модель данных, которая является расширением концептуальной модели данных (см. рисунок 18).

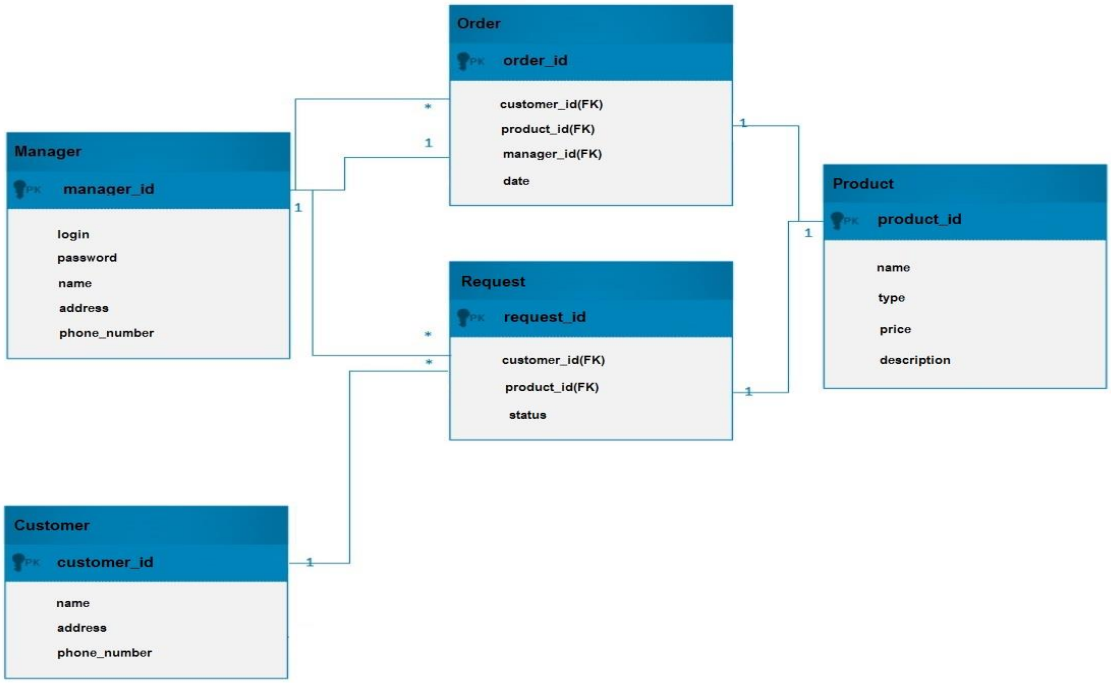


Рисунок 18 - Логическая модель данных

Потенциальный клиент формирует запрос с потребностью в определенном товаре. Менеджер, обрабатывая запрос клиента - формирует заказ с определенным товаром, который обеспечивается (резервируется) на складе, если он отсутствует, то направляется заказ на производство (создание данного продукта для клиента).

Итак, дано описание концептуальной и логической модели данных.

Выводы по главе 2

Проведено исследование предметной области автоматизации, изучена деятельность ООО ТД «Купеческий», определены недостатки в существующих процессах компании, построены структурные схемы в нотации IDEF0 «as is» и «to be», построена концептуальная и логическая модель данных проектируемой к внедрению ERP-системы.

Глава 3 Разработка реализации проектируемой ERP-системы

3.1 Требования к проектируемой ERP-системе

Анализ бизнес-процессов организации в ситуации «как есть» показал следующее:

– в процессе сделки производится регистрация операции только в бухгалтерском контуре, история общения с клиентом надлежащим образом не фиксируется, информация представлена в виде разрозненных массивов данных у разных менеджеров по продажам;

– теряется информация о первоначальных требованиях клиента к заказу, так как фиксируется только то, что он выбрал в итоге, невозможность проследить степень удовлетворенности клиента по его исходным требованиям к заказу;

– информация о клиентах содержится в разных источниках (отсутствие единой базы данных о клиентах), существуют риски потери информации о клиентах при смене менеджеров по продажам, затруднен комплексный анализ данных о клиентах и продажах;

– информация при передаче между подразделениями может искажаться или теряться;

– недостаточная автоматизация бизнес-процессов, затруднен оперативный контроль о исполнении бизнес-процесса;

– история взаимоотношений с клиентами не фиксируется, как следствие отсутствие возможности строить отчеты по продажам в связке с историей и стадиями прохождениями сделок;

– инструмент для прогнозирования продаж отсутствует, как следствие, затруднены прогнозы закупок материалов и формирования потребностей в производственных мощностях;

– ответы на типовые вопросы, составление коммерческого предложения или типового договора занимает много времени у сотрудников предприятия;

– отсутствует база знаний по процессам производства и продаж;

– отсутствует система ключевых показателей для контроля работы сотрудников;

– отсутствуют оптимальные процедуры планирования ресурсов предприятия, что приводит к повышению издержек предприятия и снижению объема продаж.

Для устранения недостатков в бизнес-процессах требуется внедрить ERP-систему, которая позволит:

– хранить в единой базе данных информацию о клиентах, поставщиках, конкурентах;

– вести историю взаимодействий с клиентами;

– организовать электронный обмен информацией между подразделениями и сотрудниками предприятия;

– использовать инструменты прогнозирования и аналитическую отчетность;

– использовать инструменты для анализа эффективности работы предприятия;

– осуществлять оптимальное планирования ресурсов предприятия исходя из планов продаж, закупок и производства (согласование данных планов);

– использовать инструменты по управлению качеством;

– структурированно хранить накопленные знания по производству и продаже одежды.

3.2 Обоснование выбора конкретной ERP системы для управления сетью территориально распределенных филиалов

Осуществим сравнение ERP-систем по критериям основных функциональных модулей. Это даст понимание насколько та или иная система соответствует современным требованиям к системам планирования

ресурсов предприятия. Итак, сравнение систем выполним по следующим аспектам:

- планирование продаж и производства;
- управление кадрами;
- управление проектами и программами;
- управление складом;
- управление производством;
- контроль материальных потоков;
- финансы;
- отраслевые решения;
- управление потоком операций (Workflow);
- моделирование и прогнозирование;
- управление и оптимизация цепочек поставок;
- управление взаимоотношениями с клиентами (CRM);
- поддержка MS Office;
- управление взаимоотношениями с клиентами (CRM);
- «облачные» технологии и мобильное использование;
- бизнес-аналитика (BI) и OLAP;
- электронный документооборот;
- безопасность и администрирование;
- средняя продолжительность внедрения.

По результатам сравнения подготовлена таблица А.1 - Сравнение функциональных возможностей российских ERP-систем, которая представлена в Приложении А.

Система 1С:ERP является лидером отечественных ERP-решений. В качестве системы для реализации проекта выбираем программный продукт 1С:ERP Управление предприятием 2.

3.3 Особенности организации и хранения информации в распределенных корпоративных информационных системах

Корпоративная информационная система (КИС) предприятия с сетью филиалов должна обеспечивать поддержку работы территориально-распределенных подразделений. Требование о бесперебойности работы и временной недоступности интернета на канале связи делает невозможным использование централизованной архитектуры базы данных, информационные ресурсы должны стать распределенными.

Совокупность логически взаимосвязанных баз данных, распределенных в компьютерной сети, называется распределенной базой данных. Распределенная информационная база обладает следующими характеристиками:

– локальная автономность: локальные данные должны быть под локальным владением и управлением, включая функции представления данных, целостности и безопасности (исключения: управление внешним узлом распределенной транзакцией, несколько узлов включены в ограничения целостности);

– принцип децентрализации функций (означает, что никакой определенный сервис не возлагается на выделенный центральный узел);

– принцип непрерывности функционирования (работает даже при изменении количества узлов, смены СУБД на конкретном узле, удалении некоторых данных в распределенной среде);

– принцип независимости от местоположения: пользователи и приложения о физическом расположении данных могут не иметь информации;

– принцип независимости от фрагментации: пользователи и приложения не знают (или могут не знать) как обрабатываются фрагменты БД;

– независимость от тиражирования;

– распределенная обработка запросов, управление распределенными транзакциями;

– принцип независимости от оборудования, сети, операционных систем, локальных СУБД.

В практических приложениях сложно построить систему сразу со всеми вышеперечисленными характеристиками.

Распределенные базы данных классифицируются на однородные и неоднородные, построенные по принципу "сверху-вниз" и "снизу-вверх".

Распределение данных может принимать разные формы:

– фрагментация (разделение данных на порции, распределенные между множеством физических ресурсов);

– тиражирование (дублирование данных на нескольких узлах).

Однородные распределенные системы в большинстве случаев проектируются методом "сверху-вниз" - являются сильно связанными системами (оптимизированы для достижения максимальной пропускной способности и производительности), в основе своей содержат одну и ту же СУБД.

Неоднородные распределенные системы управляются разными СУБД - такие системы часто проектируются методом "снизу-вверх" методом «снизу-вверх».

На рисунке 19 представлена структура однородной системы.

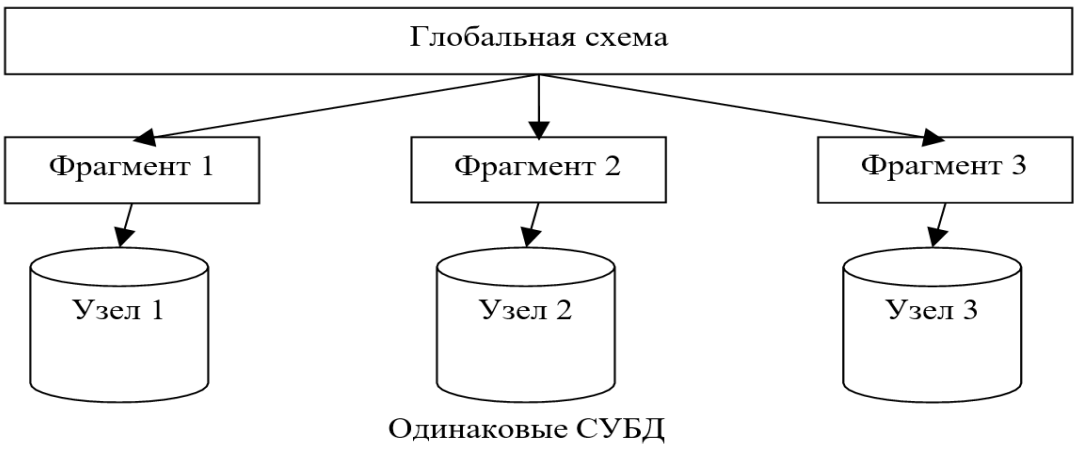


Рисунок 19 - Структура однородной распределенной системы

На рисунке 20 представлена структура неоднородной системы.

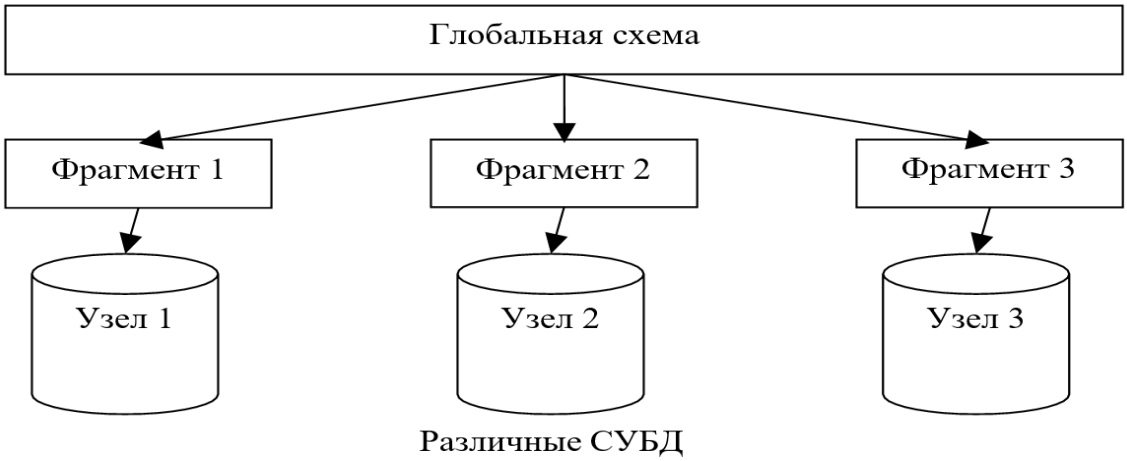


Рисунок 20 - Структура неоднородной распределенной системы

Метод проектирования распределенных баз данных "сверху-вниз" похож на проектирование централизованных БД, но предполагается что объекты системы распределены по нескольким вычислительным комплексам. Распределение данных осуществляется фрагментацией и тиражированием.

Фрагментация бывает двух видов:

– горизонтальная (фрагментация по строкам) - распределение по разным узлам строк таблицы, может осуществляться по значению параметров (к примеру, продажи одного магазина на конкретном узле) или по принципу "карусели";

– вертикальная (фрагментация по столбцам) - столбцы таблицы распределяются по узлам.

Фрагментация означает декомпозицию объектов базы данных, таких, как реляционные таблицы, на две или более частей, которые размещаются на разных компьютерных системах.

Различают два вида фрагментации:

– горизонтальная фрагментация (фрагментация по строкам) означает распределение по узлам строк таблицы. Распределение записей может осуществляется в соответствии со значением параметров

(например, на каждом узле магазина хранятся продажи только этого магазина) или не на основе значений (по принципу «карусели»).

– вертикальная фрагментация (фрагментация по столбцам) означает распределение по узлам столбцов таблицы.

При этом предполагается, что независимо от используемого вида фрагментации поддерживается глобальная схема, воссоздающая из фрагментов логически централизованную таблицу.

Тиражирование (или репликация) - создание дубликатов данных. Репликаты - множество физических копий некоторого объекта БД, для которого с помощью специальных правил осуществляется синхронизация с центральной копией.

Модели тиражирования бывают различны (см. рисунки 21, 22, 23).

Рисунок 21 - Тиражирование с одновременным обновлением

Рисунок 22 - Тиражирование с распространением обновлений

Рисунок 23 - Тиражирование с запланированной синхронизацией дубликатов только для чтения

Проектирование "сверху-вниз" подходит при создании новых систем и обычно применяется к однородным распределенным системам.

При интеграции существующих систем прибегают к методу "снизу-вверх" - здесь основная проблема объединить существующие схемы баз данных (предоставить и новым, и старым приложениям доступ к старым и новым ресурсам данных, рисунок 24).

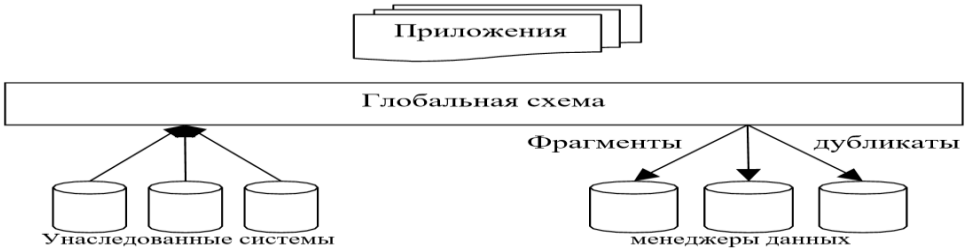


Рисунок 24 - Интеграция распределенных баз данных «снизу-вверх»

Чтобы повысить эффективность работы корпоративной информационной системы (минимизировать траффик, распределить вычислительные мощности и т.п.) применяют технологии распределения вычислений. Распределение вычислений заключается в частичном или полном переносе вычислений на сервер базы данных (рисунок 25).

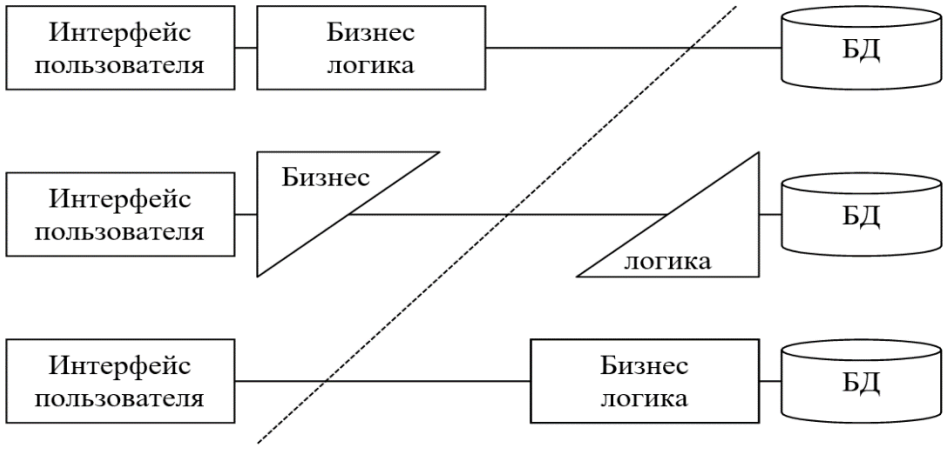


Рисунок 25 - Распределенные вычисления

Наиболее эффективной технологией распределения вычислений является построение трехуровневой (или многоуровневой) архитектуры при помощи сервера приложений (рисунок 26).

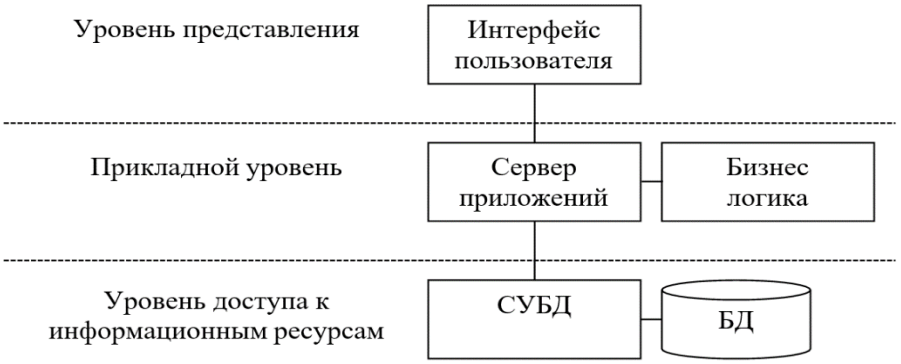


Рисунок 26 - Трехуровневая архитектура
с использованием севера приложений

Система разбивается на три уровня: уровень представления (реализующий функции ввода и отображения данных); прикладной уровень (реализующий универсальные сервисы, а также функции, специфичные дляопределенной предметной области); уровень доступа к информационным ресурсам (реализующий фундаментальные функции

хранения и управления информационно-вычислительными ресурсами). В качестве СУБД для центральной базы и баз филиалов выбирается Microsoft SQL Server, которая хорошо зарекомендовала себя в связке с продуктами 1С.

3.4. Структура распределенной базы данных для корпоративной информационной системы управления сетью филиалов

Рассматриваемое предприятие состоит из N-филиалов и центрального офиса (N однотипных филиалов и одного центра управления).

Распределенная база данных, таким образом, состоит из совокупности логически взаимосвязанных БД:

- базы центрального офиса;
- N баз данных филиалов (однотипных).

Известно, что транзакции изменения данных не выходят за пределы локальной сети, поэтому методом их оптимизации служит распределение вычислений как для случая локальной БД. Сосредоточим внимание на оптимизации запросов к БД.

Прямые связи между филиалами отсутствуют поэтому все запросы к распределенной БД можно представить в виде следующих запросов:

- запросы к данным центра в пределах локальной сети центра;
- запросы к данным филиала из центра;
- запросы к данным центра из филиала;
- запросы к данным филиала в пределах локальной сети филиала.

Это показано на рисунке 27.

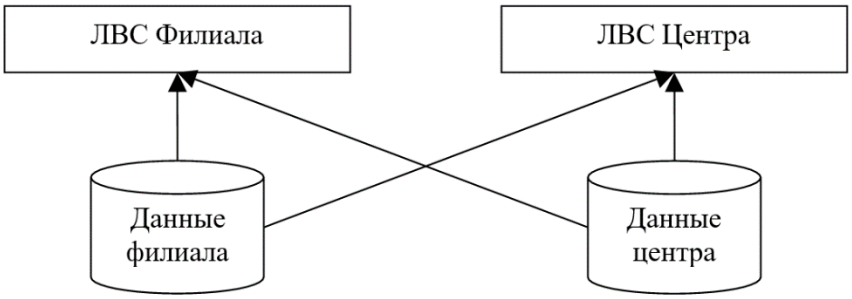


Рисунок 27 - Виды запросов между центром и филиалом

Запросы также подразделяются на:

– единовременные (данные передаются единовременно);

– многократные (одни и те же данные передаются многократно).

Удаленные запросы (между филиалом и центром) самые дорогостоящие - их необходимо оптимизировать. Именно они должны быть существенно оптимизированы. Целесообразно применить распределенные вычисления чтобы передавать только конечные данные, а предварительную обработку осуществлять на сервере удаленной базы данных. Необходимо понимать, что при этом возрастает нагрузка на сервер данной БД, но это дает выигрыш за счет ускорения выполнения запросов и уменьшения сетевого траффика. Скорость обработки многократных запросов можно повысить при помощи хранилищ данных (агрегирование данных). Агрегирование требует дополнительных вычислительных ресурсов и объема памяти на жестких дисках. Если обработка осуществляется после изменения исходных данных, то процесс агрегирования можно осуществить асинхронно. Это дает возможность равномернее распределять вычислительные ресурсы во времени. Однако появляются задачи достоверности и непротиворечивости данных. Для оптимизации многократных удаленных запросов применяют тиражирование. При особых требованиях к скорости при однократных запросах также применяют тиражирование. При тиражировании вновь возникают задачи непротиворечивости и достоверности данных. Применение тиражирования проиллюстрировано на рисунке 28.

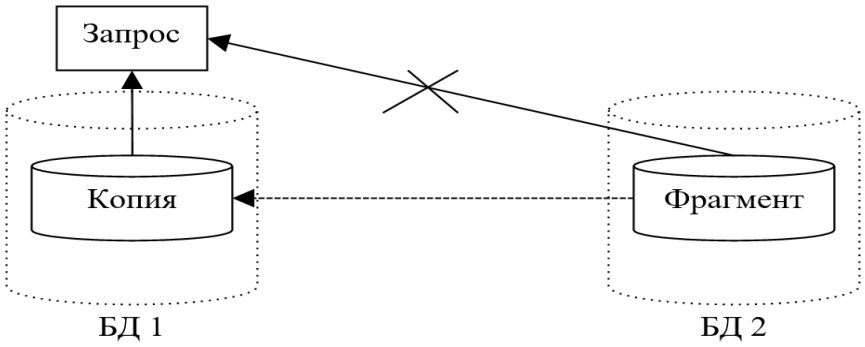


Рисунок 28 - Применение тиражирования для оптимизации многократных удаленных запросов

Данные в базе данных можно условно отнести к трем группам: справочники (классификаторы, настройки и т.п.), статические данные (состояние объектов), динамические данные (история изменения состояний объектов). Каждая группа обладает свойствами, которые определяют стратегии для оптимизации доступа. Данные справочников редко добавляются, изменяются и удаляются. Эта группа данных имеет малый объем по сравнению со всей базой данных. Данные справочников участвуют в большинстве запросов и высока частота многократного обращения. Примером подобных данных могут служить классификаторы или настройки. Оптимальным методом оптимизации доступа является тиражирование в силу редкого изменения этой группы данных и высокой частоты обращения (в том числе повторного обращения). Учитывая, незначительный объем информации этой группы и требований к достоверности и непротиворечивости данных можно использовать одновременное (параллельное) тиражирование (или отложенное тиражирование). К группе статических данных относится информация, которая также редко добавляется и удаляется, но претерпевает очень частое изменение. Такие данные описывают текущее состояние объектов (например, балансовые счета). Данная группа данных имеет незначительную часть от объема базы данных, имеет высокую частоту обращения. На группу статических данных

накладывают большую часть условий по ограничениям целостности и непротиворечивости. Для оптимизации доступа к таким данным подходит параллельное тиражирование. Группа динамических данных включает информацию об изменении состоянии объектов. Такие данные не обновляются и не удаляются, но часто добавляются. Динамические данные составляют большую часть базы данных. Примером может служить история продаж. Именно к динамическим данным применяют агрегирование (очень часто интересно не точное время изменения объекта, а принадлежность этого изменения к некоторому интервалу времени). Применение агрегирования приводит к уменьшению использования вычислительных ресурсов и сокращение объема передаваемой по сети информации. Примером может служить ситуация когда несущественна информация о продажах внутри дня, тогда вместо операций чтения нескольких записей и их суммирования применяется чтение одной записи Это проиллюстрировано на рисунке 29.

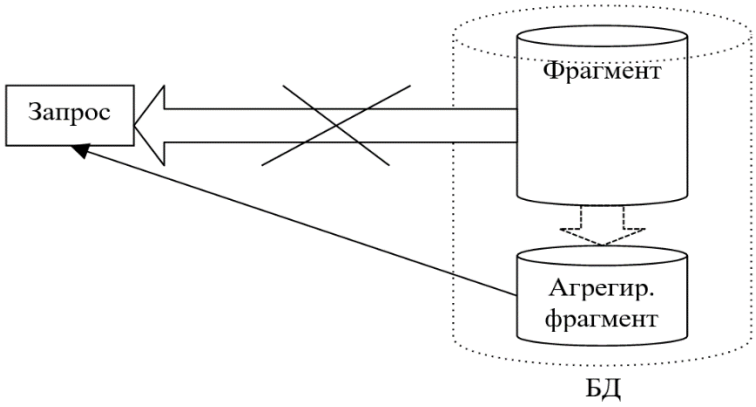


Рисунок 29 - Применение агрегирования для оптимизации запросов

Повторное (многократное) обращение к одним и тем же динамическим данным довольно редко явление, но стоимость таких запросов высокая, потому что приходится передавать большие объемы информации. Именно поэтому применяется тиражирование исходных или агрегированных данных. С течением времени динамические данные

теряют свою актуальность и при ограничениях на размер базы данных возникает задача их архивирования.

Данные по частоте использования сведены в таблицу 4.

Таблица 4 - Характеристика групп данных

	Справочники	Статические данные	Динамические данные
Обновление	редко	часто	не обновляются
Добавление	редко	редко	часто
Удаление	редко	редко	не удаляются[1]
Объем информации	маленький	маленький	большой
Частота обращения к блоку (одному и тому же)	большая	большая	маленькая
Методы оптимизации	тиражирование	тиражирование	агрегирование, тиражирование

Рассмотрим случай, когда центр осуществляет управление филиалами путем выработки правил и планов функционирования. Причем управленческие решения принимаются на основании стандартизированной информации по филиалам. Тогда модель распределенной базы данных корпоративной системы управления сетью филиалов будет выглядеть следующим образом (рисунок 30), где С - справочники, SD - статические данные, DD - динамические данные, AD - агрегированные данные, * - копия, branch office - филиал, head office - центр.

[1] Динамические данные не удаляются, существуют сроки полезности, при превышении которых данные помещаются в архив

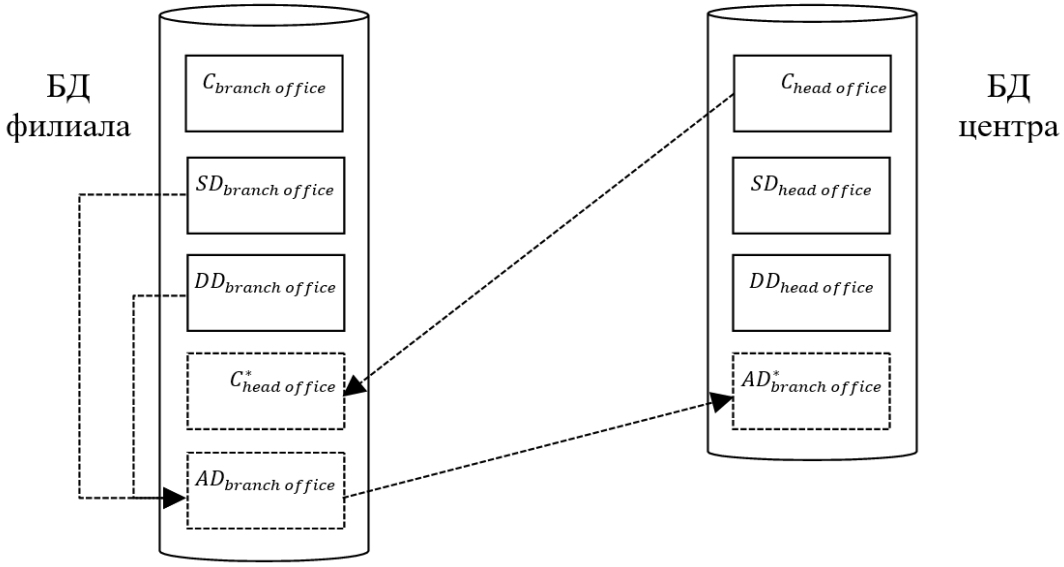


Рисунок 30 - Распределенная база данных корпоративной информационной системы управления сетью филиалов

В данном разделе описана структура распределенной базы данных для корпоративной информационной системы управления сетью филиалов. Показаны методы оптимизации хранения и обмена данными для повышения производительности распределенной системы. Перейдем к задаче выбора оптимальных параметров функционирования распределенной базы данных корпоративной информационной системы для управления сетью филиалов.

3.5 Задача выбора оптимальных параметров функционирования распределенной базы данных корпоративной информационной системы управления сетью филиалов

Имеем N-однотипных локальных баз филиалов и одну базу данных центра управления.

Будем рассматривать случай, когда филиалы не взаимодействуют между собой, центр и филиалы связаны посредством каналов интернета. Тиражирование справочников центра на филиалы, по сути, есть передача

глобального управления. Кроме глобального управления осуществляется локальное самоуправление филиалом. Введем обозначения:

$T^{\Ф}$- период агрегирования оперативной базы данных филиала;

$T^{\Ц}$ - период тиражирования хранилища филиала;

$k_{\text{агр}}^{\Ф}$- коэффициент агрегирования базы данных филиала;

$w^{\Ф}$- мощность потока исходных данных базы данных филиала;

$w^{\П}$- мощность потока запросов данных пользователями филиала;

$q^{\П}$- число запросов пользователей;

N - число филиалов;

$V^{\Ц}$ - объем справочников центра (глобального управления);

$T_{\text{у}}^{\Ф}$- период локального управления;

$T_{\text{у}}^{\Ц}$ - период тиражирования из центра (глобального управления);

$V_{\text{иу}}^{\Ф}$ - обрабатываемый объем данных для расчета локального управления;

$V_{\text{у}}^{\Ф}$ - объем коррекции данных после расчета локального управления;

$s_{\text{бд}}^{\ф}$ - время обработки единицы данных СУБД филиала;

$s^{\text{п}}$ - время обработки единицы данных ЦП рабочей станции пользователя;

z - параметр распределения вычислений

$k_{\text{сп}}^{\ф}$ - относительное сокращение объема данных при выполнении запроса

через сервер приложений.

СУБД филиала выполняет задачи записи исходных данных, записи данных агрегирования в хранилище, записи данных тиражирования из центра, записи локального управления, чтения данных, запрашиваемых пользователями, чтения данных для агрегирования, чтения данных для тиражирования в центр и чтения данных для локального управления. При этом одна операция записи соответствует $k_{\text{бд}}^{\ф}$ операции чтения.

$$k_{\text{бд}}^{\text{ф}} \cdot \left(w^{\text{Ф}} + w^{\text{Ф}} \cdot k_{\text{агр}}^{\text{Ф}} + \frac{V^{\text{Ц}}}{T_{\text{у}}^{\text{Ц}}+T^{\text{Ц}}+T^{\text{Ф}}} + \frac{V_{\text{у}}^{\text{Ф}}}{T_{\text{у}}^{\text{Ф}}+T^{\text{Ф}}}\right) + w^{\text{П}} + w^{\text{Ф}} + \frac{w^{\text{Ф}} \cdot k_{\text{агр}}^{\text{Ф}}}{T^{\text{Ц}}+T^{\text{Ф}}} + \frac{V_{\text{иу}}^{\text{Ф}}}{T_{\text{у}}^{\text{Ф}}+T^{\text{Ф}}} \leq \frac{1}{s_{\text{бд}}^{\text{ф}}} \qquad (1)$$

Локальная сеть филиала выполняет задачи передачи результатов прямых запросов и передачи результатов запросов через сервер приложений.

$$z \cdot w^{\text{П}} + (1-z) \cdot k_{\text{сп}}^{\text{ф}} \cdot w^{\text{П}} \leq \frac{1}{s_{\text{сет}}^{\text{ф}}} \qquad (2)$$

Глобальная сеть выполняет задачи передачи данных тиражирования от филиала центру и от центра филиалу. При этом входящая скорость в $k^{\text{инт}}$ раз больше исходящей.

$$N \cdot k^{\text{инт}} \cdot \frac{V^{\text{Ц}}}{T_{\text{у}}^{\text{Ц}}+T^{\text{Ц}}+T^{\text{Ф}}} + N \cdot \frac{w^{\text{Ф}} \cdot k_{\text{агр}}^{\text{Ф}}}{T^{\text{Ц}}+T^{\text{Ф}}} \leq \frac{1}{s_{\text{сет}}^{\text{инт}}} \qquad (3)$$

$$k^{\text{инт}} \cdot \frac{w^{\text{Ф}} \cdot k_{\text{агр}}^{\text{Ф}}}{T^{\text{Ц}}+T^{\text{Ф}}} + \frac{V^{\text{Ц}}}{T_{\text{у}}^{\text{Ц}}+T^{\text{Ц}}+T^{\text{Ф}}} \leq \frac{1}{s_{\text{сет}}^{\text{инт}}} \qquad (4)$$

ЦП рабочей станции пользователя выполняет задачу обработки данных прямых запросов.

$$z \cdot w^{\text{П}} \leq \frac{1}{s^{\text{п}}} \qquad (5)$$

ЦП сервера филиала помимо обслуживания работы СУБД выполняет задачи расчета агрегирования, расчета локального управления и обработки данных запросов через сервер приложений.

$$g_{\text{бд}}^{\text{ф}} \cdot \left(k_{\text{бд}}^{\text{ф}} \cdot \left(w^{\text{Ф}} + w^{\text{Ф}} \cdot k_{\text{агр}}^{\text{Ф}} + \frac{V^{\text{Ц}}}{T_{\text{у}}^{\text{Ц}}+T^{\text{Ц}}+T^{\text{Ф}}} + \frac{V_{\text{у}}^{\text{Ф}}}{T_{\text{у}}^{\text{Ф}}+T^{\text{Ф}}}\right) + w^{\text{П}} + w^{\text{Ф}} + \frac{w^{\text{Ф}} \cdot k_{\text{агр}}^{\text{Ф}}}{T^{\text{Ц}}+T^{\text{Ф}}} + \frac{V_{\text{иу}}^{\text{Ф}}}{T_{\text{у}}^{\text{Ф}}+T^{\text{Ф}}}\right) + g_{\text{агр}}^{\text{ф}} \cdot w^{\text{Ф}} + \frac{G_{\text{агр}}^{\text{ф}}}{T^{\text{Ф}}} + \frac{G_{\text{у}}^{\text{ф}}}{T_{\text{у}}^{\text{Ф}}+T^{\text{Ф}}} + g_{\text{сп}}^{\text{ф}} \cdot (1-z) \cdot w^{\text{П}} \leq \frac{1}{s_{\text{обр}}^{\text{ф}}} \qquad (6)$$

Среднее время обработки запроса пользователей:

$$t_{\text{ср}}^{\text{п}} = \frac{T}{q^{\text{П}}} \cdot (s_{\text{бд}}^{\text{ф}} \cdot \left(k_{\text{бд}}^{\text{ф}} \cdot \left(w^{\text{Ф}} + w^{\text{Ф}} \cdot k_{\text{агр}}^{\text{Ф}} + \frac{V^{\text{Ц}}}{T_{\text{у}}^{\text{Ц}}+T^{\text{Ц}}+T^{\text{Ф}}} + \frac{V_{\text{у}}^{\text{Ф}}}{T_{\text{у}}^{\text{Ф}}+T^{\text{Ф}}}\right) + w^{\text{П}} + w^{\text{Ф}} + \frac{w^{\text{Ф}} \cdot k_{\text{агр}}^{\text{Ф}}}{T^{\text{Ц}}+T^{\text{Ф}}} + \frac{V_{\text{иу}}^{\text{Ф}}}{T_{\text{у}}^{\text{Ф}}+T^{\text{Ф}}}\right) + s^{\text{п}} \cdot z \cdot w^{\text{П}} + s_{\text{агр}}^{\text{ф}} \cdot w^{\text{Ф}} + \frac{S_{\text{агр}}^{\text{ф}}}{T^{\text{Ф}}} + \frac{S_{\text{у}}^{\text{ф}}}{T_{\text{у}}^{\text{Ф}}+T^{\text{Ф}}} + s_{\text{сп}}^{\text{ф}} \cdot (1-z) \cdot w^{\text{П}} + s_{\text{сет}}^{\text{ф}} \cdot \left(z \cdot w^{\text{П}} + (1-z) \cdot k_{\text{сп}}^{\text{ф}} \cdot w^{\text{П}}\right)) \qquad (7)$$

Штраф за превышение трафика глобальной сети:

$$C^{\text{инт}} = \begin{cases} 0, \ w^{\text{инт}} \cdot T < V_{\text{абон}}^{\text{инт}} \\ \text{с}^{\text{инт}} \cdot (w^{\text{инт}} \cdot T - V_{\text{абон}}^{\text{инт}}) \end{cases}, \text{где } w^{\text{инт}} = N \cdot \left(\frac{V^{\text{Ц}}}{T_{\text{у}}^{\text{Ц}} + T^{\text{Ц}} + T^{\text{Ф}}} + \frac{w^{\text{Ф}} \cdot k_{\text{агр}}^{\text{Ф}}}{T^{\text{Ц}} + T^{\text{Ф}}}\right) \quad (8)$$

Штраф за задержки локального управления:

$$\text{C}_{\text{упр}}^{\text{ф}} = \begin{cases} 0, T_{\text{у}}^{\text{Ф}} + T^{\text{Ф}} < T_{\text{упр}}^{\text{ф}} \\ c_{\text{упр}}^{\text{ф}} \cdot T \cdot \left(1 - \frac{T_{\text{упр}}^{\text{ф}}}{T_{\text{у}}^{\text{Ф}} + T^{\text{Ф}}}\right) \end{cases} \quad (9)$$

Штраф за задержки глобального управления:

$$C_{\text{упр}}^{\text{Ц}} = \begin{cases} 0, T_{\text{у}}^{\text{Ц}} + T^{\text{Ц}} + \ T^{\text{Ф}} < T_{\text{упр}}^{\text{Ц}} \\ c_{\text{упр}}^{\text{ц}} \cdot T \cdot \left(1 - \frac{T_{\text{упр}}^{\text{Ц}}}{T_{\text{у}}^{\text{Ц}} + T^{\text{Ц}} + T^{\text{Ф}}}\right) \end{cases} \quad (10)$$

Штраф за задержки обработки запросов пользователей:

$$C_{\text{ср}}^{\text{П}} = \begin{cases} 0, t_{\text{ср}}^{\text{П}} < T_{\text{ср}}^{\text{П}} \\ c_{\text{ср}}^{\text{П}} \cdot (t_{\text{ср}}^{\text{П}} - T_{\text{ср}}^{\text{П}}) \end{cases} \quad (11)$$

$$\boldsymbol{C}^{\text{инт}} + \text{C}_{\text{упр}}^{\text{ф}} + C_{\text{упр}}^{\text{Ц}} + \boldsymbol{C}_{\text{ср}}^{\text{П}} \xrightarrow{(T^{\text{Ф}}, T^{\text{Ц}}, T_{\text{у}}^{\text{Ц}}, T_{\text{у}}^{\text{Ф}}, z)} \boldsymbol{min}$$

$$\boldsymbol{1 \leq T^{\text{ф}} < T}$$

$$\boldsymbol{1 \leq T_{\text{у}}^{\text{Ф}} + T^{\text{ф}} < T}$$

$$\boldsymbol{1 \leq T^{\text{Ц}} + T^{\text{ф}} < T}$$

$$\boldsymbol{1 \leq T_{\text{у}}^{\text{Ц}} + T^{\text{Ц}} + T^{\text{ф}} < T}$$

$$\boldsymbol{0 \leq z \leq 1}$$

Полученная задача представляет собой задачу нелинейного программирования с линейными ограничениями. Численное решение может быть найдено при помощи современных математических программных пакетов. В данном разделе описаны основные методы построения однородных и неоднородных распределенных баз данных, метод распределения вычислений с использованием трехуровневой архитектуры, структура распределенной базы данных для корпоративной информационной системы управления сетью филиалов, сформулирован критерий и поставлена

задача выбора оптимальных параметров функционирования предложенной структуры распределенной системы.

Для создания территориально распределенных систем в конфигурации 1С:ERP предусмотрен механизм распределенных информационных баз (РИБ). С помощью данного механизма осуществляются синхронизация изменений конфигурации и полная консолидация данных между центральной базой (центральным узлом РИБ) и всеми периферийными базами (периферийными узлами РИБ). Миграция данных между различными узлами РИБ происходит без применения каких-либо фильтров и включает в себя значения констант, элементы справочников и планы видов характеристик, документы и их движения, бизнес-процессы и задачи.

3.6 Устав проекта внедрения 1С:ERP Управление предприятием 2

Перед началом проекта необходимо провести подготовку к внедрению. Для этого необходимо:

- определить цели и результаты внедрения;
- определить руководителя и заказчика проекта;
- определить сроки и бюджет проекта;
- изучить возможности программы 1С:ERP Управление предприятием.

В результате мероприятий по подготовке к внедрению составим устав проекта.

1. Наименование проекта: Проект внедрения ERP-системы 1С:ERP Управление предприятием 2.

2. Цели проекта: сбор и структурирование в единую базу данных информации о клиентах, партнерах, поставщиках, конкурентах; ведение истории взаимоотношений с клиентами; организацию электронного обмена информацией между сотрудниками компании; аналитические инструменты и инструменты прогнозирования; инструменты для планирования, проведения и анализа эффективности работы предприятия; оптимальное планирования

ресурсов предприятия исходя из планов продаж, закупок и производства (согласование данных планов).

3. Результаты проекта: внедрение ERP-системы 1С:ERP Управление предприятием 2.

4. Требования к системе:

– ввод информации в систему обеспечивается операторным методом;

– информация выводится в виде: экранных форм и табличных отчетов;

– система должна функционировать в многопользовательском режиме;

– для хранения нормативной и постоянной информации общего пользования используются справочники, которые могут редактироваться и обновляться отдельными группами пользователей в соответствии с правами и привилегиями доступа.

5. Допущения и ограничения:

– время исполнения проекта: до 12 месяцев;

– затраты по проекту не более 5000000 рублей;

– организационные: внедрение всех контуров программы осуществляется единовременно, что требует серьёзной подготовительной работы и работы с персоналом (пользователей);

– время команды проекта - для решения задач данного проекта из общего рабочего времени выделяют: для руководителя проекта (начальник отдела продаж) выделяется 30% своего рабочего времени, два программиста - 80% рабочего времени, оператора - до 10% рабочего времени;

6. Критерии оценки успешности проекта: ввод в эксплуатацию ERP-системы, выполнение бюджета проекта.

7. Ключевые участники и заинтересованные стороны: генеральный директор, менеджеры по продажам, заказчик проекта (начальник отдела продаж), члены команды проекта со стороны предприятия.

8. Ресурсы проекта - команда проекта: руководитель отдела продаж (общее руководство), 2 программиста (внедрение информационного продукта), оператор (первоначальное заполнение справочников системы).

9. Сроки: информацию по срокам представим по этапам жизненного цикла в таблице 5.

Таблица 5 - Фазы жизненного цикла проекта

Фазы жизненного цикла проекта	
1. Инициация и запуск проекта	До 01.06.2020г.: – решение о целесообразности проекта; – выбор схемы финансирования; – назначение руководителя проекта; – приказ генерального директора о запуске проекта.
2. Предпроектное обследование, разработка технического задания	До 01.08.2020 г.: – проведение обследования задачи на предприятии; – составление технического задания на проект ERP-системы; – описание рисков проекта и мероприятий по их снижению; – разработка плана проекта.
3. Проектирование	До 01.12.2020 г.: – разработка технического проекта на основании согласованного технического задания
4. Реализация проекта	До 30.04.2021 г.: – настройка информационной системы согласно технического задания; – подготовка нормативно-справочной информации; – первоначальное заполнение справочников;
5. Опытно-промышленная эксплуатация	До 15.05.2021 г.: обучение администрированию базы данных; – ввод первичных документов пользователями с одновременным обучением и под контролем консультантов (оператора, программиста) по всем участкам программы; – доработка ERP-системы по результатам опытно-промышленной эксплуатации; разработка и оформление индивидуальных инструкций по наиболее сложным вопросам эксплуатации ERP-системы
6. Завершение проекта	До 31.05.2021 г.: – переход в режим промышленной эксплуатации; – консультационная поддержка пользователей в режиме «горячей линии»; – подведение итогов проекта, оценка достижения целей и задач проекта; – принятие решения о развитии ИС.

10. Риски:

– выход из бюджета проекта;

– недостаток финансовых ресурсов;

– ошибки в планировании работ по проекту;

– отсутствие необходимых технических возможностей в требуемый момент;

– недостаток квалификации пользователей;

– противодействие системы управления;

– страхи и опасения пользователей и команды проекта;

– потеря ожидаемого функционала информационной системы;

– изменения в составе команды проекта;

– ошибки в организации работы по проекту;

– низкая скорость принятия решений по проекту (затягивание процедуры согласования);

– недостаточная компетентность членов команды проекта;

– некому проводить аналитическую работу по документации проекта;

– самоустранение руководства от участия в проекте.

11. Критерии приемки проекта:

– создание информационного проекта, соответствующего поставленному техническому заданию;

– выполнение бюджета проекта.

Итак, сформирован устав проекта, перейдем к описанию этапов внедрения ERP-системы.

3.7 Этапы внедрения 1С:ERP

Проект внедрения ERP-системы включает этапы:

– предпроектное обследование (создание «Технического задания»);

– проектирование (создание «Эскизного проекта»);

– разработка;

– внедрение.

Диаграмма проекта внедрения представлена на рисунке 31.

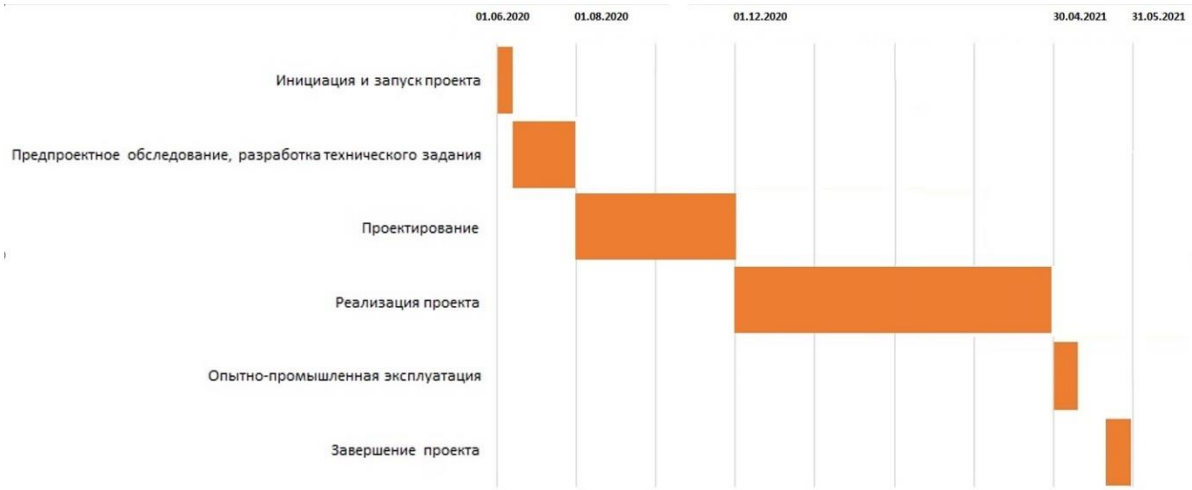


Рисунок 31 - Диаграмма проекта внедрения

Рассмотрим данные этапы более подробно.

Предпроектное обследование. На данном этапе изучается текущее состояние дел в области управления, определяются проблемные зоны и желаемое состояние бизнес-процессов.

Обследование текущего состояния выполняется на основе интервью, которое проводит проектная команда с руководителями и специалистами отдела продаж и других подразделений предприятия. Здесь изучается организационная структура компании, основные бизнес-процессы, продукция компании.

На основе анализа ситуации «Как есть» и «Как должно быть» формируются функциональные требования к ERP-системе. Данный анализ и определение требований осуществлен в научно-исследовательской работе 2. Итогом первого этапа является четкое представление, что необходимо сделать.

Проектирование. На данном этапе осуществляется проектирование внедряемой ERP-системы. Цель этапа описать способы реализации

требований к программе. Для решения данной задачи был проведен анализ функционала ERP-системы 1С:ERP Управление предприятием 2, где было выявлено, что его достаточно для покрытия сформированных требований к системе (с некоторыми доработками). В общем случае этап проектирования дает ответ на вопрос «Как нужно делать, чтобы выполнить сформированные функциональные требования к системе».

Разработка. Данный этап включает в себя следующее:

– программирование;

– настройка ERP-системы;

– разработка эксплуатационной документации (при необходимости - используем руководство от компании-разработчика ERP-системы);

– тестирование.

Внедрение. Этап включает:

– перенос накопленных данных;

– обучение пользователей;

– опытная эксплуатация;

– сдача в промышленную эксплуатацию;

– аудит проекта.

Этап программирования включает необходимые доработки в связи с добавлением подсистемы «Проект поставки» (учитывает необходимые данные для товаров, которые закупаются для перепродажи).

Настройка ERP-системы включает регистрацию пользователей, предоставление им прав доступа, настройка интерфейсов, заполнение необходимых справочников, установку основных параметров работы системы.

Разработка эксплуатационной документации подразумевает написание инструкций для пользователей системы. В нашем случае, учитывая, что программа внедряется по большей части в типовом варианте, в качестве такой документации используем руководства фирмы 1С.

Тестирование необходимо для проверки работоспособности системы. Тестирование может проводиться на сквозных заранее подготовленных примерах или же посредством тестового запуска системы в эксплуатацию. Таким образом, группа тестирования выполняет тестовый пример или свои непосредственные обязанности с помощью системы.

Этап обучения пользователей подготавливает их к работе в ERP-системе.

Опытная эксплуатация означает начало работы с ERP-системой всех пользователей. На этом этапе важно обеспечить обратную связь о проблемах, возникающих у пользователей. Это необходимо, чтобы оперативно оказать консультацию или сделать необходимую дополнительную настройку системы.

В результате опытной эксплуатации все сотрудники должны научиться работать с ERP-системой, а также должны быть сделаны дополнительные настройки системы по предложениям пользователей.

За этапом опытной эксплуатации следует этап промышленной эксплуатации и происходит сдача проекта.

Последним этапом внедрения является аудит проекта. На этом этапе проводится анализ работы сотрудников с ERP-системой и по его результатам вырабатываются корректирующие воздействия. При помощи аудита контролируется выполнение регламентов по работе с системой.

В данном разделе описаны этапы внедрения ERP-системы - 1С:ERP Управление предприятием 2. Далее перейдем к описанию подсистемы «Проект поставки», встраиваемого в конфигурацию системы.

3.8 Подсистема «Проект поставки» встроенная в конфигурацию 1С:ERP

Данная подсистема служит для описания товаров, которые закупаются от поставщиков - их характеристики, составы, коды продукции, сведения о

маркировке, штрихкодах и прочих сведениях необходимых для процесса торговли. Каждая закупка оформляется отдельным документом Проект поставки. Данная подсистема встроена в типовую конфигурацию 1С:ERP чтобы обеспечить необходимые бизнес-процессы заказчика, связанные с продажей закупных товаров.

Описание структуры метаданных данной подсистемы представлено на рисунке 32

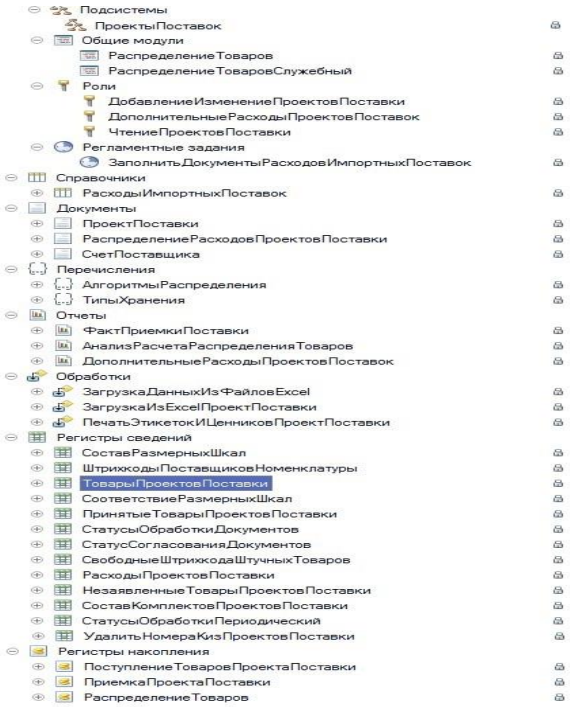


Рисунок 32 - Подсистема «Проект поставки»

Основой данной подсистемы является документ «Проект поставки» и регистр сведений «Товары проектов поставки».

Структура метаданных документа «Проект поставки» представлена на рисунке 33.

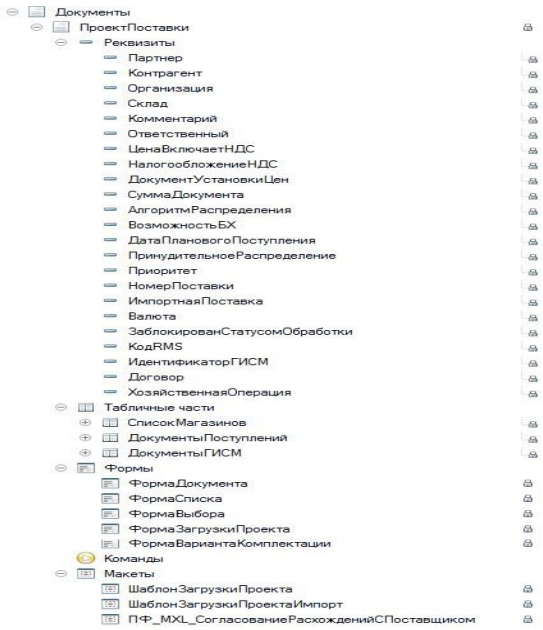


Рисунок 33 - Структура документов «Проект поставки»

Загрузка проекта поставки происходит через шаблон, в который данные копируются из excel. Состав реквизитов для загрузки представлен в таблице Б.1 Приложения Б.

Форма загрузки проекта поставки представлена на рисунке 34.

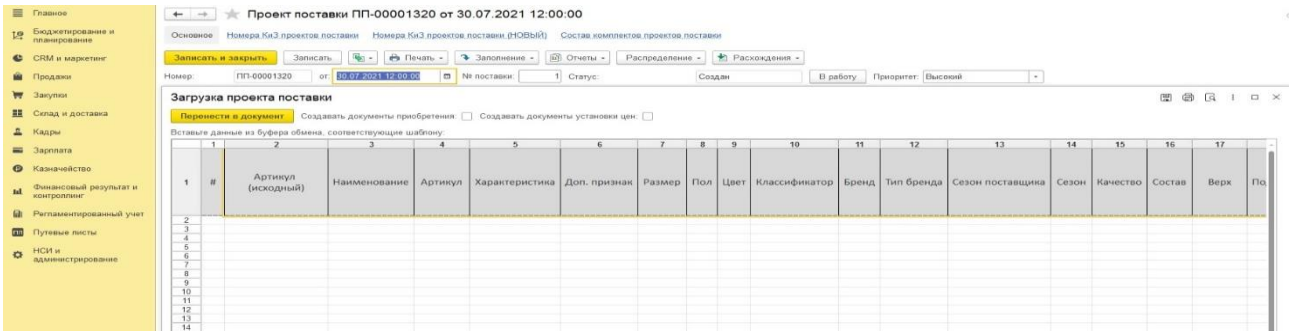


Рисунок 34 - Форма загрузки проекта поставки

Форма документа «Проект поставки» представлена на рисунке 35.

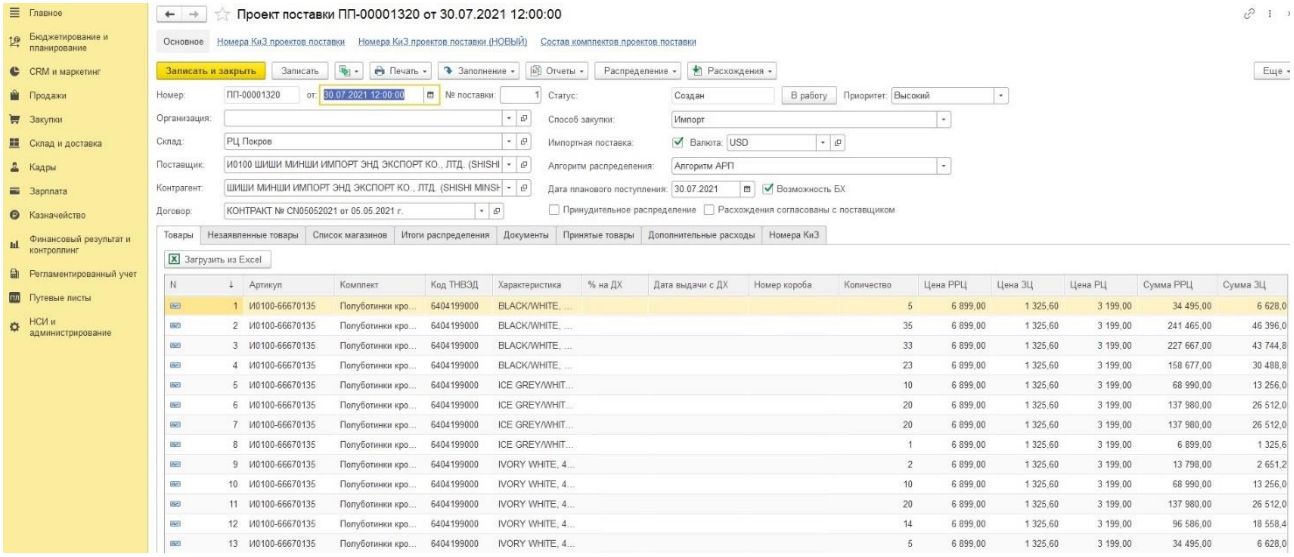


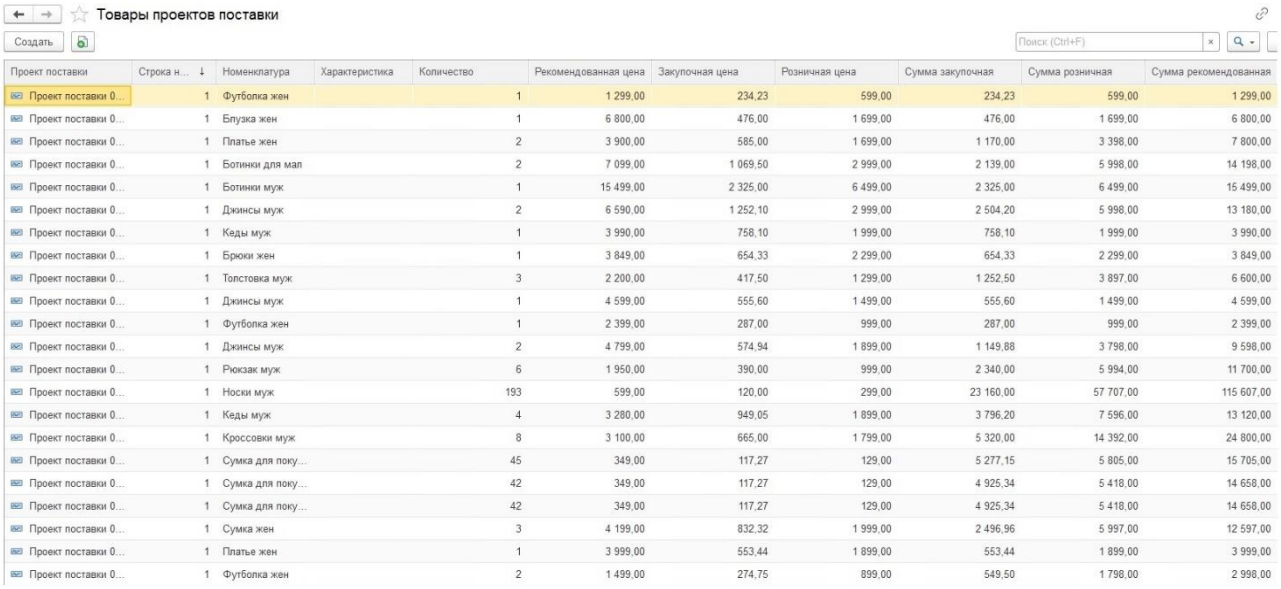
Рисунок 35 - Форма документа «Проект поставки»

Табличная часть «Товары» документа «Проект поставки» представляют собой записи регистра сведений «Товары проектов поставки».

На основании загрузки данных проекта поставки создаются нормативно-справочная информация (НСИ) в системе-справочники номенклатура, характеристики, бренды, и т.д. А также создаются документы «Установка цен

номенклатуры», «Приобретение товаров, услуг». Данная подсистема является основой для создания данных в системе по товарам для перепродажи.

Форма регистра сведений «Товары проектов поставки» представлена на рисунке 36.



Проект поставки	Строка н…	Номенклатура	Характеристика	Количество	Рекомендованная цена	Закупочная цена	Розничная цена	Сумма закупочная	Сумма розничная	Сумма рекомендованная
Проект поставки 0…	1	Футболка жен		1	1 299,00	234,23	599,00	234,23	599,00	1 299,00
Проект поставки 0…	1	Блузка жен		1	6 800,00	476,00	1 699,00	476,00	1 699,00	6 800,00
Проект поставки 0…	1	Платье жен		2	3 900,00	585,00	1 699,00	1 170,00	3 398,00	7 800,00
Проект поставки 0…	1	Ботинки для мал		2	7 099,00	1 069,50	2 999,00	2 139,00	5 998,00	14 198,00
Проект поставки 0…	1	Ботинки муж		1	15 499,00	2 325,00	6 499,00	2 325,00	6 499,00	15 499,00
Проект поставки 0…	1	Джинсы муж		2	6 590,00	1 252,10	2 999,00	2 504,20	5 998,00	13 180,00
Проект поставки 0…	1	Кеды муж		1	3 990,00	758,10	1 999,00	758,10	1 999,00	3 990,00
Проект поставки 0…	1	Брюки жен		1	3 849,00	654,33	2 299,00	654,33	2 299,00	3 849,00
Проект поставки 0…	1	Толстовка муж		3	2 200,00	417,50	1 299,00	1 252,50	3 897,00	6 600,00
Проект поставки 0…	1	Джинсы муж		1	4 599,00	555,60	1 499,00	555,60	1 499,00	4 599,00
Проект поставки 0…	1	Футболка жен		1	2 399,00	287,00	999,00	287,00	999,00	2 399,00
Проект поставки 0…	1	Джинсы муж		2	4 799,00	574,94	1 899,00	1 149,88	3 798,00	9 598,00
Проект поставки 0…	1	Рюкзак муж		6	1 950,00	390,00	999,00	2 340,00	5 994,00	11 700,00
Проект поставки 0…	1	Носки муж		193	599,00	120,00	299,00	23 160,00	57 707,00	115 607,00
Проект поставки 0…	1	Кеды муж		4	3 280,00	949,05	1 899,00	3 796,20	7 596,00	13 120,00
Проект поставки 0…	1	Кроссовки муж		8	3 100,00	665,00	1 799,00	5 320,00	14 392,00	24 800,00
Проект поставки 0…	1	Сумка для поку…		45	349,00	117,27	129,00	5 277,15	5 805,00	15 705,00
Проект поставки 0…	1	Сумка для поку…		42	349,00	117,27	129,00	4 925,34	5 418,00	14 658,00
Проект поставки 0…	1	Сумка для поку…		42	349,00	117,27	129,00	4 925,34	5 418,00	14 658,00
Проект поставки 0…	1	Сумка жен		3	4 199,00	832,32	1 999,00	2 496,96	5 997,00	12 597,00
Проект поставки 0…	1	Платье жен		1	3 999,00	553,44	1 899,00	553,44	1 899,00	3 999,00
Проект поставки 0…	1	Футболка жен		2	1 499,00	274,75	899,00	549,50	1 798,00	2 998,00

Рисунок 36 - Форма регистра сведений «Товары проектов поставки»

Тестирование как подсистемы «Проекты поставок», так и в целом всего информационного продукта осуществлялось посредством тестового запуска системы в эксплуатацию. Данный тестовый запуск завершился успешно. Проект перешел в стадию опытной, а затем промышленной эксплуатации системы.

Листинг программных модулей подсистемы «Проекты поставок» представлены в Приложении В.

В данном разделе описана подсистема «Проекты поставок» - доработка, которая была встроена в типовую конфигурацию 1С:ERP для реализации требуемых бизнес-процессов заказчика, одновременно с реализацией проекта внедрения данной корпоративной информационной системы

Выводы по главе 3

Осуществлена реализация ERP-системы на базе 1С:ERP Управление предприятием 2. Выполнена настройка данной системы, назначены права пользователям системы, осуществлен ввод первоначальных данных. Сделаны доработки типовой конфигурации 1С:ERP для соответствия требуемым бизнес-процессам заказчикам - добавлена в частности подсистема «Проекты поставок», осуществляющую загрузку данных по товарам для перепродажи. Произведена настройка синхронизации данных БД головного предприятия и территориально-распределенных баз данных филиалов. Осуществлено тестирование программного продукта путем его тестового запуска. Данный запуск прошел успешно. Проект перешел в стадию промышленной эксплуатации.

Глава 4 Анализ эффективности распределенной корпоративной информационной системы и апробация результатов

4.1Анализ эффективности внедренной ERP-системы

Оценку эффективности внедрения ERP-системы принято оценивать с помощью следующих подходов:

1) экономические методы оценки эффективности;
2) методы проектного менеджмента;
3) использование процессного подхода.

Группа экономических методов характеризуется расчетом следующих коэффициентов:

– показатель возврата инвестиций (ROI, return on investment);
– совокупная стоимость владения (TCO, total cost of ownership);
– эффективность затрат (CBA, cost-benefits analysis).

Коэффициент возврата инвестиций после внедрения ERP-системы рассчитывается по формуле:

$$\mathrm{ROI} = \frac{\mathrm{CBI} - \mathrm{TCO}}{\mathrm{TCO}} \qquad (12)$$

где ROI - коэффициент возврата инвестиций;

TCO - совокупная стоимость владения ERP-системой;

CBI - выгода от внедрения ERP-системы (прирост доходов).

При расчете полной стоимости владения (TCO) учитывают затраты на внедрение, доработку и эксплуатацию. Данный показатель позволяет учесть только затраты. Очевидно, что для многопользовательской системы данный показатель выше, однако и выгоды от ее внедрения значительны. Поэтому необходимо помимо затрат учесть прирост доходов после внедрения ERP-системы. Это учитывается с помощью ROI - коэффициента возврата инвестиций.

Существует сложность в определении выгод от внедрения в количественном выражении на стадии принятия решения о выборе ERP-

системы. Поэтому для оценки эффективности внедрения используют метод CBA (Cost Benefit Analysis) - анализ выгодности затрат. Данный метод подвергает детальному рассмотрению каждой статьи затрат. Эффективность затрат (CBA) основывается на сравнении двух альтернатив - с использованием системы, и без ее применения (учитываются и возможные потери (opportunity cost) если проект не будет реализован). Для выбора оптимального варианта сравниваются - выгода от проекта и затраты по его реализации или сохранение существующей информационной системы (выгода и издержки этого варианта).

Метод CBA предполагает экспертные оценки относительно выгодности альтернативных вариантов.

В дополнение к экономическим методам оценки эффективности относят:

– NPV (чистый приведенный доход);

– IRR (внутренняя норма рентабельности);

– экономическая добавленная стоимость (EVA, EconomicValue Added) - чистая операционная прибыль после уплаты налогов (NOP, Net Operating Profit) за вычетом затрат на капитал (методология EVA требует учета всех инвестиций в том числе первоначальные денежные вложения, расходы на техническую поддержку, затраты на обучение и др.);

– совокупный экономический эффект (TEI - расширение метода анализа преимуществ и затрат, рассматриваются такие параметры как стоимость системы, её преимущества и гибкость (для данных параметров определяется некоторый уровень риска), анализ стоимости ведется по методу TCO);

– ожидаемый экономический эффект (EEI, Expected Economic Impact), рассчитываемый до имплементации системы;

– расчёт годовой экономии (AS, Annual Savings - годовая экономия - сокращение эксплуатационных (операционных) расходов и экономия в связи с повышением производительности труда);

– фактический экономический эффект (OEI, Operating Economic Impact), который рассчитывается после имплементации проекта;

– капитальные расходы (затраты) на проект (CAPEX);

– заработная плата экспертов и специалистов на всех этапах проектирования (ASES, Aggregated Salaries of Experts and Specialists).

Оценка эффективности внедрения с позиций денежного эквивалента не отражает всю специфику бизнес-процессов и внутренних процессов проекта внедрения. Расширить взгляд на эти процессы позволяют методы проектного и процессного менеджмента.

Анализ эффективности внедрения ERP-системы с использованием методов проектного менеджмента проводится посредством поэтапной укрупненной детализации всех операций в составе проекта внедрения. Применяются методики PEST/COST-анализа (Program Evaluation Review Technique), концепция C/S CSC (Cost/Schedule Control System Criteria). Метод C/S CSC использует сетевые модели планирования стоимости и времени проекта. При использовании метода разрабатываются различные сценарии развития, которые позволяют оценить эффективность внедрения на уровне отдельных операций, так и укрупненных стадий на основе двух показателей - соотношение объема плановых и выполненных работ, а также соотношение плановых и фактических затрат.

Можно оценить эффективность внедрения ERP-системы на основе процессного подхода рассматривая проект внедрения с точки зрения эффективности внутренних процессов. Есть отличие от методов проектного менеджмента - двусторонняя направленность процессов планирования и контроля. Планирование процессов осуществляется «сверху-вниз» (исходя из главных целей, проект разбивается на основные процессы, каждый из которых состоит из подпроцессов). Контроль осуществляется «снизу-вверх» - на основе показателей эффективности выполнения отдельных подпроцессов, которые затем объединяются на более высоком уровне. Преимущество процессного подхода в возможности контроля эффективности проекта через

«центры ответственности», собирающие, анализирующие и передающие информацию вышестоящему центру.

Процессный подход позволяет дать оценку эффекта от воздействия внедрения на бизнес-процессы. В результате выделяют процессы, на которые внедрение ERP-системы оказало положительное влияние и бизнес-процессы, на которые процесс внедрения оказал негативное влияние. Положительным влиянием является увеличение производительности процесса, сокращение времени выполнения и снижение объемов использования дополнительных внутренних ресурсов. Таким образом, улучшением процесса является любое уменьшение затрат финансовых и временных ресурсов на выполнение бизнес-процесса. Негативным влиянием автоматизации является, соответственно, любое увеличение расходов на выполнение процесса.

Существуют также качественные методы оценки целесообразности внедрения. Одним из таких методов является метод информационной экономики. Метод заключается в том, что руководство компании определяет приоритеты в развитии бизнеса и расставляет приоритеты проектных критериев. Недостатком метода является высокий уровень абстракции и неформализованность процедур формирования критериев.

Еще один качественный метод оценки целесообразности внедрения- IT Scorecard (адаптированная для ИТ-службы система сбалансированных показателей). В IT Scorecard выбирают четыре сбалансированных направления влияния ИТ на бизнес: помощь в развитии бизнеса компании, повышение качества продукции, повышение качества принятия решений и повышение производительности труда. Это более конкретизированные критерии, что и в методе информационной экономики и обладает теми же недостатками.

Методы экономического анализа базируются на инвестиционных показателях, которые трудно оценить до начала процесса внедрения (любая оценка, заложенная в модель расчета ROI или NPV, базируясь на эмпирических оценках, будет сильно отличаться от реальности - не сможет

точно спрогнозировать отдачу от инвестиций в ERP-систему). Методы процессного анализа предполагают оценку эффективности через применения методов управления проектом и контроля за сроком проведения работ и уровнем их исполнения, но они не позволяют провести прединвестиционную оценку эффективности и осуществить выбор наилучшим образом удовлетворяющей задачам компании ERP-системы. Дополнительные сложности при использовании экономических методов и оценки выгод от внедрения, связаны со сроком реализации бизнес-преимуществ от внедрения, которые составляют не менее 2 лет (исследования показывают, что выгоды от внедрения можно оценить только по прошествии нескольких лет после внедрения ERP-системы).

Качественная оценка эффективности внедрения ERP-системы в компании позволяет не только оценить, насколько эффективно использование ERP, но и может использоваться как полноценный инструмент управления проектом внедрения, на всех этапах жизненного цикла проекта.

В зависимости от корпоративных требований или задач, стоящих перед компанией, она может разработать собственные показатели для анализа эффективности внедрения системы. В основе таких показателей, как правило, лежит оценка предполагаемых выгод от внедрения ERP-системы и затрат на ее использование. Рассмотрим, как оценить эти выгоды и затраты.

Цели и задачи компании, которые вы хотите решить с помощью ERP-системы, необходимо сформулировать с учетом стратегии бизнеса. Чтобы точнее оценить ожидаемый возврат от вложенных средств, нужно ответить на следующие вопросы:

1) достижение каких показателей (стратегических и тактических) наиболее важно для бизнеса компании;

2) назначены ли ответственные за достижение ожидаемых результатов, а также определен ли механизм учета вносимых изменений;

3) поможет ли внедрение системы (если да, то как, насколько и когда).

Произведем сравнение эффективности предприятия на основании опроса ключевых пользователей (и ранжирования полученных сведений) до и после внедрения ERP-системы по ряду критериев полезности. Данные до внедрения КИС представлены в таблице 6.

Таблица 6 - Эффективность предприятия до внедрения ERP-системы

Характеристика сравнения	Весовой коэффициент, ki	Рейтинговая оценка, ωi
Планирование, контроль исполнения финансовых и оперативных планов	0,16	1
Взаимоотношения с клиентами	0,14	0
Увеличение объема продаж	0,18	1
Уменьшение времени исполнения заказов	0,16	2
Снижение производственных, операционных затрат	0,18	1
Уменьшение вложений в складские запасы	0,16	2
Сокращение времени на разработку и вывод новой продукции на рынок	0,16	1
Надежность	0,18	2
Эргономичность	0,12	2
Порядок приобретения и стоимость	0,18	2
Обслуживание системы	0,09	3

Данные после внедрения ERP-системы представлены в таблице 7.

Таблица 7 - Эффективность предприятия после внедрения ERP-системы

Характеристика сравнения	Весовой коэффициент, $\boldsymbol{ki}$	Рейтинговая оценка, $\boldsymbol{\omega i}$
Планирование, контроль исполнения финансовых и оперативных планов	0,16	4

Продолжение таблицы 7

Характеристика сравнения	Весовой коэффициент, ki	Рейтинговая оценка
Взаимоотношения с клиентами	0,14	3
Увеличение объема продаж	0,18	3
Уменьшение времени исполнения заказов	0,16	4
Снижение производственных, операционных затрат	0,18	3
Уменьшение вложений в складские запасы	0,16	4
Сокращение времени на разработку и вывод новой продукции на рынок	0,16	3
Надежность	0,18	5
Эргономичность	0,12	3
Порядок приобретения и стоимость	0,18	3
Обслуживание системы	0,09	5

Значения весовых коэффициентов приведены в таблице 8.

Таблица 8 - Значения весовых коэффициентов

Качественное значение коэффициента	Числовое значение коэффициента, ki
«жизненно важно»	0,18
«необходимо»	0,16
«крайне важно»	0,14
«очень важно»	0,12
«важно»	0,11
«имеет значение»	0,09

Продолжение таблицы 8

Качественное значение коэффициента	Числовое значение коэффициента, ki
«требуется»	0,07
«весьма полезно»	0,06
«желательно»	0,04
«необязательно»	0,03

Значения рейтинговых оценок представлены в таблице 9.

Таблица 9 - Рейтинговые оценки

Рейтинговая оценка	Рейтинговый балл, ωi
отсутствует	0
неприемлемо	1
приемлемо	2
посредственно	3
хорошо	4
отлично	5

Полезность корпоративной информационной системы определим по формуле:

$$A = \sum_{i=1}^{I} k_i \omega_i \quad (13),$$

где ω_i - рейтинговая оценка характеристики; k_i - весовой коэффициент характеристики, I - количество характеристик сравнения корпоративных информационных систем.

Выполним расчет значений показателей эффективности по формуле (13):

A1 = 2,55, где A1 - оценка эффективности до внедрения ERP-системы.

A2 = 6,15, где A2 - оценка эффективности после внедрения ERP-системы.

Сравнив, данные показатели делаем вывод, что эффективность работы предприятия выросла в 2,4 раза.

Более того, необходимо отметить, что часть показателей недооценена, так как с момента внедрения ERP-системы прошло мало времени, их необходимо переоценить на горизонте двух лет эксплуатации системы.

4.2 Апробация результатов исследования

Для подтверждения гипотезы исследования среди работников предприятия ООО Торговый Дом «Купеческий» был проведен опрос, включающий следующие вопросы:

1) Используете ли вы в своей работе внедренную ERP-систему? (X)
2) Помогает ли ERP-система в поддержке принятия управленческих решений лучше, чем прежняя информационная система? (Y)

Сформулируем «нулевую гипотезу» об отсутствии поддержки в принятии управленческих решений до и после использования ERP-системы (H0) и «альтернативную гипотезу» о наличии поддержки в принятии управленческих решений в ERP-системе (H1). При обработке статистических данных опроса использован коэффициент корреляции φ.

В опросе приняло участие 40 сотрудников предприятия, результаты приведены в Приложении Г.

Вычислим коэффициент корреляции φ по формуле (14):

$$\varphi = (pxy - px \cdot py)/\sqrt{(px \cdot (1 - px) \cdot py \cdot (1 - py))} \quad (14)$$

где pxy - частота признака, который имеет значения 1 по X и по Y;

px - частота признака со значением 1 по X;

py - частота признака со значением 1 по Y;

$(1 - px)$ - частота признака со значением 0 по X;

$(1 - py)$ - частота признака со значением 0 по Y;

Значение частоты px вычисляется суммированием значения 1 по X и делением суммы на общее число элементов столбца X. Аналогично

вычисляется частота py по столбцу Y. Значение частоты px соответствует доле опрошенных у которых ответ на вопрос (X) утвердительный, $px = \frac{29}{40} = 0{,}725$.

Тогда доля опрошенных с отрицательным ответом на вопрос (X) равна $(1 - px) = 1 - 0{,}725 = 0{,}275$.

Значение частоты py соответствует доле опрошенных у которых ответ на вопрос (Y) утвердительный, $py = \frac{25}{40} = 0{,}625$.

Тогда доля опрошенных с отрицательным ответом на вопрос (Y) равна $(1 - py) = 1 - 0{,}625 = 0{,}375$.

Значение частоты pxy соответствует доле опрошенных с утвердительными ответами и на вопрос (X), и на вопрос (Y), $pxy = \frac{22}{40} = 0{,}55$.

Подставим значения в формулу (14), имеем

$$\varphi = \frac{pxy - px \cdot py}{px \cdot (1 - px) \cdot py \cdot (1 - py)} = \frac{0{,}55 - 0{,}725 \cdot 0{,}625}{0{,}725 \cdot 0{,}275 \cdot 0{,}625 \cdot 0{,}375} = 0{,}448$$

Далее вычислим выражение T_φ по формуле (4):

$$T_\varphi = \varphi \cdot \sqrt{(n-2)/(1-\varphi^2)} \qquad (15),$$

где $(n - 2)$, соответствует числу степеней свободы 40-2 = 38.

Подставим значения в формулу (15), имеем:

$T_\varphi = 0{,}448 \cdot \sqrt{38/(1 - 0{,}448^2)}$ = 3,09.

Определим по таблице критические значения критерия Стьюдента, имеем:

$t_{\text{кр}} = 2{,}024$ для $P \leq 0{,}05$

$t_{\text{кр}} = 2{,}712$ для $P \leq 0{,}01$

Построим ось значимости, отобразив значение полученного эмпирического значения T_φ относительно критических точек (рисунок 37)

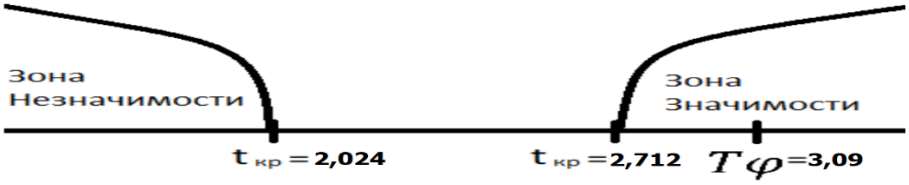


Рисунок 37 - Ось значимости значения

Из рисунка следует, что значение $T_\varphi = 3{,}09$ находится в зоне значимости.

Принимается гипотеза Н1 о сходстве коэффициента корреляции φ с единицей. Таким образом, гипотеза исследования, вынесенная на защиту диссертации, подтверждается.

Выводы по главе 4

Апробация полученных результатов внедренной ERP-системы доказала, что использование системы приводит к улучшению поддержки принятия управленческих решений - приводит к улучшению показателей качества принятия решения. Использование реализованной распределенной информационной системы дало следующие преимущества: увеличение скорости принятия решения, быстрое получение различного вида отчетов; оперативное выявление трендов, используя методы прогнозирования, работа территориально-удаленных филиалов в едином информационном пространстве, улучшены взаимоотношения с клиентами, уменьшены вложения в складские запасы, уменьшено время исполнения заказов, снижены производственные и операционные затраты, оптимизированы бизнес-процессы компании. Таким образом, внедрение ERP-системы «1С: Управление предприятием 2» оказалось эффективным и целесообразным.

Заключение

В ходе работы над магистерской диссертации была проанализирована литература, касающаяся вопросов внедрения и применения ERP-систем, изучена концепция данных систем, вопросы внедрения ERP-систем на предприятиях.

Изучена организационная структура конкретного предприятия, его основные бизнес-процессы, выявлены недостатки в данных процессах.
Построены модели бизнес-процессов предприятия, спроектирована логическая структура ERP-системы. Предложено провести реорганизацию данных процессов путем внедрения ERP-системы для территориально-распределенного предприятия.

Произведен выбор ERP-системы для внедрения, и осуществлен проект внедрения данной системы на предприятии ООО Торговый Дом «Купеческий». Разработана модель распределенной базы данных предприятия, позволяющая рассчитать оптимальные параметры функционирования распределенной базы данных предприятия.

В результате анализа бизнес-процессов предприятия и функционала внедряемой ERP-системы выявлена необходимость кастомизации типового функционала программного продукта. Осуществлена разработка и внедрение подсистемы «Проект поставки» которая необходима для осуществления закупок широкой номенклатуры товаров. Листинг кода данной подсистемы приведен в Приложении В.

По результатам внедрения проанализирована эффективность внедрения и апробация результатов внедрения. Путем статистического обследования доказана эффективность внедренной системы, доказана гипотеза магистерского исследования о том, что поддержка принятия решений в управлении ресурсами территориально-распределенного предприятия будет более эффективной при использовании ERP-системы.

Список используемых источников

1. Автоматизация деятельности предприятия розничной торговли с использованием информационной системы Microsoft Dynamics NAV : учебное пособие / В. И. Грекул, Н. Л. Коровкина, Д. А. Богословцев, Н. Н. Синайская. - 3-е изд. - Москва, Саратов : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2020. - 229 с. - Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. - URL: http://www.iprbookshop.ru/89413.html (дата обращения: 04.11.2021)

2. Аттетков, А. В. Методы оптимизации : учебное пособие / А. В. Аттетков, В. С. Зарубин, А. Н. Канатников. - Саратов : Вузовское образование, 2018. - 272 с. - Текст: электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. - URL: http://www.iprbookshop.ru/77664.html (дата обращения: 04.11.2021)

3. Баженов, Р. И. Интеллектуальные информационные технологии в управлении : учебное пособие / Р. И. Баженов. - Саратов : Ай Пи Эр Медиа, 2018. - 117 с. - Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. - URL: http://www.iprbookshop.ru/72801.html (дата обращения: 04.11.2021)

4. Балдин, К. В. Информационные системы в экономике : учебник / К. В. Балдин, В. Б. Уткин. - 8-е изд. - Москва : Дашков и К, 2019. - 395 с. - Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. - URL: http://www.iprbookshop.ru/85638.html (дата обращения: 04.11.2021)

5. Васильев, Р. Б. Управление развитием информационных систем : учебник / Р. Б. Васильев, Г. Н. Калянов, Г. А. Левочкина. - 3-е изд. - Москва : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2020. - 507 с. - Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. - URL: http://www.iprbookshop.ru/94864.html (дата обращения: 04.11.2021)

6. Введение в программные системы и их разработку : учебное пособие / С. В. Назаров, С. Н. Белоусова, И. А. Бессонова [и др.]. - 3-е изд. - Москва, Саратов : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2020. - 649 с. - Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. - URL: http://www.iprbookshop.ru/89429.html (дата обращения: 04.11.2021)

7. Википедия – свободная энциклопедия / [Электронный ресурс] - URL: https://ru.wikipedia.org/wiki/ (дата обращения 04.11.2021)

8. Головицына, М. В. Информационные технологии в экономике : учебное пособие / М. В. Головицына. - 3-е изд. - Москва, Саратов : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2020. - 589 с. - Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. - URL: http://www.iprbookshop.ru/89438.html (дата обращения: 04.11.2021)

9. Граничин, О. Н. Информационные технологии в управлении : учебное пособие / О. Н. Граничин, В. И. Кияев. - 3-е изд. - Москва, Саратов : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2020. - 400 с. - Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. - URL: http://www.iprbookshop.ru/89437.html (дата обращения: 04.11.2021)

10. Долженко, А. И. Технологии командной разработки программного обеспечения информационных систем : курс лекций / А. И. Долженко. - 3-е изд. - Москва : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Эр Медиа, 2019. - 300 с. - Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. - URL: http://www.iprbookshop.ru/79723.html (дата обращения: 04.11.2021)

11. Ильин, В. В. Внедрение ERP-систем: управление экономической эффективностью / В. В. Ильин. - 3-е изд. - Москва : Интермедиатор, 2018. - 296 с. - Текст : электронный // Электронно-библиотечная система IPR

BOOKS : [сайт]. - URL: http://www.iprbookshop.ru/89565.html (дата обращения: 04.11.2021)

12. Информационные технологии и управление предприятием / В. В. Баронов, Г. Н. Калянов, Ю. Н. Попов, И. Н. Титовский. - 2-е изд. - Саратов : Профобразование, 2019. - 327 c. - Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. - URL: http://www.iprbookshop.ru/87996.html (дата обращения: 04.11.2021)

13. Когаловский, М. Р. Перспективные технологии информационных систем / М. Р. Когаловский. - 2-е изд. - Москва : ДМК Пресс, 2018. - 285 c. - Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. - URL: http://www.iprbookshop.ru/89594.html (дата обращения: 04.11.2021)

14. Корпорация Парус, решения ERP-класса/ [Электронный ресурс] - URL: https://citk-parus.com/catalog/resheniya-dlya-opk-i-biznesa/ (дата обращения 04.11.2021)

15.Леоненков, А. В. Объектно-ориентированный анализ и проектирование с использованием UML и IBM Rational Rose. Курс лекций : учебное пособие для студентов вузов, обучающихся по специальностям в области информационных технологий / А. В. Леоненков. - Москва, Саратов : Интернет-Университет Информационных Технологий (ИНТУИТ), Вузовское образование, 2017. - 318 c. - Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. - URL: http://www.iprbookshop.ru/67388.html (дата обращения: 04.11.2021)

16. Метод Нелдера Мида / [Электронный ресурс]. - URL: http://www.machinelearning.ru/wiki/index.php?title=Метод_Нелдера-Мида (дата обращения 04.11.2021)

17. Метод множителей Лагранжа / [Электронный ресурс] - URL: http://www.math.mrsu.ru/text/courses/method/metod_mnogitelei_lagranga.htm (дата обращения 04.11.2021)

18. Обзор Microsoft Dynamics 365 / [Электронный ресурс] - URL: https://dynamics.microsoft.com/ru-ru/ (дата обращения 04.11.2021)

19. Отечественная, российская ERP система Галактика / [Электронный ресурс] - URL: https://www.galaktika.ru/erp/ (дата обращения 04.11.2021)

20. Павлова, Е. А. Технологии разработки современных информационных систем на платформе Microsoft.NET : учебное пособие / Е. А. Павлова. - 3-е изд. - Москва, Саратов : Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа, 2020. - 128 с. - Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. - URL: http://www.iprbookshop.ru/89479.html (дата обращения: 04.11.2021)

21. Сайт национальный открытый университет ИНТУИТ/ [Электронный ресурс]. - URL: http://intuit.ru (дата обращения 04.11.2021)

22. Синицын, С. В. Основы разработки программного обеспечения на примере языка С : учебное пособие для СПО / С. В. Синицын, О. И. Хлытчиев. - Саратов : Профобразование, 2019. - 212 с. - Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. - URL: http://www.iprbookshop.ru/86201.html (дата обращения: 04.11.2021)

23. Техническая документация продукта 1С: ERP / [Электронный ресурс] - URL:http://v8.1c.ru/erp/ (дата обращения 04.11.2021)

24. Тихобаев, В. М. Математические модели планирования и управления : учебное пособие / В. М. Тихобаев. - Тула : Институт законоведения и управления ВПА, 2018. - 138 с. - Текст : электронный // Электронно-библиотечная система IPR BOOKS : [сайт]. - URL: http://www.iprbookshop.ru/78623.html (дата обращения: 04.11.2021)

25. Туровец, О. Г. Организация производства и управление предприятием : учебник / О. Г. Туровец, М. И. Бухалков, В. Б. Родионов [и др.] ; под ред. О. Г. Туровца. - 3-е изд. - Москва : НИЦ ИНФРА-М, 2015. - 506 с. - (Высшее образо-вание: Бакалавриат). - Текст : электронный. - URL: https://new.znanium.com/catalog/product/472411 (дата обращения: 04.11.2021)

26. Управление промышленными предприятиями: стратегии, механизмы, системы : монография / О.В. Логиновский, А.А. Максимов, В.Н. Бурков [и др.] ; под ред. О.В. Логиновского, А.А. Максимова. - Москва : ИНФРА-М, 2018. - 410 с. - (Научная мысль). - Текст : электронный. - URL: https://new.znanium.com/catalog/product/945371 (дата обращения: 04.11.2021)

27. Фатхутдинов, Р. А. Организация производства : учебник / Р.А. Фатхутдинов. - 3-е изд., перераб. и доп. - Москва : ИНФРА-М, 2020. - 544 с. - (Высшее образование: Бакалавриат). - Текст : электронный. - URL: https://new.znanium.com/catalog/product/1043130 (дата обращения: 04.11.2021)

28. Эффективное управление организационными и производственными структурами : монография / О.В. Логиновский, А.В. Голлай, О.И. Дранко, А.Л. Шестаков, А.А. Шинкарев ; под ред. О.В. Логиновского. - Москва : ИНФРА-М, 2020. - 450 с. - (Научная мысль). - Текст : электронный. - URL: https://new.znanium.com/catalog/product/1087996 (дата обращения: 04.11.2021)

29. ERP-системы планирования ресурсов предприятия / [Электронный ресурс] - URL: http://iteranet.ru/sys/konsalting/sis/erp/ (дата обращения 04.11.2021)

30. ERP - СТАНДАРТ ПЛАНИРОВАНИЯ РЕСУРСОВ ПРЕДПРИЯТИЯ И СИСТЕМА / Itstan [Электронный ресурс]. - URL: http://www.itstan.ru/it-i-is/erp-standart-planirovanija-resursov-predprijatija-i-sistema.html (дата обращения 04.11.2021)

31. ERP for Small Businesses and Midsize Companies - / [Электронный ресурс] - URL: https://www.sap.com/products/erp-financial-management/small-business-erp.html (дата обращения 04.11.2021)

32. What is ERP? / [Электронный ресурс] - URL: https://www.sap.com/products/what-is-erp.html (дата обращения 04.11.2021)

Приложение А

Сравнение ERP-систем по функциональности

Таблица А.1 - Сравнение функциональных возможностей российских и импортных ERP-систем (обозначение: «+» - реализован полный функционал; «+/–» - только базовые функции; «–» - функционал отсутствует)

Функциональные Модули	ERP-системы			
	Галактика ERP	1С	Парус	Alfa
Планирование продаж и производства	+	+	+	+
Управление кадрами	+	+	+	+
Управление проектами и программами	+	+	+-	+
Управление производством	+	+	+	+
Управление складом	+	+	+	+
Контроль материальных потоков	+	+	+-	+
Финансы	+	+	+	+
Отраслевые решения	+	+	+-	+
Управление потоком операций (Workflow)	+	+	+	+-
Моделирование и прогнозирование	-	+-	-	+
Управление и оптимизация цепочек поставок	+-	+	+-	+
Поддержка MS Office	+-	+	+	+-
CRM-система	+	+	+	+-
Бизнес-аналитика (BI) и OLAP	+	+	+	+
«Облачные» технологии и мобильное использование	-	+	+-	-
Электронный документооборот	+	+	+	+
Безопасность и администрирование	+-	+	+-	+-
Длительность внедрения (мес.)	10	12	8	9

Приложение Б

Состав реквизитов проекта поставки

Таблица Б.1 - Состав реквизитов для загрузки проекта поставки

Наименование реквизита	
1) Артикул (исходный)	25) Количество ед.
2) Наименование	26) РРЦ
3) Артикул	27) Закупочная цена
4) Характеристика	28) Розничная цена
5) Доп. признак	29) Сумма з.ц.
6) Размер	30) Сумма р.ц.
7) Пол	31) Сумма р.р.ц
8) Цвет	32) Скидка ЗЦ от РРЦ
9) Классификатор	33) Скидка РЦ от РРЦ
10) Бренд	34) Наценка
11)Тип бренда	35) ТОРГ-12
12) Сезон поставщика	36) Номер ГТД
13)Сезон	37) Ед. изм.
14) Качество	38) Цена в валюте
15) Состав	39) Сумма в валюте
16) Верх	40) Вес
17) Подкладка	41) Наименование изготовителя
18) Подошва	42) Юр. адрес изготовителя
19) Страна производителя	43) Наименование Дистрибьютера/Импортера
20) % на ДХ	44) Юр. адрес Дистрибьютера/Импортера
21) Дата выдачи с ДХ	45) GTIN
22) Маркировка	
23) Баркод	
24) Код ТНВЭД	

Приложение В
Листинг программных модулей подсистема «Проекты поставок»

```
//Модуль формы документ «Проект поставки»
#Область ОписаниеПеременных
//&НаКлиенте
//Перем ПараметрыДляЗаписи Экспорт;
#КонецОбласти

#Область ОбработчикиСобытийФормы

&НаСервере
Процедура ПриСозданииНаСервере(Отказ, СтандартнаяОбработка)

		Если Параметры.Свойство("АвтоТест") Тогда // Возврат при получении формы для анализа.
		  Возврат;
		КонецЕсли;

		// Обработчик механизма "ВерсионированиеОбъектов"
		ВерсионированиеОбъектов.ПриСозданииНаСервере(ЭтаФорма);

		// СтандартныеПодсистемы.ПодключаемыеКоманды
		ПодключаемыеКоманды.ПриСозданииНаСервере(ЭтотОбъект);
		// Конец СтандартныеПодсистемы.ПодключаемыеКоманды

		Если Не ЗначениеЗаполнено(Объект.Ссылка) Тогда
		  ПриЧтенииСозданииНаСервере();
		КонецЕсли;

		СобытияФорм.ПриСозданииНаСервере(ЭтаФорма, Отказ, СтандартнаяОбработка);

		Элементы.СтраницаПринятыеТовары.Доступность = ЗначениеЗаполнено(Объект.Ссылка);
		Элементы.СтраницаДопРасходы.Доступность = ЗначениеЗаполнено(Объект.Ссылка);
		Элементы.РасхожденияСогласованыСПоставщиком.Доступность = ЗначениеЗаполнено(Объект.Ссылка);
		Если ЗначениеЗаполнено(Объект.Ссылка) Тогда
		  РасхожденияСогласованыСПоставщиком =
Документы.ПроектПоставки.ПолучитьСтатусСогласования(Объект.Ссылка);
		КонецЕсли;

		Элементы.ДокументыПоступленийЗагрузитьПТУ.Видимость = РольДоступна("ПолныеПрава");

		Элементы.ХозяйственнаяОперация.РежимВыбораИзСписка = Истина;
		Элементы.ХозяйственнаяОперация.СписокВыбора.Добавить(ПредопределенноеЗначение("Перечисление.Хозяйственные
Операции.ЗакупкаУПоставщика"), "Закупка в России");
		Элементы.ХозяйственнаяОперация.СписокВыбора.Добавить(ПредопределенноеЗначение("Перечисление.Хозяйственные
Операции.ЗакупкаПоИмпорту"));
		Элементы.ХозяйственнаяОперация.СписокВыбора.Добавить(ПредопределенноеЗначение("Перечисление.Хозяйственные
Операции.ЗакупкаВСтранахЕАЭС"));

КонецПроцедуры

&НаСервере
Процедура ПриЧтенииНаСервере(ТекущийОбъект)

		ФормаТолькоПросмотр = ЭтаФорма.ТолькоПросмотр;

		// Обработчик механизма "ДатыЗапретаИзменения"
		ДатыЗапретаИзменения.ОбъектПриЧтенииНаСервере(ЭтаФорма, ТекущийОбъект);

		Если ЭтаФорма.ТолькоПросмотр И Не ФормаТолькоПросмотр Тогда

		  ЭтаФорма.ТолькоПросмотр = Ложь;

		  ЗаблокироватьРеквизитыФормыПоДатеЗапрета();

		КонецЕсли;

		ПриЧтенииСозданииНаСервере();

		СобытияФорм.ПриЧтенииНаСервере(ЭтотОбъект, ТекущийОбъект);

		// СтандартныеПодсистемы.ПодключаемыеКоманды
		ПодключаемыеКомандыКлиентСервер.ОбновитьКоманды(ЭтотОбъект, Объект);
		// Конец СтандартныеПодсистемы.ПодключаемыеКоманды
		  ТоварыОбновитьИтоги();
КонецПроцедуры
```

Продолжение Приложения В

```
&НаКлиенте
Процедура ПриОткрытии(Отказ)

		АлгоритмРаспределенияПриИзменении(Неопределено);

		ИмпортнаяПоставкаПриИзменении(Неопределено);

		// СтандартныеПодсистемы.ПодключаемыеКоманды
		ПодключаемыеКомандыКлиент.НачатьОбновлениеКоманд(ЭтотОбъект);
		// Конец СтандартныеПодсистемы.ПодключаемыеКоманды

КонецПроцедуры

&НаКлиенте
Процедура ПередЗаписью(Отказ, ПараметрыЗаписи)

		СобытияФормКлиент.ПередЗаписью(ЭтотОбъект, Отказ, ПараметрыЗаписи);

		Если ПараметрыЗаписи.Свойство("ДействиеПослеЗаписи") Тогда
		  ПараметрыЗаписи.Удалить("ДействиеПослеЗаписи");
		КонецЕсли;

		//Если НеВыполнятьПроверкуПередЗаписью Тогда
		//	НеВыполнятьПроверкуПередЗаписью = Ложь;
		//	Возврат;
		//КонецЕсли;
		//
		//Если ИзмененоРаспределениеТоваров И Не Объект.ПринудительноеРаспределение Тогда
		//
		//	Отказ = Истина;
		//
		//	ДополнительныеПараметры = Новый Структура;
		//	ДополнительныеПараметры.Вставить("ПараметрыЗаписи", ПараметрыЗаписи);
		//
		//	РаспределениеТоваровКлиент.ПредложитьУстановитьПринудительноеРаспределение(ЭтаФорма,
		//				Новый
ОписаниеОповещения("ПередЗаписьюПредложитьУстановитьПринудительноеРаспределениеЗавершение", ЭтотОбъект,
ДополнительныеПараметры));
		//
		//КонецЕсли;
		//
		//ОбщегоНазначенияУТКлиент.ЗаписатьОбъектПриНеобходимости(ЭтотОбъект, ПараметрыЗаписи, Отказ);

КонецПроцедуры

&НаКлиенте
Процедура ПередЗаписьюОбработатьЗаписьОбъектаВФорме(Результат, ДополнительныеПараметры) Экспорт

		Если Результат <> КодВозвратаДиалога.Нет Тогда

		  ОбщегоНазначенияУТКлиент.ОбработатьЗаписьОбъектаВФорме(ЭтаФорма,
ДополнительныеПараметры.ПараметрыЗаписи);

		КонецЕсли;

КонецПроцедуры

&НаСервере
Процедура ПриЗаписиНаСервере(Отказ, ТекущийОбъект, ПараметрыЗаписи)

		//НаборЗаписей = РегистрыСведений.РезультатыРаспределенияПроектаПоставки.СоздатьНаборЗаписей();
		//НаборЗаписей.Отбор.ПроектПоставки.Установить(Объект.Ссылка);

		ТекущаяДата = ТекущаяДатаСеанса();

		//НаборЗаписей = РегистрыНакопления.РаспределениеТоваров.СоздатьНаборЗаписей();
		//НаборЗаписей.Отбор.Регистратор.Установить(ТекущийОбъект.Ссылка);

		НаборЗаписей = РегистрыСведений.ПринудительноеРаспределение.СоздатьНаборЗаписей();
		НаборЗаписей.Отбор.ПроектПоставки.Установить(ТекущийОбъект.Ссылка);
		Если Объект.ПринудительноеРаспределение Тогда
		  Для каждого СтрокаТЧ Из РезультатыРаспределения Цикл

			Для каждого ЭлСписка Из СписокДинамическихКолонок Цикл
```

```
                                Количество = СтрокаТЧ[ЭлСписка.Представление];

                                Если Количество > 0 Тогда

                                        Запись = НаборЗаписей.Добавить();

                                        //ЗаполнитьЗначенияСвойств(Запись, СтрокаТЧ);
                                        Запись.СтрокаНомерДанных = СтрокаТЧ.СтрокаДанных;
                                        Запись.Номенклатура = СтрокаТЧ.Номенклатура;
                                        Запись.Характеристика = СтрокаТЧ.Характеристика;

                                        Запись.Назначение = ЭлСписка.Значение;
                                        Запись.Количество = Количество;

                                        Запись.ПроектПоставки = ТекущийОбъект.Ссылка;

                                        Если Запись.Назначение = Перечисления.ТипыХранения.ДлительноеХранение Тогда
                                                Запись.ДатаВыдачи = СтрокаТЧ.ДатаВыдачи;
                                        КонецЕсли;
                                КонецЕсли;

                        КонецЦикла;

                  КонецЦикла;
                КонецЕсли;

                НаборЗаписей.Записать();
                Если Не УправлениеСтатусамиОбработки.УстановитьСтатусОбработкиДокумента(ТекущийОбъект.Ссылка,
СтатусОбработки, СкладПодУправлениемWMS, Не СкладПодУправлениемWMS) Тогда
                  Отказ = Истина;
                КонецЕсли;
КонецПроцедуры

&НаСервере
Процедура ПослеЗаписиНаСервере(ТекущийОбъект, ПараметрыЗаписи)

                Элементы.СтраницаПринятыеТовары.Доступность = ЗначениеЗаполнено(Объект.Ссылка);
                Элементы.РасхожденияСогласованыСПоставщиком.Доступность = ЗначениеЗаполнено(Объект.Ссылка);
                Элементы.СтраницаПринятыеТовары.Доступность = ЗначениеЗаполнено(Объект.Ссылка);

КонецПроцедуры

&НаКлиенте
Процедура ПослеЗаписи(ПараметрыЗаписи)

                ОбщегоНазначенияУТКлиент.ВыполнитьДействияПослеЗаписи(ЭтаФорма, Объект, ПараметрыЗаписи);

                ЗаполнитьДанныеИзРегистров(Истина, Объект.ПринудительноеРаспределение);

КонецПроцедуры

#КонецОбласти

#Область ОбработчикиКомандФормы
// СтандартныеПодсистемы.ПодключаемыеКоманды
&НаКлиенте
Процедура Подключаемый_ВыполнитьКоманду(Команда)
                ПодключаемыеКомандыКлиент.ВыполнитьКоманду(ЭтотОбъект, Команда, Объект);
КонецПроцедуры

&НаСервере
Процедура Подключаемый_ВыполнитьКомандуНаСервере(Контекст, Результат)
                ПодключаемыеКоманды.ВыполнитьКоманду(ЭтотОбъект, Контекст, Объект, Результат);
КонецПроцедуры

&НаКлиенте
Процедура Подключаемый_ОбновитьКоманды()
                ПодключаемыеКомандыКлиентСервер.ОбновитьКоманды(ЭтотОбъект, Объект);
КонецПроцедуры
// Конец СтандартныеПодсистемы.ПодключаемыеКоманды
&НаКлиенте
Процедура ЗагрузитьТоварыИзExcel(Команда)

                ПоступлениеПроверено = НЕ ПроверитьВозможностьРедактированияПоступления();

                Если ПоступлениеПроверено Тогда
```

```
                ОбщегоНазначенияКлиентСервер.СообщитьПользователю("Документы поступления являются проверенными -
редактирование их запрещено. Обратитесь в бухгалтерию.");
                Если НЕ Объект.ИмпортнаяПоставка Тогда
                        Возврат;
                КонецЕсли
            КонецЕсли;

            ДополнительныеПараметры = Новый Структура;
            ДополнительныеПараметры.Вставить("ДокументыПриобретенияПроверены", ПоступлениеПроверено);

            ОбработкаПродолжения = Новый ОписаниеОповещения("ЗагрузитьПроектПоставкиПродолжение", ЭтотОбъект,
ДополнительныеПараметры);

            ПроверитьЗаписьДокумента(ОбработкаПродолжения);

КонецПроцедуры
&НаКлиенте
Процедура СоздатьНоменклатуру(Команда)

Структура("Импорт, СозданиеНоменклатуры", Объект.ИмпортнаяПоставка, Истина),
            //ЭтаФорма, ЭтаФорма.УникальныйИдентификатор,,,Новый
ОписаниеОповещения("ЗагрузитьПроектПоставкиЗавершение", ЭтаФорма), РежимОткрытияОкнаФормы.БлокироватьОкноВладельца);

            ОбработкаПродолжения = Новый ОписаниеОповещения("СоздатьНоменклатуруПродолжение", ЭтотОбъект);

            ПроверитьЗаписьДокумента(ОбработкаПродолжения);

КонецПроцедуры

&НаСервере
Функция ФактЗагружен()

            Запрос = Новый Запрос;
            Запрос.Текст =
            "ВЫБРАТЬ
            | КОЛИЧЕСТВО(*) КАК Записи
            |ИЗ
            | РегистрСведений.ПринятыеТоварыПроектовПоставки КАК ПринятыеТоварыПроектовПоставки
            |ГДЕ
            | ПринятыеТоварыПроектовПоставки.ПроектПоставки = &ПроектПоставки";

            Запрос.УстановитьПараметр("ПроектПоставки", Объект.Ссылка);
            РезультатЗапроса = Запрос.Выполнить();
            Выборка = РезультатЗапроса.Выбрать();
            Выборка.Следующий();

            Записи = Выборка.Записи;

            Если Выборка.Записи = 0 Тогда

              Возврат Ложь;

            иначе

              Возврат Истина;

            КонецЕсли;

КонецФункции

&НаКлиенте
Процедура ОформитьРасхождения(Команда)

            Если Объект.Ссылка.Пустая() Тогда
              ПоказатьПредупреждение(, "Сначала запишите документ!");
              Возврат;
            КонецЕсли;

            ОчиститьСообщения();
            Если ФактЗагружен() Тогда

              ДополнительныеПараметры = Новый Структура;
              ОбработчикОповещения = Новый ОписаниеОповещения("ОповещениеОткрытьЗагрузкаКорректировки", ЭтотОбъект,
ДополнительныеПараметры);
              ОткрытьФорму("Обработка.ЗагрузкаИзExcelПроектПоставки.Форма.ФормаЗагрузкиАкта", Новый
Структура("ПроектПоставки", Объект.Ссылка),,,,,ОбработчикОповещения, РежимОткрытияОкнаФормы.БлокироватьВесьИнтерфейс);
```

```
                иначе

                  ОформитьРасхождения НаСервере();

                КонецЕсли;

                //Если УдалитьОбработанныеОбъекты.Количество()>0 Тогда
                //
                //          Элементы.ФормаОтменитьОформлениеРасхождений.Доступность = Истина;
                //
                //          Для каждого ЭлСписка Из УдалитьОбработанныеОбъекты Цикл
                //
                //                    ОповеститьОбИзменении(ЭлСписка.Значение);
                //
                //          КонецЦикла;
                //
                //КонецЕсли;

                Если Элементы.ГруппаСтраницы.ТекущаяСтраница.Имя = "СтраницаДокументы" Тогда

                  ЗаполнитьДеревоДокументов();

                КонецЕсли;

КонецПроцедуры

&НаКлиенте
Процедура ЗаполнитьИтогиРаспределения(Команда)

                Если РезультатыРаспределения.Количество() > 0 Тогда

                  ОписаниеОповещения = Новый ОписаниеОповещения("ЗаполнитьИтогиРаспределенияЗавершение", ЭтаФорма, Новый
Структура);
                  ПоказатьВопрос(ОписаниеОповещения, "Табличная часть ""Итоги распределения"" будет очищена. Продолжить?",
РежимДиалогаВопрос.ДаНет);

                Иначе

                  ЗаполнитьИтогиРаспределенияЗавершение(КодВозвратаДиалога.Да, Новый Структура);

                КонецЕсли;

КонецПроцедуры

&НаКлиенте
Процедура ВыгрузитьИтогиВExcel(Команда)

                РаспределениеТоваровКлиент.КомандаВыгрузитьРезультатыРаспределенияВExcel(Команда, ЭтотОбъект);

КонецПроцедуры

&НаКлиенте
Процедура ЗагрузитьИтогиИзExcel(Команда)

ОписаниеОповещения("ЗагрузитьРезультатыРаспределенияИзExcelПродолжение", ЭтаФорма, Новый Структура);

                РаспределениеТоваровКлиент.КомандаЗагрузитьРезультатыРаспределенияИзExcel(Команда, ЭтотОбъект);

КонецПроцедуры

&НаКлиенте
Процедура ТолькоСОшибкамиЗаполнения(Команда)

                Элементы.РезультатыРаспределенияТолькоСОшибкамиЗаполнения.Пометка = Не
Элементы.РезультатыРаспределенияТолькоСОшибкамиЗаполнения.Пометка;

                Если Элементы.РезультатыРаспределенияТолькоСОшибкамиЗаполнения.Пометка Тогда
                  Элементы.РезультатыРаспределения.ОтборСтрок = Новый ФиксированнаяСтруктура("ОшибкаЗаполнения", Истина);
                Иначе
                  Элементы.РезультатыРаспределения.ОтборСтрок = Неопределено;
                КонецЕсли;

КонецПроцедуры

&НаКлиенте
Процедура Распределить(Команда)
```

```
		Если НЕ ЗначениеЗаполнено(Объект.АлгоритмРаспределения) Тогда
		  Возврат;
		КонецЕсли;

		//ОбработкаПродолжения = Новый ОписаниеОповещения("РаспределениеТоваровПослеЗаписиДокумента",
ЭтотОбъект);
		ДополнительныеПараметры = Новый Структура;
		ДополнительныеПараметры.Вставить("ОбработкаПродолженияРаспределения", Новый
ОписаниеОповещения("РаспределитьПродолжение", ЭтотОбъект));

		ОбработкаПродолжения = Новый ОписаниеОповещения("РаспределениеТоваровПослеЗаписиДокумента", ЭтотОбъект,
ДополнительныеПараметры);

		ПроверитьЗаписьДокумента(ОбработкаПродолжения);

КонецПроцедуры

&НаКлиенте
Процедура ОтменитьРаспределение(Команда)

		ОписаниеОповещения = Новый ОписаниеОповещения("ОтменитьРаспределениеЗавершение", ЭтотОбъект);
		ПоказатьВопрос(ОписаниеОповещения, "Вы действительно хотите отменить распределение?",
РежимДиалогаВопрос.ДаНет, 60);

КонецПроцедуры

&НаКлиенте
Процедура ПредставлениеДокументаУстановкиЦенНажатие(Элемент, СтандартнаяОбработка)

		СтандартнаяОбработка = Ложь;

		Если ЗначениеЗаполнено(Объект.ДокументУстановкиЦен) Тогда

		  ПоказатьЗначение(Неопределено, Объект.ДокументУстановкиЦен);

		Иначе

		  ПоказатьПредупреждение(Неопределено, "Документы установки цен создаются автоматически при загрузке проекта
поставки.");

		КонецЕсли;

КонецПроцедуры

&НаКлиенте
Процедура ПроверитьИтогиРаспределения(Команда)
		ПроверитьИтогиРаспределенияНаСервере();
КонецПроцедуры

&НаСервере
Процедура ПроверитьИтогиРаспределенияНаСервере()

		Запрос = Новый Запрос;
		Запрос.Текст =
		"ВЫБРАТЬ
		| РезультатыРаспределения.СтрокаДанных КАК СтрокаДанных,
		| РезультатыРаспределения.Номенклатура КАК Номенклатура,
		| РезультатыРаспределения.Количество КАК Количество
		|ПОМЕСТИТЬ ВТ_РезультатыРаспределения
		|ИЗ
		| &РезультатыРаспределения КАК РезультатыРаспределения
		|
		|ИНДЕКСИРОВАТЬ ПО
		| СтрокаДанных
		|;
		|
		|////////////////////////////////////////////////////////////////////////////////
		|ВЫБРАТЬ
		| ТоварыПроектовПоставки.СтрокаНомер КАК СтрокаНомер,
		| ТоварыПроектовПоставки.Номенклатура КАК Номенклатура,
		| ТоварыПроектовПоставки.Количество
		|ПОМЕСТИТЬ ВТ_ДанныеОТоварах
		|ИЗ
		| РегистрСведений.ТоварыПроектовПоставки КАК ТоварыПроектовПоставки
		|ГДЕ
		| ТоварыПроектовПоставки.ПроектПоставки = &ПроектПоставки
```

```
                |
                |ИНДЕКСИРОВАТЬ ПО
                | СтрокаНомер
                |;
                |
                |////////////////////////////////////////////////////////////////////////////
                |ВЫБРАТЬ
                | КОЛИЧЕСТВО(ВТ_ДанныеОТоварах.СтрокаНомер) КАК КоличествоСтрок
                |ИЗ
                | ВТ_ДанныеОТоварах КАК ВТ_ДанныеОТоварах
                |;
                |
                |////////////////////////////////////////////////////////////////////////////
                |ВЫБРАТЬ
                | ЕСТЬNULL(ВТ_РезультатыРаспределения.СтрокаДанных, ВТ_ДанныеОТоварах.СтрокаНомер) КАК СтрокаНомер,
                | ВТ_РезультатыРаспределения.Номенклатура КАК НоменклатураРаспределения,
                | ВТ_РезультатыРаспределения.Количество КАК КоличествоРезультаты,
                | ВТ_ДанныеОТоварах.Количество КАК КоличествоДанные,
                | ВТ_ДанныеОТоварах.Номенклатура КАК НоменклатураТоваров

                |ИЗ
                | ВТ_ДанныеОТоварах КАК ВТ_ДанныеОТоварах
                |            ПОЛНОЕ СОЕДИНЕНИЕ ВТ_РезультатыРаспределения КАК ВТ_РезультатыРаспределения
                |            ПО ВТ_ДанныеОТоварах.СтрокаНомер = ВТ_РезультатыРаспределения.СтрокаДанных
                |ГДЕ
                | (ЕСТЬNULL(ВТ_ДанныеОТоварах.Номенклатура.Артикул, Значение(Справочник.Номенклатура.ПустаяСсылка)) <>
ЕСТЬNULL(ВТ_РезультатыРаспределения.Номенклатура, Значение(Справочник.Номенклатура.ПустаяСсылка))
                |                ИЛИ ЕСТЬNULL(ВТ_ДанныеОТоварах.Количество, 0) <>
ЕСТЬNULL(ВТ_РезультатыРаспределения.Количество, 0))";
                 //Запрос.УстановитьПараметр("РезультатыРаспределения", РезультатыРаспределения.Выгрузить(,
"СтрокаНомерДанных, Артикул, Количество"));
                Запрос.УстановитьПараметр("РезультатыРаспределения", РезультатыРаспределения.Выгрузить(, "СтрокаДанных,
Номенклатура, Количество"));
                Запрос.УстановитьПараметр("ПроектПоставки", Объект.Ссылка);

                РезультатыЗапроса = Запрос.ВыполнитьПакет();

                ВыборкаОбщееКоличество = РезультатыЗапроса[2].Выбрать();

                Если ВыборкаОбщееКоличество.Следующий() и ВыборкаОбщееКоличество.КоличествоСтрок <>
РезультатыРаспределения.Количество() Тогда

                  Сообщить("Не совпадает общее количество строк в таблицах ""Данные о товарах"" и ""Итоги распределения""");

                КонецЕсли;

                Если Не РезультатыЗапроса[3].Пустой() Тогда

                  Выборка = РезультатыЗапроса[3].Выбрать();

                  Пока Выборка.Следующий() Цикл

                        Если Выборка.НоменклатураРаспределения <> Выборка.НоменклатураТоваров Тогда
                              Сообщить("Номенклатура в строке № " + (Выборка.СтрокаНомер) + " не совпадает с номенклатурой в
строке данных о товарах");
                        КонецЕсли;

                        Если Выборка.КоличествоРезультаты <> Выборка.КоличествоДанные Тогда

                              Сообщить("Количество в строке № " + (Выборка.СтрокаНомер) + " не равно количеству в строке
данных о товарах");

                        КонецЕсли;

                  КонецЦикла;

                КонецЕсли;

                Для каждого СтрокаТЧ Из РезультатыРаспределения Цикл

                  Если СтрокаТЧ.ОшибкаЗаполнения Тогда

                        Сообщить("Для строки № " + (СтрокаТЧ.СтрокаДанных) + " не соблюдается правило Итог =
СуммаПоМагазинам + БХ + ДХ - Количество");

                  КонецЕсли;
```

```
		КонецЦикла;

КонецПроцедуры

#КонецОбласти

#Область ОбработчикиСобытийЭлементовШапкиФормы

&НаКлиенте
Процедура ПартнерПриИзменении(Элемент)

		ПриИзмененииПартнераСервер();

КонецПроцедуры

&НаКлиенте
Процедура АлгоритмРаспределенияПриИзменении(Элемент)

		Если Объект.АлгоритмРаспределения =
ПредопределенноеЗначение("Перечисление.АлгоритмыРаспределения.РавномерноеРаспределение") Тогда
		  Объект.ВозможностьБХ = Ложь;
		  Элементы.ВозможностьБХ.Доступность = Ложь;
		Иначе
		  Элементы.ВозможностьБХ.Доступность = Истина;
		КонецЕсли;

КонецПроцедуры

&НаКлиенте
Процедура ИмпортнаяПоставкаПриИзменении(Элемент)
		Элементы.Валюта.Видимость = Объект.ИмпортнаяПоставка;
		Элементы.ТоварыГруппаИмпорт.Видимость = Объект.ИмпортнаяПоставка;
		Элементы.ТоварыИтогСуммаВВалюте.Видимость = Объект.ИмпортнаяПоставка;
		Элементы.ГруппаНДС.Видимость = НЕ Объект.ИмпортнаяПоставка;
		Элементы.ТоварыГруппаНДС.Видимость = НЕ Объект.ИмпортнаяПоставка;
		Элементы.ТоварыИтогСуммаНДС.Видимость = НЕ Объект.ИмпортнаяПоставка;
		Элементы.СтраницаДопРасходы.Видимость = Объект.ИмпортнаяПоставка;
КонецПроцедуры

&НаКлиенте
Процедура СкладПриИзменении(Элемент)

		СкладПриИзмененииНаСервере();

		ОповеститьОбИзменении(Объект.Ссылка);

КонецПроцедуры
&НаКлиенте
Процедура ПринудительноеРаспределениеПриИзменении(Элемент)

		Если Не Объект.ПринудительноеРаспределение Тогда //Сняли флаг

		  Если РезультатыРаспределения.Количество() > 0 Тогда

			ОписаниеОповещения = Новый
ОписаниеОповещения("СнятиеФлагаПринудительногоРаспределенияЗавершение", ЭтаФорма, Новый Структура);
			ПоказатьВопрос(ОписаниеОповещения, "Табличная часть ""Итоги распределения"" будет очищена.
Продолжить?", РежимДиалогаВопрос.ДаНет);

		  Иначе

			СнятиеФлагаПринудительногоРаспределенияЗавершение(КодВозвратаДиалога.Да, Новый Структура);

		  КонецЕсли;

		КонецЕсли;

КонецПроцедуры

&НаКлиенте
Процедура СнятиеФлагаПринудительногоРаспределенияЗавершение(Результат, ДополнительныеПараметры) Экспорт

		Если Результат = КодВозвратаДиалога.Да Тогда

		  ЗаполнитьДанныеИзРегистров(Истина, Ложь);
		  //ИзмененоРаспределениеТоваров = Ложь;
```

```
                Иначе

                  Объект.ПринудительноеРаспределение = Не Объект.ПринудительноеРаспределение;

                КонецЕсли;

КонецПроцедуры

#КонецОбласти

#Область ОбработчикиСобытийЭлементовТаблицыФормыТовары
&НаКлиенте
Процедура ТоварыПриИзменении(Элемент)
                ТоварыОбновитьИтоги();
КонецПроцедуры

&НаКлиенте
Процедура ТоварыВыбор(Элемент, ВыбраннаяСтрока, Поле, СтандартнаяОбработка)

                СтандартнаяОбработка = Ложь;

                ТекущиеДанные = Элемент.ТекущиеДанные;

                Если ТекущиеДанные <> Неопределено Тогда

                  Если Поле.Имя = "ТоварыНоменклатура" Тогда

                        ПоказатьЗначение(, ТекущиеДанные.Номенклатура);

                  ИначеЕсли Поле.Имя = "ТоварыХарактеристика" Тогда

                        ПоказатьЗначение(, ТекущиеДанные.Характеристика);

                  ИначеЕсли Поле.Имя = "ТоварыВариантКомплектации" Тогда
                        СтруктураПараметров = Новый Структура;
                        СтруктураПараметров.Вставить("ПроектПоставки",          Объект.Ссылка);
                        СтруктураПараметров.Вставить("Номенклатура",            ТекущиеДанные.Номенклатура);
                        СтруктураПараметров.Вставить("Характеристика",          ТекущиеДанные.Характеристика);
                        СтруктураПараметров.Вставить("СтрокаНомер",             ТекущиеДанные.СтрокаНомер);
                        //СтруктураПараметров.Вставить("ВариантКомплектации", ТекущиеДанные.ВариантКомплектации);

                        //ОткрытьФорму("Документ.ПроектПоставки.Форма.ФормаВариантаКомплектации", СтруктураПараметров,
ЭтаФорма, ЭтаФорма.УникальныйИдентификатор,,, ОбработчикЗакрытия, РежимОткрытияОкнаФормы.БлокироватьОкноВладельца);
                        ОткрытьФорму("Документ.ПроектПоставки.Форма.ФормаВариантаКомплектации", СтруктураПараметров,
ЭтаФорма, ЭтаФорма.УникальныйИдентификатор,,, , РежимОткрытияОкнаФормы.БлокироватьОкноВладельца);

                  КонецЕсли;

                КонецЕсли;

КонецПроцедуры
#КонецОбласти

#Область ОбработчикиСобытийЭлементовТаблицыФормыРезультатыРаспределения

&НаКлиенте
Процедура РезультатыРаспределенияПередНачаломДобавления(Элемент, Отказ, Копирование, Родитель, Группа, Параметр)

                Отказ = Истина;

КонецПроцедуры

&НаКлиенте
Процедура РезультатыРаспределенияПередУдалением(Элемент, Отказ)

                Отказ = Истина;

КонецПроцедуры

&НаКлиенте
Процедура РезультатыРаспределенияРассчитываемыеПоляПриИзменении(Элемент)

                ТекДанные = Элементы.РезультатыРаспределения.ТекущиеДанные;

                РасчитанныйИтог = 0;
```

```
                Для Каждого ЭлементСписка Из СписокДинамическихКолонок Цикл

                  ИмяКолонки = ЭлементСписка.Представление;
                  РасчитанныйИтог = РасчитанныйИтог + ТекДанные[ИмяКолонки];

                КонецЦикла;

                ТекДанные.Итог = РасчитанныйИтог - ТекДанные.Количество;
                ТекДанные.ОшибкаЗаполнения = ТекДанные.Итог <> 0;

                УстановитьПризнакПринудительногоРаспределения(Истина);

КонецПроцедуры
#КонецОбласти

#Область ОбработчикиСобытийЭлементовТаблицыФормыДокументыПоступлений

&НаКлиенте
Процедура ДокументыПоступленийВыбор(Элемент, ВыбраннаяСтрока, Поле, СтандартнаяОбработка)

                ТекущиеДанные = Объект.ДокументыПоступлений.НайтиПоИдентификатору(ВыбраннаяСтрока);

                Если ТекущиеДанные <> Неопределено Тогда
                  ПоказатьЗначение(,ТекущиеДанные.ПриобретениеТоваровУслуг);
                КонецЕсли;

КонецПроцедуры

#КонецОбласти

#Область СлужебныеПроцедурыИФункции

&НаСервере
Процедура ПриЧтенииСозданииНаСервере()

                УстановитьПредставлениеДокументаУстановкиЦен();

                ЗаполнитьДанныеИзРегистров(Истина, Объект.ПринудительноеРаспределение);

                СкладПодУправлениемWMS = СкладыСервер.СкладПодУправлениемWMS(Объект.Склад);

                СтатусОбработки = УправлениеСтатусамиОбработки.ПолучитьСтатусОбработки(Объект.Ссылка);

                Если Не ЗначениеЗаполнено(СтатусОбработки) Тогда
                  СтатусОбработки = Перечисления.СтатусыОбработкиДокументов.Создан;
                КонецЕсли;

                НастроитьФормуПоСтатусуОбработки();

                ОбщегоНазначенияКлиентСервер.УстановитьЭлементОтбораДинамическогоСписка(НезаявленныеТовары,
"ПроектПоставки", Объект.Ссылка, ВидСравненияКомпоновкиДанных.Равно, , Истина);

КонецПроцедуры

&НаСервере
Процедура УстановитьПредставлениеДокументаУстановкиЦен()

                Если ЗначениеЗаполнено(Объект.ДокументУстановкиЦен) Тогда
                  ПредставлениеДокументаУстановкиЦен = Объект.ДокументУстановкиЦен;
                Иначе
                  ПредставлениеДокументаУстановкиЦен = "Загрузите проект поставки для создания документа установки цен";
                КонецЕсли;

                Элементы.ПредставлениеДокументаУстановкиЦен.ЦветТекста = ЦветаСтиля.ЦветГиперссылки;

КонецПроцедуры
&НаСервере

Процедура ЗаполнитьДанныеИзРегистров(ЗаполнениеПриСозданииФормы, ПоРегиструПринудительногоРаспределения)

                РезультатыРаспределения.Очистить();

                Если СписокДинамическихКолонок.Количество() > 0 Тогда

                  МассивРеквизитов = Новый Массив;
```

```
                Для каждого ЭлСписка Из СписокДинамическихКолонок Цикл

                        Если ЭлСписка.Значение = Перечисления.ТипыХранения.БуферноеХранение
                                или ЭлСписка.Значение = Перечисления.ТипыХранения.ДлительноеХранение Тогда
                                Продолжить;
                        КонецЕсли;

                        Элементы.Удалить(Элементы["РезультатыРаспределения" + ЭлСписка.Представление]);
                        МассивРеквизитов.Добавить("РезультатыРаспределения." + ЭлСписка.Представление);

                КонецЦикла;

                ИзменитьРеквизиты(,МассивРеквизитов);
                СписокДинамическихКолонок.Очистить();

            КонецЕсли;

            //Товары проекта поставки
            ОбщегоНазначенияКлиентСервер.УстановитьЭлементОтбораДинамическогоСписка(Товары, "ПроектПоставки",
Объект.Ссылка, ВидСравненияКомпоновкиДанных.Равно, , Истина);


            ОбщегоНазначенияКлиентСервер.УстановитьЭлементОтбораДинамическогоСписка(НомераКиЗ, "ПроектПоставки",
Объект.Ссылка, ВидСравненияКомпоновкиДанных.Равно, , Истина);

            //Результаты распределения
            ТаблицаРезультатыРаспределения = РеквизитФормыВЗначение("РезультатыРаспределения");

            Запрос = Новый Запрос;

            Запрос.УстановитьПараметр("ПроектПоставки", Объект.Ссылка);

            Запрос.МенеджерВременныхТаблиц = Новый МенеджерВременныхТаблиц;

            Если ПоРегиструПринудительногоРаспределения Тогда

                ТекстСозданияТаблицыРаспределенныхТоваров =
                "ВЫБРАТЬ
                |       ПринудительноеРаспределение.СтрокаНомерДанных КАК СтрокаНомерДанных,
                |       ПринудительноеРаспределение.Номенклатура КАК Номенклатура,
                |       ПринудительноеРаспределение.Характеристика КАК Характеристика,
                |       ПринудительноеРаспределение.Назначение КАК Назначение,
                |       ПринудительноеРаспределение.Количество КАК Количество
                |ПОМЕСТИТЬ РаспределениеТоваров
                |ИЗ
                |       РегистрСведений.ПринудительноеРаспределение КАК ПринудительноеРаспределение
                |ГДЕ
                |       ПринудительноеРаспределение.ПроектПоставки = &ПроектПоставки
                |
                |ИНДЕКСИРОВАТЬ ПО
                |       СтрокаНомерДанных,
                |       Номенклатура,
                |       Характеристика";

            Иначе

                ТекстСозданияТаблицыРаспределенныхТоваров =
                "ВЫБРАТЬ
                |       РаспределениеТоваров.СтрокаДанных КАК СтрокаНомерДанных,
                |       РаспределениеТоваров.Номенклатура КАК Номенклатура,
                |       РаспределениеТоваров.Характеристика КАК Характеристика,
                |       РаспределениеТоваров.Назначение КАК Назначение,
                |       РаспределениеТоваров.Количество КАК Количество
                |ПОМЕСТИТЬ РаспределениеТоваров
                |ИЗ
                |       РегистрНакопления.РаспределениеТоваров КАК РаспределениеТоваров
                |ГДЕ
                |       РаспределениеТоваров.Регистратор = &ПроектПоставки
                |       И РаспределениеТоваров.Количество >= 0
                |
                |ИНДЕКСИРОВАТЬ ПО
                |       СтрокаНомерДанных,
                |       Номенклатура,
                |       Характеристика";

            КонецЕсли;
```

```
Запрос.Текст = ТекстСозданияТаблицыРаспределенныхТоваров;
Запрос.Выполнить();

Запрос.Текст =

"ВЫБРАТЬ
| ТоварыПроектовПоставки.СтрокаНомер КАК СтрокаНомер,
| ТоварыПроектовПоставки.Номенклатура КАК Номенклатура,
| ТоварыПроектовПоставки.Характеристика КАК Характеристика,
| ТоварыПроектовПоставки.ПроцентНаДХ КАК ПроцентНаДХ,

| ТоварыПроектовПоставки.ДатаВыдачиСДХ КАК ДатаВыдачи,

| ТоварыПроектовПоставки.Количество КАК Количество,
| СпрНоменклатура.Артикул КАК Артикул,
| СпрНоменклатура.Наименование КАК Наименование,
| СпрНоменклатура.АртикулИсходный КАК АртикулИсходный,
| СпрНоменклатура.Цвет КАК Цвет,
| СпрНоменклатура.Размер КАК Размер,
| СпрНоменклатура.ТоварнаяКатегория КАК ТоварнаяКатегория,
| СпрНоменклатура.КоллекцияНоменклатуры КАК Сезон
|ПОМЕСТИТЬ ВТ_ТоварыПроектовПоставки
|ИЗ
| РегистрСведений.ТоварыПроектовПоставки КАК ТоварыПроектовПоставки
|       ВНУТРЕННЕЕ СОЕДИНЕНИЕ Справочник.Номенклатура КАК СпрНоменклатура
|       ПО ТоварыПроектовПоставки.Номенклатура = СпрНоменклатура.Ссылка
|ГДЕ
| ТоварыПроектовПоставки.ПроектПоставки = &ПроектПоставки
| И НЕ СпрНоменклатура.ВидНоменклатуры.ИспользоватьХарактеристики
|
|ОБЪЕДИНИТЬ ВСЕ
|
|ВЫБРАТЬ
| ТоварыПроектовПоставки.СтрокаНомер,
| ТоварыПроектовПоставки.Номенклатура,
| ТоварыПроектовПоставки.Характеристика,
| ТоварыПроектовПоставки.ПроцентНаДХ,
| ТоварыПроектовПоставки.ДатаВыдачиСДХ КАК ДатаВыдачи,
| ТоварыПроектовПоставки.Количество,
| СпрНоменклатура.Артикул,
| СпрНоменклатура.Наименование,
| СпрНоменклатура.АртикулИсходный,
| ЕСТЬNULL(ХарактеристикаЦвета.Значение, ЗНАЧЕНИЕ(Справочник.Цвета.ПустаяСсылка)),
| ЕСТЬNULL(ХарактеристикаРазмера.Значение, ЗНАЧЕНИЕ(Справочник.ор_Размеры.ПустаяСсылка)),
| СпрНоменклатура.ТоварнаяКатегория,
| ЕСТЬNULL(ХарактеристикаСезона.Значение, ЗНАЧЕНИЕ(Справочник.КоллекцииНоменклатуры.ПустаяСсылка))
|ИЗ
| РегистрСведений.ТоварыПроектовПоставки КАК ТоварыПроектовПоставки
|       ВНУТРЕННЕЕ СОЕДИНЕНИЕ Справочник.Номенклатура КАК СпрНоменклатура
|       ПО ТоварыПроектовПоставки.Номенклатура = СпрНоменклатура.Ссылка
|       ЛЕВОЕ СОЕДИНЕНИЕ Справочник.ХарактеристикиНоменклатуры.ДополнительныеРеквизиты КАК
ХарактеристикаЦвета
|       ПО ТоварыПроектовПоставки.Характеристика = ХарактеристикаЦвета.Ссылка
|           И (ХарактеристикаЦвета.Свойство =
ЗНАЧЕНИЕ(ПланВидовХарактеристик.ДополнительныеРеквизитыИСведения.Цвет))
|       ЛЕВОЕ СОЕДИНЕНИЕ Справочник.ХарактеристикиНоменклатуры.ДополнительныеРеквизиты КАК
ХарактеристикаРазмера
|       ПО ТоварыПроектовПоставки.Характеристика = ХарактеристикаРазмера.Ссылка
|           И (ХарактеристикаРазмера.Свойство =
ЗНАЧЕНИЕ(ПланВидовХарактеристик.ДополнительныеРеквизитыИСведения.Размер))
|       ЛЕВОЕ СОЕДИНЕНИЕ Справочник.ХарактеристикиНоменклатуры.ДополнительныеРеквизиты КАК
ХарактеристикаСезона
|       ПО ТоварыПроектовПоставки.Характеристика = ХарактеристикаСезона.Ссылка
|           И (ХарактеристикаСезона.Свойство =
ЗНАЧЕНИЕ(ПланВидовХарактеристик.ДополнительныеРеквизитыИСведения.Сезон))
|ГДЕ
| ТоварыПроектовПоставки.ПроектПоставки = &ПроектПоставки
| И СпрНоменклатура.ВидНоменклатуры.ИспользоватьХарактеристики
|
|ИНДЕКСИРОВАТЬ ПО
| СтрокаНомер,
| Номенклатура,
| Характеристика
|;
|
|////////////////////////////////////////////////////////////////////////////////
```

```
|ВЫБРАТЬ
| ТоварыПроектовПоставки.СтрокаНомер КАК СтрокаДанных,
| ТоварыПроектовПоставки.Номенклатура КАК Номенклатура,
| ТоварыПроектовПоставки.Характеристика КАК Характеристика,
| ТоварыПроектовПоставки.ПроцентНаДХ КАК ПроцентНаДХ,
| ТоварыПроектовПоставки.ДатаВыдачи КАК ДатаВыдачи,
| ТоварыПроектовПоставки.Количество КАК КоличествоДанные,
| ТоварыПроектовПоставки.Артикул КАК Артикул,
| ТоварыПроектовПоставки.Наименование КАК Наименование,
| ТоварыПроектовПоставки.АртикулИсходный КАК АртикулИсходный,
| ТоварыПроектовПоставки.Цвет КАК Цвет,
| ТоварыПроектовПоставки.Размер КАК Размер,
| ТоварыПроектовПоставки.ТоварнаяКатегория КАК ТоварнаяКатегория,
| ТоварыПроектовПоставки.Сезон КАК Сезон,
| ВЫБОР
|           КОГДА РаспределениеТоваров.Назначение ССЫЛКА Справочник.Склады
|                   ТОГДА ВЫРАЗИТЬ(РаспределениеТоваров.Назначение КАК Справочник.Склады).НомерМагазина
|           КОГДА РаспределениеТоваров.Назначение = ЗНАЧЕНИЕ(Перечисление.ТипыХранения.БуферноеХранение)
|                   ТОГДА ""БХ""
|           КОГДА РаспределениеТоваров.Назначение = ЗНАЧЕНИЕ(Перечисление.ТипыХранения.ДлительноеХранение)
|                   ТОГДА ""ДХ""
|           ИНАЧЕ """"
| КОНЕЦ КАК ИмяКолонкиНазначение,
| РаспределениеТоваров.Назначение,
| ЕСТЬNULL(РаспределениеТоваров.Количество, 0) КАК Количество
|ПОМЕСТИТЬ ВТ_Товары
|ИЗ

| ВТ_ТоварыПроектовПоставки КАК ТоварыПроектовПоставки

КАК РаспределениеТоваров
РаспределениеТоваров.СтрокаНомерДанных

//|                   ЛЕВОЕ СОЕДИНЕНИЕ РегистрНакопления.РаспределениеТоваров КАК РаспределениеТоваров
//|                   ПО РаспределениеТоваров.Регистратор = &ПроектПоставки
//|                           И ТоварыПроектовПоставки.СтрокаНомер = РаспределениеТоваров.СтрокаДанных

//|                   ЛЕВОЕ СОЕДИНЕНИЕ РегистрСведений.ПринудительноеРаспределение КАК РаспределениеТоваров
//|                   ПО РаспределениеТоваров.ПроектПоставки = &ПроектПоставки И
|           ЛЕВОЕ СОЕДИНЕНИЕ РаспределениеТоваров КАК РаспределениеТоваров ПО

|                   ТоварыПроектовПоставки.СтрокаНомер = РаспределениеТоваров.СтрокаНомерДанных

|                   И ТоварыПроектовПоставки.Номенклатура = РаспределениеТоваров.Номенклатура
|                   И ТоварыПроектовПоставки.Характеристика = РаспределениеТоваров.Характеристика

//|ГДЕ
//|           ТоварыПроектовПоставки.ПроектПоставки = &ПроектПоставки
|;
|
|////////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ РАЗЛИЧНЫЕ
| Склады.Ссылка КАК Назначение,
| Склады.НомерМагазина КАК ИмяКолонкиНазначение
|ИЗ
| Справочник.Склады КАК Склады
|ГДЕ
| Склады.Ссылка В(&Магазины)
| И Склады.НомерМагазина <> """"
|
|ОБЪЕДИНИТЬ
|
|ВЫБРАТЬ РАЗЛИЧНЫЕ
| Склады.Ссылка,
| Склады.НомерМагазина
|ИЗ
| Справочник.Склады КАК Склады
|ГДЕ
| Склады.Ссылка В
|                   (ВЫБРАТЬ РАЗЛИЧНЫЕ
|                           ВТ_Товары.Назначение
|                   ИЗ
|                           ВТ_Товары КАК ВТ_Товары)
| И Склады.НомерМагазина <> """"
|
|УПОРЯДОЧИТЬ ПО
```

```
| ИмяКолонкиНазначение
|;
|
|////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
| ВТ_Товары.СтрокаДанных КАК СтрокаДанных,
| ВТ_Товары.Номенклатура КАК Номенклатура,
| ВТ_Товары.Характеристика КАК Характеристика,
| ВТ_Товары.ПроцентНаДХ КАК ПроцентНаДХ,
| ВТ_Товары.ДатаВыдачи КАК ДатаВыдачи,

| ВТ_Товары.Артикул КАК Артикул,
| ВТ_Товары.Наименование КАК Наименование,
| ВТ_Товары.АртикулИсходный КАК АртикулИсходный,
| ВТ_Товары.Цвет КАК Цвет,
| ВТ_Товары.Размер КАК Размер,
| ВТ_Товары.ТоварнаяКатегория КАК ТоварнаяКатегория,
| ВТ_Товары.Сезон КАК Сезон,
| ВТ_Товары.ИмяКолонкиНазначение КАК ИмяКолонкиНазначение,
| ВТ_Товары.Назначение,
| ВТ_Товары.КоличествоДанные КАК КоличествоДанные,
| ВТ_Товары.Количество КАК Количество
|ИЗ
| ВТ_Товары КАК ВТ_Товары
|
|УПОРЯДОЧИТЬ ПО
| СтрокаДанных
|ИТОГИ
| МАКСИМУМ(СтрокаДанных),
| МАКСИМУМ(Номенклатура),
| МАКСИМУМ(Характеристика),
| МАКСИМУМ(ПроцентНаДХ),
| МАКСИМУМ(ДатаВыдачи),

| МАКСИМУМ(Артикул),
| МАКСИМУМ(Наименование),
| МАКСИМУМ(АртикулИсходный),
| МАКСИМУМ(Цвет),
| МАКСИМУМ(Размер),
| МАКСИМУМ(ТоварнаяКатегория),
| МАКСИМУМ(Сезон),
| МАКСИМУМ(ИмяКолонкиНазначение),
| МАКСИМУМ(КоличествоДанные),
| СУММА(Количество)
|ПО
| ОБЩИЕ,
| СтрокаДанных";

Строки = Объект.СписокМагазинов.НайтиСтроки(Новый Структура("Распределение", Истина));

Запрос.УстановитьПараметр("Магазины", Объект.СписокМагазинов.Выгрузить(Строки,
"Магазин").ВыгрузитьКолонку("Магазин"));
Запрос.УстановитьПараметр("ПроектПоставки", Объект.Ссылка);

МассивРезультатов = Запрос.ВыполнитьПакет();

РезультатМестаХранения = МассивРезультатов[МассивРезультатов.ВГраница() - 1];
РезультатЗапроса = МассивРезультатов[МассивРезультатов.ВГраница()];


//создадим структуру таблицы
МассивИменСлужебныхКолонок = Новый Массив;
МассивИменСлужебныхКолонок.Добавить("Назначение");

СоответствиеИменИЗаголовковПолей = Новый Соответствие; //имя заголовок
СоответствиеИменИЗаголовковПолей.Вставить("ПроцентНаДХ", "% на ДХ");

КолонкиРезультата = РезультатЗапроса.Колонки;
ВыборкаМестаХранения = РезультатМестаХранения.Выбрать();

ЭлементТЗ = Элементы.РезультатыРаспределения; ; //элФорм

ОписаниеКоличества = Новый ОписаниеТипов("Число", Новый КвалификаторыЧисла(15, 0));

МассивРеквизитов = Новый Массив;
Пока ВыборкаМестаХранения.Следующий() Цикл
```

```
                ИмяКолонки = "к" + СтрЗаменить(ВыборкаМестаХранения.ИмяКолонкиНазначение, " ", "");

                ТаблицаРезультатыРаспределения.Колонки.Добавить(ИмяКолонки, ОписаниеКоличества,
ВыборкаМестаХранения.ИмяКолонкиНазначение); //ТабЗнач

                НоваяКолонка = Новый РеквизитФормы(ИмяКолонки, ОписаниеКоличества, "РезультатыРаспределения");//табФорм
                НоваяКолонка.Заголовок = ВыборкаМестаХранения.ИмяКолонкиНазначение;
                МассивРеквизитов.Добавить(НоваяКолонка);//табФорм

                СписокДинамическихКолонок.Добавить(ВыборкаМестаХранения.Назначение, ИмяКолонки);
              КонецЦикла;

              СписокДинамическихКолонок.Добавить(Перечисления.ТипыХранения.БуферноеХранение, "кБХ");
              СписокДинамическихКолонок.Добавить(Перечисления.ТипыХранения.ДлительноеХранение, "кДХ");

              ИзменитьРеквизиты(МассивРеквизитов);

              ВыборкаМестаХранения.Сбросить();
              Пока ВыборкаМестаХранения.Следующий() Цикл

                ИмяКолонки = "к" + СтрЗаменить(ВыборкаМестаХранения.ИмяКолонкиНазначение, " ", "");

                НовыйЭлементФормы = Элементы.Вставить("РезультатыРаспределения"+ИмяКолонки, Тип("ПолеФормы"),
ЭлементТЗ, Элементы.РезультатыРаспределенияКБХ); //элФорм
                НовыйЭлементФормы.Вид = ВидПоляФормы.ПолеВвода; //элФорм
                НовыйЭлементФормы.Заголовок = ВыборкаМестаХранения.ИмяКолонкиНазначение;

                НовыйЭлементФормы.ПутьКДанным = "РезультатыРаспределения." + ИмяКолонки; //элФорм
                НовыйЭлементФормы.Ширина = 6;

                НовыйЭлементФормы.УстановитьДействие("ПриИзменении",
"РезультатыРаспределенияРассчитываемыеПоляПриИзменении");

              КонецЦикла;

              ВыборкаОбщиеИтоги = РезультатЗапроса.Выбрать(ОбходРезультатаЗапроса.ПоГруппировкам);
              ВыборкаОбщиеИтоги.Следующий();

              Если ЗаполнениеПриСозданииФормы и ВыборкаОбщиеИтоги.Количество = 0 Тогда
                Возврат;
              КонецЕсли;

              ВыборкаСтрокаДанных = ВыборкаОбщиеИтоги.Выбрать(ОбходРезультатаЗапроса.ПоГруппировкам);

              Пока ВыборкаСтрокаДанных.Следующий() Цикл

                // Добавляем строку
                СтрокаРезультатовРаспределения = ТаблицаРезультатыРаспределения.Добавить();
//РезультатыРаспределения.Добавить();
                ЗаполнитьЗначенияСвойств(СтрокаРезультатовРаспределения, ВыборкаСтрокаДанных);

                СтрокаРезультатовРаспределения.Бренд = СтрокаРезультатовРаспределения.Номенклатура.Марка;

                СтрокаРезультатовРаспределения.Количество = ВыборкаСтрокаДанных.КоличествоДанные;

                Если ВыборкаСтрокаДанных.ИмяКолонкиНазначение = "" Тогда
                        //СтрокаРезультатовРаспределения.Итог = -ВыборкаСтрокаНомерДанных.КоличествоПлан;
                        СтрокаРезультатовРаспределения.Итог = -СтрокаРезультатовРаспределения.Количество;
                        СтрокаРезультатовРаспределения.ОшибкаЗаполнения = Истина;
                        Продолжить;
                КонецЕсли;

                План = СтрокаРезультатовРаспределения.Количество;
                Распределено = 0;

                Если ЗаполнениеПриСозданииФормы Тогда

                        ВыборкаДетальныеЗаписи = ВыборкаСтрокаДанных.Выбрать();

                        Пока ВыборкаДетальныеЗаписи.Следующий() Цикл

                                // заполняем колонки назначений (+ итог). не добавлять каждый раз, а искать есть ли такая колонка уже
(как вариант, хранить в соответсвии значение Назначения и индекс колонки)
                                //при добавлении колонки она должна быть доступна для редактирования
```

```
                                    Если ПустаяСтрока(ВыборкаДетальныеЗаписи.ИмяКолонкиНазначение) Тогда
                                        ВызватьИсключение "Не указан номер магазина в карточке склада";
                                    КонецЕсли;

                                    //Поиск колонки
                                    ИмяКолонки = "к" + СтрЗаменить(ВыборкаДетальныеЗаписи.ИмяКолонкиНазначение, " ", "");

                                    //СтрокаРезультатовРаспределения[ИмяКолонки] = ВыборкаДетальныеЗаписи.Количество;
                                    СтрокаРезультатовРаспределения[ИмяКолонки] = СтрокаРезультатовРаспределения[ИмяКолонки] +
ВыборкаДетальныеЗаписи.Количество;
                                    Распределено = Распределено + ВыборкаДетальныеЗаписи.Количество;
                                    //итоговую посчитать
                                КонецЦикла;

                            КонецЕсли;

                            СтрокаРезультатовРаспределения.Итог = Распределено - План;
                            СтрокаРезультатовРаспределения.ОшибкаЗаполнения = СтрокаРезультатовРаспределения.Итог <> 0;

                        КонецЦикла;

                        ЗначениеВРеквизитФормы(ТаблицаРезультатыРаспределения, "РезультатыРаспределения");

КонецПроцедуры

&НаКлиенте
Процедура ПроверитьЗаписьДокумента(ОбработкаПродолжения)

                        Если Модифицированность ИЛИ Объект.Ссылка.Пустая() Тогда
                            ТекстВопроса = НСтр("ru = 'Для продолжения операции необходимо записать документ.
                            |Записать?'");
                            ДопПараметры = Новый Структура("ИмяПроцедуры", ОбработкаПродолжения);
                            ОписаниеОповещения = Новый ОписаниеОповещения("ЗакончитьПроверкуЗаписиДокумента", ЭтотОбъект,
ДопПараметры);
                            ПоказатьВопрос(ОписаниеОповещения, ТекстВопроса, РежимДиалогаВопрос.ДаНет, , КодВозвратаДиалога.Да);
                        Иначе
                            ВыполнитьОбработкуОповещения(ОбработкаПродолжения, КодВозвратаДиалога.Да);
                        КонецЕсли;

КонецПроцедуры

&НаКлиенте
Процедура ЗакончитьПроверкуЗаписиДокумента(Результат, ДополнительныеПараметры) Экспорт

                        Если Результат = КодВозвратаДиалога.Да Тогда

                            Если ЗаписатьНовыйДокумент() Тогда

                                ОбработкаПродолжения = "";
                                Если ТипЗнч(ДополнительныеПараметры) = Тип("Структура")
                                    И ДополнительныеПараметры.Свойство("ИмяПроцедуры", ОбработкаПродолжения)
                                    И ТипЗнч(ОбработкаПродолжения) = Тип("ОписаниеОповещения") Тогда

                                    ВыполнитьОбработкуОповещения(ОбработкаПродолжения, Результат);


                                КонецЕсли;

                            КонецЕсли;

                        КонецЕсли;

КонецПроцедуры

&НаСервере
Функция ЗаписатьНовыйДокумент()

                        Если ПроверитьЗаполнение() Тогда
                            Записать();
                            возврат Истина;
                        КонецЕсли;

                        возврат Ложь;

КонецФункции
```

```
//&НаКлиенте
//Процедура ПередЗаписьюПредложитьУстановитьПринудительноеРаспределениеЗавершение(Результат, ДополнительныеПараметры)
Экспорт
//
//		ОбщегоНазначенияУТКлиент.ЗаписатьОбъектПриНеобходимости(ЭтотОбъект,
ДополнительныеПараметры.ПараметрыЗаписи,,Истина);
//
//КонецПроцедуры

//&НаКлиенте
//Процедура ОбработатьЗаписьОбъекта()
//
//		ОбщегоНазначенияУТКлиент.ОбработатьЗаписьОбъектаВФорме(ЭтотОбъект, ПараметрыДляЗаписи);
//
//КонецПроцедуры

#Область РаспределениеПоставки
&НаКлиенте
Процедура ЗаполнитьИтогиРаспределенияЗавершение(Результат, Параметры) Экспорт

		Если Результат = КодВозвратаДиалога.Да Тогда

		 РезультатыРаспределения.Очистить();

		 //ЗаполнитьДанныеИзРегистров();
		 ЗаполнитьДанныеИзРегистров(Ложь, Объект.ПринудительноеРаспределение);

		 //ИзмененоРаспределениеТоваров = Ложь;

		КонецЕсли;

КонецПроцедуры

&НаКлиенте
Процедура РаспределениеТоваровПослеЗаписиДокумента(Результат, ДополнительныеПараметры) Экспорт

		Если Результат = КодВозвратаДиалога.Да Тогда

		 ОбработкаПродолжения = ДополнительныеПараметры.ОбработкаПродолженияРаспределения;

		 Если РаспределениеУжеВыполнялось() Тогда

			ТекстВопроса = НСтр("ru = 'По документу уже выполнялось распределение. Вы хотите выполнить его
повторно?'");
			ПоказатьВопрос(ОбработкаПродолжения, ТекстВопроса, РежимДиалогаВопрос.ДаНет, ,
КодВозвратаДиалога.Да);

		 Иначе

			ВыполнитьОбработкуОповещения(ОбработкаПродолжения, КодВозвратаДиалога.Да);

		 КонецЕсли;

		КонецЕсли;

КонецПроцедуры

&НаКлиенте
Процедура РаспределитьПродолжение(Результат, ДополнительныеПараметры) Экспорт

		Если Результат = КодВозвратаДиалога.Да Тогда

		 ДлительнаяОперация = Неопределено;

		 УстановитьПризнакПринудительногоРаспределения(Ложь);

		 РаспределитьНаСервере(ДлительнаяОперация);

		 Если ДлительнаяОперация <> Неопределено Тогда

			ПараметрыОжидания = ДлительныеОперацииКлиент.ПараметрыОжидания(ЭтотОбъект);
			ПараметрыОжидания.Интервал = 2;

			ОповещениеОЗавершении = Новый ОписаниеОповещения("РаспределитьЗавершение", ЭтотОбъект);
			ДлительныеОперацииКлиент.ОжидатьЗавершение(ДлительнаяОперация, ОповещениеОЗавершении,
ПараметрыОжидания);
```

```
                КонецЕсли;

            КонецЕсли;

КонецПроцедуры

&НаСервере
Функция РаспределитьНаСервере(ДлительнаяОперация)

            АдресВХранилище = ПоместитьВоВременноеХранилище(Неопределено, УникальныйИдентификатор);

            ПараметрыЗадания = Новый Структура;
            ПараметрыЗадания.Вставить("Документ", Объект.Ссылка);
            ПараметрыЗадания.Вставить("АлгоритмРаспределения", Объект.АлгоритмРаспределения);

            ПараметрыВыполнения =
ДлительныеОперации.ПараметрыВыполненияВФоне(ЭтотОбъект.УникальныйИдентификатор);
            ПараметрыВыполнения.НаименованиеФоновогоЗадания = "Распределение проекта поставки";
            ПараметрыВыполнения.АдресРезультата = АдресВХранилище;
            ПараметрыВыполнения.ЗапуститьВФоне = Истина;

            //ДлительнаяОперация =
ДлительныеОперации.ВыполнитьВФоне("МеханизмыРаспределения.ВыполнитьАлгоритмРаспределения", ПараметрыЗадания,
ПараметрыВыполнения);
            ДлительнаяОперация = ДлительныеОперации.ВыполнитьВФоне("РаспределениеТоваров.РаспределитьВФоне",
ПараметрыЗадания, ПараметрыВыполнения);

КонецФункции

&НаКлиенте
Процедура РаспределитьЗавершение(Результат, ДополнительныеПараметры) Экспорт

            Если ТипЗнч(Результат) <> Тип("Структура") Тогда
              Возврат;
            КонецЕсли;

            Если Результат.Статус = "Выполняется" Тогда
              Возврат;
            КонецЕсли;

            РезультатВыполнения = ПолучитьИзВременногоХранилища(Результат.АдресРезультата);
            УдалитьИзВременногоХранилища(Результат.АдресРезультата);

            ОтключитьОбработчикОжидания("РаспределитьЗавершение");

            Если Результат.Статус = "Ошибка" Тогда

              ОбщегоНазначенияКлиентСервер.СообщитьПользователю("При выполнении распределения произошла ошибка: "
              + Результат.КраткоеПредставлениеОшибки + Символы.ПС
              + "Подробное описание ошибки: " + Результат.ПодробноеПредставлениеОшибки);
              // Если ошибка, то отменяем проведение

              ПараметрыЗаписи = Новый Структура("РежимЗаписи", РежимЗаписиДокумента.ОтменаПроведения);
              ЭтаФорма.Записать(ПараметрыЗаписи);
            Иначе

              РезультатыРаспределения.Очистить();

              ЗаполнитьДанныеИзРегистров(Истина, Ложь);

              ОбщегоНазначенияКлиентСервер.СообщитьПользователю("Распределение завершено");

            КонецЕсли;

КонецПроцедуры

&НаКлиенте
Процедура ОтменитьРаспределениеЗавершение(РезультатВопроса, ДополнительныеПараметры) Экспорт

            Если РезультатВопроса = КодВозвратаДиалога.Да Тогда
              ОтменитьРаспределениеНаСервере();
              ОбщегоНазначенияКлиентСервер.СообщитьПользователю("Распределение было успешно отменено.");
            КонецЕсли;

КонецПроцедуры
```

```
&НаСервере
Процедура ОтменитьРаспределениеНаСервере()

		Документы.ПроектПоставки.УдалитьДвиженияИзРегистровРаспределения(Объект.Ссылка);

		Если Объект.ПринудительноеРаспределение Тогда
		  УстановитьПризнакПринудительногоРаспределения(Ложь);
		  ЭтаФорма.Записать();
		КонецЕсли;
		РезультатыРаспределения.Очистить();

		ЗаполнитьДанныеИзРегистров(Истина, Объект.ПринудительноеРаспределение);

КонецПроцедуры

&НаСервере
Функция РаспределениеУжеВыполнялось()

		Запрос = Новый Запрос;
		Запрос.Текст =
		"ВЫБРАТЬ ПЕРВЫЕ 1

		| РаспределениеТоваров.Регистратор
		|ИЗ
		| РегистрНакопления.РаспределениеТоваров КАК РаспределениеТоваров
		|ГДЕ
		| РаспределениеТоваров.Регистратор = &ПроектПоставки";

		Запрос.УстановитьПараметр("ПроектПоставки", Объект.Ссылка);

		РезультатЗапроса = Запрос.Выполнить();

		Возврат НЕ РезультатЗапроса.Пустой();

КонецФункции

&НаСервере
Процедура УстановитьПризнакПринудительногоРаспределения(НовоеЗначение)

		Объект.ПринудительноеРаспределение = НовоеЗначение;

КонецПроцедуры
#КонецОбласти

#Область ПриИзмененииРеквизитов

&НаСервере
Процедура ПриИзмененииПартнераСервер()

		ПартнерыИКонтрагенты.ЗаполнитьКонтрагентаПартнераПоУмолчанию(Объект.Партнер, Объект.Контрагент);

КонецПроцедуры

&НаСервере
Процедура СкладПриИзмененииНаСервере()

		СкладПодУправлениемWMS = СкладыСервер.СкладПодУправлениемWMS(Объект.Склад);

		СтатусОбработки = Перечисления.СтатусыОбработкиДокументов.Создан;

		НастроитьФормуПоСтатусуОбработки();

КонецПроцедуры

#КонецОбласти

#Область ВыгрузкаЗагрузкаExcel
&НаКлиенте
Процедура ЗагрузитьРезультатыРаспределенияИзExcelПродолжение(ВыбранныеФайлы, ДополнительныеПараметры) Экспорт

		Если ВыбранныеФайлы = Неопределено Тогда
		  Возврат;
		КонецЕсли;

		Попытка
		  ДвоичныеДанные = Новый ДвоичныеДанные(ВыбранныеФайлы[0]);
```

```
		Исключение
		  Сообщить("Не удалось получить доступ к файлу. Проверьте, что он закрыт.");
		  Возврат;
		КонецПопытки;

		ЗагрузитьРезультатыРаспределенияНаСервере(ПоместитьВоВременноеХранилище(ДвоичныеДанные));

		//ИзмененоРаспределениеТоваров = Истина;

		УстановитьПризнакПринудительногоРаспределения(Истина);

КонецПроцедуры

&НаСервере
Процедура ЗагрузитьРезультатыРаспределенияНаСервере(АдресХранилища)

		//ДвоичныеДанные = ПолучитьИзВременногоХранилища(Адрес);
		//Путь = ПолучитьИмяВременногоФайла("xlsx");
		//ДвоичныеДанные.Записать(Путь);
		//ТабДок = Новый ТабличныйДокумент;
		//ТабДок.Прочитать(Путь);
		//УдалитьФайлы(Путь);
		//
		//Сч = 1;
		//
		//Для каждого Элемент Из Элементы.РезультатыРаспределения.ПодчиненныеЭлементы Цикл
		//
		//	Если ТипЗнч(Элемент) = Тип("ПолеФормы") Тогда
		//
		//		Имя = СтрЗаменить(Элемент.Имя, "РезультатыРаспределения", "");
		//		Заголовок = ?(Элемент.Заголовок = "", Имя, Элемент.Заголовок);
		//
		//		Если ТабДок.Область(1, Сч).Текст <> Заголовок Тогда
		//
		//			Сообщить("Формат полей файла не соответствует формату табличной части документа.
		//			|Колонка в файле - """ + ТабДок.Область(1, Сч).Текст + """, колонка в документе - """ +
Заголовок + """");
		//			Возврат;
		//
		//		КонецЕсли;
		//
		//		Сч = Сч+1
		//
		//	КонецЕсли;
		//
		//КонецЦикла;
		//
		//РезультатыРаспределения.Очистить();
		//
		//ВсегоКолонок = Сч;
		//
		//Кэши = Новый Структура("РодительНоменклатура, Размер, Пол, Цвет, ТоварнаяКатегория, Бренд, ТипБренда,
СезонПоставщика, Сезон, СтранаПроизводителя",
		//		Новый Соответствие, Новый Соответствие, Новый Соответствие, Новый
Соответствие, Новый Соответствие, Новый Соответствие, Новый Соответствие,
		//		Новый Соответствие, Новый Соответствие, Новый Соответствие
		//		);
		//
		//Для НомерСтроки = 2 По ТабДок.ВысотаТаблицы Цикл
		//
		//	Если ПустаяСтрока(ТабДок.Область(НомерСтроки, 1).Текст) Тогда
		//		Возврат;
		//	КонецЕсли;
		//
		//	НовСтр = РезультатыРаспределения.Добавить();
		//	НовСтр.СтрокаДанных = СтроковыеФункцииКлиентСервер.СтрокаВЧисло(ТабДок.Область(НомерСтроки,
1).Текст);
		//	НовСтр.Номенклатура = Справочники.Номенклатура.НайтиПоРеквизиту("Артикул",
ТабДок.Область(НомерСтроки, 4).Текст);
		//	НовСтр.Характеристика = НайтиХарактеристику(ТабДок.Область(НомерСтроки, 3).Текст,
НовСтр.Номенклатура);
		//	НовСтр.Артикул = ТабДок.Область(НомерСтроки, 4).Текст;
		//	НовСтр.АртикулИсходный = ТабДок.Область(НомерСтроки, 5).Текст;
		//	НовСтр.Цвет = НайтиЭлементСправочника(ТабДок.Область(НомерСтроки, 6).Текст, "Цвета", Кэши.Цвет);
		//	НовСтр.Размер = НайтиЭлементСправочника(ТабДок.Область(НомерСтроки, 7).Текст, "ор_Размеры",
Кэши.Размер);
```

```
		//	НовСтр.ТоварнаяКатегория = НайтиЭлементСправочника(ТабДок.Область(НомерСтроки, 8).Текст,
"ТоварныеКатегории", Кэши.ТоварнаяКатегория);
		//	НовСтр.Сезон = НайтиЭлементСправочника(ТабДок.Область(НомерСтроки, 9).Текст, "ор_Сезоны",
Кэши.Сезон);
		//	НовСтр.ПроцентНаДХ = СтроковыеФункцииКлиентСервер.СтрокаВЧисло(ТабДок.Область(НомерСтроки,
10).Текст);
		//	НовСтр.Количество = СтроковыеФункцииКлиентСервер.СтрокаВЧисло(ТабДок.Область(НомерСтроки,
11).Текст);
		//	//НовСтр.КоличествоПлан = СтроковыеФункцииКлиентСервер.СтрокаВЧисло(ТабДок.Область(НомерСтроки,
12).Текст);
		//
		//	Сч = ВсегоКолонок - СписокДинамическихКолонок.Количество() - 1;
		//	СуммаПоМагазинам = 0;
		//	Для каждого ЭлСписка Из СписокДинамическихКолонок Цикл
		//
		//		НовСтр[ЭлСписка.Представление] =
СтроковыеФункцииКлиентСервер.СтрокаВЧисло(ТабДок.Область(НомерСтроки, Сч).Текст);
		//		НовСтр.Назначение = ЭлСписка.Значение;
		//		СуммаПоМагазинам = СуммаПоМагазинам + НовСтр[ЭлСписка.Представление];
		//		Сч = Сч + 1;
		//
		//	КонецЦикла;
		//
		//	НовСтр.Итог = СтроковыеФункцииКлиентСервер.СтрокаВЧисло(ТабДок.Область(НомерСтроки, Сч).Текст);
		//
		//	Если НовСтр.Номенклатура.Пустая() Тогда
		//
		//		Сообщить("Номенклатура с артикулом " + НовСтр.Артикул + " не найдена");
		//		НовСтр.ОшибкаЗаполнения = Истина;
		//
		//	КонецЕсли;
		//
		//	//Если СуммаПоМагазинам - НовСтр.КоличествоПлан <> НовСтр.Итог Тогда
		//	Если НовСтр.Итог < 0 или СуммаПоМагазинам - НовСтр.Количество <> НовСтр.Итог Тогда
		//
		//		Сообщить("Для строки № " + (НомерСтроки-1) + " не соблюдается правило Итог = СуммаПоМагазинам
+ БХ + ДХ - Количество");
		//		НовСтр.ОшибкаЗаполнения = Истина;
		//
		//	КонецЕсли;
		//
		//КонецЦикла;

		РаспределениеТоваровВызовСервера.ЗагрузитьРезультатыРаспределенияИзТабличногоДокумента(АдресХранилища,
РезультатыРаспределения, СписокДинамическихКолонок);

КонецПроцедуры
		&НаСервере
Процедура ВыгрузитьРезультатыРаспределенияНаСервере(АдресРезультата) Экспорт

		РаспределениеТоваров.ВывестиРезультатыРаспределенияВТабличныйДокумент(АдресРезультата,
РезультатыРаспределения, СписокДинамическихКолонок);

КонецПроцедуры

#КонецОбласти

#Область ЗагрузкаПроектаПоставки

&НаКлиенте
Процедура ЗагрузитьПроектПоставкиПродолжение(Результат, ДополнительныеПараметры) Экспорт

		ОчиститьСообщения();

		ОбработчикЗакрытия = Новый ОписаниеОповещения("ЗагрузитьПроектПоставкиЗавершение", ЭтаФорма);

		////ОткрытьФорму("Документ.ПроектПоставки.Форма.ФормаЗагрузкиПроекта", , ЭтаФорма,
ЭтаФорма.УникальныйИдентификатор, , , ОбработчикЗакрытия, РежимОткрытияОкнаФормы.БлокироватьОкноВладельца);
		//ОткрытьФорму("Документ.ПроектПоставки.Форма.ФормаЗагрузкиПроекта", Новый Структура("Импорт,
ПоступлениеПроверено, ЕстьПоступление", Объект.ИмпортнаяПоставка, ПоступлениеПроверено,
Объект.ДокументыПоступлений.Количество()),
		//
			ЭтаФорма, ЭтаФорма.УникальныйИдентификатор, , , ОбработчикЗакрытия,
РежимОткрытияОкнаФормы.БлокироватьОкноВладельца);

		СтруктураПараметров = Новый Структура;
```

```
		СтруктураПараметров.Вставить("ПроектПоставки"		, Объект.Ссылка);
		СтруктураПараметров.Вставить("ИмпортнаяПоставка"		, Объект.ИмпортнаяПоставка);
		СтруктураПараметров.Вставить("ДокументыПриобретенияСозданы"	, Объект.ДокументыПоступлений.Количество());
		СтруктураПараметров.Вставить("ДокументыПриобретенияПроверены" ,
ДополнительныеПараметры.ДокументыПриобретенияПроверены);

		ОткрытьФорму("Документ.ПроектПоставки.Форма.ФормаЗагрузкиПроекта", СтруктураПараметров, ЭтаФорма,
ЭтаФорма.УникальныйИдентификатор,,, ОбработчикЗакрытия, РежимОткрытияОкнаФормы.БлокироватьОкноВладельца);

КонецПроцедуры

&НаКлиенте
Процедура ЗагрузитьПроектПоставкиЗавершение(РезультатЗакрытия, ДополнительныеПараметры) Экспорт

		Если РезультатЗакрытия = Неопределено Тогда
		  Возврат;
		КонецЕсли;

		РезультатыРаспределения.Очистить();

		ЗагрузитьПроектПоставкиНаСервере(Объект.Ссылка, РезультатЗакрытия);

		ОбщегоНазначенияКлиентСервер.УстановитьЭлементОтбораДинамическогоСписка(Товары, "ПроектПоставки",
Объект.Ссылка);
		Элементы.Товары.Обновить();
		РезультатыРаспределения.Очистить();

		ЭтаФорма.Прочитать();

КонецПроцедуры

&НаСервере
Процедура ЗагрузитьПроектПоставкиНаСервере(ПроектПоставки, РезультатЗагрузкиИзExcel)

		ТаблицаТоваров = ПолучитьИзВременногоХранилища(РезультатЗагрузкиИзExcel.АдресТоваровВХранилище);

		СтруктураПолучаемыхРеквизитов = Новый Структура;
		СтруктураПолучаемыхРеквизитов.Вставить("КодПоставщика"	, "Партнер.Код");
		СтруктураПолучаемыхРеквизитов.Вставить("Поставщик"		, "Партнер");
		СтруктураПолучаемыхРеквизитов.Вставить("ИмпортнаяПоставка"	, "ИмпортнаяПоставка");

		РеквизитыПроекта = ОбщегоНазначения.ЗначенияРеквизитовОбъекта(ПроектПоставки,
СтруктураПолучаемыхРеквизитов);

		ПараметрыОбработки = Новый Структура;
		ПараметрыОбработки.Вставить("ПроектПоставки"	, ПроектПоставки);
		ПараметрыОбработки.Вставить("КодПоставщика"	, СокрЛП(РеквизитыПроекта.КодПоставщика));
		ПараметрыОбработки.Вставить("Поставщик"		, РеквизитыПроекта.Поставщик);
		ПараметрыОбработки.Вставить("ИмпортнаяПоставка"	, РеквизитыПроекта.ИмпортнаяПоставка);

		Ошибки = Документы.ПроектПоставки.ПолучитьСтруктуруОшибокПроектаПоставки();

		//Если РезультатЗагрузкиИзExcel.СоздаватьДокументыПриобретения Тогда
		//		Документы.ПроектПоставки.ОбработатьЗагруженныеТовары(ТаблицаТоваров, Ошибки,
ПараметрыОбработки);
		//КонецЕсли;
		Документы.ПроектПоставки.ОбработатьЗагруженныеТовары(ТаблицаТоваров, Ошибки, ПараметрыОбработки);

		// Зачем дублировать один и тот же код в разных местах?
		ОчиститьРегистрыТабличныхЧастей();

		Если Ошибки.СписокОшибок.Количество() = 0 Тогда

		  ЗаписатьРезультатыЗагрузкиПроектаПоставки(ТаблицаТоваров,
РезультатЗагрузкиИзExcel.СоздаватьДокументыПриобретения, РезультатЗагрузкиИзExcel.СоздаватьДокументыУстановкиЦен);

		Иначе

		  Документы.ПроектПоставки.ВывестиСписокОшибокПриЗагрузкеПроекта(ТаблицаТоваров, Ошибки);

		КонецЕсли;

		ИмяСобытия = Документы.ПроектПоставки.ИмяСобытияЗагрузкиЖурналаРегистрации();

		ЗаписьЖурналаРегистрации(ИмяСобытия, УровеньЖурналаРегистрации.Информация, Объект.Ссылка.Метаданные(),
Объект.Ссылка,
```

```
НСтр("ru = 'Завершение загрузки проекта поставки.'"));

КонецПроцедуры

&НаКлиенте
Процедура СоздатьНоменклатуруПродолжение(Результат, ДополнительныеПараметры) Экспорт

	ОчиститьСообщения();

	ОбработчикЗакрытия = Новый ОписаниеОповещения("СоздатьНоменклатуруЗавершение", ЭтаФорма);

	СтруктураПараметров = Новый Структура;
	СтруктураПараметров.Вставить("ПроектПоставки"       , Объект.Ссылка);
	СтруктураПараметров.Вставить("ИмпортнаяПоставка"    , Объект.ИмпортнаяПоставка);
	СтруктураПараметров.Вставить("Поставщик"            , Объект.Партнер);
	СтруктураПараметров.Вставить("СозданиеНоменклатуры" , Истина);

	ОткрытьФорму("Документ.ПроектПоставки.Форма.ФормаЗагрузкиПроекта", СтруктураПараметров, ЭтаФорма,
ЭтаФорма.УникальныйИдентификатор,,, ОбработчикЗакрытия, РежимОткрытияОкнаФормы.БлокироватьОкноВладельца);

КонецПроцедуры

&НаКлиенте
Процедура СоздатьНоменклатуруЗавершение(РезультатЗакрытия, ДополнительныеПараметры) Экспорт

	Если РезультатЗакрытия = Неопределено Тогда
	  Возврат;
	КонецЕсли;

	СоздатьНоменклатуруНаСервере(Объект.Ссылка, РезультатЗакрытия);

	Элементы.НезаявленныеТовары.Обновить();

КонецПроцедуры

&НаСервере
Процедура СоздатьНоменклатуруНаСервере(ПроектПоставки, РезультатЗагрузкиИзExcel)

	ТаблицаТоваров = ПолучитьИзВременногоХранилища(РезультатЗагрузкиИзExcel.АдресТоваровВХранилище);

	СтруктураПолучаемыхРеквизитов = Новый Структура;
	СтруктураПолучаемыхРеквизитов.Вставить("КодПоставщика"       , "Партнер.Код");
	СтруктураПолучаемыхРеквизитов.Вставить("Поставщик"           , "Партнер");
	СтруктураПолучаемыхРеквизитов.Вставить("ИмпортнаяПоставка"   , "ИмпортнаяПоставка");

	РеквизитыПроекта = ОбщегоНазначения.ЗначенияРеквизитовОбъекта(ПроектПоставки,
СтруктураПолучаемыхРеквизитов);

	ПараметрыОбработки = Новый Структура;
	ПараметрыОбработки.Вставить("ПроектПоставки"       , ПроектПоставки);
	ПараметрыОбработки.Вставить("КодПоставщика"        , СокрЛП(РеквизитыПроекта.КодПоставщика));
	ПараметрыОбработки.Вставить("Поставщик"            , РеквизитыПроекта.Поставщик);
	ПараметрыОбработки.Вставить("ИмпортнаяПоставка"    , РеквизитыПроекта.ИмпортнаяПоставка);

	Ошибки = Документы.ПроектПоставки.ПолучитьСтруктуруОшибокПроектаПоставки();

	Документы.ПроектПоставки.ОбработатьЗагруженныеТовары(ТаблицаТоваров, Ошибки, ПараметрыОбработки);

	Если Ошибки.СписокОшибок.Количество() > 0 Тогда

	  Документы.ПроектПоставки.ВывестиСписокОшибокПриЗагрузкеПроекта(ТаблицаТоваров, Ошибки);

	Иначе

	  НачатьТранзакцию();

	  НаборЗаписей = РегистрыСведений.НезаявленныеТоварыПроектовПоставки.СоздатьНаборЗаписей();
	  НаборЗаписей.Отбор.ПроектПоставки.Установить(Объект.Ссылка);
	  НаборЗаписей.Прочитать();

	  ТаблицаНезаявленныхТоваров = НаборЗаписей.Выгрузить();

	  Отбор = Новый Структура("Номенклатура, Характеристика");
	  Для Каждого СтрокаТовара Из ТаблицаТоваров Цикл

	        Отбор.Номенклатура = СтрокаТовара.Номенклатура;
```

```
                    Отбор.Характеристика = СтрокаТовара.Характеристика;

                    НайденныеСтроки = ТаблицаНезаявленныхТоваров.НайтиСтроки(Отбор);

                    Если НайденныеСтроки.Количество() > 0 Тогда
                        Запись = НайденныеСтроки[0];
                    Иначе
                        Запись = ТаблицаНезаявленныхТоваров.Добавить();
                    КонецЕсли;

                    Запись.ПроектПоставки = Объект.Ссылка;
                    ЗаполнитьЗначенияСвойств(Запись, СтрокаТовара);

                КонецЦикла;

                НаборЗаписей.Загрузить(ТаблицаНезаявленныхТоваров);
                НаборЗаписей.Записать();

                Если РезультатЗагрузкиИзExcel.СоздаватьДокументыУстановкиЦен Тогда

                    Ценообразование.СоздатьУстановкуЦенПроектаПоставки(Объект.Ссылка, ТаблицаТоваров);

                КонецЕсли;

                ЗафиксироватьТранзакцию();

            КонецЕсли;

КонецПроцедуры

&НаСервере
Процедура ЗаписатьРезультатыЗагрузкиПроектаПоставки(ТаблицаТоваров, СоздаватьДокументыПриобретения,
СоздаватьДокументыУстановкиЦен)

            НачатьТранзакцию();

            //Итоги распределения

            // этот же код выполняется в ОчиститьРегистрыТабличныхЧастей()

            //Товары проекта поставки
            НаборЗаписей = РегистрыСведений.ТоварыПроектовПоставки.СоздатьНаборЗаписей();
            НаборЗаписей.Отбор.ПроектПоставки.Установить(Объект.Ссылка);

            Для Каждого СтрокаТовара Из ТаблицаТоваров Цикл

                Запись = НаборЗаписей.Добавить();
                ЗаполнитьЗначенияСвойств(Запись, СтрокаТовара);

                Запись.ПроектПоставки = Объект.Ссылка;
                Запись.СтрокаНомер = СтрокаТовара.НомерСтроки;

            КонецЦикла;

            НаборЗаписей.Записать();

            //Поступления товаров и услуг

            Если СоздаватьДокументыПриобретения Тогда

                СоздатьОбновитьДокументыПриобретения(ТаблицаТоваров);

            КонецЕсли;

            //Установка цен номенклатуры
            Если СоздаватьДокументыУстановкиЦен Тогда

                Ценообразование.СоздатьУстановкуЦенПроектаПоставки(Объект.Ссылка, ТаблицаТоваров);

            КонецЕсли;

            Объект.СуммаДокумента = ТаблицаТоваров.Итог("СуммаЗакупочная");

            Записать();

            ЗафиксироватьТранзакцию();
```

```
КонецПроцедуры

&НаСервере
Процедура _СоздатьОбновитьДокументыПриобретения(ТаблицаТоваров)

		СозданныеДокументыПоступлений = Новый Массив;

		Запрос = Новый Запрос;
		Запрос.Текст =
		"ВЫБРАТЬ РАЗЛИЧНЫЕ
		| ТоварыПроектовПоставки.ПризнакТОРГ12
		|ИЗ
		| РегистрСведений.ТоварыПроектовПоставки КАК ТоварыПроектовПоставки
		|ГДЕ
		| ТоварыПроектовПоставки.ПроектПоставки = &ТекущийДокумент
		|;
		|
		|////////////////////////////////////////////////////////////////////////////////
		|ВЫБРАТЬ
		| ПроектПоставкиДокументыПоступлений.ПриобретениеТоваровУслуг,
		| ПроектПоставкиДокументыПоступлений.ПризнакТОРГ12,
		| ЛОЖЬ КАК Использование
		|ИЗ
		| Документ.ПроектПоставки.ДокументыПоступлений КАК ПроектПоставкиДокументыПоступлений
		|ГДЕ
		| ПроектПоставкиДокументыПоступлений.Ссылка = &ТекущийДокумент";

		Запрос.УстановитьПараметр("ТекущийДокумент", Объект.Ссылка);

		Результат = Запрос.ВыполнитьПакет();

		ПризнакиТОРГ12 = Результат[0].Выгрузить().ВыгрузитьКолонку("ПризнакТОРГ12");
		ТаблицаДокументовПоступлений = Результат[1].Выгрузить();

		Для Каждого ПризнакТОРГ12 Из ПризнакиТОРГ12 Цикл

		  СтрокаДокумента = ТаблицаДокументовПоступлений.Найти(ПризнакТОРГ12, "ПризнакТОРГ12");
		  Если СтрокаДокумента = Неопределено Тогда
		  		ДокументПоступленияОбъект = Документы.ПриобретениеТоваровУслуг.СоздатьДокумент();
		  Иначе
		  		ДокументПоступленияОбъект = СтрокаДокумента.ПриобретениеТоваровУслуг.ПолучитьОбъект();
		  		СтрокаДокумента.Использование = Истина;
		  КонецЕсли;

		  СтруктураЗаполнения = Новый Структура;
		  СтруктураЗаполнения.Вставить("ДокументОснование", Объект.Ссылка);
		  СтруктураЗаполнения.Вставить("ПризнакТОРГ12"    , ПризнакТОРГ12);

		  ДокументПоступленияОбъект.Дата = Объект.ДатаПлановогоПоступления;

		  ДокументПоступленияОбъект.Автор = Пользователи.ТекущийПользователь();

		  ДокументПоступленияОбъект.ЗакупкаПодДеятельность =
Перечисления.ТипыНалогообложенияНДС.ПродажаОблагаетсяНДС;

		  ДокументПоступленияОбъект.Заполнить(СтруктураЗаполнения);

		  ДокументПоступленияОбъект.Записать(РежимЗаписиДокумента.Проведение);

		  Если СтрокаДокумента = Неопределено Тогда

		  		НоваяСтрока = ТаблицаДокументовПоступлений.Добавить();
		  		НоваяСтрока.ПриобретениеТоваровУслуг = ДокументПоступленияОбъект.Ссылка;
		  		НоваяСтрока.ПризнакТОРГ12 = ПризнакТОРГ12;
		  		НоваяСтрока.Использование = Истина;

		  КонецЕсли;

		КонецЦикла;

		Индекс = ТаблицаДокументовПоступлений.Количество() - 1;
		Пока Индекс >= 0 Цикл

		  СтрокаПоступления = ТаблицаДокументовПоступлений[Индекс];
```

```
                Если Не СтрокаПоступления.Использование Тогда

                    ДокументПоступленияОбъект = СтрокаПоступления.ПриобретениеТоваровУслуг.ПолучитьОбъект();
                    ДокументПоступленияОбъект.УстановитьПометкуУдаления(Истина);

                    ТаблицаДокументовПоступлений.Удалить(Индекс);

                КонецЕсли;

                Индекс = Индекс - 1;

            КонецЦикла;

            Объект.ДокументыПоступлений.Загрузить(ТаблицаДокументовПоступлений);

КонецПроцедуры
//
&НаСервере
Процедура СоздатьОбновитьДокументыПриобретения(ТаблицаТоваров=Неопределено)

            ДополнительныеРеквизиты = Новый Структура("ТоварыВПути,КорректировкаТоваровВПути");
            Для Каждого КлючЗначение Из ДополнительныеРеквизиты Цикл
              ИмяРеквизита = КлючЗначение.Ключ;
              СвойствоСсылка = ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.НайтиПоРеквизиту("Имя",
ИмяРеквизита);
              Если НЕ СвойствоСсылка =
ПредопределенноеЗначение("ПланВидовХарактеристик.ДополнительныеРеквизитыИСведения.ПустаяСсылка") Тогда
                  ДополнительныеРеквизиты.Вставить(ИмяРеквизита, СвойствоСсылка);
              КонецЕсли;
            КонецЦикла;

            ИспользоватьТоварыВПути = НЕ Неопределено = ДополнительныеРеквизиты.ТоварыВПути
                                                    И НЕ Неопределено =
ДополнительныеРеквизиты.КорректировкаТоваровВПути;

            ИспользоватьТоварыВПути = Ложь;


            СозданныеДокументыПоступлений = Новый Массив;

            Запрос = Новый Запрос;
            Запрос.Текст =
            "ВЫБРАТЬ РАЗЛИЧНЫЕ
            | ТоварыПроектовПоставки.ПризнакТОРГ12
            |ИЗ
            | РегистрСведений.ТоварыПроектовПоставки КАК ТоварыПроектовПоставки
            |ГДЕ
            | ТоварыПроектовПоставки.ПроектПоставки = &ТекущийДокумент
            |;
            |
            |////////////////////////////////////////////////////////////////////////////////
            |ВЫБРАТЬ
            | ПроектПоставкиДокументыПоступлений.ПриобретениеТоваровУслуг,
            | ПроектПоставкиДокументыПоступлений.ПризнакТОРГ12,
            | ЛОЖЬ КАК Использование
            |ИЗ
            | Документ.ПроектПоставки.ДокументыПоступлений КАК ПроектПоставкиДокументыПоступлений
            |ГДЕ
            | ПроектПоставкиДокументыПоступлений.Ссылка = &ТекущийДокумент";

            Запрос.УстановитьПараметр("ТекущийДокумент", Объект.Ссылка);

            Результат = Запрос.ВыполнитьПакет();

            ПризнакиТОРГ12 = Результат[0].Выгрузить().ВыгрузитьКолонку("ПризнакТОРГ12");
            ТаблицаДокументовПоступлений = Результат[1].Выгрузить();

            Для Каждого ПризнакТОРГ12 Из ПризнакиТОРГ12 Цикл

              СтрокаДокумента = ТаблицаДокументовПоступлений.Найти(ПризнакТОРГ12, "ПризнакТОРГ12");

              ДокументПоступленияСсылка = Неопределено;

              Если ИспользоватьТоварыВПути И Объект.ИмпортнаяПоставка Тогда

                  Если СтрокаДокумента = Неопределено Тогда
```

```
                        // ПТУ не существует
                        //ВызватьИсключение "Не найден документ перехода права собственности";
                        ДокументПоступленияСсылка = СоздатьОбновитьПриобретениеТоваровУслуг(Неопределено,
Объект.Ссылка, ПризнакТОРГ12, Истина);

                Иначе

                        ЭтоТоварыВПути =
УправлениеСвойствами.ЗначениеСвойства(СтрокаДокумента.ПриобретениеТоваровУслуг, ДополнительныеРеквизиты.ТоварыВПути);
                        Если НЕ ЭтоТоварыВПути = ИСТИНА Тогда
                                // Обычная поставка, обновим ПТУ
                                ДокументПоступленияСсылка =
СоздатьОбновитьПриобретениеТоваровУслуг(СтрокаДокумента.ПриобретениеТоваровУслуг, Объект.Ссылка, ПризнакТОРГ12);

                        Иначе
                                // А это уже "Товары в пути"
                                Сообщить("ЭтоТоварыВПути = "+ЭтоТоварыВПути);

                                // Найдем последнюю корректировку
                                Запрос = Новый Запрос;
                                Запрос.УстановитьПараметр("ДокументОснование",
СтрокаДокумента.ПриобретениеТоваровУслуг);
                                Запрос.Текст =
                                "ВЫБРАТЬ ПЕРВЫЕ 1
                                |       КорректировкаПриобретения.Ссылка КАК Ссылка
                                |ИЗ
                                |       Документ.КорректировкаПриобретения КАК КорректировкаПриобретения
                                |ГДЕ
                                |       КорректировкаПриобретения.Проведен
                                |       И КорректировкаПриобретения.ДокументОснование = &ДокументОснование
                                |
                                |УПОРЯДОЧИТЬ ПО
                                |       КорректировкаПриобретения.МоментВремени УБЫВ";
                                Выборка = Запрос.Выполнить().Выбрать();
                                Если Выборка.Следующий() Тогда
                                        КорректировкаСсылка = Выборка.Ссылка;
                                Иначе
                                        КорректировкаСсылка = Неопределено;
                                КонецЕсли;

                                Если НЕ ЗначениеЗаполнено(КорректировкаСсылка) И НачалоМесяца(ТекущаяДата()) =
НачалоМесяца(СтрокаДокумента.ПриобретениеТоваровУслуг.Дата) Тогда
                                        // Изменения в том же месяце, что и первичный документ, и нет корректировок,
                                        // поэтому просто перезаполним существующий документ "Приобретение товаров услуг"
                                        ДокументПоступленияСсылка =
СоздатьОбновитьПриобретениеТоваровУслуг(СтрокаДокумента.ПриобретениеТоваровУслуг, Объект.Ссылка, ПризнакТОРГ12);

                                Иначе

                                        Если ЗначениеЗаполнено(КорректировкаСсылка) Тогда
                                                // Есть корректировка, проверим есть ли возможность использовать ее или
придется создать новую

                                                ЭтоКорректировкаТоваровВПути =
УправлениеСвойствами.ЗначениеСвойства(КорректировкаСсылка, ДополнительныеРеквизиты.КорректировкаТоваровВПути);

                                                Если ЭтоКорректировкаТоваровВПути = Ложь Тогда
                                                        // То есть последняя корректировка к "Товарам в пути" не имеет
никакого отношения, значит с поставкой производились иные манипуляции,
                                                        // например кредит-нота, какие именно - одному богу известно. Поэтому
в этом случае посылаем пользователю сообщение об ошибке и ждем,
                                                        // когда он прибежит жаловаться...
                                                        ВызватьИсключение "Существуют  прочие корректировки
приобретения";
                                                КонецЕсли;

                                                Если НЕ НачалоМесяца(ТекущаяДата()) =
НачалоМесяца(КорректировкаСсылка.Дата) Тогда
                                                        // Последняя корректировка, была в месяце отличном от текущего,
поэтому создаем новую
                                                        КорректировкаСсылка = Неопределено;
                                                КонецЕсли;

                                        КонецЕсли;
```

```
                                        СоздатьОбновитьКорректировкуПриобретения(КорректировкаСсылка,
СтрокаДокумента.ПриобретениеТоваровУслуг, Объект.Ссылка, ПризнакТОРГ12);

                                    КонецЕсли;

                                КонецЕсли;

                                СтрокаДокумента.Использование = Истина;

                            КонецЕсли;

                        Иначе

                            Если СтрокаДокумента = Неопределено Тогда
                                ДокументПоступленияСсылка = СоздатьОбновитьПриобретениеТоваровУслуг(Неопределено,
Объект.Ссылка, ПризнакТОРГ12);
                            Иначе
                                ДокументПоступленияСсылка =
СоздатьОбновитьПриобретениеТоваровУслуг(СтрокаДокумента.ПриобретениеТоваровУслуг, Объект.Ссылка, ПризнакТОРГ12);
                                СтрокаДокумента.Использование = Истина;
                            КонецЕсли;

                        КонецЕсли;
                        // дальше все по старому коду

                        Если СтрокаДокумента = Неопределено Тогда
                            НоваяСтрока = ТаблицаДокументовПоступлений.Добавить();
                            НоваяСтрока.ПриобретениеТоваровУслуг    = ДокументПоступленияСсылка;
                            НоваяСтрока.ПризнакТОРГ12                    = ПризнакТОРГ12;
                            НоваяСтрока.Использование                    = Истина;
                        КонецЕсли;

                    КонецЦикла;

                    Индекс = ТаблицаДокументовПоступлений.Количество() - 1;
                    Пока Индекс >= 0 Цикл

                        СтрокаПоступления = ТаблицаДокументовПоступлений[Индекс];

                        Если Не СтрокаПоступления.Использование Тогда

                            ДокументПоступленияОбъект = СтрокаПоступления.ПриобретениеТоваровУслуг.ПолучитьОбъект();
                            ДокументПоступленияОбъект.УстановитьПометкуУдаления(Истина);

                            ТаблицаДокументовПоступлений.Удалить(Индекс);

                        КонецЕсли;

                        Индекс = Индекс - 1;

                    КонецЦикла;

                    Объект.ДокументыПоступлений.Загрузить(ТаблицаДокументовПоступлений);

КонецПроцедуры

&НаСервере
Функция СоздатьОбновитьПриобретениеТоваровУслуг(ДокументСсылка, ПроектПоставки, ПризнакТОРГ12, ПризнакТоварыВПути =
Неопределено)

                    Если НЕ ЗначениеЗаполнено(ДокументСсылка) Тогда
                      ДокументОбъект = Документы.ПриобретениеТоваровУслуг.СоздатьДокумент();
                    Иначе
                      ДокументОбъект = ДокументСсылка.ПолучитьОбъект();
                    КонецЕсли;

                    СтруктураЗаполнения = Новый Структура;
                    СтруктураЗаполнения.Вставить("ДокументОснование", ПроектПоставки);
                    СтруктураЗаполнения.Вставить("ПризнакТОРГ12"     , ПризнакТОРГ12);


                    Если ПризнакТоварыВПути = Истина Тогда
                      ДокументОбъект.Дата = ТекущаяДата();
                    Иначе
                      ДокументОбъект.Дата = ПроектПоставки.ДатаПлановогоПоступления;
                      /
```

```
		КонецЕсли;

		ДокументОбъект.Автор = Пользователи.ТекущийПользователь();
		ДокументОбъект.ЗакупкаПодДеятельность = Перечисления.ТипыНалогообложенияНДС.ПродажаОблагаетсяНДС;
		ДокументОбъект.Заполнить(СтруктураЗаполнения);

		Если ПризнакТоварыВПути <> Неопределено Тогда
		  ДопРеквизитСсылка = ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.НайтиПоРеквизиту("Имя",
"ТоварыВПути");
		  Если ЗначениеЗаполнено(ДопРеквизитСсылка) Тогда
			СтрокаДопРеквизиты = ДокументОбъект.ДополнительныеРеквизиты.Найти(ДопРеквизитСсылка, "Свойство");
			Если СтрокаДопРеквизиты = Неопределено Тогда
				СтрокаДопРеквизиты = ДокументОбъект.ДополнительныеРеквизиты.Добавить();
				СтрокаДопРеквизиты.Свойство = ДопРеквизитСсылка;
			КонецЕсли;
			СтрокаДопРеквизиты.Значение = Истина;
		  КонецЕсли;
		КонецЕсли;

		ДокументОбъект.Записать(РежимЗаписиДокумента.Проведение);

		Возврат ДокументОбъект.Ссылка;

КонецФункции

&НаСервере
Функция СоздатьОбновитьКорректировкуПриобретения(ДокументСсылка, Основание, ПроектПоставки, ПризнакТОРГ12)

		Если ДокументСсылка = Неопределено Тогда
		  ДокументОбъект = Документы.КорректировкаПриобретения.СоздатьДокумент();
		  ДокументОбъект.Дата = ТекущаяДата();
		Иначе
		  ДокументОбъект = ДокументСсылка.ПолучитьОбъект();
		  ДокументОбъект.Записать(РежимЗаписиДокумента.ОтменаПроведения);
		КонецЕсли;

		СтруктураЗаполнения = Новый Структура;
		СтруктураЗаполнения.Вставить("ДокументОснование", Основание);
		СтруктураЗаполнения.Вставить("ПроектПоставки", ПроектПоставки);
		СтруктураЗаполнения.Вставить("ПризнакТОРГ12", ПризнакТОРГ12);

		ДокументОбъект.Заполнить(СтруктураЗаполнения);

		ДопРеквизитСсылка = ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.НайтиПоРеквизиту("Имя",
"КорректировкаТоваровВПути");
		Если ЗначениеЗаполнено(ДопРеквизитСсылка) Тогда
		  СтрокаДопРеквизиты = ДокументОбъект.ДополнительныеРеквизиты.Найти(ДопРеквизитСсылка, "Свойство");
		  Если СтрокаДопРеквизиты = Неопределено Тогда
			СтрокаДопРеквизиты = ДокументОбъект.ДополнительныеРеквизиты.Добавить();
			СтрокаДопРеквизиты.Свойство = ДопРеквизитСсылка;
		  КонецЕсли;
		  СтрокаДопРеквизиты.Значение = Истина;
		КонецЕсли;

		ДокументОбъект.Записать(РежимЗаписиДокумента.Проведение);

		Возврат ДокументОбъект.Ссылка;

КонецФункции

&НаСервере
Процедура ОчиститьРегистрыТабличныхЧастей()

		НачатьТранзакцию();

		//Итоги распределения


		//НаборЗаписей = РегистрыСведений.РезультатыРаспределенияПроектаПоставки.СоздатьНаборЗаписей();
		//НаборЗаписей.Отбор.ПроектПоставки.Установить(Объект.Ссылка);
		//
		//НаборЗаписей.Записать();

		Документы.ПроектПоставки.УдалитьДвиженияИзРегистровРаспределения(Объект.Ссылка);

		//Данные о товарах поставки
```

```
		НаборЗаписей = РегистрыСведений.ТоварыПроектовПоставки.СоздатьНаборЗаписей();
		НаборЗаписей.Отбор.ПроектПоставки.Установить(Объект.Ссылка);

		НаборЗаписей.Записать();

		//Состав комплектов
		НаборЗаписей = РегистрыСведений.СоставКомплектовПроектовПоставки.СоздатьНаборЗаписей();
		НаборЗаписей.Отбор.ПроектПоставки.Установить(Объект.Ссылка);

		НаборЗаписей.Записать();

		//КиЗы
		НаборЗаписей = РегистрыСведений.НомераКизПроектовПоставки.СоздатьНаборЗаписей();
		НаборЗаписей.Отбор.ПроектПоставки.Установить(Объект.Ссылка);

		НаборЗаписей.Записать();

		ЗафиксироватьТранзакцию();

КонецПроцедуры

#КонецОбласти

#Область ОформлениеРасхождений

&НаСервере
Процедура ОформитьРасхожденияНаСервере()

		#Область Шаблон
		#КонецОбласти

		#Область ОсновнойЗапрос

		ТаблицаФактаПриемки = Расхождения Сервер.ПолучитьФактПриемкиПроектаПоставки(Объект.Ссылка);

		Запрос = Новый Запрос;
		Запрос.Текст =
		"ВЫБРАТЬ
		| ПриобретениеТоваровУслугТовары.Ссылка КАК ДокументПриобретения,
		| ПриобретениеТоваровУслугТовары.НомерСтроки КАК НомерСтроки,
		| ПриобретениеТоваровУслугТовары.Номенклатура КАК Номенклатура,
		| ПриобретениеТоваровУслугТовары.Характеристика КАК Характеристика,
		| ПриобретениеТоваровУслугТовары.Количество КАК КоличествоПлан,
		| 0 КАК КоличествоФакт
		|ИЗ
		| Документ.ПриобретениеТоваровУслуг.Товары КАК ПриобретениеТоваровУслугТовары
		|ГДЕ
		| ПриобретениеТоваровУслугТовары.Ссылка.Проведен
		| И ПриобретениеТоваровУслугТовары.Ссылка.ПроектПоставки = &ПроектПоставки
		|
		|УПОРЯДОЧИТЬ ПО
		| ПриобретениеТоваровУслугТовары.Ссылка.Дата
		|ИТОГИ ПО
		| ДокументПриобретения";

		Запрос.УстановитьПараметр("ПроектПоставки", Объект.Ссылка);

		ДеревоПриобретений = Запрос.Выполнить().Выгрузить(ОбходРезультатаЗапроса.ПоГруппировкам);

		РаспределениеОрдернойСхемы = Новый Соответствие;
		Для Каждого СтрокаПриобретения Из ДеревоПриобретений.Строки Цикл

		  СтруктураРеквизитов = Новый Структура;
		  СтруктураРеквизитов.Вставить("Дата"      , "Дата");
		  СтруктураРеквизитов.Вставить("Склад"     , "Склад");
		  СтруктураРеквизитов.Вставить("ТипСклада" , "Склад.ТипСклада");

		  РеквизитыПриобретения =
ОбщегоНазначения.ЗначенияРеквизитовОбъекта(СтрокаПриобретения.ДокументПриобретения, СтруктураРеквизитов);

		  Если СкладыСервер.ИспользоватьОрдернуюСхемуПриПоступлении(РеквизитыПриобретения.Склад,
РеквизитыПриобретения.Дата) И РеквизитыПриобретения.ТипСклада = Перечисления.ТипыСкладов.ОптовыйСклад Тогда

		    ТаблицаПриходногоОрдера = ТаблицаФактаПриемки.СкопироватьКолонки();
		    РаспределениеОрдернойСхемы.Вставить(СтрокаПриобретения.ДокументПриобретения,
ТаблицаПриходногоОрдера);
```

```
                КонецЕсли;

            КонецЦикла;

            Если Объект.ИмпортнаяПоставка Тогда
              РасхожденияСервер.РаспределитьКомплектыПроектаПоставкиПоПриобретениям(ДеревоПриобретений,
Объект.Ссылка);
            КонецЕсли;

            РасхожденияСервер.РаспределитьФактПриемкиПоПриобретениям(ДеревоПриобретений, ТаблицаФактаПриемки,
РаспределениеОрдернойСхемы);

            #КонецОбласти

            Запрос = Новый Запрос;
            Запрос.УстановитьПараметр("ПроектПоставки"       , Объект.Ссылка);
            Запрос.Текст =
  "ВЫБРАТЬ
            | ПриходныйОрдерНаТовары.Ссылка КАК Ссылка,
            | ПриходныйОрдерНаТовары.Ссылка.Дата КАК Дата,
            | ПриходныйОрдерНаТовары.Ссылка.Склад.ТипСклада КАК ТипСклада,
            | ПриходныйОрдерНаТовары.Номенклатура КАК Номенклатура,
            | ПриходныйОрдерНаТовары.Характеристика КАК Характеристика,
  |  ПриходныйОрдерНаТовары.ХешСуммаШтрихкода КАК ХешСуммаШтрихкода,
  |  ПриходныйОрдерНаТовары.УШК КАК УникальныйШтрихкод
            |ИЗ
            | Документ.ПриходныйОрдерНаТовары.УникальныйШтрихкод КАК ПриходныйОрдерНаТовары
            |ГДЕ
            | НЕ ПриходныйОрдерНаТовары.Ссылка.ПометкаУдаления И
            | (ВЫРАЗИТЬ(ПриходныйОрдерНаТовары.Ссылка.Распоряжение КАК Документ.ПроектПоставки)) =
&ПроектПоставки";

            ТаблПулкодовУШК = Запрос.Выполнить().Выгрузить();
            //не занятые коды
            ТаблПулкодовУШК.Колонки.Добавить("Пометка");
            Для каждого СтрокаУШК Из ТаблПулкодовУШК Цикл
              СтрокаУШК.Пометка = Ложь;
            КонецЦикла;

            НачатьТранзакцию();

            #Область УдалениеСтарыхДокументов
            //Обновим связанные документы
            Запрос = Новый Запрос;
            Запрос.УстановитьПараметр("ПроектПоставки"       , Объект.Ссылка);
            Запрос.УстановитьПараметр("ДокументыПриобретения" ,
ОбщегоНазначенияКлиентСервер.СвернутьМассив(ДеревоПриобретений.Строки.ВыгрузитьКолонку("ДокументПриобретения")));
            Запрос.Текст =
            "ВЫБРАТЬ
            | ПриходныйОрдерНаТовары.Ссылка КАК Ссылка,
            | ПриходныйОрдерНаТовары.Дата КАК Дата,
            | ПриходныйОрдерНаТовары.Склад.ТипСклада КАК ТипСклада
            |ИЗ
            | Документ.ПриходныйОрдерНаТовары КАК ПриходныйОрдерНаТовары
            |ГДЕ
            | НЕ ПриходныйОрдерНаТовары.ПометкаУдаления
            | И (ВЫРАЗИТЬ(ПриходныйОрдерНаТовары.Распоряжение КАК Документ.ПроектПоставки)) = &ПроектПоставки
            |
            |ОБЪЕДИНИТЬ ВСЕ
            |
            |ВЫБРАТЬ
            | ПриходныйОрдерНаТовары.Ссылка,
            | ПриходныйОрдерНаТовары.Дата,
            | ПриходныйОрдерНаТовары.Склад.ТипСклада
            |ИЗ
            | Документ.ПриходныйОрдерНаТовары КАК ПриходныйОрдерНаТовары
            |ГДЕ
            | НЕ ПриходныйОрдерНаТовары.ПометкаУдаления
            | И ВЫРАЗИТЬ(ПриходныйОрдерНаТовары.Распоряжение КАК Документ.ПриобретениеТоваровУслуг) В
(&ДокументыПриобретения)";

            ДатаПриемкиТоваров = Неопределено;

            Выборка = Запрос.Выполнить().Выбрать();
            Пока Выборка.Следующий() Цикл
```

```
  Если ДатаПриемкиТоваров = Неопределено Тогда
        ДатаПриемкиТоваров = Выборка.Дата;
  КонецЕсли;

  Если Выборка.ТипСклада = Перечисления.ТипыСкладов.ОптовыйСклад Тогда

        ДокументОбъект = Выборка.Ссылка.ПолучитьОбъект();
        ДокументОбъект.УстановитьПометкуУдаления(Истина);

        ДатаПриемкиТоваров = Мин(Выборка.Дата, ДатаПриемкиТоваров);

  Иначе

        ДатаПриемкиТоваров = Макс(Выборка.Дата, ДатаПриемкиТоваров);

  КонецЕсли;

КонецЦикла;

Если ДатаПриемкиТоваров = Неопределено Тогда
  ДатаПриемкиТоваров = '00010101';
КонецЕсли;

Запрос.Текст =
"ВЫБРАТЬ
| АктОРасхожденияхПослеПриемкиТовары.Ссылка КАК Ссылка
|ПОМЕСТИТЬ втАктыОРасхождениях
|ИЗ
| Документ.АктОРасхожденияхПослеПриемки.Товары КАК АктОРасхожденияхПослеПриемкиТовары
|ГДЕ
| АктОРасхожденияхПослеПриемкиТовары.ДокументОснование В(&ДокументыПриобретения)
|;
|
|////////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
| втАктыОРасхождениях.Ссылка КАК Ссылка
|ИЗ
| втАктыОРасхождениях КАК втАктыОРасхождениях
|ГДЕ
| НЕ втАктыОРасхождениях.Ссылка.ПометкаУдаления
|
|ОБЪЕДИНИТЬ ВСЕ
|
|ВЫБРАТЬ
| ОприходованиеИзлишковТоваров.Ссылка
|ИЗ
| Документ.ОприходованиеИзлишковТоваров КАК ОприходованиеИзлишковТоваров
|ГДЕ
| ОприходованиеИзлишковТоваров.Основание В(&ДокументыПриобретения)
| И НЕ ОприходованиеИзлишковТоваров.ПометкаУдаления
|
|ОБЪЕДИНИТЬ ВСЕ
|
|ВЫБРАТЬ
| СписаниеНедостачТоваров.Ссылка
|ИЗ
| Документ.СписаниеНедостачТоваров КАК СписаниеНедостачТоваров
|ГДЕ
| СписаниеНедостачТоваров.Основание В(&ДокументыПриобретения)
| И НЕ СписаниеНедостачТоваров.ПометкаУдаления
| И НЕ СписаниеНедостачТоваров.ТоварыВПути
|
|ОБЪЕДИНИТЬ ВСЕ
|
|ВЫБРАТЬ
| ПересортицаТоваров.Ссылка
|ИЗ
| Документ.ПересортицаТоваров КАК ПересортицаТоваров
|       ВНУТРЕННЕЕ СОЕДИНЕНИЕ втАктыОРасхождениях КАК втАктыОРасхождениях
|       ПО ПересортицаТоваров.АктОРасхожденияхОснование = втАктыОРасхождениях.Ссылка
|ГДЕ
| НЕ ПересортицаТоваров.ПометкаУдаления
|
|ОБЪЕДИНИТЬ ВСЕ
|
|ВЫБРАТЬ
```

```
| ПересчетТоваров.Ссылка
|ИЗ
| Документ.ПересчетТоваров КАК ПересчетТоваров
|ГДЕ
| НЕ ПересчетТоваров.ПометкаУдаления
| И ПересчетТоваров.ДокументОснование В(&ДокументыПриобретения)";

Выборка = Запрос.Выполнить().Выбрать();
Пока Выборка.Следующий() Цикл

  ДокументОбъект = Выборка.Ссылка.ПолучитьОбъект();
  ДокументОбъект.УстановитьПометкуУдаления(Истина);

КонецЦикла;

#КонецОбласти

#Область СозданиеПриходныхОрдеров

Для Каждого КлючЗначение Из РаспределениеОрдернойСхемы Цикл

  Распоряжение = КлючЗначение.Ключ;
  ТаблицаПриходногоОрдера = КлючЗначение.Значение;

  РеквизитыРаспоряжения = ОбщегоНазначения.ЗначенияРеквизитовОбъекта(Распоряжение, "Дата, Партнер, Склад,
ХозяйственнаяОперация");

  ОбъектПриходногоОрдера = Документы.ПриходныйОрдерНаТовары.СоздатьДокумент();
  ОбъектПриходногоОрдера.Дата = Макс(ДатаПриемкиТоваров, РеквизитыРаспоряжения.Дата);
  ОбъектПриходногоОрдера.Склад = РеквизитыРаспоряжения.Склад;
  ОбъектПриходногоОрдера.Распоряжение = Распоряжение;
  ОбъектПриходногоОрдера.Ответственный = Пользователи.ТекущийПользователь();
  ОбъектПриходногоОрдера.Статус = Перечисления.СтатусыПриходныхОрдеров.Принят;
  ОбъектПриходногоОрдера.СкладскаяОперация = Перечисления.СкладскиеОперации.ПриемкаОтПоставщика;
  ОбъектПриходногоОрдера.Отправитель = РеквизитыРаспоряжения.Партнер;
  ОбъектПриходногоОрдера.ХозяйственнаяОперация = РеквизитыРаспоряжения.ХозяйственнаяОперация;

  ОбщегоНазначенияУТ.СвернутьТаблицуЗначений(ТаблицаПриходногоОрдера, , "Количество");


  ТаблицаПоУШК = Новый ТаблицаЗначений;
  ТаблицаПоУШК.Колонки.Добавить("Номенклатура");
  ТаблицаПоУШК.Колонки.Добавить("Характеристика");
  ТаблицаПоУШК.Колонки.Добавить("КоличествоУШК");

  Для Каждого СтрокаПриходногоОрдера Из ТаблицаПриходногоОрдера Цикл

        НоваяСтрока = ОбъектПриходногоОрдера.Товары.Добавить();
        ЗаполнитьЗначенияСвойств(НоваяСтрока, СтрокаПриходногоОрдера);

        НоваяСтрока.Количество = СтрокаПриходногоОрдера.Количество;
        НоваяСтрока.КоличествоУпаковок = СтрокаПриходногоОрдера.Количество;

        //добавить другой цикл учитывающий количество по строке
        Для А = 1 По (СтрокаПриходногоОрдера.Количество) Цикл
              НоваяСтрока = ТаблицаПоУШК.Добавить();
              НоваяСтрока.Номенклатура = СтрокаПриходногоОрдера.Номенклатура;
              НоваяСтрока.Характеристика = СтрокаПриходногоОрдера.Характеристика;
              НоваяСтрока.КоличествоУШК = 1;
        КонецЦикла;

  КонецЦикла;

  Для каждого СтрокаУШК Из ТаблицаПоУШК Цикл
        Отбор = Новый Структура;
        Отбор.Вставить("Номенклатура",СтрокаУШК.Номенклатура);
        Отбор.Вставить("Характеристика",СтрокаУШК.Характеристика);
        Отбор.Вставить("Пометка",Ложь);
        НайденныеСтроки = ТаблПулкодовУШК.НайтиСтроки(Отбор);
        Если НайденныеСтроки.Количество() > 0 Тогда
              Если ЗначениеЗаполнено(НайденныеСтроки[0].УникальныйШтрихкод) Тогда
                    НоваяСтрокаУШК = ОбъектПриходногоОрдера.УникальныйШтрихкод.Добавить();
                    ЗаполнитьЗначенияСвойств(НоваяСтрокаУШК, СтрокаУШК);
                    НоваяСтрокаУШК.УШК = НайденныеСтроки[0].УникальныйШтрихкод;
                    НайденныеСтроки[0].Пометка = Истина;
              КонецЕсли;
```

```
                КонецЕсли;
            КонецЦикла;

            ОбъектПриходногоОрдера.Записать(РежимЗаписиДокумента.Проведение);

        КонецЦикла;

        #КонецОбласти

        Для Каждого СтрокаПриобретения Из ДеревоПриобретений.Строки Цикл

            ЕстьРасхождения = Ложь;
            Для Каждого СтрокаТовараПриобретения Из СтрокаПриобретения.Строки Цикл

                Если СтрокаТовараПриобретения.КоличествоПлан <> СтрокаТовараПриобретения.КоличествоФакт Тогда
                    ЕстьРасхождения = Истина;
                    прервать;
                КонецЕсли;

            КонецЦикла;

            ОформитьКорректировкуПриобретения = РасхожденияСогласованыСПоставщиком И ЕстьРасхождения;
            ОформитьОприходованиеТоваров = Ложь;
            ОформитьСписаниеТоваров = Ложь;

            ОформитьПересортицуТоваров = Ложь;

            РеквизитыПриобретения =
ОбщегоНазначения.ЗначенияРеквизитовОбъекта(СтрокаПриобретения.ДокументПриобретения, "Дата, Склад, Подразделение");

            ПодразделениеПоУмолчанию = Справочники.СтруктураПредприятия.НайтиПоНаименованию("Коммерческая
дирекция",Истина);
            Если ЗначениеЗаполнено(ПодразделениеПоУмолчанию) Тогда
                РеквизитыПриобретения.Подразделение = ПодразделениеПоУмолчанию;
            КонецЕсли;

            #Область СозданиеАктовОРасхождениях

            Если ЕстьРасхождения Тогда

                ОбъектАктаРасхождений = Документы.АктОРасхожденияхПослеПриемки.СоздатьДокумент();

                ОбъектАктаРасхождений.Дата = Макс(ДатаПриемкиТоваров, РеквизитыПриобретения.Дата) + 1;
                ОбъектАктаРасхождений.ОснованиеДляСоставленияАкта = "Приказ";

                //Типовое заполнение на основании в релизе 2.4.6.207 полностью переносит таблицу товаров из ПТУ
                ОбъектАктаРасхождений.Заполнить(СтрокаПриобретения.ДокументПриобретения);

                ОбъектАктаРасхождений.Подразделение = РеквизитыПриобретения.Подразделение;

                ОтразитьСписанияВАктеОРасхождениях(ОбъектАктаРасхождений,
СтрокаПриобретения.ДокументПриобретения);

                ТаблицаТоваров = ОбъектАктаРасхождений.Товары.Выгрузить();
                ТаблицаТоваров.ЗаполнитьЗначения(0, "Количество, КоличествоУпаковок, Сумма, СуммаНДС, СуммаСНДС");

                //Поэтому можем просто воспользоваться переносом распределенного кол-ва
                Для Каждого СтрокаТовараПриобретения Из СтрокаПриобретения.Строки Цикл

                    Если СтрокаТовараПриобретения.НомерСтроки = 0 Тогда

                        НоваяСтрока = ТаблицаТоваров.Добавить();
                        НоваяСтрока.Номенклатура = СтрокаТовараПриобретения.Номенклатура;
                        НоваяСтрока.Характеристика = СтрокаТовараПриобретения.Характеристика;
                        НоваяСтрока.Количество = СтрокаТовараПриобретения.КоличествоФакт;
                        НоваяСтрока.ДокументОснование = СтрокаПриобретения.ДокументПриобретения;
                        НоваяСтрока.Склад = РеквизитыПриобретения.Склад;

                        Если
ОбщегоНазначения.ЗначениеРеквизитаОбъекта(СтрокаТовараПриобретения.Номенклатура, "Комплект") Тогда

                            НоваяСтрока.ЗаполненоПоОснованию = Истина;
                            НоваяСтрока.КоличествоПоДокументу = СтрокаТовараПриобретения.КоличествоПлан;

                            Комплектация = Новый Структура;
```

```
                                        Комплектация.Вставить("ОсновнойВариант",
Справочники.ВариантыКомплектацииНоменклатуры.ПолучитьОсновнуюКомплектацию(СтрокаТовараПриобретения.Номенклатура,
СтрокаТовараПриобретения.Характеристика));
                                        Комплектация.Вставить("Комплектующие" ,
Справочники.ВариантыКомплектацииНоменклатуры.ПолучитьКомплектующиеНоменклатуры(Комплектация.ОсновнойВариант));

                                        Отбор = Новый Структура("Номенклатура, Характеристика");
                                        Для Каждого СтрокаКомплектации Из Комплектация.Комплектующие Цикл

                                            Потребность = НоваяСтрока.КоличествоПоДокументу *
СтрокаКомплектации.Количество;

                                            ЗаполнитьЗначенияСвойств(Отбор, СтрокаКомплектации);

                                            НайденныеСтрокиТоваров = ТаблицаТоваров.НайтиСтроки(Отбор);
                                            Для Каждого СтрокаТовара Из НайденныеСтрокиТоваров Цикл

                                                Если Не ЗначениеЗаполнено(НоваяСтрока.СтавкаНДС) Тогда
                                                    НоваяСтрока.СтавкаНДС = СтрокаТовара.СтавкаНДС;
                                                ИначеЕсли НоваяСтрока.СтавкаНДС <> СтрокаТовара.СтавкаНДС И
СтрокаТовара.СтавкаНДС = Перечисления.СтавкиНДС.НДС20 Тогда
                                                    НоваяСтрока.СтавкаНДС = СтрокаТовара.СтавкаНДС;
                                                КонецЕсли;

                                                КРаспределению = Мин(СтрокаТовара.КоличествоПоДокументу,
Потребность);

                                                НоваяСтрока.Цена = НоваяСтрока.Цена + СтрокаТовара.Цена;
                                                НоваяСтрока.СуммаПоДокументу = НоваяСтрока.СуммаПоДокументу +
КРаспределению / СтрокаТовара.КоличествоПоДокументу * СтрокаТовара.СуммаПоДокументу;
                                                НоваяСтрока.СуммаНДСПоДокументу =
НоваяСтрока.СуммаНДСПоДокументу + КРаспределению / СтрокаТовара.КоличествоПоДокументу *
СтрокаТовара.СуммаНДСПоДокументу;
                                                НоваяСтрока.СуммаСНДСПоДокументу =
НоваяСтрока.СуммаСНДСПоДокументу + КРаспределению / СтрокаТовара.КоличествоПоДокументу *
СтрокаТовара.СуммаСНДСПоДокументу;

                                                СтрокаТовара.КоличествоПоДокументу =
СтрокаТовара.КоличествоПоДокументу - КРаспределению;

                                                Если СтрокаТовара.КоличествоПоДокументу = 0 Тогда
                                                    ТаблицаТоваров.Удалить(СтрокаТовара);
                                                КонецЕсли;

                                                Потребность = Потребность - КРаспределению;

                                            КонецЦикла;

                                        КонецЦикла;

                                    КонецЕсли;

                                Иначе

                                    ТаблицаТоваров[СтрокаТовараПриобретения.НомерСтроки - 1].Количество =
СтрокаТовараПриобретения.КоличествоФакт;

                                КонецЕсли;

                            КонецЦикла;

                            Для Каждого СтрокаТоваров Из ТаблицаТоваров Цикл

                                СтрокаТоваров.КоличествоУпаковок = СтрокаТоваров.Количество;

                                СтрокаТоваров.КоличествоУпаковокПоДокументу = СтрокаТоваров.КоличествоПоДокументу;

                                Если СтрокаТоваров.КоличествоПоДокументу <> 0 Тогда
                                    СтрокаТоваров.Сумма = СтрокаТоваров.СуммаПоДокументу /
СтрокаТоваров.КоличествоПоДокументу * СтрокаТоваров.Количество;
                                    СтрокаТоваров.СуммаНДС = СтрокаТоваров.СуммаНДСПоДокументу /
СтрокаТоваров.КоличествоПоДокументу * СтрокаТоваров.Количество;
                                    СтрокаТоваров.СуммаСНДС = СтрокаТоваров.СуммаСНДСПоДокументу /
СтрокаТоваров.КоличествоПоДокументу * СтрокаТоваров.Количество;
                                КонецЕсли;
```

```
                    Если РасхожденияСогласованыСПоставщиком Тогда

                        Если СтрокаТоваров.Количество > СтрокаТоваров.КоличествоПоДокументу Тогда
                            СтрокаТоваров.Действие =
Перечисления.ВариантыДействийПоРасхождениямВАктеПослеПриемки.ОформитьПерепоставленное;
                        ИначеЕсли СтрокаТоваров.Количество < СтрокаТоваров.КоличествоПоДокументу Тогда
                            СтрокаТоваров.Действие =
Перечисления.ВариантыДействийПоРасхождениямВАктеПослеПриемки.ОформитьНедостачу;
                        КонецЕсли;

                    Иначе

                        Если СтрокаТоваров.Количество > СтрокаТоваров.КоличествоПоДокументу Тогда

                            СтрокаТоваров.Действие =
Перечисления.ВариантыДействийПоРасхождениямВАктеПослеПриемки.ОтнестиПерепоставленноеНаПрочиеДоходы;

                        ИначеЕсли СтрокаТоваров.Количество < СтрокаТоваров.КоличествоПоДокументу Тогда

                            СтрокаТоваров.Действие =
Перечисления.ВариантыДействийПоРасхождениямВАктеПослеПриемки.ОтнестиНедостачуНаПрочиеРасходы;

                        КонецЕсли;

                    КонецЕсли;

                КонецЦикла;

                ОбъектАктаРасхождений.Товары.Загрузить(ТаблицаТоваров);

                //Для товара которого не было в приобретениях заполним цены
                СтруктураПересчетаСуммы =
ОбработкаТабличнойЧастиКлиентСервер.ПараметрыПересчетаСуммыНДСВСтрокеТЧ(ОбъектАктаРасхождений);

                СтруктураЗаполненияЦены =
ОбработкаТабличнойЧастиКлиентСервер.ПолучитьСтруктуруЗаполненияЦеныВСтрокеТЧ(ОбъектАктаРасхождений);
                СтруктураЗаполненияЦены.Вставить("ВидЦены", Справочники.ВидыЦен.ЗакупочнаяЦена);

                СтруктураДействий = Новый Структура;

                СтруктураДействий.Вставить("ЗаполнитьСтавкуНДС", Новый Структура("НалогообложениеНДС, Дата",
ОбъектАктаРасхождений.НалогообложениеНДС, ОбъектАктаРасхождений.Дата));
                СтруктураДействий.Вставить("ПересчитатьСуммуНДС", СтруктураПересчетаСуммы);
                СтруктураДействий.Вставить("ПересчитатьСуммуСНДС", СтруктураПересчетаСуммы);
                СтруктураДействий.Вставить("ПересчитатьСумму", "Количество");

                КэшированныеЗначения =
ОбработкаТабличнойЧастиКлиентСервер.ПолучитьСтруктуруКэшируемыеЗначения();

                НовыеСтроки = ОбъектАктаРасхождений.Товары.НайтиСтроки(Новый Структура("Цена", 0));
                Для Каждого СтрокаТоваров Из НовыеСтроки Цикл
                    ОбработкаТабличнойЧастиСервер.ОбработатьСтрокуТЧ(СтрокаТоваров, СтруктураДействий,
КэшированныеЗначения);
                КонецЦикла;

                УдалитьПустыеСтрокиВАктеОРасхождениях(ОбъектАктаРасхождений);

                Если НЕ РасхожденияСогласованыСПоставщиком Тогда
                    ОпределитьПересортицу(ОбъектАктаРасхождений, ОформитьПересортицуТоваров);
                КонецЕсли;

                ОбъектАктаРасхождений.Статус = Перечисления.СтатусыАктаОРасхождениях.Отработано;
                ОбъектАктаРасхождений.Записать(РежимЗаписиДокумента.Проведение);


                Если НЕ РасхожденияСогласованыСПоставщиком Тогда

            Документы.ПроектПоставки.ОпределитьДействияПоАктуОРасхождениях(ОбъектАктаРасхождений.Ссылка,
ОформитьПересортицуТоваров, ОформитьОприходованиеТоваров, ОформитьСписаниеТоваров);
                КонецЕсли;


            КонецЕсли;

            #КонецОбласти
```

```
		#Область СозданиеПересортицы

		Если ОформитьПересортицуТоваров Тогда

			//ОбъектПересортицыТоваров = Документы.ПересортицаТоваров.СоздатьДокумент();
			//ОбъектПересортицыТоваров.Дата		= Макс(ДатаПриемкиТоваров, РеквизитыПриобретения.Дата) +
1;
			//ОбъектПересортицыТоваров.ВидЦены		= Справочники.ВидыЦен.ЗакупочнаяЦена;
			//ОбъектПересортицыТоваров.Валюта  =
ДоходыИРасходыСервер.ПолучитьВалютуУправленческогоУчета(Справочники.ВидыЦен.ПолучитьРеквизитыВидаЦены(Справочники
.ВидыЦен.ЗакупочнаяЦена).ВалютаЦены);
			//
			//ОбъектПересортицыТоваров.Заполнить(ОбъектАктаРасхождений.Ссылка);
			//
			//ОбъектПересортицыТоваров.Комментарий =
СтроковыеФункцииКлиентСервер.ПодставитьПараметрыВСтроку("Создан на основании документа: %1", Объект.Ссылка);
			//ОбъектПересортицыТоваров.Записать(РежимЗаписиДокумента.Проведение);

			Основание = ОбъектАктаРасхождений.Ссылка;
			//
			ДокОбъект = Документы.ПересортицаТоваров.СоздатьДокумент();

			ДокОбъект.Дата 					= Основание.Дата;
			ДокОбъект.Организация 				= Основание.Организация;
			ДокОбъект.Подразделение 			= Основание.Подразделение;
			ДокОбъект.ВидДеятельностиНДС 		=
Перечисления.ТипыНалогообложенияНДС.ПродажаОблагаетсяНДС;
			ДокОбъект.Валюта 				= Основание.Валюта;
			ДокОбъект.ВидЦены 				= Справочники.ВидыЦен.ЗакупочнаяЦена;

			ДокОбъект.СтатьяРасходов 		= ПланыВидовХарактеристик.СтатьиРасходов.НайтиПоКоду("УП-
000136");
			ДокОбъект.АналитикаРасходов 		=
ДокОбъект.СтатьяРасходов.ТипЗначения.ПривестиЗначение(ДокОбъект.Подразделение);

			ДокОбъект.СтатьяДоходов 			=
ПланыВидовХарактеристик.СтатьиДоходов.НайтиПоКоду("УП-000002");
			ДокОбъект.АналитикаДоходов 		=
ДокОбъект.СтатьяДоходов.ТипЗначения.ПривестиЗначение(ДокОбъект.Подразделение);

			ДокОбъект.Ответственный 			= Пользователи.ТекущийПользователь();
			ДокОбъект.АктОРасхожденияхОснование = Основание;
			ДокОбъект.Комментарий 			= ТекстКомментария(Основание);

			Запрос = Новый Запрос;
			Запрос.УстановитьПараметр("Ссылка", Основание);
			Запрос.Текст =
			"ВЫБРАТЬ
			|	Недопоставка.Номенклатура.Наименование КАК Наименование,
			|	Недопоставка.Номенклатура КАК Номенклатура,
			|	Недопоставка.Характеристика КАК Характеристика,
			|	Недопоставка.Серия КАК Серия,
			|	Недопоставка.КоличествоПоДокументу - Недопоставка.Количество КАК Количество,
			|	Недопоставка.Цена КАК Цена,
			|	Недопоставка.НомерГТД КАК НомерГТД,
			|	Недопоставка.ДокументОснование КАК ДокументОснование,
			|	Недопоставка.СтатусУказанияСерий КАК СтатусУказанияСерий,
			|	Недопоставка.Склад КАК Склад
			|ИЗ
			|	Документ.АктОРасхожденияхПослеПриемки.Товары КАК Недопоставка
			|ГДЕ
			|	Недопоставка.Ссылка = &Ссылка
			|	И Недопоставка.Количество < Недопоставка.КоличествоПоДокументу
			|	И Недопоставка.Действие =
ЗНАЧЕНИЕ(Перечисление.ВариантыДействийПоРасхождениямВАктеПослеПриемки.ОжидатьДопоставкуБезОформления)
			|	И Недопоставка.Пересортица = ИСТИНА
			|;
			|
			|////////////////////////////////////////////////////////////////////////////////
			|ВЫБРАТЬ
			|	Перепоставка.Номенклатура.Наименование КАК Наименование,
			|	Перепоставка.Номенклатура КАК Номенклатура,
			|	Перепоставка.Характеристика КАК Характеристика,
			|	Перепоставка.Серия КАК Серия,
			|	Перепоставка.Количество - Перепоставка.КоличествоПоДокументу КАК Количество,
			|	Перепоставка.Цена КАК Цена,
```

```
                        |       Перепоставка.НомерГТД КАК НомерГТД,
                        |       Перепоставка.ДокументОснование КАК ДокументОснование,
                        |       Перепоставка.СтатусУказанияСерий КАК СтатусУказанияСерий,
                        |       Перепоставка.Склад КАК Склад
                        |ИЗ
                        |       Документ.АктОРасхожденияхПослеПриемки.Товары КАК Перепоставка
                        |ГДЕ
                        |       Перепоставка.Ссылка = &Ссылка
                        |       И Перепоставка.Количество > Перепоставка.КоличествоПоДокументу
                        |       И Перепоставка.Действие =
ЗНАЧЕНИЕ(Перечисление.ВариантыДействийПоРасхождениямВАктеПослеПриемки.ВернутьПерепоставленноеБезОформления)
                        |       И Перепоставка.Пересортица = ИСТИНА";
                    МассивРезультатов = Запрос.ВыполнитьПакет();

                    ТаблНедопоставка = МассивРезультатов[0].Выгрузить();
                    ТаблПерепоставка = МассивРезультатов[1].Выгрузить();

                    Для Каждого СтрокаНедопоставка Из ТаблНедопоставка Цикл


                        //СтруктураОтбора = Новый Структура("Номенклатура,Цена", СтрокаНедопоставка.Номенклатура,
СтрокаНедопоставка.Цена);
                        ////СтруктураОтбора = Новый Структура("Наименование,Цена", СтрокаНедопоставка.Наименование,
СтрокаНедопоставка.Цена);
                        СтруктураОтбора = Новый Структура("Номенклатура", СтрокаНедопоставка.Номенклатура);

                        мПерепоставка = ТаблПерепоставка.НайтиСтроки(СтруктураОтбора);
                        Индекс = 0;
                        Пока Индекс < мПерепоставка.Количество() И СтрокаНедопоставка.Количество > 0 Цикл
                            СтрокаПерепоставка = мПерепоставка[Индекс];
                            Индекс = Индекс + 1;
                            Если СтрокаПерепоставка.Количество > 0 Тогда
                                НоваяСтрока = ДокОбъект.Товары.Добавить();

                                НоваяСтрока.Номенклатура            = СтрокаНедопоставка.Номенклатура;
                                НоваяСтрока.Характеристика          = СтрокаНедопоставка.Характеристика;
                                НоваяСтрока.Серия                   = СтрокаНедопоставка.Серия;
                                НоваяСтрока.СтатусУказанияСерий = СтрокаНедопоставка.СтатусУказанияСерий;

                                НоваяСтрока.НоменклатураОприходование     = СтрокаПерепоставка.Номенклатура;
                                НоваяСтрока.ХарактеристикаОприходование   = СтрокаПерепоставка.Характеристика;
                                НоваяСтрока.СерияОприходование            =
СтрокаПерепоставка.Серия;
                                НоваяСтрока.СтатусУказанияСерийОприходование=
СтрокаПерепоставка.СтатусУказанияСерий;

                                НоваяСтрока.Количество      = Мин(СтрокаНедопоставка.Количество,
СтрокаПерепоставка.Количество);
                                НоваяСтрока.Цена            = СтрокаНедопоставка.Цена;
                                НоваяСтрока.НомерГТД        = СтрокаПерепоставка.НомерГТД;

                                СтрокаНедопоставка.Количество = СтрокаНедопоставка.Количество -
НоваяСтрока.Количество;
                                СтрокаПерепоставка.Количество = СтрокаПерепоставка.Количество -
НоваяСтрока.Количество;

                                Если НЕ ЗначениеЗаполнено(ДокОбъект.Склад) Тогда
                                    ДокОбъект.Склад = СтрокаНедопоставка.Склад;
                                КонецЕсли;

                            КонецЕсли;
                        КонецЦикла;

                    КонецЦикла;

                    СтруктураКолонок = Новый Структура;
                    СтруктураКолонок.Вставить("НомерСтроки");
                    СтруктураКолонок.Вставить("Номенклатура", "Номенклатура");
                    СтруктураКолонок.Вставить("Характеристика", "Характеристика");

                    оф_ЗапасыСервер.ЗаполнитьНомераГТДПоПоступлениям(ДокОбъект.Товары, СтруктураКолонок);

                    ДокОбъект.Записать(РежимЗаписиДокумента.Проведение);

                КонецЕсли;
```

```
		#КонецОбласти
		#Область СозданиеПересчета

		Если ОформитьОприходованиеТоваров ИЛИ ОформитьСписаниеТоваров Тогда

			ДанныеЗаполнения = Новый Структура;
			ДанныеЗаполнения.Вставить("АктОРасхождениях"      , ОбъектАктаРасхождений.Ссылка);
			ДанныеЗаполнения.Вставить("ОснованиеАкта"         , СтрокаПриобретения.ДокументПриобретения);
			ДанныеЗаполнения.Вставить("Склад"                 , РеквизитыПриобретения.Склад);
			ДанныеЗаполнения.Вставить("ПричинаИнвентаризации" ,
Перечисления.ПричиныИнвентаризации.ВыборочнаяПроверка);

			ОбъектПересчетаТоваров = Документы.ПересчетТоваров.СоздатьДокумент();
			ОбъектПересчетаТоваров.УчетныеДанныеЗаполнены = Истина;

			ОбъектПересчетаТоваров.Заполнить(ДанныеЗаполнения);

			ОбъектПересчетаТоваров.Дата = Макс(ДатаПриемкиТоваров, РеквизитыПриобретения.Дата) + 1;
			ОбъектПересчетаТоваров.Записать(РежимЗаписиДокумента.Проведение); //Проведение в статусе "В работе"

			ОбъектПересчетаТоваров.Статус = Перечисления.СтатусыПересчетовТоваров.Выполнено;
			ОбъектПересчетаТоваров.Записать(РежимЗаписиДокумента.Проведение);

		КонецЕсли;

		#КонецОбласти

		#Область СозданиеОприходования

		Если ОформитьОприходованиеТоваров Тогда

			ДанныеЗаполнения = Новый Структура;
			ДанныеЗаполнения.Вставить("АктОРасхождениях", ОбъектАктаРасхождений.Ссылка);
			ДанныеЗаполнения.Вставить("ОснованиеАкта",    СтрокаПриобретения.ДокументПриобретения);
			ДанныеЗаполнения.Вставить("Склад",            РеквизитыПриобретения.Склад);

			ОбъектОприходованияТоваров = Документы.ОприходованиеИзлишковТоваров.СоздатьДокумент();
			ОбъектОприходованияТоваров.Дата = Макс(ДатаПриемкиТоваров, РеквизитыПриобретения.Дата) + 1;
			ОбъектОприходованияТоваров.ВидЦены = Справочники.ВидыЦен.ЗакупочнаяЦена;
			ОбъектОприходованияТоваров.Валюта =
ДоходыИРасходыСервер.ПолучитьВалютуУправленческогоУчета(Справочники.ВидыЦен.ПолучитьРеквизитыВидаЦены(Справочники
.ВидыЦен.ЗакупочнаяЦена).ВалютаЦены);

			ОбъектОприходованияТоваров.ПересчетТоваров = ОбъектПересчетаТоваров.Ссылка;

			ОбъектОприходованияТоваров.Заполнить(ДанныеЗаполнения);

			ОбъектОприходованияТоваров.ОприходованиеПодДеятельность =
Перечисления.ТипыНалогообложенияНДС.ПродажаОблагаетсяНДС;

			//ОбъектОприходованияТоваров.АналитикаДоходов = РеквизитыПриобретения.Подразделение;
			ОбъектОприходованияТоваров.Комментарий =
СтроковыеФункцииКлиентСервер.ПодставитьПараметрыВСтроку("Создан на основании документа: %1", Объект.Ссылка);

		Документы.ОприходованиеИзлишковТоваров.ЗаполнитьНомераГТДПоПоступлениям(ОбъектОприходованияТоваров.То
вары);

			ОбъектОприходованияТоваров.Записать(РежимЗаписиДокумента.Проведение);

		КонецЕсли;

		#КонецОбласти

		#Область СозданиеСписания

		Если ОформитьСписаниеТоваров Тогда

			ДанныеЗаполнения = Новый Структура;
			ДанныеЗаполнения.Вставить("АктОРасхождениях", ОбъектАктаРасхождений.Ссылка);
			ДанныеЗаполнения.Вставить("ОснованиеАкта",    СтрокаПриобретения.ДокументПриобретения);
			ДанныеЗаполнения.Вставить("Склад",            РеквизитыПриобретения.Склад);

			ОбъектСписанияТоваров = Документы.СписаниеНедостачТоваров.СоздатьДокумент();
			ОбъектСписанияТоваров.Дата = Макс(ДатаПриемкиТоваров, РеквизитыПриобретения.Дата) + 1;

			ОбъектСписанияТоваров.Заполнить(ДанныеЗаполнения);
```

```
                    ОбъектСписанияТоваров.Комментарий =
СтроковыеФункцииКлиентСервер.ПодставитьПараметрыВСтроку("Создан на основании документа: %1", Объект.Ссылка);

                    ОбъектСписанияТоваров.СтатьяРасходов = Константы.СтатьяРасходовСписанияТоваров.Получить();

                    ОбъектСписанияТоваров.АналитикаРасходов = РеквизитыПриобретения.Подразделение;
                    ОбъектСписанияТоваров.ВидДеятельностиНДС = Перечисления.ТипыНалогообложенияНДС.ПустаяСсылка();

                    ОбъектСписанияТоваров.ПересчетТоваров = ОбъектПересчетаТоваров.Ссылка;

                    ОбъектСписанияТоваров.Записать(РежимЗаписиДокумента.Проведение);

                КонецЕсли;

                #КонецОбласти

            КонецЦикла;

            Документы.ПроектПоставки.СоздатьОбновитьДокументыСборкиКомплектов(Объект.Ссылка);

            ЗафиксироватьТранзакцию();

КонецПроцедуры

#КонецОбласти

#Область Прочее

&НаСервере
Процедура ТоварыОбновитьИтоги()

            Запрос = Новый Запрос;
            Запрос.Текст =
            "ВЫБРАТЬ
            | СУММА(ТоварыПроектовПоставки.СуммаЗакупочная) КАК СуммаЗакупочная,
            | СУММА(ТоварыПроектовПоставки.СуммаНДС) КАК СуммаНДС,
            | СУММА(ТоварыПроектовПоставки.СуммаРекомендованная) КАК СуммаРекомендованная,
            | СУММА(ТоварыПроектовПоставки.СуммаРозничная) КАК СуммаРозничная,
            | СУММА(ТоварыПроектовПоставки.Количество) КАК Количество,
            | СУММА(ТоварыПроектовПоставки.СуммаВВалюте) КАК СуммаВВалюте
            |ИЗ
            | РегистрСведений.ТоварыПроектовПоставки КАК ТоварыПроектовПоставки
            |ГДЕ
            | ТоварыПроектовПоставки.ПроектПоставки = &ПроектПоставки";

            Запрос.УстановитьПараметр("ПроектПоставки", Объект.Ссылка);

            Выборка = Запрос.Выполнить().Выбрать();

            Если Выборка.Следующий() Тогда

              ТоварыИтогСуммаЗакупочная = Выборка.СуммаЗакупочная;
              ТоварыИтогСуммаНДС = Выборка.СуммаНДС;
              ТоварыИтогСуммаРекомендованная = Выборка.СуммаРекомендованная;
              ТоварыИтогСуммаРозничная = Выборка.СуммаРозничная;
              ТоварыИтогКоличество = Выборка.Количество;
              ТоварыИтогСуммаВВалюте = Выборка.СуммаВВалюте;

            КонецЕсли;

КонецПроцедуры

&НаСервере
Функция ПроверитьВозможностьРедактированияПоступления()

            Запрос = Новый Запрос;
            Запрос.Текст =
            "ВЫБРАТЬ ПЕРВЫЕ 1
            | СтатусыПроверкиДокументов.Документ
            |ИЗ
            | РегистрСведений.СтатусыПроверкиДокументов КАК СтатусыПроверкиДокументов
            |ГДЕ
            | СтатусыПроверкиДокументов.Документ В(&СписокПоступлений)
            | И СтатусыПроверкиДокументов.СтатусПроверки =
ЗНАЧЕНИЕ(Перечисление.ЭтапыПроверкиДокументаВРеглУчете.Проверен)";
```

```
		Запрос.УстановитьПараметр("СписокПоступлений",
Объект.ДокументыПоступлений.Выгрузить().ВыгрузитьКолонку("ПриобретениеТоваровУслуг"));

		РезультатЗапроса = Запрос.Выполнить();

		Возврат РезультатЗапроса.Пустой();

КонецФункции

&НаКлиенте
Процедура ГруппаСтраницыПриСменеСтраницы(Элемент, ТекущаяСтраница)

		Если ТекущаяСтраница.Имя = "СтраницаПринятыеТовары" Тогда

		 ПринятыеТовары.Параметры.УстановитьЗначениеПараметра("ПроектПоставки", Объект.Ссылка);


		ИначеЕсли ТекущаяСтраница.Имя = "СтраницаДопРасходы" Тогда

		 ДополнительныеРасходы.Параметры.УстановитьЗначениеПараметра("ПроектПоставки", Объект.Ссылка);
		 ОбновитьИтогиДопРасходы();

		ИначеЕсли ТекущаяСтраница.Имя = "СтраницаДокументы" Тогда

		 ЗаполнитьДеревоДокументов();

		КонецЕсли;

КонецПроцедуры

&НаКлиенте
Процедура ОповещениеОткрытьФормуЗагрузкиExcel(РезультатОткрытияФормы, ДополнительныеПараметры) Экспорт

		Элементы.ПринятыеТовары.Обновить();

КонецПроцедуры

&НаКлиенте
Процедура ЗагрузитьФактExcel(Команда)

		ДополнительныеПараметры = Новый Структура;
		ОбработчикОповещения = Новый ОписаниеОповещения("ОповещениеОткрытьФормуЗагрузкиExcel", ЭтотОбъект,
ДополнительныеПараметры);

		ОткрытьФорму("Обработка.ЗагрузкаИзExcelПроектПоставки.Форма", Новый Структура("ПроектПоставки",
Объект.Ссылка),,,,,ОбработчикОповещения, РежимОткрытияОкнаФормы.БлокироватьВесьИнтерфейс);

КонецПроцедуры

&НаСервере
Процедура РасхожденияСогласованыСПоставщикомПриИзмененииНаСервере()

		Документы.ПроектПоставки.УстановитьСтатусСогласования(Объект.Ссылка,РасхожденияСогласованыСПоставщиком);

КонецПроцедуры

&НаКлиенте
Процедура РасхожденияСогласованыСПоставщикомПриИзменении(Элемент)

		РасхожденияСогласованыСПоставщикомПриИзмененииНаСервере();

КонецПроцедуры

&НаКлиенте
Процедура ОповещениеОткрытьЗагрузкаКорректировки(РезультатОткрытияФормы, ДополнительныеПараметры) Экспорт

		Если Элементы.ГруппаСтраницы.ТекущаяСтраница.Имя = "СтраницаДокументы" Тогда

		 ЗаполнитьДеревоДокументов();

		КонецЕсли;

КонецПроцедуры

&НаКлиенте
Процедура ЗагрузитьКорректировкуExcel(Команда)
```

```
		ДополнительныеПараметры = Новый Структура;
		ОбработчикОповещения = Новый ОписаниеОповещения("ОповещениеОткрытьЗагрузкаКорректировки", ЭтотОбъект,
ДополнительныеПараметры);

		ОткрытьФорму("Обработка.ЗагрузкаИзExcelПроектПоставки.Форма.ФормаЗагрузкиСЧФ", Новый
Структура("ПроектПоставки", Объект.Ссылка),,,,,ОбработчикОповещения, РежимОткрытияОкнаФормы.БлокироватьВесьИнтерфейс);

КонецПроцедуры
#КонецОбласти

#Область СтатусыОбработкиДокумента
&НаСервере
Процедура НастроитьФормуПоСтатусуОбработки()

		Элементы.Статус.ТолькоПросмотр = СкладПодУправлениемWMS;

		Если СкладПодУправлениемWMS Тогда

		 Элементы.Статус.СписокВыбора.Очистить();
		 Элементы.Статус.РежимВыбораИзСписка = Ложь;

		 //Заголовок и видимость кнопки смены статуса
		 СменяемыеСтатусы = Новый Массив;
		 СменяемыеСтатусы.Добавить(Перечисления.СтатусыОбработкиДокументов.Создан);

		 //Нельзя отзывать пока идет обмен - возможна рассинхронизация данных в случае если данные попали в менеджер
заданий
		 //СменяемыеСтатусы.Добавить(Перечисления.СтатусыОбработкиДокументов.ОбменДанными);
		 СменяемыеСтатусы.Добавить(Перечисления.СтатусыОбработкиДокументов.ОжидаетОбработку);
		 СменяемыеСтатусы.Добавить(Перечисления.СтатусыОбработкиДокументов.Обработан);
		 СменяемыеСтатусы.Добавить(Перечисления.СтатусыОбработкиДокументов.Завершен);

		 Если СменяемыеСтатусы.Найти(СтатусОбработки) <> Неопределено Тогда

			Если СтатусОбработки = Перечисления.СтатусыОбработкиДокументов.Создан Тогда
				Элементы.СменитьСтатусОбработки.Заголовок = "В работу";
			ИначеЕсли СтатусОбработки = Перечисления.СтатусыОбработкиДокументов.Обработан Тогда
				Элементы.СменитьСтатусОбработки.Заголовок = "Завершить";
			ИначеЕсли СтатусОбработки = Перечисления.СтатусыОбработкиДокументов.Завершен Тогда
				Элементы.СменитьСтатусОбработки.Заголовок = "Продолжить";
			Иначе
				Элементы.СменитьСтатусОбработки.Заголовок = "Отозвать";
			КонецЕсли;

			Элементы.СменитьСтатусОбработки.Видимость = Истина;

		 Иначе

			Элементы.СменитьСтатусОбработки.Видимость = Ложь;

		 КонецЕсли;

		 //Блокировка формы
		 Если Объект.ЗаблокированСтатусомОбработки Тогда

			//Документ находится в нередактируемом статусе
			Если ТолькоПросмотр Тогда
				// Форма уже заблокирована, но не из-за статуса.
				СтатусыОбработки_БлокировкаФормы = Ложь;
			Иначе
				// Форма не заблокирована, заблокируем ее и сделаем соответствующую пометку.
				ТолькоПросмотр = Истина;
				СтатусыОбработки_БлокировкаФормы = Истина;
			КонецЕсли;

		 Иначе

			// Форма не нуждается в блокировке

			Если СтатусыОбработки_БлокировкаФормы Тогда
				// Форма ранее была заблокирована по статусу - разблокируем форму и снимем пометку о блокировке.
				ТолькоПросмотр = Ложь;
				СтатусыОбработки_БлокировкаФормы = Ложь;

			КонецЕсли;
```

```
            КонецЕсли;

        Иначе

            ДоступныеСтатусы = Новый Массив;
            ДоступныеСтатусы.Добавить(Перечисления.СтатусыОбработкиДокументов.Создан);
            ДоступныеСтатусы.Добавить(Перечисления.СтатусыОбработкиДокументов.ВОбработке);
            ДоступныеСтатусы.Добавить(Перечисления.СтатусыОбработкиДокументов.Обработан);
            ДоступныеСтатусы.Добавить(Перечисления.СтатусыОбработкиДокументов.Завершен);

            Элементы.Статус.СписокВыбора.ЗагрузитьЗначения(ДоступныеСтатусы);
            Элементы.Статус.РежимВыбораИзСписка = Истина;

            Элементы.СменитьСтатусОбработки.Видимость = Ложь;

        КонецЕсли;

КонецПроцедуры

&НаСервере
Функция СменитьСтатусОбработкиНаСервере() Экспорт

        возврат УправлениеСтатусамиОбработки.СменитьСтатусОбработки(ЭтотОбъект, РеквизитФормыВЗначение("Объект"));

КонецФункции

&НаКлиенте
Процедура СменитьСтатусОбработки(Команда)

        УправлениеСтатусамиОбработкиКлиент.СменитьСтатусОбработки(ЭтотОбъект, Объект, Ложь);

КонецПроцедуры
#КонецОбласти

&НаКлиентеНаСервереБезКонтекста
Функция НомерКартинки(СтрокаДерева)

        Если СтрокаДерева.ПометкаУдаления Тогда
          Возврат 1;
        ИначеЕсли СтрокаДерева.Проведен Тогда
          Возврат 0;
        Иначе
          Возврат 2;
        КонецЕсли;

КонецФункции

&НаСервере
Процедура ВывестиДеревоДокументов()

        ДеревоДокументов.ПолучитьЭлементы().Очистить();

        Для каждого ДокПоступления ИЗ Объект.ДокументыПоступлений Цикл

          ДокументСсылка = ДокПоступления.ПриобретениеТоваровУслуг;

          Запрос = Новый Запрос;
          Запрос.Текст =
          //"ВЫБРАТЬ РАЗРЕШЕННЫЕ
          //|    СвязанныеДокументы.Ссылка КАК Ссылка,
          //|    СвязанныеДокументы.Ссылка.Проведен КАК Проведен,
          //|    СвязанныеДокументы.Ссылка.ПометкаУдаления КАК ПометкаУдаления,
          //|    СвязанныеДокументы.Ссылка.Валюта КАК Валюта
          //|ИЗ
          //|    КритерийОтбора.СвязанныеДокументы(&Поступление) КАК СвязанныеДокументы";

          "ВЫБРАТЬ РАЗРЕШЕННЫЕ
          |    АктОРасхожденияхПослеПриемкиТовары.Ссылка КАК Ссылка,
          |    СУММА(АктОРасхожденияхПослеПриемкиТовары.СуммаСНДС) КАК СуммаСНДС,
          |    СУММА(АктОРасхожденияхПослеПриемкиТовары.Количество) КАК Количество,
          |    СУММА(АктОРасхожденияхПослеПриемкиТовары.СуммаНДС) КАК СуммаНДС
          |ИЗ
          |    Документ.АктОРасхожденияхПослеПриемки.Товары КАК АктОРасхожденияхПослеПриемкиТовары
          |ГДЕ
          |    АктОРасхожденияхПослеПриемкиТовары.ДокументОснование = &ДокументОснование
          |
```

```
|СГРУППИРОВАТЬ ПО
|	АктОРасхожденияхПослеПриемкиТовары.Ссылка
|
|ОБЪЕДИНИТЬ ВСЕ
|
|ВЫБРАТЬ
|	КорректировкаПриобретения.Ссылка,
|	КорректировкаПриобретения.СуммаДокумента,
|	0,
|	0
|ИЗ
|	Документ.КорректировкаПриобретения КАК КорректировкаПриобретения
|ГДЕ
|	КорректировкаПриобретения.ДокументОснование = &ДокументОснование";

Запрос.УстановитьПараметр("ДокументОснование", ДокументСсылка);
Результат = Запрос.Выполнить();
Выборка = Результат.Выбрать();

СтрокиДерева = ДеревоДокументов.ПолучитьЭлементы();

СтрДерева = СтрокиДерева.Добавить();
СтрДерева.Ссылка= ДокументСсылка;
СтрДерева.ДокументПредставление = Строка(ДокументСсылка);
СтрДерева.Валюта = ДокументСсылка.Валюта;
СтрДерева.СуммаДокумента = ДокументСсылка.СуммаДокумента;
СтрДерева.Количество = ДокументСсылка.Товары.Итог("Количество");
СтрДерева.НДС = ДокументСсылка.Товары.Итог("СуммаНДС");
СтрДерева.Проведен = ДокументСсылка.Проведен;
СтрДерева.ПометкаУдаления = ДокументСсылка.ПометкаУдаления;
СтрДерева.Картинка = НомерКартинки(СтрДерева);
СтрДерева.ПризнакТОРГ12 = ДокПоступления.ПризнакТОРГ12;

Пока Выборка.Следующий() Цикл

	СтрокиДерева2 = СтрДерева.ПолучитьЭлементы();

	ВторойУр = СтрокиДерева2.Добавить();
	ВторойУр.Ссылка= Выборка.Ссылка;
	ВторойУр.ДокументПредставление = Строка(Выборка.Ссылка);
	ВторойУр.Валюта = Выборка.Ссылка.Валюта;
	ВторойУр.Проведен = Выборка.Ссылка.Проведен;
	ВторойУр.ПометкаУдаления = Выборка.Ссылка.ПометкаУдаления;
	ВторойУр.Картинка = НомерКартинки(ВторойУр);
	ВторойУр.СуммаДокумента = Выборка.СуммаСНДС;
	ВторойУр.Количество = Выборка.Количество;
	ВторойУр.НДС = Выборка.СуммаНДС;
	ВторойУр.Картинка = НомерКартинки(ВторойУр);

	Если ТипЗнч(Выборка.Ссылка)= Тип("ДокументСсылка.КорректировкаПриобретения") Тогда

		ВторойУр.Количество = Выборка.Ссылка.Товары.Итог("Количество");
		ВторойУр.НДС = Выборка.Ссылка.Товары.Итог("СуммаНДС");

	КонецЕсли;

 КонецЦикла;

КонецЦикла;

КонецПроцедуры

&НаСервере
Процедура ЗаполнитьПараметрыДерева()

	Если Объект.ДокументыПоступлений.Количество()>0 Тогда

 ВывестиДеревоДокументов();

	КонецЕсли;

КонецПроцедуры // УстановитьПараметрыДинамическогоСписка()

&НаКлиенте
Процедура ЗаполнитьДеревоДокументов()
```

```
		ЗаполнитьПараметрыДерева();

		Строки = ДеревоДокументов.ПолучитьЭлементы();
		Для Каждого Строка Из Строки Цикл
		  Элементы.ДеревоДокументов.Развернуть(Строка.ПолучитьИдентификатор(), Истина);
		КонецЦикла;

КонецПроцедуры

//&НаКлиенте
//Процедура ДокументыПоступленийПриАктивизацииСтроки(Элемент)
//
//		ПодключитьОбработчикОжидания("ОбработатьАктивизациюСтрокиСписка", 0.2, Истина);
//
//КонецПроцедуры

&НаСервере
Функция ВернутьМетаданные(ТекДанные)

		Имя = ТекДанные.Метаданные().Имя;

		Возврат Имя;

КонецФункции

&НаКлиенте
Процедура ДеревоДокументовВыбор(Элемент, ВыбраннаяСтрока, Поле, СтандартнаяОбработка)

		ТекДанные = Элементы.ДеревоДокументов.ТекущиеДанные;

		Если Поле.Имя = "ДеревоДокументовСсылка" Тогда
		  ТекДокумент = ТекДанные.Ссылка;
		  Если ЗначениеЗаполнено(ТекДокумент) Тогда
			СтруктураОткрытия = Новый Структура("Ключ", ТекДокумент);
			Форма = ПолучитьФорму("Документ." + ВернутьМетаданные(ТекДокумент) + ".ФормаОбъекта",
СтруктураОткрытия);
			Форма.Открыть();
		  КонецЕсли;
		КонецЕсли;

КонецПроцедуры
		&НаКлиенте
Процедура ЗагрузитьПТУ(Команда)
		ЗагрузитьПТУНаСервере();
КонецПроцедуры

&НаСервере
Процедура ЗагрузитьПТУНаСервере()

		Запрос = Новый Запрос(
		"ВЫБРАТЬ
		| ПТУ.Ссылка КАК ПриобретениеТоваровУслуг,
		| ПТУ.НомерВходящегоДокумента КАК ПризнакТОРГ12
		|ИЗ
		| Документ.ПриобретениеТоваровУслуг КАК ПТУ
		|ГДЕ
		| ПТУ.Проведен
		| И ПТУ.Дата = &Дата
		| И ПТУ.Контрагент = &Контрагент"
		);
		Запрос.УстановитьПараметр("Дата", Объект.ДатаПлановогоПоступления);
		Запрос.УстановитьПараметр("Контрагент", Объект.Контрагент);
		Объект.ДокументыПоступлений.Загрузить(Запрос.Выполнить().Выгрузить());

КонецПроцедуры

&НаСервере
Процедура ЗаполнитьДопРасходыНаСервере()

		Элементы.ДополнительныеРасходы.Обновить();

КонецПроцедуры

&НаКлиенте
Процедура ЗаполнитьДопРасходы(Команда)
```

```
		ЗаполнитьДопРасходыНаСервере();

КонецПроцедуры

&НаКлиенте
Процедура СоздатьДопРасход(Команда)

		Если СозданыДопРасходы()=Ложь Тогда

		  ПараметрыОткрытия = Новый Структура;
		  ПараметрыОткрытия.Вставить("ПроектПоставки",Объект.Ссылка);

		  ОткрытьФорму("Документ.РаспределениеРасходовПроектовПоставки.Форма.ФормаДокумента", ПараметрыОткрытия);

		иначе

		  Сообщить("Существуют проведенные документы по доп.расходам. Внесите изменения в существующие документы.");


		КонецЕсли;

КонецПроцедуры

&НаСервере
Функция СозданыДопРасходы()

		Запрос = Новый Запрос;
		Запрос.Текст =
		"ВЫБРАТЬ
		| КОЛИЧЕСТВО(*) КАК Записи
		|ИЗ
		| РегистрСведений.РасходыПроектовПоставки КАК ПринятыеТоварыПроектовПоставки
		|ГДЕ
		| ПринятыеТоварыПроектовПоставки.ПроектПоставки = &ПроектПоставки";

		Запрос.УстановитьПараметр("ПроектПоставки", Объект.Ссылка);
		РезультатЗапроса = Запрос.Выполнить();
		Выборка = РезультатЗапроса.Выбрать();
		Выборка.Следующий();

		Записи = Выборка.Записи;

		Если Выборка.Записи = 0 Тогда

		  Возврат Ложь;

		иначе

		  Возврат Истина;

		КонецЕсли;

КонецФункции

&НаСервере
Функция НайтиДопРасход(Расход=Неопределено)

		Запрос = Новый Запрос;
		Если Расход <> Неопределено Тогда

		  Запрос.Текст =
		  "ВЫБРАТЬ ПЕРВЫЕ 1
		  |	РасходыПроектовПоставки.Ссылка КАК Ссылка
		  |ИЗ
		  |	Документ.РаспределениеРасходовПроектовПоставки.РасходыПроектовПоставки КАК
РасходыПроектовПоставки
		  |ГДЕ
		  |	РасходыПроектовПоставки.Документ = &ПроектПоставки
		  |	И РасходыПроектовПоставки.РасходНаПоставку = &РасходНаПоставку";

		  Запрос.УстановитьПараметр("ПроектПоставки", Объект.Ссылка);
		  Запрос.УстановитьПараметр("РасходНаПоставку", Расход);

		иначе

		  Запрос.Текст =
```

```
                "ВЫБРАТЬ ПЕРВЫЕ 1
                |    РаспределениеРасходовПроектовПоставки.Ссылка КАК Ссылка
                |ИЗ
                |    Документ.РаспределениеРасходовПроектовПоставки КАК РаспределениеРасходовПроектовПоставки
                |ГДЕ
                |    РаспределениеРасходовПроектовПоставки.ПроектПоставки = &ПроектПоставки";

                Запрос.УстановитьПараметр("ПроектПоставки",Объект.Ссылка);

            КонецЕсли;

            РезультатЗапроса = Запрос.Выполнить();

            ВыборкаДетальныеЗаписи = РезультатЗапроса.Выбрать();

            Если ВыборкаДетальныеЗаписи.Следующий() Тогда

              Возврат ВыборкаДетальныеЗаписи.Ссылка;

            иначе

              Возврат Документы.РаспределениеРасходовПроектовПоставки.ПустаяСсылка();

            КонецЕсли;

КонецФункции

&НаКлиенте
Процедура ИзменитьДопРасход(Команда)

            Если СозданыДопРасходы() Тогда

              ТекущиеДанные = Элементы.ДополнительныеРасходы.ТекущиеДанные;

              ПараметрыФормы = Новый Структура("Ключ", НайтиДопРасход(ТекущиеДанные.Расход));
              ОткрытьФорму("Документ.РаспределениеРасходовПроектовПоставки.Форма.ФормаДокумента", ПараметрыФормы);

            иначе

              Сообщить("Нет документов дополнительных расходов по проекту поставке");

            КонецЕсли;

КонецПроцедуры

&НаСервере
Процедура ОбновитьИтогиДопРасходы()

            Запрос = Новый Запрос;
            Запрос.Текст =
            "ВЫБРАТЬ
            | СУММА(РасходыПроектовПоставки.Сумма) КАК СуммаРасходов
            |ИЗ
            | РегистрСведений.РасходыПроектовПоставки КАК РасходыПроектовПоставки
            |ГДЕ
            | РасходыПроектовПоставки.ПроектПоставки = &ПроектПоставки";

            Запрос.УстановитьПараметр("ПроектПоставки", Объект.Ссылка);

            Выборка = Запрос.Выполнить().Выбрать();

            Если Выборка.Следующий() Тогда

              ДопРасходыИтогСумма = Выборка.СуммаРасходов;

            КонецЕсли;

КонецПроцедуры

&НаСервере
Процедура ДобавитьДокументПриходаНаСервере(ПриобретениеТоваровУслуг)

            ПараметрыОтбора = Новый Структура;
  ПараметрыОтбора.Вставить("ПриобретениеТоваровУслуг", ПриобретениеТоваровУслуг);
            НайденныеСтроки = Объект.ДокументыПоступлений.НайтиСтроки(ПараметрыОтбора);
```

```
                Если НайденныеСтроки.Количество()= 0 Тогда

                  НовСтр = Объект.ДокументыПоступлений.Добавить();
                  НовСтр.ПриобретениеТоваровУслуг = ПриобретениеТоваровУслуг;
                  НовСтр.ПризнакТОРГ12 = ПриобретениеТоваровУслуг.Комментарий;

                КонецЕсли;

КонецПроцедуры

&НаКлиенте
Процедура ВыбратьПриобретениеОкончание(Значение, ДополнительныеПараметры) Экспорт

                Если Значение = Неопределено Тогда
                  Возврат;
                КонецЕсли;

                ДобавитьДокументПриходаНаСервере(Значение);
                ЗаполнитьДеревоДокументов();

КонецПроцедуры

&НаКлиенте
Процедура ДобавитьДокументПоступления(Команда)

                ПараметрыФормы  = Новый Структура("РежимВыбора, ЗакрыватьПриВыборе", Истина, Истина);
                ОбработкаВыбора = Новый ОписаниеОповещения("ВыбратьПриобретениеОкончание", ЭтаФорма);

                ОткрытьФорму("Документ.ПриобретениеТоваровУслуг.ФормаВыбора", ПараметрыФормы,
                Элементы.ДеревоДокументов,,,,ОбработкаВыбора,РежимОткрытияОкнаФормы.БлокироватьОкноВладельца);

КонецПроцедуры
                &НаСервере
                Процедура ОпределитьПересортицу(ОбъектЗаполнения, ЕстьПересортица)
                УстановитьПривилегированныйРежим(Истина);

                ИсточникДанных = ОбъектЗаполнения.Товары.Выгрузить();

                Запрос = Новый Запрос;
                Запрос.МенеджерВременныхТаблиц = Новый МенеджерВременныхТаблиц;
                Запрос.Текст = "ВЫБРАТЬ * ПОМЕСТИТЬ втТовары ИЗ &ИсточникДанных КАК ИсточникДанных";
                Запрос.УстановитьПараметр("ИсточникДанных", ИсточникДанных);
                Запрос.Выполнить();

                // Массив действий, которые можно исправить
                мДопустимыеДействия = Новый Массив;
                мДопустимыеДействия.Добавить(Перечисления.ВариантыДействийПоРасхождениямВАктеПослеПриемки.ПустаяСсылк
а());
                мДопустимыеДействия.Добавить(Перечисления.ВариантыДействийПоРасхождениямВАктеПослеПриемки.ОтнестиНедо
стачуНаПрочиеРасходы);
                мДопустимыеДействия.Добавить(Перечисления.ВариантыДействийПоРасхождениямВАктеПослеПриемки.ОтнестиПере
поставленноеНаПрочиеДоходы);

                Запрос.УстановитьПараметр("ДопустимыеДействия", мДопустимыеДействия);
                Запрос.Текст =
                "ВЫБРАТЬ
                | *,
                | 0 КАК Порядок
                |ИЗ
                | Документ.АктОРасхожденияхПослеПриемки.Товары КАК втТовары
                |ГДЕ
                | втТовары.КоличествоУпаковок = втТовары.КоличествоУпаковокПоДокументу
                |;
                |
                |////////////////////////////////////////////////////////////////////////////////
                |ВЫБРАТЬ
                | *,
                | 0 КАК Порядок,
                | втТовары.КоличествоУпаковок - втТовары.КоличествоУпаковокПоДокументу КАК КоличествоИзлишки
                |ИЗ
                | Документ.АктОРасхожденияхПослеПриемки.Товары КАК втТовары
                |ГДЕ
                | втТовары.КоличествоУпаковок > втТовары.КоличествоУпаковокПоДокументу
                |;
                |
                |////////////////////////////////////////////////////////////////////////////////
```

```
|ВЫБРАТЬ
| *,
| 0 КАК Порядок,
| втТовары.КоличествоУпаковокПоДокументу - втТовары.КоличествоУпаковок КАК КоличествоНедостача
|ИЗ
| Документ.АктОРасхожденияхПослеПриемки.Товары КАК втТовары
|ГДЕ
| втТовары.КоличествоУпаковокПоДокументу > втТовары.КоличествоУпаковок";

Запрос.Текст = СтрЗаменить(Запрос.Текст, "Документ.АктОРасхожденияхПослеПриемки.Товары КАК втТовары",
"втТовары КАК втТовары");

МассивРезультатов = Запрос.ВыполнитьПакет();

КолонкиГруппировки  = Новый Массив;
КолонкиГруппировки.Добавить("Порядок");
КолонкиГруппировки.Добавить("НомерСтроки");
КолонкиГруппировки.Добавить("НоменклатураПартнера");
КолонкиГруппировки.Добавить("Номенклатура");
КолонкиГруппировки.Добавить("Характеристика");
КолонкиГруппировки.Добавить("Назначение");
КолонкиГруппировки.Добавить("Упаковка");
КолонкиГруппировки.Добавить("ВидЦеныПоставщика");
КолонкиГруппировки.Добавить("Цена");
КолонкиГруппировки.Добавить("ЗаказПоставщику");
КолонкиГруппировки.Добавить("КодСтроки");
КолонкиГруппировки.Добавить("Склад");
КолонкиГруппировки.Добавить("Серия");
КолонкиГруппировки.Добавить("СтатусУказанияСерий");
КолонкиГруппировки.Добавить("ДокументОснование");
КолонкиГруппировки.Добавить("ЗаполненоПоОснованию");
КолонкиГруппировки.Добавить("НомерПаспорта");
КолонкиГруппировки.Добавить("Действие");
КолонкиГруппировки.Добавить("КомментарийПоставщика");
КолонкиГруппировки.Добавить("КомментарийМенеджера");
КолонкиГруппировки.Добавить("ДокументРеализации");
КолонкиГруппировки.Добавить("ПоВинеСтороннейКомпании");
КолонкиГруппировки.Добавить("СписатьНаРасходы");
КолонкиГруппировки.Добавить("СтатьяРасходов");
КолонкиГруппировки.Добавить("АналитикаРасходов");
КолонкиГруппировки.Добавить("Подразделение");
КолонкиГруппировки.Добавить("Сделка");
КолонкиГруппировки.Добавить("НомерГТД");
КолонкиГруппировки.Добавить("Сертификат");
КолонкиГруппировки.Добавить("Пересортица");
КолонкиГруппировки.Добавить("СтавкаНДС");
КолонкиСуммирования = Новый Массив;
КолонкиСуммирования.Добавить("КоличествоУпаковок");
КолонкиСуммирования.Добавить("Количество");
КолонкиСуммирования.Добавить("Сумма");
КолонкиСуммирования.Добавить("СуммаНДС");
КолонкиСуммирования.Добавить("СуммаСНДС");
КолонкиСуммирования.Добавить("КоличествоУпаковокПоДокументу");
КолонкиСуммирования.Добавить("КоличествоПоДокументу");
КолонкиСуммирования.Добавить("СуммаПоДокументу");
КолонкиСуммирования.Добавить("СуммаНДСПоДокументу");
КолонкиСуммирования.Добавить("СуммаСНДСПоДокументу");

ТаблТовары                  = МассивРезультатов[0].Выгрузить();
ТаблИзлишки     = МассивРезультатов[1].Выгрузить();
ТаблНедостача   = МассивРезультатов[2].Выгрузить();

//Сообщить(ТаблТовары.Количество()+ТаблИзлишки.Количество()+ТаблНедостача.Количество());
//Если Отказ Тогда
//            Возврат;
//КонецЕсли;

Для Каждого СтрокаНедостача Из ТаблНедостача Цикл
  ВариантыПоиска = Новый Массив;
  ВариантыПоиска.Добавить(Новый Структура("Номенклатура,Цена",
СтрокаНедостача.Номенклатура,СтрокаНедостача.Цена));
  ВариантыПоиска.Добавить(Новый Структура("Номенклатура,Цена", СтрокаНедостача.Номенклатура,0));
  ВариантыПоиска.Добавить(Новый Структура("Номенклатура",            СтрокаНедостача.Номенклатура));

  НомерВарианта = 0;
  Пока СтрокаНедостача.КоличествоНедостача > 0 И НомерВарианта < ВариантыПоиска.Количество() Цикл
```

```
                    СтруктураПоиска = ВариантыПоиска[НомерВарианта];
                    НомерВарианта = НомерВарианта + 1;

                    мИзлишки = ТаблИзлишки.НайтиСтроки(СтруктураПоиска);
                    Индекс = 0;
                    Пока СтрокаНедостача.КоличествоНедостача > 0 И Индекс < мИзлишки.Количество() Цикл
                        СтрокаИзлишки = мИзлишки[Индекс];
                        Индекс = Индекс + 1;
                        Если СтрокаИзлишки.КоличествоИзлишки > 0 Тогда

                            Если СтрокаИзлишки.Цена = 0 Тогда
                                СтрокаИзлишки.Цена = СтрокаНедостача.Цена;

                            СтрокаИзлишки.Сумма                 = СтрокаИзлишки.КоличествоУпаковок *
СтрокаНедостача.СуммаПоДокументу  / СтрокаНедостача.КоличествоУпаковокПоДокументу;
                            СтрокаИзлишки.СуммаНДС              = СтрокаИзлишки.КоличествоУпаковок *
СтрокаНедостача.СуммаНДСПоДокументу  / СтрокаНедостача.КоличествоУпаковокПоДокументу;
                            СтрокаИзлишки.СуммаСНДС             = СтрокаИзлишки.КоличествоУпаковок *
СтрокаНедостача.СуммаСНДСПоДокументу / СтрокаНедостача.КоличествоУпаковокПоДокументу;

                            СтрокаИзлишки.СуммаПоДокументу      =
СтрокаИзлишки.КоличествоУпаковокПоДокументу * СтрокаНедостача.СуммаПоДокументу   /
СтрокаНедостача.КоличествоУпаковокПоДокументу;
                            СтрокаИзлишки.СуммаНДСПоДокументу   =
СтрокаИзлишки.КоличествоУпаковокПоДокументу * СтрокаНедостача.СуммаНДСПоДокументу  /
СтрокаНедостача.КоличествоУпаковокПоДокументу;
                            СтрокаИзлишки.СуммаСНДСПоДокументу  =
СтрокаИзлишки.КоличествоУпаковокПоДокументу * СтрокаНедостача.СуммаСНДСПоДокументу /
СтрокаНедостача.КоличествоУпаковокПоДокументу;
                            КонецЕсли;

                            КоличествоПересорт =
Мин(СтрокаНедостача.КоличествоНедостача,СтрокаИзлишки.КоличествоИзлишки);

                            НоваяСтрока = ТаблТовары.Добавить();
                            ЗаполнитьЗначенияСвойств(НоваяСтрока, СтрокаНедостача);

                            //Недостача.КоличествоУпаковокПоДокументу > Недостача.КоличествоУпаковок
                            НоваяСтрока.Порядок                 = 1;
                            НоваяСтрока.Действие                =
ПредопределенноеЗначение("Перечисление.ВариантыДействийПоРасхождениямВАктеПослеПриемки.ОжидатьДопоставкуБезОформле
ния");
                            НоваяСтрока.СписатьНаРасходы        = Ложь;
                            НоваяСтрока.ПоВинеСтороннейКомпании  = Ложь;
                            НоваяСтрока.КомментарийМенеджера = "Пересортица";
                            НоваяСтрока.Пересортица             = Истина;

                            НоваяСтрока.КоличествоУпаковокПоДокументу  = КоличествоПересорт;
// НоваяСтрока.КоличествоПоДокументу                = НоваяСтрока.КоличествоПоДокументу;
                            НоваяСтрока.КоличествоПоДокументу   =
НоваяСтрока.КоличествоУпаковокПоДокументу;

                            НоваяСтрока.СуммаПоДокументу                =
СтрокаНедостача.СуммаПоДокументу      *КоличествоПересорт/СтрокаНедостача.КоличествоУпаковокПоДокументу;
                            НоваяСтрока.СуммаНДСПоДокументу             =
СтрокаНедостача.СуммаНДСПоДокументу   *КоличествоПересорт/СтрокаНедостача.КоличествоУпаковокПоДокументу;
                            НоваяСтрока.СуммаСНДСПоДокументу            =
СтрокаНедостача.СуммаСНДСПоДокументу  *КоличествоПересорт/СтрокаНедостача.КоличествоУпаковокПоДокументу;

                            НоваяСтрока.КоличествоУпаковок  = 0;
                            НоваяСтрока.Количество          = 0;
                            НоваяСтрока.Сумма               = 0;
                            НоваяСтрока.СуммаНДС            = 0;
                            НоваяСтрока.СуммаСНДС           = 0;

                            СтрокаНедостача.КоличествоУпаковокПоДокументу  =
СтрокаНедостача.КоличествоУпаковокПоДокументу - КоличествоПересорт;
                            СтрокаНедостача.КоличествоПоДокументу       =
СтрокаНедостача.КоличествоУпаковокПоДокументу;
                            СтрокаНедостача.СуммаПоДокументу            =
СтрокаНедостача.СуммаПоДокументу      - НоваяСтрока.СуммаПоДокументу;
                            СтрокаНедостача.СуммаНДСПоДокументу         =
СтрокаНедостача.СуммаНДСПоДокументу   - НоваяСтрока.СуммаНДСПоДокументу;
                            СтрокаНедостача.СуммаСНДСПоДокументу        =
СтрокаНедостача.СуммаСНДСПоДокументу  - НоваяСтрока.СуммаСНДСПоДокументу;
```

```
                                    СтрокаНедостача.КоличествоНедостача = СтрокаНедостача.КоличествоНедостача -
КоличествоПересорт;


                                    //Излишки.КоличествоУпаковокПоДокументу < Излишки.КоличествоУпаковок
                                    НоваяСтрока = ТаблТовары.Добавить();
                                    ЗаполнитьЗначенияСвойств(НоваяСтрока, СтрокаИзлишки);

                                    НоваяСтрока.Порядок                 = 1;
                                    НоваяСтрока.Действие                =
ПредопределенноеЗначение("Перечисление.ВариантыДействийПоРасхождениямВАктеПослеПриемки.ВернутьПерепоставленноеБезО
формления");
                                    НоваяСтрока.СписатьНаРасходы        = Ложь;
                                    НоваяСтрока.ПоВинеСтороннейКомпании = Ложь;
                                    НоваяСтрока.КомментарийМенеджера = "Пересортица";
                                    НоваяСтрока.Пересортица             = Истина;
                                    ЕстьПересортица = Истина;


                                    НоваяСтрока.КоличествоУпаковокПоДокументу = 0;
                                    НоваяСтрока.КоличествоПоДокументу         = 0;
                                    НоваяСтрока.СуммаПоДокументу              = 0;
                                    НоваяСтрока.СуммаНДСПоДокументу           = 0;
                                    НоваяСтрока.СуммаСНДСПоДокументу          = 0;

                                    НоваяСтрока.КоличествоУпаковок  = КоличествоПересорт;
                                    НоваяСтрока.Количество          = НоваяСтрока.КоличествоУпаковок;
                                    НоваяСтрока.Сумма               = СтрокаИзлишки.Сумма      *
КоличествоПересорт / СтрокаИзлишки.КоличествоУпаковок;
                                    НоваяСтрока.СуммаНДС            = СтрокаИзлишки.СуммаНДС  *
КоличествоПересорт / СтрокаИзлишки.КоличествоУпаковок;
                                    НоваяСтрока.СуммаСНДС           = СтрокаИзлишки.СуммаСНДС *
КоличествоПересорт/ СтрокаИзлишки.КоличествоУпаковок;

                                    СтрокаИзлишки.КоличествоУпаковок= СтрокаИзлишки.КоличествоУпаковок -
КоличествоПересорт;

            // СтрокаИзлишки.Количество     = СтрокаИзлишки.Количество;
            СтрокаИзлишки.Количество        = СтрокаИзлишки.КоличествоУпаковок;

                                    СтрокаИзлишки.Сумма             = СтрокаИзлишки.Сумма          -
НоваяСтрока.Сумма;
                                    СтрокаИзлишки.СуммаНДС          = СтрокаИзлишки.СуммаНДС  -
НоваяСтрока.СуммаНДС;
                                    СтрокаИзлишки.СуммаСНДС         = СтрокаИзлишки.СуммаСНДС      -
НоваяСтрока.СуммаСНДС;

                                    СтрокаИзлишки.КоличествоИзлишки = СтрокаИзлишки.КоличествоИзлишки -
КоличествоПересорт;

                        КонецЕсли;
                    КонецЦикла;
                КонецЦикла;

            КонецЦикла;

            Для Каждого СтрокаНедостача Из ТаблНедостача Цикл
              Если СтрокаНедостача.КоличествоУпаковок <> 0 ИЛИ СтрокаНедостача.КоличествоУпаковокПоДокументу <> 0 Тогда
                    ЗаполнитьЗначенияСвойств(ТаблТовары.Добавить(), СтрокаНедостача);
              КонецЕсли;
            КонецЦикла;

            Для Каждого СтрокаИзлишки Из ТаблИзлишки Цикл
              Если СтрокаИзлишки.КоличествоУпаковок <> 0 ИЛИ СтрокаИзлишки.КоличествоУпаковокПоДокументу <> 0 Тогда
                    ЗаполнитьЗначенияСвойств(ТаблТовары.Добавить(), СтрокаИзлишки);
              КонецЕсли;
            КонецЦикла;

            ТаблТовары.Свернуть(СтрСоединить(КолонкиГруппировки,","), СтрСоединить(КолонкиСуммирования,","));
            ТаблТовары.Сортировать("НомерСтроки ВОЗР, Порядок ВОЗР");


            ОбъектЗаполнения.Товары.Очистить();
            Для Каждого СтрокаТовары Из ТаблТовары Цикл
              НоваяСтрока = ОбъектЗаполнения.Товары.Добавить();
              ЗаполнитьЗначенияСвойств(НоваяСтрока, СтрокаТовары);
```

```
                ОбновитьСтрокуТабличнойЧасти(НоваяСтрока);
            КонецЦикла;

КонецПроцедуры

&НаСервере
Процедура ОбновитьСтрокуТабличнойЧасти(ЭлементКоллекции)

            СтруктураЭлемента = Новый Структура;

            СтруктураЭлемента.Вставить("ЕстьКомментарийМенеджера",  НЕ
ПустаяСтрока(ЭлементКоллекции.КомментарийМенеджера));
            СтруктураЭлемента.Вставить("ЕстьРасхождения",                НЕ(ЭлементКоллекции.КоличествоУпаковок =
ЭлементКоллекции.КоличествоУпаковокПоДокументу));

            СтруктураЭлемента.Вставить("СуммаРасхождения",          ЭлементКоллекции.Сумма          -
ЭлементКоллекции.СуммаПоДокументу);
            СтруктураЭлемента.Вставить("СуммаНДСРасхождения",       ЭлементКоллекции.СуммаНДС       -
ЭлементКоллекции.СуммаНДСПоДокументу);
            СтруктураЭлемента.Вставить("СуммаСНДСРасхождения",      ЭлементКоллекции.СуммаСНДС      -
ЭлементКоллекции.СуммаСНДСПоДокументу);

            СтруктураЭлемента.Вставить("КоличествоУпаковокРасхождения", ЭлементКоллекции.КоличествоУпаковок -
ЭлементКоллекции.КоличествоУпаковокПоДокументу);

            СтруктураЭлемента.Вставить("Артикул",
            ЭлементКоллекции.Номенклатура.Артикул);
            СтруктураЭлемента.Вставить("ТипНоменклатуры",
            ЭлементКоллекции.Номенклатура.ТипНоменклатуры);
            СтруктураЭлемента.Вставить("ХарактеристикиИспользуются",ЭлементКоллекции.Номенклатура.ВидНоменклатуры.Исп
ользоватьХарактеристики);
            СтруктураЭлемента.Вставить("ВедетсяУчетПоГТД",
            ЭлементКоллекции.Номенклатура.ВестиУчетПоГТД);
            СтруктураЭлемента.Вставить("СтранаПроисхождения",
            ЭлементКоллекции.НомерГТД.СтранаПроисхождения);

            ЗаполнитьЗначенияСвойств(ЭлементКоллекции, СтруктураЭлемента);

            Если ЭлементКоллекции.ЗаполненоПоОснованию И НЕ
ЗначениеЗаполнено(ЭлементКоллекции.КоличествоУпаковокПоДокументу) Тогда
                ЭлементКоллекции.ЗаполненоПоОснованию = Ложь;
            КонецЕсли;
            //СтруктураПересчетаСуммы =
ОбработкаТабличнойЧастиКлиентСервер.ПараметрыПересчетаСуммыНДСВСтрокеТЧ(Форма.Объект);

            //СтруктураДействий = Новый Структура;
            //СтруктураДействий.Вставить("ПересчитатьСуммуНДС", СтруктураПересчетаСуммы);
            //СтруктураДействий.Вставить("ПересчитатьСуммуСНДС", СтруктураПересчетаСуммы);
            //СтруктураДействий.Вставить("ЗаполнитьСтавкуНДС");
            //
            //ОбработкаТабличнойЧастиКлиент.ОбработатьСтрокуТЧ(ЭлементКоллекции, СтруктураДействий, Неопределено);
КонецПроцедуры

&НаСервере
Функция ТекстКомментария(Основание)
            Перем Результат;

            Запрос = Новый Запрос;
            Запрос.УстановитьПараметр("Ссылка", Основание);
            Запрос.Текст =
            "ВЫБРАТЬ РАЗЛИЧНЫЕ
            | АктОРасхожденияхПослеПриемкиТовары.ДокументОснование.ПроектПоставки КАК ПроектПоставки
            |ИЗ
            | Документ.АктОРасхожденияхПослеПриемки.Товары КАК АктОРасхожденияхПослеПриемкиТовары
            |ГДЕ
            | АктОРасхожденияхПослеПриемкиТовары.Ссылка = &Ссылка
            | И АктОРасхожденияхПослеПриемкиТовары.ДокументОснование ССЫЛКА Документ.ПриобретениеТоваровУслуг
            | И НЕ АктОРасхожденияхПослеПриемкиТовары.ДокументОснование =
ЗНАЧЕНИЕ(Документ.ПриобретениеТоваровУслуг.ПустаяСсылка)
            | И НЕ АктОРасхожденияхПослеПриемкиТовары.ДокументОснование.ПроектПоставки =
ЗНАЧЕНИЕ(Документ.ПроектПоставки.ПустаяСсылка)";
            РезЗапроса = Запрос.Выполнить();
            Если РезЗапроса.Пустой() Тогда
              Возврат "";
            КонецЕсли;
```

```
                Массив = РезЗапроса.Выгрузить().ВыгрузитьКолонку("ПроектПоставки");
                Возврат "Создан автоматически из "+?(Массив.Количество()=1, "документа ", "документов ") +

                  //СтроковыеФункцииКлиентСервер.СтрокаИзМассиваПодстрок(Массив, ",", Истина);
                        СтрСоединить(Массив, ",");
КонецФункции

&НаСервере
Процедура ЗаблокироватьРеквизитыФормыПоДатеЗапрета()

                ПропускаемыеРеквизиты = Документы.ПроектПоставки.РеквизитыДоступныеВЗакрытомПериоде();

                Для Каждого ЭлементФормы Из Элементы Цикл

                  Если ПропускаемыеРеквизиты.Найти(ЭлементФормы.Имя) <> Неопределено Тогда
                        продолжить;
                  КонецЕсли;

                  ТипЭлемента = ТипЗнч(ЭлементФормы);

                  Если ТипЭлемента = Тип("ПолеФормы") Тогда

                        Если Не ЭтоЭлементТаблицыФормы(ЭлементФормы.Родитель)
                          И Не (ЭлементФормы.Вид = ВидПоляФормы.ПолеНадписи И ЭлементФормы.Гиперссылка)
                        Тогда
                              ЭлементФормы.Доступность = Ложь;
                        КонецЕсли;

                  ИначеЕсли ТипЭлемента = Тип("КнопкаФормы") Тогда

                        Команда = ЭтотОбъект.Команды.Найти(ЭлементФормы.ИмяКоманды);

                        Если Команда <> Неопределено Тогда
                              ИзменяетДанные = Команда.ИзменяетСохраняемыеДанные;
                        Иначе
                              ИзменяетДанные = Ложь;
                        КонецЕсли;

                        ЭлементФормы.Доступность = Не ИзменяетДанные;

                  ИначеЕсли ТипЭлемента = Тип("ТаблицаФормы") Тогда

                        ЭлементФормы.ТолькоПросмотр = Истина;

                  КонецЕсли;

                КонецЦикла;

КонецПроцедуры

Функция ЭтоЭлементТаблицыФормы(РодительЭлемента)

                Если ТипЗнч(РодительЭлемента) = Тип("УправляемаяФорма") Тогда
                  возврат Ложь;
                ИначеЕсли ТипЗнч(РодительЭлемента) = Тип("ТаблицаФормы") Тогда
                  возврат Истина;
                Иначе
                  возврат ЭтоЭлементТаблицыФормы(РодительЭлемента.Родитель);
                КонецЕсли;

КонецФункции

&НаСервере
Процедура ПересчитатьЗЦНаСервере()

                Запрос = Новый Запрос;
                Запрос.Текст =
                        "ВЫБРАТЬ
                        |       ТоварыПроектовПоставки.ПроектПоставки КАК ПроектПоставки,
                        |       ТоварыПроектовПоставки.Номенклатура КАК Номенклатура,
                        |       ТоварыПроектовПоставки.Характеристика КАК Характеристика,
                        |       ЕСТЬNULL(ЦеныНоменклатурыСрезПоследних.Цена,0) КАК ЦенаРУБ
                        |ИЗ
                        |       РегистрСведений.ТоварыПроектовПоставки КАК ТоварыПроектовПоставки
                        |               ЛЕВОЕ СОЕДИНЕНИЕ РегистрСведений.ЦеныНоменклатуры.СрезПоследних(&Дата,
ВидЦены = &ВидЦены) КАК ЦеныНоменклатурыСрезПоследних
```

```
                        |                    ПО ТоварыПроектовПоставки.Номенклатура =
ЦеныНоменклатурыСрезПоследних.Номенклатура
                        |                        И ТоварыПроектовПоставки.Характеристика =
ЦеныНоменклатурыСрезПоследних.Характеристика
                        |ГДЕ
                        |        ТоварыПроектовПоставки.ПроектПоставки = &ПроектПоставки";

        Запрос.УстановитьПараметр("ВидЦены", Справочники.ВидыЦен.ЗакупочнаяЦена);
        Запрос.УстановитьПараметр("Дата", ТекущаяДата());
        Запрос.УстановитьПараметр("ПроектПоставки", Объект.Ссылка);

        РезультатЗапроса = Запрос.Выполнить().Выгрузить();

        НаборЗаписей = РегистрыСведений.ТоварыПроектовПоставки.СоздатьНаборЗаписей();
        НаборЗаписей.Отбор.ПроектПоставки.Установить(Объект.Ссылка);
        НаборЗаписей.Прочитать();

        СтруктураПоиска = Новый Структура;
        СтруктураПоиска.Вставить("Номенклатура");
        СтруктураПоиска.Вставить("Характеристика");

        Для каждого Запись Из НаборЗаписей Цикл
          СтруктураПоиска.Номенклатура = Запись.Номенклатура;
          СтруктураПоиска.Характеристика = Запись.Характеристика;
          МассивСтрок = РезультатЗапроса.НайтиСтроки(СтруктураПоиска);

          Если МассивСтрок.Количество()>0 Тогда

                ЗакупочнаяЦенаРуб=МассивСтрок[0].ЦенаРуб;
                Если ЗакупочнаяЦенаРуб>0 Тогда
                        Запись.ЗакупочнаяЦена = ЗакупочнаяЦенаРуб;
                        Запись.СуммаЗакупочная = Запись.Количество*ЗакупочнаяЦенаРуб;
                        Запись.Наценка = (Запись.РозничнаяЦена/ЗакупочнаяЦенаРуб-1)*100;
                иначе
                        Сообщить("Обнаружена нулевая цена по артикулу "+Запись.Номенклатура.Артикул+" в проекте
поставки");
                КонецЕсли;
          КонецЕсли;

        КонецЦикла;

        НаборЗаписей.Записать(Истина);

КонецПроцедуры

&НаКлиенте
Процедура ПересчитатьЗЦ(Команда)
        ПересчитатьЗЦНаСервере();
КонецПроцедуры

&НаСервере
Процедура ОтразитьСписанияВАктеОРасхожденияхАктОРасхожденияхОбъект, ДокументПриобретения)

        Запрос = Новый Запрос;
        Запрос.УстановитьПараметр("ДокументПриобретения", ДокументПриобретения);
        Запрос.Текст =
        "ВЫБРАТЬ
        | СписаниеНедостачТоваровТовары.Номенклатура КАК Номенклатура,
        | СписаниеНедостачТоваровТовары.Характеристика КАК Характеристика,
        | СписаниеНедостачТоваровТовары.Ссылка.Основание КАК ДокументОснование,
        | СУММА(СписаниеНедостачТоваровТовары.Количество) КАК Количество
        |ИЗ
        | Документ.СписаниеНедостачТоваров.Товары КАК СписаниеНедостачТоваровТовары
        |ГДЕ
        | СписаниеНедостачТоваровТовары.Ссылка.Проведен
        | И СписаниеНедостачТоваровТовары.Ссылка.Основание = &ДокументПриобретения
        | И СписаниеНедостачТоваровТовары.Ссылка.ТоварыВПути
        |
        |СГРУППИРОВАТЬ ПО
        | СписаниеНедостачТоваровТовары.Номенклатура,
        | СписаниеНедостачТоваровТовары.Характеристика,
        | СписаниеНедостачТоваровТовары.Ссылка.Основание";
        РезЗапроса = Запрос.Выполнить();
        Если РезЗапроса.Пустой() Тогда
          Возврат;
        КонецЕсли;
```

```
		ТаблицаТовары 		= АктОРасхожденияхОбъект.Товары.Выгрузить();
		ТаблицаСписанныеТовары 	= РезЗапроса.Выгрузить();

		Для Каждого СтрокаСписанныеТовары Из ТаблицаСписанныеТовары Цикл
		 ПараметрыОтбора = Новый Структура;
		 ПараметрыОтбора.Вставить("Номенклатура", 		СтрокаСписанныеТовары.Номенклатура);
		 ПараметрыОтбора.Вставить("Характеристика", 		СтрокаСписанныеТовары.Характеристика);
		 ПараметрыОтбора.Вставить("ДокументОснование", 	СтрокаСписанныеТовары.ДокументОснование);

		 МассивСтрок = ТаблицаТовары.НайтиСтроки(ПараметрыОтбора);
		 Индекс = 0;
		 Пока Индекс < МассивСтрок.Количество() И СтрокаСписанныеТовары.Количество > 0 Цикл
			СтрокаТовары = МассивСтрок[Индекс];
			Индекс = Индекс + 1;

			Если СтрокаТовары.КоличествоПоДокументу > 0 Тогда

				СписанноеКоличествоПоДокументу 		= Мин(СтрокаТовары.КоличествоПоДокументу,
СтрокаСписанныеТовары.Количество);
				КоэффициентПересчетаСуммыПоДокументу = 1 - СписанноеКоличествоПоДокументу /
СтрокаСписанныеТовары.Количество;

				СтрокаТовары.СуммаПоДокументу 		= Мин(СтрокаТовары.СуммаПоДокументу,
		КоэффициентПересчетаСуммыПоДокументу * СтрокаТовары.СуммаПоДокументу);
				СтрокаТовары.СуммаНДСПоДокументу 	= Мин(СтрокаТовары.СуммаНДСПоДокументу,
		КоэффициентПересчетаСуммыПоДокументу * СтрокаТовары.СуммаНДСПоДокументу);
				СтрокаТовары.СуммаСНДСПоДокументу 	= Мин(СтрокаТовары.СуммаПоДокументу,
		КоэффициентПересчетаСуммыПоДокументу * СтрокаТовары.СуммаСНДСПоДокументу);
				СтрокаТовары.КоличествоПоДокументу 	= СтрокаТовары.КоличествоПоДокументу -
СписанноеКоличествоПоДокументу;

				СтрокаТовары.Сумма 		= СтрокаТовары.СуммаПоДокументу;
				СтрокаТовары.СуммаНДС 		= СтрокаТовары.СуммаНДСПоДокументу;
				СтрокаТовары.СуммаСНДС 		= СтрокаТовары.СуммаСНДСПоДокументу;
				СтрокаТовары.Количество = СтрокаТовары.КоличествоПоДокументу;

				СтрокаСписанныеТовары.Количество = СтрокаСписанныеТовары.Количество -
СписанноеКоличествоПоДокументу;

			КонецЕсли;
		 КонецЦикла;

		КонецЦикла;

		АктОРасхожденияхОбъект.Товары.Загрузить(ТаблицаТовары);

		//////Запрос = Новый Запрос;
		//////Запрос.МенеджерВременныхТаблиц = Новый МенеджерВременныхТаблиц;
		//////Запрос.Текст = "ВЫБРАТЬ * ПОМЕСТИТЬ втТовары ИЗ &ИсточникДанных КАК ИсточникДанных";
		//////Запрос.УстановитьПараметр("ИсточникДанных", ТаблицаТовары);
		//////Запрос.Выполнить();
		//////
		//////Запрос.Текст =
		//////"ВЫБРАТЬ
		//////|	*
		//////|ИЗ
		//////|	втТовары КАК втТовары
		//////|ГДЕ
		//////|	втТовары.КоличествоПоДокументу > 0";
		//////ТаблицаТовары = Запрос.Выполнить().Выгрузить();
		//////
		//////АктОРасхожденияхОбъект.Товары.Загрузить(ТаблицаТовары);

КонецПроцедуры

&НаСервере
Процедура УдалитьПустыеСтрокиВАктеОРасхождениях(АктОРасхожденияхОбъект)

		Запрос = Новый Запрос;
		Запрос.МенеджерВременныхТаблиц = Новый МенеджерВременныхТаблиц;
		Запрос.Текст = "ВЫБРАТЬ * ПОМЕСТИТЬ втТовары ИЗ &ИсточникДанных КАК ИсточникДанных";
		Запрос.УстановитьПараметр("ИсточникДанных", АктОРасхожденияхОбъект.Товары.Выгрузить());
		Запрос.Выполнить();

		Запрос.Текст =
		"ВЫБРАТЬ
```

```
                | *
                |ИЗ
                | втТовары КАК втТовары
                |ГДЕ
                | НЕ(втТовары.КоличествоПоДокументу = 0
                |                 И втТовары.Количество = 0)";

                ТаблицаТовары = Запрос.Выполнить().Выгрузить();

                Если ТаблицаТовары.Количество() <> АктОРасхожденияхОбъект.Товары.Количество() Тогда
                  АктОРасхожденияхОбъект.Товары.Загрузить(ТаблицаТовары);
                КонецЕсли;

КонецПроцедуры

#КонецОбласти

//Модуль объекта док.ПроектПоставки

#Область ОбработчикиСобытий

Процедура ПередЗаписью(Отказ, РежимЗаписи, РежимПроведения)

                Если ОбменДанными.Загрузка Тогда
                  Возврат;
                КонецЕсли;

                Если ЭтотОбъект.ПометкаУдаления И ОбщегоНазначения.ЗначениеРеквизитаОбъекта(ЭтотОбъект.Ссылка,
"ПометкаУдаления") = Ложь  Тогда
                  ПроверитьСвязанныеДокументы(Отказ);
                КонецЕсли;

                Если РежимЗаписи = РежимЗаписиДокумента.ОтменаПроведения Тогда
                  УправлениеСтатусамиОбработки.ПроверитьБлокировкуПоСтатусуОбработки(ЭтотОбъект, РежимЗаписи, Отказ);
                КонецЕсли;

                Если НЕ(ПометкаУдаления ИЛИ РежимЗаписи = РежимЗаписиДокумента.ОтменаПроведения) Тогда
                  СтатусДоговора = ОбщегоНазначения.ЗначениеРеквизитаОбъекта(Договор,"Статус");
                  Если НЕ СтатусДоговора = Предопределенное Значение("Перечисление.СтатусыДоговоровКонтрагентов.Действует")
Тогда
                        ТекстСообщения = СтрШаблон("Договор %1 %2",
                                                                                Договор,
                                                                                ?(ЗначениеЗаполнено(СтатусДоговора),
СтатусДоговора, Перечисления.СтатусыДоговоровКонтрагентов.НеСогласован)
                                                                                );
                        ОбщегоНазначения.СообщитьПользователю(ТекстСообщения,,,,Отказ);
                  КонецЕсли;
                КонецЕсли;


КонецПроцедуры

Процедура ОбработкаПроверкиЗаполнения(Отказ, ПроверяемыеРеквизиты)

                Если ОбщегоНазначенияУТ.ПроверитьЗаполнениеРеквизитовОбъекта(ЭтотОбъект, ПроверяемыеРеквизиты) Тогда
                  Отказ = Истина;
                КонецЕсли;

                ПроверяемыеРеквизиты.Добавить("Договор");
                ПроверяемыеРеквизиты.Добавить("ХозяйственнаяОперация");
                ПроверитьИспользованиеНомераПоставки(Отказ);

КонецПроцедуры

Процедура ОбработкаЗаполнения(ДанныеЗаполнения, СтандартнаяОбработка)

                ИнициализироватьДокумент(ДанныеЗаполнения);

                ЗаполнениеСвойствПоСтатистикеСервер.ЗаполнитьСвойстваОбъекта(ЭтотОбъект, ДанныеЗаполнения);

                ЗаполнитьСписокМагазинов();

КонецПроцедуры

#КонецОбласти
```

```
#Область СлужебныеПроцедурыИФункции

#Область ИнициализацияИЗаполнение

Процедура ИнициализироватьДокумент(ДанныеЗаполнения = Неопределено)

		Ответственный	= Пользователи.ТекущийПользователь();
		Организация	= ЗначениеНастроекПовтИсп.ПолучитьОрганизациюПоУмолчанию(Организация);
		Склад		= ЗначениеНастроекПовтИсп.ПолучитьСкладПоУмолчанию(Склад, Ложь, Истина);

		Если НЕ ЗначениеЗаполнено(ХозяйственнаяОперация) Тогда
		  ХозяйственнаяОперация =
ПредопределенноеЗначение("Перечисление.ХозяйственныеОперации.ЗакупкаУПоставщика");
		КонецЕсли;


КонецПроцедуры

Процедура ПриКопировании(ОбъектКопирования)

		ДокументУстановкиЦен = Неопределено;
		Ответственный = Неопределено;

		СуммаДокумента = 0;
		ДокументыПоступлений.Очистить();

		ЗаблокированСтатусомОбработки = Ложь;

		Приоритет = Перечисления.Приоритеты.Средний;
		ДатаПлановогоПоступления = '00010101';
		ПринудительноеРаспределение = Ложь;
		АлгоритмРаспределения = Перечисления.АлгоритмыРаспределения.АлгоритмАРП;
		РасхожденияСогласованыСПоставщиком = Ложь;
		Комментарий = "";

		ИнициализироватьДокумент();

КонецПроцедуры



// Процедура ЗаполнитьСписокМагазинов
//
Процедура ЗаполнитьСписокМагазинов()

		Запрос = Новый Запрос;
		Запрос.Текст =
		  "ВЫБРАТЬ
		  |	Склады.Ссылка КАК Магазин,
		  |	Склады.ИспользоватьРаспределение КАК Распределение
		  |ИЗ
		  |	Справочник.Склады КАК Склады
		  |ГДЕ
		  |	НЕ Склады.ПометкаУдаления
		  |	И НЕ Склады.ЭтоГруппа
		  |	И Склады.ИспользоватьРаспределение
		  |	И Склады.ТипСклада = ЗНАЧЕНИЕ(Перечисление.ТипыСкладов.РозничныйМагазин)
		  |
		  |УПОРЯДОЧИТЬ ПО
		  |	Магазин
		  |АВТОУПОРЯДОЧИВАНИЕ";

		СписокМагазинов.Загрузить(Запрос.Выполнить().Выгрузить());

КонецПроцедуры

#КонецОбласти

Процедура ПроверитьСвязанныеДокументы(Отказ)

		УстановитьПривилегированныйРежим(Истина);

		Запрос = Новый Запрос;
		Запрос.Текст =
		"ВЫБРАТЬ
		| СвязанныеДокументы.Ссылка
```

```
                |ИЗ
                | КритерийОтбора.СвязанныеДокументы(&Ссылка) КАК СвязанныеДокументы
                |ГДЕ
                | НЕ СвязанныеДокументы.Ссылка.ПометкаУдаления";

                Запрос.УстановитьПараметр("Ссылка", ЭтотОбъект.Ссылка);

                РезультатЗапроса = Запрос.Выполнить();
                Если РезультатЗапроса.Пустой() Тогда
                  Возврат;
                КонецЕсли;

                Список = РезультатЗапроса.Выгрузить().ВыгрузитьКолонку("Ссылка");

                Сообщение = "Невозможно пометить на удаление: " + ЭтотОбъект.Ссылка + "
                  |Так как не помечены к удалению связанные документы: " + СтрСоединить(Список, ", ");
                ОбщегоНазначенияКлиентСервер.СообщитьПользователю(Сообщение, , , , Отказ);

КонецПроцедуры

Процедура ПроверитьИспользованиеНомераПоставки(Отказ)

                Запрос = Новый Запрос;
                Запрос.Текст =
                "ВЫБРАТЬ ПЕРВЫЕ 1
                | ПроектПоставки.Ссылка КАК Ссылка
                |ИЗ
                | Документ.ПроектПоставки КАК ПроектПоставки
                |ГДЕ
                | ПроектПоставки.Партнер = &Поставщик
                | И ПроектПоставки.НомерПоставки = &НомерПоставки
                | И ПроектПоставки.Ссылка <> &ТекущийДокумент";

                Запрос.УстановитьПараметр("Поставщик"       , ЭтотОбъект.Партнер);
                Запрос.УстановитьПараметр("НомерПоставки"   , ЭтотОбъект.НомерПоставки);
                Запрос.УстановитьПараметр("ТекущийДокумент" , ЭтотОбъект.Ссылка);

                Результат = Запрос.Выполнить();

                Если Не Результат.Пустой() Тогда

                  Выборка = Результат.Выбрать();
                  Выборка.Следующий();

                  Текст = НСтр("ru = 'Номер поставки уже использовался. Укажите другой номер.'");
                  ОбщегоНазначенияКлиентСервер.СообщитьПользователю(
                            Текст,
                            ЭтотОбъект,
                            "НомерПоставки",
                            ,
                            Отказ);

                КонецЕсли;

КонецПроцедуры

#КонецОбласти

//Модуль менеджера Док.ПроектПоставки

#Если Сервер Или ТолстыйКлиентОбычноеПриложение Или ВнешнееСоединение Тогда

#Область ПрограммныйИнтерфейс

// СтандартныеПодсистемы.ВерсионированиеОбъектов

// Определяет настройки объекта для подсистемы ВерсионированиеОбъектов.
//
// Параметры:
//  Настройки - Структура - настройки подсистемы.
Процедура ПриОпределенииНастроекВерсионированияОбъектов(Настройки) Экспорт

КонецПроцедуры

// Конец СтандартныеПодсистемы.ВерсионированиеОбъектов
```

```
// Определяет список команд создания на основании.
//
// Параметры:
//   КомандыСозданияНаОсновании - ТаблицаЗначений - Таблица с командами создания на основании. Для изменения.
//       См. описание 1 параметра процедуры
СозданиеНаОснованииПереопределяемый.ПередДобавлениемКомандСозданияНаОсновании().
//   Параметры - Структура - Вспомогательные параметры. Для чтения.
//       См. описание 2 параметра процедуры
СозданиеНаОснованииПереопределяемый.ПередДобавлениемКомандСозданияНаОсновании().
//
Процедура ДобавитьКомандыСозданияНаОсновании(КомандыСозданияНаОсновании, Параметры) Экспорт

		Документы.ПриобретениеТоваровУслуг.ДобавитьКомандуСоздатьНаОсновании(КомандыСозданияНаОсновании);
		Документы.УстановкаЦенНоменклатуры.ДобавитьКомандуСоздатьНаОсновании(КомандыСозданияНаОсновании);

		Документы.Предрасчет.ДобавитьКомандуСоздатьНаОсновании(КомандыСозданияНаОсновании);

КонецПроцедуры

// Добавляет команду создания документа "Проект поставки".
//
// Параметры:
//   КомандыСозданияНаОсновании - ТаблицаЗначений - Таблица с командами создания на основании. Для изменения.
//       См. описание 1 параметра процедуры
СозданиеНаОснованииПереопределяемый.ПередДобавлениемКомандСозданияНаОсновании().
//
Функция ДобавитьКомандуСоздатьНаОсновании(КомандыСозданияНаОсновании) Экспорт
		Если ПравоДоступа("Добавление", Метаданные.Документы.ПроектПоставки) Тогда
		  КомандаСоздатьНаОсновании = КомандыСозданияНаОсновании.Добавить();
		  КомандаСоздатьНаОсновании.Менеджер = Метаданные.Документы.ПроектПоставки.ПолноеИмя();
		  КомандаСоздатьНаОсновании.Представление =
ОбщегоНазначенияУТ.ПредставлениеОбъекта(Метаданные.Документы.ПроектПоставки);
		  КомандаСоздатьНаОсновании.РежимЗаписи = "Проводить";

		  Возврат КомандаСоздатьНаОсновании;

		КонецЕсли;

		Возврат Неопределено;
КонецФункции

// Определяет список команд отчетов.
//
// Параметры:
//   КомандыОтчетов - ТаблицаЗначений - Таблица с командами отчетов. Для изменения.
//       См. описание 1 параметра процедуры ВариантыОтчетовПереопределяемый.ПередДобавлениемКомандОтчетов().
//   Параметры - Структура - Вспомогательные параметры. Для чтения.
//       См. описание 2 параметра процедуры ВариантыОтчетовПереопределяемый.ПередДобавлениемКомандОтчетов().
//
Процедура ДобавитьКомандыОтчетов(КомандыОтчетов, Параметры) Экспорт

		КомандаОтчет =
ВариантыОтчетовУТПереопределяемый.ДобавитьКомандуСтруктураПодчиненности(КомандыОтчетов);
		Если КомандаОтчет <> Неопределено Тогда

		КонецЕсли;

		КомандаОтчет =
ВариантыОтчетовУТПереопределяемый.ДобавитьКомандуПланФактныйАнализРасхожденийПоставки(КомандыОтчетов);
		Если КомандаОтчет <> Неопределено Тогда

		КонецЕсли;

		КомандаОтчет = ВариантыОтчетовУТПереопределяемый.ДобавитьКомандуФактПриемкиПоставки(КомандыОтчетов);
		Если КомандаОтчет <> Неопределено Тогда

		КонецЕсли;

КомандаОтчет = ВариантыОтчетовУТПереопределяемый.ДобавитьКомандуАнализРасчетаРаспределенияТоваров(КомандыОтчетов);
		Если КомандаОтчет <> Неопределено Тогда

		КонецЕсли;

		КомандаОтчет =
ВариантыОтчетовУТПереопределяемый.ДобавитьКомандуАнализПредрасчетаТоваров(КомандыОтчетов);
		Если КомандаОтчет <> Неопределено Тогда
```

```
		КонецЕсли;

КонецПроцедуры

Процедура УдалитьДвиженияИзРегистровРаспределения(ПроектПоставки) Экспорт

		УстановитьПривилегированныйРежим(Истина);

		НаборЗаписей = РегистрыНакопления.РаспределениеТоваров.СоздатьНаборЗаписей();
		НаборЗаписей.Отбор.Регистратор.Установить(ПроектПоставки);
		НаборЗаписей.Записать();

		НаборЗаписей = РегистрыСведений.ПринудительноеРаспределение.СоздатьНаборЗаписей();
		НаборЗаписей.Отбор.ПроектПоставки.Установить(ПроектПоставки);
		НаборЗаписей.Записать();

		УстановитьПривилегированныйРежим(Ложь);

КонецПроцедуры

Функция ПолучитьСтатусСогласования(ПроектПоставки) Экспорт

		Запрос = Новый Запрос;
		Запрос.Текст =
		"ВЫБРАТЬ
		| СтатусСогласованияДокументов.Согласовано КАК Согласовано
		|ИЗ
		| РегистрСведений.СтатусСогласованияДокументов КАК СтатусСогласованияДокументов
		|ГДЕ
		| СтатусСогласованияДокументов.Документ = &Документ";

		Запрос.УстановитьПараметр("Документ", ПроектПоставки);
		РезультатЗапроса = Запрос.Выполнить();

		Выборка = РезультатЗапроса.Выбрать();

		Если Выборка.Следующий() Тогда

		  Статус = Выборка.Согласовано;

		иначе

		  Статус = Ложь;

		КонецЕсли;

		Возврат Статус;

КонецФункции

Процедура УстановитьСтатусСогласования(ПроектПоставки, Статус) Экспорт

		Запись = РегистрыСведений.СтатусСогласованияДокументов.СоздатьМенеджерЗаписи();
		Запись.Документ = ПроектПоставки;

		Запись.Согласовано = Статус;
		Запись.Записать();

КонецПроцедуры

Функция РеквизитыДоступныеВЗакрытомПериоде() Экспорт

		ДоступныеРеквизиты = Новый Массив;
		ДоступныеРеквизиты.Добавить("ФормаЗаписатьИЗакрыть");
		ДоступныеРеквизиты.Добавить("ФормаЗаписать");
		ДоступныеРеквизиты.Добавить("ФормаРаспределить");
		ДоступныеРеквизиты.Добавить("ФормаОтменитьРаспределение");
		ДоступныеРеквизиты.Добавить("РезультатыРаспределенияЗаполнитьИтогиРаспределения");
		ДоступныеРеквизиты.Добавить("РезультатыРаспределенияВыгрузитьИтогиВExcel");
		ДоступныеРеквизиты.Добавить("РезультатыРаспределенияЗагрузитьИтогиИзExcel");
		ДоступныеРеквизиты.Добавить("ДатаПлановогоПоступления");
		ДоступныеРеквизиты.Добавить("АлгоритмРаспределения");
		ДоступныеРеквизиты.Добавить("ВозможностьБХ");
		ДоступныеРеквизиты.Добавить("ПринудительноеРаспределение");
		ДоступныеРеквизиты.Добавить("СписокМагазинов");
```

```
		возврат ДоступныеРеквизиты;

КонецФункции


#Область ЗагрузкаПроектаПоставки

Процедура ОбработатьЗагруженныеТовары(ТаблицаТоваров, Ошибки, ПараметрыОбработки) Экспорт

		КэшированныеЗначения = Новый Структура;
		КэшированныеЗначения.Вставить("СвободныеШтрихкода",
РегистрыСведений.СвободныеШтрихкодаШтучныхТоваров.ДобавитьСвободныеШтрихкода(ТаблицаТоваров.Количество()));
		КэшированныеЗначения.Вставить("Номенклатура", Новый Соответствие);
		КэшированныеЗначения.Вставить("Штрихкоды", Новый Соответствие);

		НачатьТранзакцию();

		//Блокируем справочник Номенклатура от изменений другими сеансами
		Блокировка = Новый БлокировкаДанных;

		ЭлементБлокировки = Блокировка.Добавить("Справочник.Номенклатура");
		ЭлементБлокировки.ИсточникДанных = ТаблицаТоваров;
		ЭлементБлокировки.УстановитьЗначение("Поставщик", ПараметрыОбработки.Поставщик);
		ЭлементБлокировки.ИспользоватьИзИсточникаДанных("Марка", "Марка");

		Блокировка.Заблокировать();

		ПолучитьДанныеНоменклатуры(ТаблицаТоваров, КэшированныеЗначения);

		//1. Поиск в базе подходящей номенклатуры
		// - для обычной номенклатуры поиск осуществляется по артикулу
		// - для импортной поиск по совокупности реквизитов

		Если ПараметрыОбработки.ИмпортнаяПоставка Тогда
		  ПодобратьСуществующуюИмпортнуюНоменклатуру(ТаблицаТоваров, ПараметрыОбработки.Поставщик,
КэшированныеЗначения);
		Иначе
		  ПодобратьСуществующуюНоменклатуру(ТаблицаТоваров, ПараметрыОбработки.Поставщик,
КэшированныеЗначения);
		КонецЕсли;

		//2. Создание номенклатуры и характеристик
		СоздатьНоменклатуруХарактеристики(ТаблицаТоваров, Ошибки, КэшированныеЗначения,
ПараметрыОбработки.КодПоставщика);

		Если Ошибки.СтрокиОшибок.Количество() > 0 Тогда
		  ОтменитьТранзакцию();
		  возврат;
		КонецЕсли;

		//3. Проверка штрикходов поставщиков

		//Штрихкода поставщиков используемые в качестве внутренних штрихкодов
		ТаблицаТоваров.Колонки.Добавить("МаркируетсяШтрихкодомПоставщика", Новый ОписаниеТипов("Булево"));

		СтрокиТоваровМаркируемыхПоставщиком = Новый Массив;
		Для Каждого СтрокаТовара Из ТаблицаТоваров Цикл

		  СтрокаТовара.МаркируетсяШтрихкодомПоставщика =
СтрокаТовара.ПараметрыУчета.МаркируетсяШтрихкодомПоставщика;

		  Если СтрокаТовара.МаркируетсяШтрихкодомПоставщика Тогда
			СтрокиТоваровМаркируемыхПоставщиком.Добавить(СтрокаТовара);
		  КонецЕсли;

		КонецЦикла;

		МассивВнутреннихШтриходов = ОбщегоНазначения.ВыгрузитьКолонку(СтрокиТоваровМаркируемыхПоставщиком,
"Штрихкод", Истина);

		Блокировка = Новый БлокировкаДанных;

		ЭлементБлокировки = Блокировка.Добавить("РегистрСведений.ШтрихкодыПоставщиковНоменклатуры");
		ЭлементБлокировки.ИсточникДанных = ТаблицаТоваров;
		ЭлементБлокировки.ИспользоватьИзИсточникаДанных("Номенклатура"   , "Номенклатура");
		ЭлементБлокировки.ИспользоватьИзИсточникаДанных("Характеристика" , "Характеристика");
```

```
		Если МассивВнутреннихШтриходов.Количество() > 0 Тогда

		  ТаблицаДляБлокировки = ТаблицаТоваров.Скопировать(СтрокиТоваровМаркируемыхПоставщиком, "Штрихкод");

		  ЭлементБлокировки = Блокировка.Добавить("РегистрСведений.ШтрихкодыНоменклатуры");
		  ЭлементБлокировки.ИсточникДанных = ТаблицаДляБлокировки;
		  ЭлементБлокировки.ИспользоватьИзИсточникаДанных("Штрихкод", "Штрихкод");

		КонецЕсли;

		Блокировка.Заблокировать();

		ПолучитьДанныеПоВнутреннимШтрихкодам(МассивВнутреннихШтриходов, КэшированныеЗначения);

		ПроверитьШтрихкодаПоставщиков(СтрокиТоваровМаркируемыхПоставщиком, Ошибки, КэшированныеЗначения);

		Если Ошибки.СтрокиОшибок.Количество() > 0 Тогда
		  ОтменитьТранзакцию();
		  возврат;
		КонецЕсли;

		//4. Создание штрихкодов
		СоздатьШтрихкода(ТаблицаТоваров, Ошибки, КэшированныеЗначения);

		//5. Освобождение неиспользованных свободных штрихкодов
		ОсвободитьСвободныеШтрихкода(КэшированныеЗначения);

		Если Ошибки.СтрокиОшибок.Количество() = 0 Тогда
		  ЗафиксироватьТранзакцию();
		Иначе
		  ОтменитьТранзакцию();
		КонецЕсли;

КонецПроцедуры

Процедура ОсвободитьСвободныеШтрихкода(КэшированныеЗначения)

		УстановитьПривилегированныйРежим(Истина);

		Для Каждого Штрихкод Из КэшированныеЗначения.СвободныеШтрихкода Цикл

		  МенеджерЗаписи = РегистрыСведений.СвободныеШтрихкодаШтучныхТоваров.СоздатьМенеджерЗаписи();
		  МенеджерЗаписи.Штрихкод = Штрихкод;
		  МенеджерЗаписи.Удалить();

		КонецЦикла;

КонецПроцедуры

Процедура ПодобратьСуществующуюИмпортнуюНоменклатуру(ТаблицаТоваров, Поставщик, КэшированныеЗначения) Экспорт

		Запрос = Новый Запрос;
		Запрос.Текст =
		"ВЫБРАТЬ РАЗЛИЧНЫЕ
		| ТаблицаТоваров.ИдентификаторТовара КАК ИдентификаторТовара,
		| &Поставщик КАК Поставщик,
		| ТаблицаТоваров.АртикулИсходный КАК АртикулИсходный,
		| ТаблицаТоваров.КодТНВЭД КАК КодТНВЭД,
		| ТаблицаТоваров.СтранаПроисхождения КАК СтранаПроисхождения
		|ПОМЕСТИТЬ ТаблицаТоваров
		|ИЗ
		| &ТаблицаТоваров КАК ТаблицаТоваров
		|;
		|
		|////////////////////////////////////////////////////////////////////////////////
		|ВЫБРАТЬ РАЗЛИЧНЫЕ
		| ТаблицаТоваров.ИдентификаторТовара КАК ИдентификаторТовара,
		| Номенклатура.Ссылка КАК Номенклатура,
		| Номенклатура.Артикул КАК Артикул,
		| Номенклатура.Наименование КАК Наименование,
		| Номенклатура.Родитель КАК ГруппаНоменклатуры,
		| Номенклатура.Поставщик КАК Поставщик,
		| Номенклатура.СтавкаНДС КАК СтавкаНДС,
		| Номенклатура.ВидНоменклатуры КАК ВидНоменклатуры,
		| Номенклатура.Марка КАК Марка,
		| Номенклатура.ТоварнаяКатегория КАК ТоварнаяКатегория,
```

```
                | Номенклатура.КоллекцияНоменклатуры КАК КоллекцияНоменклатуры,
                | Номенклатура.АртикулИсходный КАК АртикулИсходный,
                | Номенклатура.Состав КАК Состав,
                | Номенклатура.СтранаПроисхождения КАК СтранаПроисхождения,
                | Номенклатура.ор_Пол КАК Пол,
                | Номенклатура.Цвет КАК Цвет,
                | Номенклатура.Размер КАК РоссийскийРазмер,
                | Номенклатура.ЕдиницаИзмерения КАК ЕдиницаИзмерения,
                | Номенклатура.КодТНВЭД КАК КодТНВЭД,
                | Номенклатура.ВестиУчетПоГТД КАК ВестиУчетПоГТД,
                | ЕСТЬNULL(ДополнительныйРеквизитСезонПоставщика.Значение,
ЗНАЧЕНИЕ(Справочник.ЗначенияСвойствОбъектов.ПустаяСсылка)) КАК СезонПоставщика,
                | ЕСТЬNULL(ДополнительныйРеквизитДопПризнак.Значение, """") КАК ДополнительныйПризнак

                |,Номенклатура.ПометкаУдаления КАК ПометкаУдаления

                |ИЗ
                | Справочник.Номенклатура КАК Номенклатура
                |       ВНУТРЕННЕЕ СОЕДИНЕНИЕ ТаблицаТоваров КАК ТаблицаТоваров
                |       ПО Номенклатура.Поставщик = ТаблицаТоваров.Поставщик
                |               И Номенклатура.АртикулИсходный = ТаблицаТоваров.АртикулИсходный
                |               И Номенклатура.КодТНВЭД = ТаблицаТоваров.КодТНВЭД
                |               И Номенклатура.СтранаПроисхождения = ТаблицаТоваров.СтранаПроисхождения
                |       ЛЕВОЕ СОЕДИНЕНИЕ Справочник.Номенклатура.ДополнительныеРеквизиты КАК
ДополнительныйРеквизитСезонПоставщика
                |       ПО (ДополнительныйРеквизитСезонПоставщика.Ссылка = Номенклатура.Ссылка)
                |               И (ДополнительныйРеквизитСезонПоставщика.Свойство = &СвойствоСезонПоставщика)
                |       ЛЕВОЕ СОЕДИНЕНИЕ Справочник.Номенклатура.ДополнительныеРеквизиты КАК
ДополнительныйРеквизитДопПризнак
                |       ПО (ДополнительныйРеквизитДопПризнак.Ссылка = Номенклатура.Ссылка)
                |               И (ДополнительныйРеквизитДопПризнак.Свойство = &СвойствоДопПризнак)
                |ГДЕ
                | Номенклатура.ВидНоменклатуры.ИспользоватьХарактеристики
                | И Номенклатура.ВидНоменклатуры.РассчитыватьСебестоимостьБезУчетаХарактеристик";

                ТаблицаТоваровБезНоменклатуры = ТаблицаТоваров.Скопировать(Новый Структура("Номенклатура",
Справочники.Номенклатура.ПустаяСсылка()), "ИдентификаторТовара, АртикулИсходный, КодТНВЭД, СтранаПроисхождения");

                Запрос.УстановитьПараметр("ТаблицаТоваров"          , ТаблицаТоваровБезНоменклатуры);
                Запрос.УстановитьПараметр("Поставщик"               , Поставщик);
                Запрос.УстановитьПараметр("СвойствоСезонПоставщика" ,
ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.НайтиПоНаименованию("Сезон поставщика", Истина));
                Запрос.УстановитьПараметр("СвойствоДопПризнак"      ,
ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.НайтиПоНаименованию("Доп. признак", Истина));

                ОтборСтрок = Новый Структура("ИдентификаторТовара");

                Выборка = Запрос.Выполнить().Выбрать();
                Пока Выборка.Следующий() Цикл

                  ОтборСтрок.ИдентификаторТовара = Выборка.ИдентификаторТовара;

                  //Таблица товаров
                  НайденныеСтроки = ТаблицаТоваров.НайтиСтроки(ОтборСтрок);
                  Для Каждого СтрокаТовара Из НайденныеСтроки Цикл
                        СтрокаТовара.Номенклатура = Выборка.Номенклатура;
                  КонецЦикла;

                  ИнформацияПоНоменклатуре = СтуктураКешируемойИнформацииПоНоменклатуре();
                  ЗаполнитьЗначенияСвойств(ИнформацияПоНоменклатуре, Выборка);

                  КэшированныеЗначения.Номенклатура.Вставить(Выборка.Номенклатура, ИнформацияПоНоменклатуре);

                КонецЦикла;

КонецПроцедуры

Процедура ПодобратьСуществующуюНоменклатуру(ТаблицаТоваров, Поставщик, КэшированныеЗначения) Экспорт

                Запрос = Новый Запрос;
                Запрос.Текст =
                "ВЫБРАТЬ РАЗЛИЧНЫЕ
                | ТаблицаТоваров.ИдентификаторТовара КАК ИдентификаторТовара,
                | &Поставщик КАК Поставщик,
                | ТаблицаТоваров.АртикулИсходный КАК АртикулИсходный
                |ПОМЕСТИТЬ ТаблицаТоваров
```

```
|ИЗ
| &ТаблицаТоваров КАК ТаблицаТоваров
|;
|
|////////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
| ТаблицаТоваров.ИдентификаторТовара КАК ИдентификаторТовара,
| Номенклатура.Ссылка КАК Номенклатура,
| Номенклатура.Артикул КАК Артикул,
| Номенклатура.Наименование КАК Наименование,
| Номенклатура.Родитель КАК ГруппаНоменклатуры,
| Номенклатура.Поставщик КАК Поставщик,
| Номенклатура.СтавкаНДС КАК СтавкаНДС,
| Номенклатура.ВидНоменклатуры КАК ВидНоменклатуры,
| Номенклатура.Марка КАК Марка,
| Номенклатура.ТоварнаяКатегория КАК ТоварнаяКатегория,
| Номенклатура.КоллекцияНоменклатуры КАК КоллекцияНоменклатуры,
| Номенклатура.АртикулИсходный КАК АртикулИсходный,
| Номенклатура.Состав КАК Состав,
| Номенклатура.СтранаПроисхождения КАК СтранаПроисхождения,
| Номенклатура.ор_Пол КАК Пол,
| Номенклатура.Цвет КАК Цвет,
| Номенклатура.Размер КАК РоссийскийРазмер,
| ЕСТЬNULL(ДополнительныйРеквизитСезонПоставщика.Значение,
ЗНАЧЕНИЕ(Справочник.ЗначенияСвойствОбъектов.ПустаяСсылка)) КАК СезонПоставщика,
| ЕСТЬNULL(ДополнительныйРеквизитДопПризнак.Значение, """") КАК ДополнительныйПризнак

|,Номенклатура.ПометкаУдаления КАК ПометкаУдаления

|ИЗ
| Справочник.Номенклатура КАК Номенклатура
|       ЛЕВОЕ СОЕДИНЕНИЕ Справочник.Номенклатура.ДополнительныеРеквизиты КАК
ДополнительныйРеквизитСезонПоставщика
|       ПО (ДополнительныйРеквизитСезонПоставщика.Ссылка = Номенклатура.Ссылка)
|           И (ДополнительныйРеквизитСезонПоставщика.Свойство = &СвойствоСезонПоставщика)
|       ЛЕВОЕ СОЕДИНЕНИЕ Справочник.Номенклатура.ДополнительныеРеквизиты КАК
ДополнительныйРеквизитДопПризнак
|       ПО (ДополнительныйРеквизитДопПризнак.Ссылка = Номенклатура.Ссылка)
|           И (ДополнительныйРеквизитДопПризнак.Свойство = &СвойствоДопПризнак)
|       ВНУТРЕННЕЕ СОЕДИНЕНИЕ ТаблицаТоваров КАК ТаблицаТоваров
|       ПО Номенклатура.АртикулИсходный = ТаблицаТоваров.АртикулИсходный
|           И Номенклатура.Поставщик = ТаблицаТоваров.Поставщик
|ГДЕ
| Номенклатура.ВидНоменклатуры.ИспользоватьХарактеристики
| И Номенклатура.ВидНоменклатуры.РассчитыватьСебестоимостьБезУчетаХарактеристик";

ТаблицаТоваровБезНоменклатуры = ТаблицаТоваров.Скопировать(Новый Структура("Номенклатура",
Справочники.Номенклатура.ПустаяСсылка()), "ИдентификаторТовара, АртикулИсходный");

Запрос.УстановитьПараметр("ТаблицаТоваров"          , ТаблицаТоваровБезНоменклатуры);
Запрос.УстановитьПараметр("Поставщик"               , Поставщик);
Запрос.УстановитьПараметр("СвойствоСезонПоставщика" ,
ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.НайтиПоНаименованию("Сезон поставщика", Истина));
Запрос.УстановитьПараметр("СвойствоДопПризнак"      ,
ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.НайтиПоНаименованию("Доп. признак", Истина));

ОтборСтрок = Новый Структура("ИдентификаторТовара");

Выборка = Запрос.Выполнить().Выбрать();
Пока Выборка.Следующий() Цикл

  ОтборСтрок.ИдентификаторТовара = Выборка.ИдентификаторТовара;

  //Таблица товаров
  НайденныеСтроки = ТаблицаТоваров.НайтиСтроки(ОтборСтрок);
  Для Каждого СтрокаТовара Из НайденныеСтроки Цикл
        СтрокаТовара.Номенклатура = Выборка.Номенклатура;
  КонецЦикла;

  ИнформацияПоНоменклатуре = СтуктураКешируемойИнформацииПоНоменклатуре();
  ЗаполнитьЗначенияСвойств(ИнформацияПоНоменклатуре, Выборка);

  КэшированныеЗначения.Номенклатура.Вставить(Выборка.Номенклатура, ИнформацияПоНоменклатуре);

КонецЦикла;
```

```
КонецПроцедуры

Процедура ПолучитьДанныеНоменклатуры(ТаблицаТоваров, КэшированныеЗначения)

		Запрос = Новый Запрос;
		Запрос.Текст =
		"ВЫБРАТЬ
		| Номенклатура.Ссылка КАК Номенклатура,
		| Номенклатура.Артикул КАК Артикул,
		| Номенклатура.Наименование КАК Наименование,
		| Номенклатура.Родитель КАК ГруппаНоменклатуры,
		| Номенклатура.Поставщик КАК Поставщик,
		| Номенклатура.СтавкаНДС КАК СтавкаНДС,
		| Номенклатура.ВидНоменклатуры КАК ВидНоменклатуры,
		| Номенклатура.Марка КАК Марка,
		| Номенклатура.ТоварнаяКатегория КАК ТоварнаяКатегория,
		| Номенклатура.КоллекцияНоменклатуры КАК КоллекцияНоменклатуры,
		| Номенклатура.АртикулИсходный КАК АртикулИсходный,
		| Номенклатура.Состав КАК Состав,
		| Номенклатура.СтранаПроисхождения КАК СтранаПроисхождения,
		| Номенклатура.ор_Пол КАК Пол,
		| Номенклатура.Цвет КАК Цвет,
		| Номенклатура.Размер КАК РоссийскийРазмер,
		| Номенклатура.ЕдиницаИзмерения КАК ЕдиницаИзмерения,
		| Номенклатура.КодТНВЭД КАК КодТНВЭД,
		| Номенклатура.ВестиУчетПоГТД КАК ВестиУчетПоГТД,
		| ЕСТЬNULL(ДополнительныйРеквизитСезонПоставщика.Значение,
ЗНАЧЕНИЕ(Справочник.ЗначенияСвойствОбъектов.ПустаяСсылка)) КАК СезонПоставщика,
		| ЕСТЬNULL(ДополнительныйРеквизитДопПризнак.Значение, """") КАК ДополнительныйПризнак
		|ИЗ
		| Справочник.Номенклатура КАК Номенклатура
		|	ЛЕВОЕ СОЕДИНЕНИЕ Справочник.Номенклатура.ДополнительныеРеквизиты КАК
ДополнительныйРеквизитСезонПоставщика
		|	ПО (ДополнительныйРеквизитСезонПоставщика.Ссылка = Номенклатура.Ссылка)
		|		И (ДополнительныйРеквизитСезонПоставщика.Свойство = &СвойствоСезонПоставщика)
		|	ЛЕВОЕ СОЕДИНЕНИЕ Справочник.Номенклатура.ДополнительныеРеквизиты КАК
ДополнительныйРеквизитДопПризнак
		|	ПО (ДополнительныйРеквизитДопПризнак.Ссылка = Номенклатура.Ссылка)
		|		И (ДополнительныйРеквизитДопПризнак.Свойство = &СвойствоДопПризнак)
		|ГДЕ
		| Номенклатура.Ссылка В(&МассивНоменклатуры)";

		МассивНоменклатуры = ОбщегоНазначения.ВыгрузитьКолонку(ТаблицаТоваров, "Номенклатура", Истина);
		ОбщегоНазначенияКлиентСервер.УдалитьЗначениеИзМассива(МассивНоменклатуры,
Справочники.Номенклатура.ПустаяСсылка());

		Запрос.УстановитьПараметр("МассивНоменклатуры"     , МассивНоменклатуры);
		Запрос.УстановитьПараметр("СвойствоСезонПоставщика",
ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.НайтиПоНаименованию("Сезон поставщика", Истина));
		Запрос.УстановитьПараметр("СвойствоДопПризнак"     ,
ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.НайтиПоНаименованию("Доп. признак", Истина));

		Выборка = Запрос.Выполнить().Выбрать();
		Пока Выборка.Следующий() Цикл

		  ИнформацияПоНоменклатуре = СтуктураКешируемойИнформацииПоНоменклатуре();
		  ЗаполнитьЗначенияСвойств(ИнформацияПоНоменклатуре, Выборка);

		  КэшированныеЗначения.Номенклатура.Вставить(Выборка.Номенклатура, ИнформацияПоНоменклатуре);

		КонецЦикла;

КонецПроцедуры

Процедура ВывестиСписокОшибокПриЗагрузкеПроекта(ТаблицаТоваров, Ошибки) Экспорт

		Если Ошибки = Неопределено Тогда
		  возврат;
		КонецЕсли;

		СтрокиОшибок = Ошибки.СтрокиОшибок;

		Для Каждого СтрокаОшибки Из СтрокиОшибок Цикл

		  //Заголовок ошибки
		  Если СтрокаОшибки = 0 Тогда
```

```
                    ЗаголовокОшибки = "Общие ошибки проекта:"

                Иначе

                    СтрокаТовара = ТаблицаТоваров[СтрокаОшибки - 1];

                    ШаблонЗаголовкаОшибки = "Ошибки в строке %1 (Исходный артикул: %2):";
                    ЗаголовокОшибки =
СтроковыеФункцииКлиентСервер.ПодставитьПараметрыВСтроку(ШаблонЗаголовкаОшибки,
                    Формат(СтрокаТовара.НомерСтроки, "ЧГ="),
                    СтрокаТовара.АртикулИсходный);

                КонецЕсли;

                ОбщегоНазначенияКлиентСервер.СообщитьПользователю(ЗаголовокОшибки);

                //Ошибка
                ТекстыОшибок = Ошибки.СписокОшибок.Получить(СтрокаОшибки);
                Для Каждого ТекстОшибки Из ТекстыОшибок Цикл
                    ОбщегоНазначенияКлиентСервер.СообщитьПользователю(ТекстОшибки);
                КонецЦикла;

            КонецЦикла;

КонецПроцедуры

Процедура ДобавитьОшибкуЗагрузкиТоваров(Ошибки, ТекстОшибки, НомерСтроки = 0) Экспорт

            Если Ошибки.СтрокиОшибок.Найти(НомерСтроки) = Неопределено Тогда

                Ошибки.СтрокиОшибок.Добавить(НомерСтроки);

                МассивОшибок = Новый Массив;
                Ошибки.СписокОшибок.Вставить(НомерСтроки, МассивОшибок);

            КонецЕсли;

            ТекстыОшибок = Ошибки.СписокОшибок.Получить(НомерСтроки);
            ТекстыОшибок.Добавить(ТекстОшибки);

КонецПроцедуры

Функция ИмяСобытияЗагрузкиЖурналаРегистрации() Экспорт
            Возврат НСтр("ru = 'Проект поставки.Загрузка данных'", ОбщегоНазначенияКлиентСервер.КодОсновногоЯзыка());
КонецФункции

Функция ПолучитьСтруктуруОшибокПроектаПоставки() Экспорт

            Ошибки = Новый Структура;
            Ошибки.Вставить("СписокОшибок", Новый Соответствие);
            Ошибки.Вставить("СтрокиОшибок", Новый Массив);

            возврат Ошибки;

КонецФункции

Процедура ПроставитьСоответствияРазмеров(ТаблицаТоваров, Поставщик) Экспорт

            Запрос = Новый Запрос;
            Запрос.Текст =

            "ВЫБРАТЬ
            | ТаблицаТоваров.НомерСтроки КАК НомерСтроки,
            | ТаблицаТоваров.Марка КАК Марка,
            | ТаблицаТоваров.ТоварнаяКатегория КАК ТоварнаяКатегория,
            | ТаблицаТоваров.РазмерПоставщика КАК РазмерПоставщика
            |ПОМЕСТИТЬ ТаблицаТоваров
            |ИЗ
            | &ТаблицаТоваров КАК ТаблицаТоваров
            |;
            |
            ////////////////////////////////////////////////////////////////////////////////

            |ВЫБРАТЬ
            | РазмерныеШкалы.Ссылка КАК РазмернаяШкала,
            | ВидыРазмерныхШкал.Марка КАК Марка,
```

```
                | РазмерныеШкалыТоварныеКатегории.ТоварнаяКатегория КАК ТоварнаяКатегория
                |ПОМЕСТИТЬ РазмерныеШкалыПоставщика
                |ИЗ
                | Справочник.РазмерныеШкалы.ТоварныеКатегории КАК РазмерныеШкалыТоварныеКатегории
                |       ВНУТРЕННЕЕ СОЕДИНЕНИЕ Справочник.РазмерныеШкалы КАК РазмерныеШкалы
                |               ВНУТРЕННЕЕ СОЕДИНЕНИЕ Справочник.ВидыРазмерныхШкал КАК ВидыРазмерныхШкал
                |               ПО РазмерныеШкалы.ВидРазмернойШкалы = ВидыРазмерныхШкал.Ссылка
                |       ПО РазмерныеШкалыТоварныеКатегории.Ссылка = РазмерныеШкалы.Ссылка
                |ГДЕ
                | ВидыРазмерныхШкал.Поставщик = &Поставщик
                | И ВидыРазмерныхШкал.Марка В(&Марки)
                | И РазмерныеШкалыТоварныеКатегории.ТоварнаяКатегория В(&ТоварныеКатегории)
                |;
                |
                |////////////////////////////////////////////////////////////////////////////////
                |ВЫБРАТЬ
                | РазмерныеШкалы.Ссылка КАК РазмернаяШкала,
                | РазмерныеШкалыТоварныеКатегории.ТоварнаяКатегория КАК ТоварнаяКатегория
                |ПОМЕСТИТЬ РоссийскиеРазмерныеШкалы
                |ИЗ
                | Справочник.РазмерныеШкалы.ТоварныеКатегории КАК РазмерныеШкалыТоварныеКатегории
                |       ВНУТРЕННЕЕ СОЕДИНЕНИЕ Справочник.РазмерныеШкалы КАК РазмерныеШкалы
                |       ПО РазмерныеШкалыТоварныеКатегории.Ссылка = РазмерныеШкалы.Ссылка
                |ГДЕ
                | РазмерныеШкалы.ВидРазмернойШкалы = ЗНАЧЕНИЕ(Справочник.ВидыРазмерныхШкал.Российская)
                |;
                |
                |////////////////////////////////////////////////////////////////////////////////
                |ВЫБРАТЬ
                | РазмерныеШкалыПоставщика.РазмернаяШкала КАК РазмернаяШкалаПоставщика,
                | РазмерныеШкалыПоставщика.Марка КАК Марка,
                | РазмерныеШкалыПоставщика.ТоварнаяКатегория КАК ТоварнаяКатегория,
                | РоссийскиеРазмерныеШкалы.РазмернаяШкала КАК РоссийскаяРазмернаяШкала
                |ПОМЕСТИТЬ СоответствияРазмерныхШкал
                |ИЗ
                | РазмерныеШкалыПоставщика КАК РазмерныеШкалыПоставщика
                |       ВНУТРЕННЕЕ СОЕДИНЕНИЕ РоссийскиеРазмерныеШкалы КАК РоссийскиеРазмерныеШкалы
                |       ПО РазмерныеШкалыПоставщика.ТоварнаяКатегория = РоссийскиеРазмерныеШкалы.ТоварнаяКатегория
                |;
                |
                |////////////////////////////////////////////////////////////////////////////////
                |ВЫБРАТЬ
                | СоответствияРазмерныхШкал.Марка КАК Марка,
                | СоответствияРазмерныхШкал.ТоварнаяКатегория КАК ТоварнаяКатегория,
                | СоответствияРазмеровРазмерныхШкал.РазмерСоответствия КАК РазмерПоставщика,
                | СоответствияРазмеровРазмерныхШкал.Размер КАК РоссийскийРазмер
                |ПОМЕСТИТЬ СоответствияРазмеров
                |ИЗ
                | СоответствияРазмерныхШкал КАК СоответствияРазмерныхШкал
                |       ВНУТРЕННЕЕ СОЕДИНЕНИЕ РегистрСведений.СоответствиеРазмерныхШкал КАК
СоответствияРазмеровРазмерныхШкал
                |       ПО СоответствияРазмерныхШкал.РазмернаяШкалаПоставщика =
СоответствияРазмеровРазмерныхШкал.РазмернаяШкалаСоответствия
                |               И СоответствияРазмерныхШкал.РоссийскаяРазмернаяШкала =
СоответствияРазмеровРазмерныхШкал.РазмернаяШкала
                |;
                |
                |////////////////////////////////////////////////////////////////////////////////
                |ВЫБРАТЬ
                | ТаблицаТоваров.НомерСтроки КАК НомерСтроки,
                | СоответствияРазмеров.РоссийскийРазмер КАК РоссийскийРазмер
                |ИЗ
                | ТаблицаТоваров КАК ТаблицаТоваров
                |       ВНУТРЕННЕЕ СОЕДИНЕНИЕ СоответствияРазмеров КАК СоответствияРазмеров
                |       ПО ТаблицаТоваров.Марка = СоответствияРазмеров.Марка
                |               И ТаблицаТоваров.ТоварнаяКатегория = СоответствияРазмеров.ТоварнаяКатегория
                |               И ТаблицаТоваров.РазмерПоставщика = СоответствияРазмеров.РазмерПоставщика";

                Запрос.УстановитьПараметр("Поставщик"        , Поставщик);
                Запрос.УстановитьПараметр("Марки"            , ОбщегоНазначения.ВыгрузитьКолонку(ТаблицаТоваров, "Марка",
Истина));
                Запрос.УстановитьПараметр("ТоварныеКатегории" , ОбщегоНазначения.ВыгрузитьКолонку(ТаблицаТоваров,
"ТоварнаяКатегория", Истина));

                Запрос.УстановитьПараметр("ТаблицаТоваров", ТаблицаТоваров);
```

```
		Выборка = Запрос.Выполнить().Выбрать();
		Пока Выборка.Следующий() Цикл

		  СтрокаТовара = ТаблицаТоваров[Выборка.НомерСтроки - 1];
		  СтрокаТовара.РоссийскийРазмер = Выборка.РоссийскийРазмер;

		КонецЦикла;

		Для Каждого СтрокаТовара Из ТаблицаТоваров Цикл

		  Если ЗначениеЗаполнено(СтрокаТовара.ПараметрыУчета.ВидНоменклатуры) И Не
СтрокаТовара.ПараметрыУчета.ИспользоватьСоответствияРоссийскимРазмерам Тогда
			СтрокаТовара.РоссийскийРазмер = СтрокаТовара.РазмерПоставщика;
		  КонецЕсли;

		КонецЦикла;

КонецПроцедуры

&НаСервере
Процедура ОпределитьДействияПоАктуОРасхождениях(АктОРасхождениях, ОформитьПересортицу=Ложь,
ОформитьОприходование=Ложь, ОформитьСписание=Ложь) Экспорт

		Запрос = Новый Запрос;
		Запрос.УстановитьПараметр("Ссылка", АктОРасхождениях);
		Запрос.Текст =
		"ВЫБРАТЬ
		| АктОРасхожденияхПослеПриемкиТовары.НомерСтроки КАК НомерСтроки,
		| АктОРасхожденияхПослеПриемкиТовары.Номенклатура КАК Номенклатура,
		| АктОРасхожденияхПослеПриемкиТовары.Характеристика КАК Характеристика,
		| АктОРасхожденияхПослеПриемкиТовары.Действие КАК Действие
		|ИЗ
		| Документ.АктОРасхожденияхПослеПриемки.Товары КАК АктОРасхожденияхПослеПриемкиТовары
		|ГДЕ
		| АктОРасхожденияхПослеПриемкиТовары.Ссылка = &Ссылка
		| И (АктОРасхожденияхПослеПриемкиТовары.Количество >
АктОРасхожденияхПослеПриемкиТовары.КоличествоПоДокументу
		|                                  И АктОРасхожденияхПослеПриемкиТовары.Действие =
ЗНАЧЕНИЕ(Перечисление.ВариантыДействийПоРасхождениямВАктеПослеПриемки.ВернутьПерепоставленноеБезОформления)
		|                        ИЛИ АктОРасхожденияхПослеПриемкиТовары.Количество <
АктОРасхожденияхПослеПриемкиТовары.КоличествоПоДокументу
		|                                  И АктОРасхожденияхПослеПриемкиТовары.Действие =
ЗНАЧЕНИЕ(Перечисление.ВариантыДействийПоРасхождениямВАктеПослеПриемки.ОжидатьДопоставкуБезОформления))
		| И АктОРасхожденияхПослеПриемкиТовары.Пересортица
		|;
		|
		|////////////////////////////////////////////////////////////////////////////////
		|ВЫБРАТЬ
		| АктОРасхожденияхПослеПриемкиТовары.НомерСтроки КАК НомерСтроки,
		| АктОРасхожденияхПослеПриемкиТовары.Номенклатура КАК Номенклатура,
		| АктОРасхожденияхПослеПриемкиТовары.Характеристика КАК Характеристика,
		| АктОРасхожденияхПослеПриемкиТовары.Действие КАК Действие
		|ИЗ
		| Документ.АктОРасхожденияхПослеПриемки.Товары КАК АктОРасхожденияхПослеПриемкиТовары
		|ГДЕ
		| АктОРасхожденияхПослеПриемкиТовары.Ссылка = &Ссылка
		| И АктОРасхожденияхПослеПриемкиТовары.Количество >
АктОРасхожденияхПослеПриемкиТовары.КоличествоПоДокументу
		| И АктОРасхожденияхПослеПриемкиТовары.Действие =
ЗНАЧЕНИЕ(Перечисление.ВариантыДействийПоРасхождениямВАктеПослеПриемки.ОтнестиПерепоставленноеНаПрочиеДоходы)
		|;
		|
		|////////////////////////////////////////////////////////////////////////////////
		|ВЫБРАТЬ
		| АктОРасхожденияхПослеПриемкиТовары.НомерСтроки КАК НомерСтроки,
		| АктОРасхожденияхПослеПриемкиТовары.Номенклатура КАК Номенклатура,
		| АктОРасхожденияхПослеПриемкиТовары.Характеристика КАК Характеристика,
		| АктОРасхожденияхПослеПриемкиТовары.Действие КАК Действие
		|ИЗ
		| Документ.АктОРасхожденияхПослеПриемки.Товары КАК АктОРасхожденияхПослеПриемкиТовары
		|ГДЕ
		| АктОРасхожденияхПослеПриемкиТовары.Ссылка = &Ссылка
		| И АктОРасхожденияхПослеПриемкиТовары.Количество <
АктОРасхожденияхПослеПриемкиТовары.КоличествоПоДокументу
		| И АктОРасхожденияхПослеПриемкиТовары.Действие =
ЗНАЧЕНИЕ(Перечисление.ВариантыДействийПоРасхождениямВАктеПослеПриемки.ОтнестиНедостачуНаПрочиеРасходы)";
```

```
		МассивРезультатов = Запрос.ВыполнитьПакет();

		ОформитьПересортицу		= НЕ МассивРезультатов[0].Пустой();
		ОформитьОприходование		= НЕ МассивРезультатов[1].Пустой();
		ОформитьСписание		= НЕ МассивРезультатов[2].Пустой();

КонецПроцедуры

	//	ВариантКомплектации	- Структура, копирующая справочник ВариантыКомплектации
//		ПроектПоставки			- ДокументСсылка.ПроектПоставки
//		КоличествоКомплектов	- Число
&НаСервере
Функция СоздатьОбновитьСборкуТоваров(ВариантКомплектации, ПроектПоставки, КоличествоКомплектов)

		Запрос = Новый Запрос;
		Запрос.УстановитьПараметр("ПроектПоставки", ПроектПоставки);
		Запрос.УстановитьПараметр("Номенклатура",	ВариантКомплектации.Владелец);
		Запрос.УстановитьПараметр("Характеристика", ВариантКомплектации.Характеристика);
		Запрос.Текст =
		"ВЫБРАТЬ
		| СборкаТоваров.Ссылка КАК Ссылка
		|ИЗ
		| Документ.СборкаТоваров КАК СборкаТоваров
		|ГДЕ
		| НЕ СборкаТоваров.ПометкаУдаления
		| И СборкаТоваров.ПроектПоставки = &ПроектПоставки
		| И СборкаТоваров.Номенклатура = &Номенклатура
		| И СборкаТоваров.Характеристика = &Характеристика";
		Выборка = Запрос.Выполнить().Выбрать();

		Если Выборка.Следующий() Тогда
		  ТекстСообщения = "Обновлен документ";
		  СборкаТоваровОбъект = Выборка.Ссылка.ПолучитьОбъект();
		  СборкаТоваровОбъект.Товары.Очистить();
		Иначе
		  ТекстСообщения = "Создан документ";
		  СборкаТоваровОбъект = Документы.СборкаТоваров.СоздатьДокумент();
		КонецЕсли;

		ВариантКомплектации.Вставить("ПроектПоставки",		ПроектПоставки);
		ВариантКомплектации.Вставить("Дата",
		ОпределитьДатуСборкиКомплекта(ПроектПоставки));
		ВариантКомплектации.Вставить("Количество",		КоличествоКомплектов);
		ВариантКомплектации.Вставить("ВариантКомплектации",
		НайтиВариантКомплектации(ВариантКомплектации.Владелец, ВариантКомплектации.Характеристика));

		СборкаТоваровОбъект.Заполнить(ВариантКомплектации);

		Попытка
		  СборкаТоваровОбъект.Записать(РежимЗаписиДокумента.Проведение);
		  Сообщить(ТекстСообщения+" "+СборкаТоваровОбъект.Ссылка);
		Исключение
		  Возврат Неопределено;
		КонецПопытки;

		Возврат СборкаТоваровОбъект.Ссылка;

КонецФункции


&НаСервере
Функция СоздатьОбновитьПриобретениеТоваровУслуг(ДокументСсылка, ПроектПоставки, ПризнакТОРГ12, ТоварыВПути = Ложь)

		Если НЕ ЗначениеЗаполнено(ДокументСсылка) Тогда
		  ДокументОбъект = Документы.ПриобретениеТоваровУслуг.СоздатьДокумент();
		Иначе
		  ДокументОбъект = ДокументСсылка.ПолучитьОбъект();
		  ДокументОбъект.Товары.Очистить();
		КонецЕсли;

		СтруктураЗаполнения = Новый Структура;
		СтруктураЗаполнения.Вставить("ДокументОснование", ПроектПоставки);
		СтруктураЗаполнения.Вставить("ПризнакТОРГ12"	, ПризнакТОРГ12);


		ДокументОбъект.Дата = ПроектПоставки.ДатаПлановогоПоступления;
```

```
		Если ТоварыВПути Тогда
		  ДокументОбъект.Дата = ТекущаяДата();
		КонецЕсли;

		ДокументОбъект.Автор = Пользователи.ТекущийПользователь();


		ДокументОбъект.ЗакупкаПодДеятельность = Перечисления.ТипыНалогообложенияНДС.ПродажаОблагаетсяНДС;

		ДокументОбъект.ТоварыВПути = ТоварыВПути;


		ДокументОбъект.Заполнить(СтруктураЗаполнения);

		ДокументОбъект.Записать(РежимЗаписиДокумента.Проведение);

		Возврат ДокументОбъект.Ссылка;

КонецФункции

&НаСервере
Функция СоздатьОбновитьКорректировкуПриобретения(ДокументСсылка, Основание, ПроектПоставки, ПризнакТОРГ12)

		Если ДокументСсылка = Неопределено Тогда
		  ДокументОбъект                = Документы.КорректировкаПриобретения.СоздатьДокумент();
		  ДокументОбъект.Дата = ТекущаяДата();
		Иначе
		  ДокументОбъект                = ДокументСсылка.ПолучитьОбъект();
		  ДокументОбъект.Записать(РежимЗаписиДокумента.ОтменаПроведения);
		КонецЕсли;

		СтруктураЗаполнения = Новый Структура;
		СтруктураЗаполнения.Вставить("ДокументОснование", Основание);
		СтруктураЗаполнения.Вставить("ПроектПоставки",          ПроектПоставки);
		СтруктураЗаполнения.Вставить("ПризнакТОРГ12",          ПризнакТОРГ12);

		ДокументОбъект.Заполнить(СтруктураЗаполнения);

		ДокументОбъект.КорректировкаТоваровВПути = Основание.ТоварыВПути;

		ДокументОбъект.Записать(РежимЗаписиДокумента.Проведение);

		Возврат ДокументОбъект.Ссылка;

КонецФункции



&НаСервере
Процедура СоздатьОбновитьДокументыСборкиКомплектов(ПроектПоставкиСсылка, ПоФактическомуПоступлению=Ложь) Экспорт

		ДокументыКУдалению = Новый Соответствие;

		Запрос = Новый Запрос;
		Запрос.УстановитьПараметр("ПроектПоставки", ПроектПоставкиСсылка);
		Запрос.Текст =
		"ВЫБРАТЬ РАЗЛИЧНЫЕ
		| СборкаТоваров.Ссылка КАК Ссылка
		|ИЗ
		| Документ.СборкаТоваров КАК СборкаТоваров
		|ГДЕ
		| НЕ СборкаТоваров.ПометкаУдаления
		| И СборкаТоваров.ПроектПоставки = &ПроектПоставки";
		Выборка = Запрос.Выполнить().Выбрать();
		Пока Выборка.Следующий() Цикл
		  ДокументыКУдалению.Вставить(Выборка.Ссылка, Истина);
		КонецЦикла;

		Запрос = Новый Запрос;
		Запрос.УстановитьПараметр("ПроектПоставки", ПроектПоставкиСсылка);
		Запрос.УстановитьПараметр("ПоФакту",                ПоФактическомуПоступлению);
		Запрос.Текст =
		"ВЫБРАТЬ
		| СоставКомплектовПроектовПоставки.ПроектПоставки КАК ПроектПоставки,
		| СоставКомплектовПроектовПоставки.СтрокаНомер КАК СтрокаНомер,
```

```
		| СоставКомплектовПроектовПоставки.Номенклатура КАК Комплект,
		| СоставКомплектовПроектовПоставки.Характеристика КАК ХарактеристикаКомплекта,
		| СоставКомплектовПроектовПоставки.Комплектующая КАК Номенклатура,
		| СоставКомплектовПроектовПоставки.ХарактеристикаКомплектующей КАК Характеристика,
		| СоставКомплектовПроектовПоставки.Количество КАК Количество,
		| СоставКомплектовПроектовПоставки.ДоляСтоимости КАК ДоляСтоимости,
		| СоставКомплектовПроектовПоставки.ОсновнаяКомплектующая КАК ОсновнойКомпонент
		|ПОМЕСТИТЬ втКомплектующие
		|ИЗ
		| РегистрСведений.СоставКомплектовПроектовПоставки КАК СоставКомплектовПроектовПоставки
		|ГДЕ
		| СоставКомплектовПроектовПоставки.ПроектПоставки = &ПроектПоставки
		|;
		|
		////////////////////////////////////////////////////////////////////////////////
		|ВЫБРАТЬ
		| втКомплектующие.Комплект КАК Комплект,
		| втКомплектующие.ХарактеристикаКомплекта КАК ХарактеристикаКомплекта,
		| СУММА(ПоступлениеТоваровПроектаПоставкиОбороты.КоличествоФактОборот) КАК КоличествоКомплектовФакт
		|ПОМЕСТИТЬ втПоступлениеФакт
		|ИЗ
		| втКомплектующие КАК втКомплектующие
		|		ВНУТРЕННЕЕ СОЕДИНЕНИЕ РегистрНакопления.ПоступлениеТоваровПроектаПоставки.Обороты(, , ,
ПроектПоставки = &ПроектПоставки) КАК ПоступлениеТоваровПроектаПоставкиОбороты
		|		ПО втКомплектующие.Комплект = ПоступлениеТоваровПроектаПоставкиОбороты.Номенклатура
		|			И втКомплектующие.ХарактеристикаКомплекта =
ПоступлениеТоваровПроектаПоставкиОбороты.Характеристика
		|
		|СГРУППИРОВАТЬ ПО
		| втКомплектующие.Комплект,
		| втКомплектующие.ХарактеристикаКомплекта
		|;
		|
		////////////////////////////////////////////////////////////////////////////////
		|ВЫБРАТЬ
		| втКомплектующие.Комплект КАК Комплект,
		| втКомплектующие.ХарактеристикаКомплекта КАК ХарактеристикаКомплекта,
		| ТоварыПроектовПоставки.Количество КАК КоличествоКомплектовПлан
		|ПОМЕСТИТЬ втПоступлениеПлан
		|ИЗ
		| втКомплектующие КАК втКомплектующие
		|		ВНУТРЕННЕЕ СОЕДИНЕНИЕ РегистрСведений.ТоварыПроектовПоставки КАК ТоварыПроектовПоставки
		|		ПО втКомплектующие.ПроектПоставки = ТоварыПроектовПоставки.ПроектПоставки
		|			И втКомплектующие.Комплект = ТоварыПроектовПоставки.Номенклатура
		|			И втКомплектующие.ХарактеристикаКомплекта = ТоварыПроектовПоставки.Характеристика
		|
		|СГРУППИРОВАТЬ ПО
		| втКомплектующие.Комплект,
		| втКомплектующие.ХарактеристикаКомплекта,
		| ТоварыПроектовПоставки.Количество
		|;
		|
		////////////////////////////////////////////////////////////////////////////////
		|ВЫБРАТЬ
		| втКомплектующие.Комплект КАК Комплект,
		| втКомплектующие.ХарактеристикаКомплекта КАК ХарактеристикаКомплекта,
		| втКомплектующие.Номенклатура КАК Номенклатура,
		| втКомплектующие.Характеристика КАК Характеристика,
		| втКомплектующие.Количество КАК Количество,
		| ВЫБОР
		|		КОГДА &ПоФакту
		|			ТОГДА ЕСТЬNULL(втПоступлениеФакт.КоличествоКомплектовФакт, 0)
		|		ИНАЧЕ ЕСТЬNULL(втПоступлениеПлан.КоличествоКомплектовПлан, 0)
		| КОНЕЦ КАК КоличествоКомплектов,
		| втКомплектующие.ДоляСтоимости КАК ДоляСтоимости,
		| втКомплектующие.ОсновнойКомпонент КАК ОсновнойКомпонент
		|ИЗ
		| втКомплектующие КАК втКомплектующие
		|		ЛЕВОЕ СОЕДИНЕНИЕ втПоступлениеФакт КАК втПоступлениеФакт
		|		ПО втКомплектующие.Комплект = втПоступлениеФакт.Комплект
		|			И втКомплектующие.ХарактеристикаКомплекта = втПоступлениеФакт.ХарактеристикаКомплекта
		|		ЛЕВОЕ СОЕДИНЕНИЕ втПоступлениеПлан КАК втПоступлениеПлан
		|		ПО втКомплектующие.Комплект = втПоступлениеПлан.Комплект
		|			И втКомплектующие.ХарактеристикаКомплекта = втПоступлениеПлан.ХарактеристикаКомплекта
		|ГДЕ
		| ВЫБОР
```

```
                |                   КОГДА &ПоФакту
                |                       ТОГДА ЕСТЬNULL(втПоступлениеФакт.КоличествоКомплектовФакт, 0)
                |                   ИНАЧЕ ЕСТЬNULL(втПоступлениеПлан.КоличествоКомплектовПлан, 0)
                |           КОНЕЦ <> 0
                |ИТОГИ
                | СРЕДНЕЕ(КоличествоКомплектов)
                |ПО
                | Комплект,
                | ХарактеристикаКомплекта";
                РезЗапроса = Запрос.Выполнить();

                ВыборкаКомплект = РезЗапроса.Выбрать(ОбходРезультатаЗапроса.ПоГруппировкам);
                Пока ВыборкаКомплект.Следующий() Цикл
                  ВыборкаХарактеристикаКомплекта = ВыборкаКомплект.Выбрать(ОбходРезультатаЗапроса.ПоГруппировкам);
                  Пока ВыборкаХарактеристикаКомплекта.Следующий() Цикл

                        Комплектация = Новый Структура;
                        Комплектация.Вставить("Владелец",               ВыборкаКомплект.Комплект);
                        Комплектация.Вставить("Характеристика", ВыборкаХарактеристикаКомплекта.ХарактеристикаКомплекта);
                        Комплектация.Вставить("Количество",             1);
                        Комплектация.Вставить("СодержитТовары", Истина);
                        Комплектация.Вставить("НоменклатураОсновногоКомпонента");
                        Комплектация.Вставить("ХарактеристикаОсновногоКомпонента");

                        Комплектующие = Новый ТаблицаЗначений;
                        Комплектующие.Колонки.Добавить("Номенклатура",              Новый
ОписаниеТипов("СправочникСсылка.Номенклатура"));
                        Комплектующие.Колонки.Добавить("Характеристика",            Новый
ОписаниеТипов("СправочникСсылка.ХарактеристикиНоменклатуры"));
                        Комплектующие.Колонки.Добавить("Количество",                ОбщегоНазначения.ОписаниеТипаЧисло(15,3));
                        Комплектующие.Колонки.Добавить("ДоляСтоимости",             ОбщегоНазначения.ОписаниеТипаЧисло(10,0));
                        Комплектующие.Колонки.Добавить("ОсновнойКомпонент", Новый ОписаниеТипов("Булево"));

                        Выборка = ВыборкаХарактеристикаКомплекта.Выбрать();
                        Пока Выборка.Следующий() Цикл
                                ЗаполнитьЗначенияСвойств(Комплектующие.Добавить(), Выборка);
                                Если Выборка.ОсновнойКомпонент Тогда
                                        Комплектация.Вставить("НоменклатураОсновногоКомпонента",        Выборка.Номенклатура);
                                        Комплектация.Вставить("ХарактеристикаОсновногоКомпонента",
                Выборка.Характеристика);
                                КонецЕсли;
                        КонецЦикла;
                        Комплектация.Вставить("Товары", Комплектующие);

                        СборкаТоваровСсылка = СоздатьОбновитьСборкуТоваров(Комплектация, ПроектПоставкиСсылка,
ВыборкаХарактеристикаКомплекта.КоличествоКомплектов);

                        Если НЕ СборкаТоваровСсылка = Неопределено Тогда
                                ДокументыКУдалению.Вставить(СборкаТоваровСсылка, Ложь);
                        КонецЕсли;

                  КонецЦикла;
                КонецЦикла;

                Для Каждого КлючЗначение Из ДокументыКУдалению Цикл
                  Если КлючЗначение.Значение = Истина Тогда
                        СборкаТоваровОбъект = КлючЗначение.Ключ.ПолучитьОбъект();
                        СборкаТоваровОбъект.УстановитьПометкуУдаления(Истина);
                  КонецЕсли;
                КонецЦикла;

КонецПроцедуры

&НаСервере
Функция СоздатьОбновитьДокументыПриобретения(ПроектПоставки) Экспорт

                ИспользоватьТоварыВПути = Ложь;

                Запрос = Новый Запрос;
                Запрос.УстановитьПараметр("ПроектПоставки", ПроектПоставки);
                Запрос.Текст =
                "ВЫБРАТЬ РАЗЛИЧНЫЕ
                | ПроектПоставкиДокументыПоступлений.ПриобретениеТоваровУслуг КАК ПриобретениеТоваровУслуг
                |ИЗ
                | Документ.ПроектПоставки.ДокументыПоступлений КАК ПроектПоставкиДокументыПоступлений
                |ГДЕ
```

```
                | ПроектПоставкиДокументыПоступлений.Ссылка = &ПроектПоставки
                |;
                |
                |////////////////////////////////////////////////////////////////////////////
                |ВЫБРАТЬ
                | ТоварныеНакладныеПроекта.ПризнакТОРГ12 КАК ПризнакТОРГ12,
                | ЕСТЬNULL(ДокументыПоступленийПроекта.ПриобретениеТоваровУслуг, НЕОПРЕДЕЛЕНО) КАК
ПриобретениеТоваровУслуг
                |ИЗ
                | (ВЫБРАТЬ РАЗЛИЧНЫЕ
                |       ТоварыПроектовПоставки.ПризнакТОРГ12 КАК ПризнакТОРГ12
                | ИЗ
                |       РегистрСведений.ТоварыПроектовПоставки КАК ТоварыПроектовПоставки
                | ГДЕ
                |       ТоварыПроектовПоставки.ПроектПоставки = &ПроектПоставки) КАК ТоварныеНакладныеПроекта
                |       ЛЕВОЕ СОЕДИНЕНИЕ (ВЫБРАТЬ РАЗЛИЧНЫЕ
                |               ПроектПоставкиДокументыПоступлений.ПризнакТОРГ12 КАК ПризнакТОРГ12,
                |               ПроектПоставкиДокументыПоступлений.ПриобретениеТоваровУслуг КАК ПриобретениеТоваровУслуг
                |       ИЗ
                |               Документ.ПроектПоставки.ДокументыПоступлений КАК ПроектПоставкиДокументыПоступлений
                |       ГДЕ
                |               ПроектПоставкиДокументыПоступлений.Ссылка = &ПроектПоставки) КАК
ДокументыПоступленийПроекта
                |       ПО ТоварныеНакладныеПроекта.ПризнакТОРГ12 = ДокументыПоступленийПроекта.ПризнакТОРГ12";
                МассивРезультатов = Запрос.ВыполнитьПакет();

                ДокументыКУдалению = Новый Соответствие;
                Для Каждого ТекСтрока Из МассивРезультатов[0].Выгрузить() Цикл
                  ДокументыКУдалению.Вставить(ТекСтрока.ПриобретениеТоваровУслуг, Истина);
                КонецЦикла;

                ДокументыПоступлений = МассивРезультатов[1].Выгрузить();
                Для Каждого СтрокаДокумента Из ДокументыПоступлений Цикл

                  Если ИспользоватьТоварыВПути И ПроектПоставки.ИмпортнаяПоставка Тогда

                        Если СтрокаДокумента.ПриобретениеТоваровУслуг = Неопределено Тогда
                                // ПТУ не существует
                                //ВызватьИсключение "Не найден документ перехода права собственности";
                                СтрокаДокумента.ПриобретениеТоваровУслуг =
СоздатьОбновитьПриобретениеТоваровУслуг(Неопределено, ПроектПоставки.Ссылка, СтрокаДокумента.ПризнакТОРГ12, Истина);

                        Иначе
                                // ПТУ существует
                                // Возможны 2 варианта развития событий:
                                // 1. Обновляем ПТУ
                                // 2. Создаем корректировку приобретения
                                // 3. Обновляем корректировку приобретения

                                Если НЕ СтрокаДокумента.ПриобретениеТоваровУслуг.ТоварыВПути Тогда
                                        // 1. Обычная поставка, без вариантов, обновляем ПТУ
                                        СтрокаДокумента.ПриобретениеТоваровУслуг =
СоздатьОбновитьПриобретениеТоваровУслуг(СтрокаДокумента.ПриобретениеТоваровУслуг, ПроектПоставки,
СтрокаДокумента.ПризнакТОРГ12, Ложь);

                                Иначе
                                        // Товары в пути. Здесь уже возможны варианты - или обновляем ПТУ или создаем/обновляем
корректировку

                                        КорректировкаПриобретения   = Неопределено;

                                        // Найдем последнюю корректировку
                                        Запрос = Новый Запрос;
                                        Запрос.УстановитьПараметр("ДокументОснование",
СтрокаДокумента.ПриобретениеТоваровУслуг);
                                        Запрос.Текст =
                                        "ВЫБРАТЬ ПЕРВЫЕ 1
                                        |       КорректировкаПриобретения.Ссылка КАК Ссылка
                                        |ИЗ
                                        |       Документ.КорректировкаПриобретения КАК КорректировкаПриобретения
                                        |ГДЕ
                                        |       КорректировкаПриобретения.Проведен
                                        |       И КорректировкаПриобретения.ДокументОснование = &ДокументОснование
                                        |
                                        |УПОРЯДОЧИТЬ ПО
                                        |       КорректировкаПриобретения.МоментВремени УБЫВ";
```

```
                                        Выборка = Запрос.Выполнить().Выбрать();
                                        Если Выборка.Следующий() Тогда
                                            КорректировкаПриобретения = Выборка.Ссылка;
                                        КонецЕсли;

                                        ТекущийМесяц = НачалоМесяца(ТекущаяДата());

                                        Если КорректировкаПриобретения = Неопределено И ТекущийМесяц =
НачалоМесяца(СтрокаДокумента.ПриобретениеТоваровУслуг.Дата) Тогда
                                            // 1. Обновляем ПТУ
                                            //    Корректировок приобретения нет, изменения делаем в том же месяце, что и
первичный документ,
                                            //    поэтому просто перезаполним существующий документ "Приобретение товаров
услуг"
                                            СтрокаДокумента.ПриобретениеТоваровУслуг =
СоздатьОбновитьПриобретениеТоваровУслуг(СтрокаДокумента.ПриобретениеТоваровУслуг, ПроектПоставки,
СтрокаДокумента.ПризнакТОРГ12, Истина);

                                        Иначе

                                            // Работем с корректировками - или создаем новую или обновляем существующую
                                            Если НЕ КорректировкаПриобретения = Неопределено Тогда
                                                // Есть корректировка, проверим есть ли возможность обновить ее или придется
создать новую

                                                Если НЕ КорректировкаПриобретения.КорректировкаТоваровВПути Тогда
                                                    // То есть последняя корректировка к "Товарам в пути" не имеет
никакого отношения, значит с поставкой производились иные манипуляции,
                                                    // например кредит-нота, какие именно - одному богу известно. Поэтому
в этом случае посылаем пользователю сообщение об ошибке и ждем,
                                                    // когда он прибежит жаловаться...
                                                    ВызватьИсключение "Существуют  прочие корректировки
приобретения";
                                                КонецЕсли;

                                                Если НЕ ТекущийМесяц = НачалоМесяца(КорректировкаПриобретения.Дата)
Тогда
                                                    // 2. Создаем новую корректировку,
                                                    // т.к. последняя корректировка, была в месяце отличном от текущего
                                                    КорректировкаПриобретения = Неопределено;
                                                КонецЕсли;

                                            КонецЕсли;

                                            СоздатьОбновитьКорректировкуПриобретения(КорректировкаПриобретения,
СтрокаДокумента.ПриобретениеТоваровУслуг, ПроектПоставки, СтрокаДокумента.ПризнакТОРГ12);

                                        КонецЕсли;

                                    КонецЕсли;

                                КонецЕсли;

                            Иначе

                                // Товары в пути не используются или поставка не импортная
                                СтрокаДокумента.ПриобретениеТоваровУслуг =
СоздатьОбновитьПриобретениеТоваровУслуг(СтрокаДокумента.ПриобретениеТоваровУслуг, ПроектПоставки,
СтрокаДокумента.ПризнакТОРГ12, Ложь);

                            КонецЕсли;

                            ДокументыКУдалению.Вставить(СтрокаДокумента.ПриобретениеТоваровУслуг, Ложь);

                        КонецЦикла;

                        Для Каждого КлючЗначение Из ДокументыКУдалению Цикл
                            Если КлючЗначение.Значение = Истина Тогда
                                ДокументОбъект = КлючЗначение.Ключ.ПолучитьОбъект();
                                ДокументОбъект.УстановитьПометкуУдаления(Истина);
                            КонецЕсли;
                        КонецЦикла;

                        Возврат ОбщегоНазначения.ТаблицаЗначенийВМассив(ДокументыПоступлений);

КонецФункции
```

```
#КонецОбласти

#КонецОбласти

#Область СлужебныеПроцедурыИФункции

Функция КоличествоСвязанныхОбъектовОжидающихОбмена(ДокументПроектаПоставки, УзелИнформационнойБазы) Экспорт

		Запрос = Новый Запрос;
		Запрос.Текст =
		"ВЫБРАТЬ ПЕРВЫЕ 1
		| ПроектПоставкиИзменения.Ссылка КАК Ссылка
		|ИЗ
		| Документ.ПроектПоставки.Изменения КАК ПроектПоставкиИзменения
		|ГДЕ
		| ПроектПоставкиИзменения.Ссылка = &ДокументПроектаПоставки
		| И ПроектПоставкиИзменения.Узел = &УзелИнформационнойБазы";

		Запрос.УстановитьПараметр("ДокументПроектаПоставки"     , ДокументПроектаПоставки);

		Запрос.УстановитьПараметр("ПринудительноеРаспределение" ,
ОбщегоНазначения.ЗначениеРеквизитаОбъекта(ДокументПроектаПоставки, "ПринудительноеРаспределение"));

		Запрос.УстановитьПараметр("УзелИнформационнойБазы"      , УзелИнформационнойБазы);

		РезультатЗапроса = Запрос.Выполнить();

		Если РезультатЗапроса.Пустой() Тогда

		 Запрос.Текст =
		 "ВЫБРАТЬ
		 |	СУММА(КоличествоСвязанныхОбъектов.Количество) КАК Количество
		 |ИЗ
		 |	(ВЫБРАТЬ
		 |		КОЛИЧЕСТВО(ТоварыПроектовПоставкиИзменения.ПроектПоставки) КАК Количество
		 |	ИЗ
		 |		РегистрСведений.ТоварыПроектовПоставки.Изменения КАК ТоварыПроектовПоставкиИзменения
		 |	ГДЕ
		 |		ТоварыПроектовПоставкиИзменения.ПроектПоставки = &ДокументПроектаПоставки

		 |		И ТоварыПроектовПоставкиИзменения.Узел = &УзелИнформационнойБазы

		 |
		 |	ОБЪЕДИНИТЬ ВСЕ
		 |
		 |	ВЫБРАТЬ
		 |		КОЛИЧЕСТВО(РезультатыРаспределенияПроектаПоставкиИзменения.ПроектПоставки)
		 |	ИЗ

		 //|		РегистрСведений.РезультатыРаспределенияПроектаПоставки.Изменения КАК
РезультатыРаспределенияПроектаПоставкиИзменения
		 |		РегистрСведений.ПринудительноеРаспределение.Изменения КАК
РезультатыРаспределенияПроектаПоставкиИзменения

		 |	ГДЕ
		 |		РезультатыРаспределенияПроектаПоставкиИзменения.ПроектПоставки = &ДокументПроектаПоставки
		 \
		 //|		И
РезультатыРаспределенияПроектаПоставкиИзменения.ПроектПоставки.ПринудительноеРаспределение
		 |		И &ПринудительноеРаспределение


		 |		И РезультатыРаспределенияПроектаПоставкиИзменения.Узел = &УзелИнформационнойБазы

		 |	) КАК КоличествоСвязанныхОбъектов";

		Иначе

		 Запрос.Текст =
		 "ВЫБРАТЬ
		 |	СУММА(КоличествоСвязанныхОбъектов.Количество) +1 КАК Количество
		 |ИЗ
		 |	(ВЫБРАТЬ
		 |		КОЛИЧЕСТВО(ТоварыПроектовПоставки.ПроектПоставки) КАК Количество
		 |	ИЗ
		 |		РегистрСведений.ТоварыПроектовПоставки КАК ТоварыПроектовПоставки
```

```
|	ГДЕ
|		ТоварыПроектовПоставки.ПроектПоставки = &ДокументПроектаПоставки
|
|	ОБЪЕДИНИТЬ ВСЕ
|
|	ВЫБРАТЬ

//|		КОЛИЧЕСТВО(РаспределениеТоваров.Регистратор)
|		КОЛИЧЕСТВО(РаспределениеТоваров.ПроектПоставки)

|	ИЗ

//|		РегистрНакопления.РаспределениеТоваров КАК РаспределениеТоваров
//|	ГДЕ
//|		РаспределениеТоваров.Регистратор = &ДокументПроектаПоставки
|		РегистрСведений.ПринудительноеРаспределение КАК РаспределениеТоваров
|	ГДЕ
|		РаспределениеТоваров.ПроектПоставки = &ДокументПроектаПоставки

|		И РаспределениеТоваров.Количество <> 0
|		И &ПринудительноеРаспределение) КАК КоличествоСвязанныхОбъектов";


	КонецЕсли;

	возврат Запрос.Выполнить().Выгрузить()[0].Количество;

КонецФункции

&НаСервере
Функция НайтиВариантКомплектации(Номенклатура, Характеристика)

	Запрос = Новый Запрос;
	Запрос.УстановитьПараметр("Номенклатура",    Номенклатура);
	Запрос.УстановитьПараметр("Характеристика", Характеристика);
	Запрос.Текст =
	"ВЫБРАТЬ РАЗРЕШЕННЫЕ РАЗЛИЧНЫЕ ПЕРВЫЕ 1
	| ВариантыКомплектацииНоменклатуры.Ссылка КАК Ссылка,
	| ВариантыКомплектацииНоменклатуры.Основной КАК Основной
	|ИЗ
	| Справочник.ВариантыКомплектацииНоменклатуры КАК ВариантыКомплектацииНоменклатуры
	|ГДЕ
	| НЕ ВариантыКомплектацииНоменклатуры.ПометкаУдаления
	| И ВариантыКомплектацииНоменклатуры.Владелец = &Номенклатура
	| И ВариантыКомплектацииНоменклатуры.Характеристика = &Характеристика
	|
	|УПОРЯДОЧИТЬ ПО
	| Основной УБЫВ";
	Выборка = Запрос.Выполнить().Выбрать();
	Пока Выборка.Следующий() Цикл
	  Возврат Выборка.Ссылка;
	КонецЦикла;

	Возврат Неопределено;

КонецФункции

// Считаем что дата сборки - дата первого ПКО
//
&НаСервере
Функция ОпределитьДатуСборкиКомплекта(ПроектПоставки)

	Запрос = Новый Запрос;
	Запрос.УстановитьПараметр("ПроектПоставки", ПроектПоставки);
	Запрос.Текст =
	"ВЫБРАТЬ РАЗРЕШЕННЫЕ РАЗЛИЧНЫЕ ПЕРВЫЕ 1
	| ПриходныйОрдерНаТовары.Дата КАК Дата
	|ИЗ
	| Документ.ПриходныйОрдерНаТовары КАК ПриходныйОрдерНаТовары
	|ГДЕ
	| ПриходныйОрдерНаТовары.Проведен
	| И (ПриходныйОрдерНаТовары.Распоряжение ССЫЛКА Документ.ПроектПоставки
	|		И ПриходныйОрдерНаТовары.Распоряжение = &ПроектПоставки
	|	ИЛИ ПриходныйОрдерНаТовары.Распоряжение ССЫЛКА Документ.ПриобретениеТоваровУслуг
	|		И ПриходныйОрдерНаТовары.Распоряжение.ПроектПоставки = &ПроектПоставки)
	|
```

```
		|УПОРЯДОЧИТЬ ПО
		| Дата";

		Выборка = Запрос.Выполнить().Выбрать();
		Если Выборка.Следующий() Тогда
		  Возврат Выборка.Дата;
		КонецЕсли;

		Возврат ТекущаяДата();

КонецФункции

#Область Печать

// Заполняет список команд печати.
//
// Параметры:
//   КомандыПечати - ТаблицаЗначений - состав полей см. в функции УправлениеПечатью.СоздатьКоллекциюКомандПечати
//
Процедура ДобавитьКомандыПечати(КомандыПечати) Экспорт

		// Таблица товаров проектов поставки
		КомандаПечати = КомандыПечати.Добавить();
		КомандаПечати.Идентификатор = "ТоварыПроектаПоставки";
		КомандаПечати.Представление = НСтр("ru = 'Товары проекта поставки'");


		КомандаПечати = КомандыПечати.Добавить();
		КомандаПечати.Идентификатор = "СогласованиеРасхожденийСПоставщиком";
		КомандаПечати.Представление = НСтр("ru = 'Согласование расхождений с поставщиком'");

КонецПроцедуры

// Сформировать печатные формы объектов
//
// ВХОДЯЩИЕ:
//   ИменаМакетов    - Строка    - Имена макетов, перечисленные через запятую
//   МассивОбъектов  - Массив    - Массив ссылок на объекты которые нужно распечатать
//   ПараметрыПечати - Структура - Структура дополнительных параметров печати
//
// ИСХОДЯЩИЕ:
//   КоллекцияПечатныхФорм - Таблица значений - Сформированные табличные документы
//   ПараметрыВывода       - Структура        - Параметры сформированных табличных документов
//
Процедура Печать(МассивОбъектов, ПараметрыПечати, КоллекцияПечатныхФорм, ОбъектыПечати, ПараметрыВывода) Экспорт

		Если УправлениеПечатью.НужноПечататьМакет(КоллекцияПечатныхФорм, "ТоварыПроектаПоставки") Тогда
		  УправлениеПечатью.ВывестиТабличныйДокументВКоллекцию(КоллекцияПечатныхФорм, "ТоварыПроектаПоставки",
НСтр("ru='Товары проекта поставки'"), СформироватьПечатнуюФормуТоваровПроектаПоставки(МассивОбъектов, ОбъектыПечати));
		КонецЕсли;


		Если УправлениеПечатью.НужноПечататьМакет(КоллекцияПечатныхФорм,
"СогласованиеРасхожденийСПоставщиком") Тогда
		  УправлениеПечатью.ВывестиТабличныйДокументВКоллекцию(КоллекцияПечатныхФорм,
"СогласованиеРасхожденийСПоставщиком", НСтр("ru='Согласование расхождений с поставщиком'"),
СформироватьПечатнуюФормуСогласованиеРасхожденийСПоставщиком(МассивОбъектов, ОбъектыПечати),,,"Согласование
расхождений");
		КонецЕсли;

		ФормированиеПечатныхФорм.ЗаполнитьПараметрыОтправки(ПараметрыВывода.ПараметрыОтправки,
МассивОбъектов, КоллекцияПечатныхФорм);

КонецПроцедуры

Функция СформироватьПечатнуюФормуТоваровПроектаПоставки(МассивОбъектов, ОбъектыПечати)

		УстановитьПривилегированныйРежим(Истина);

		ТабличныйДокумент = Новый ТабличныйДокумент;
		ТабличныйДокумент.АвтоМасштаб = Истина;
		ТабличныйДокумент.ИмяПараметровПечати = "ПАРАМЕТРЫ_ПЕЧАТИ_ПроектПоставки_Товары";

		Запрос = Новый Запрос;
		Запрос.Текст =
		"ВЫБРАТЬ РАЗЛИЧНЫЕ
```

```
| ТоварыПроектовПоставки.ПроектПоставки КАК ПроектПоставки,
| ТоварыПроектовПоставки.СтрокаНомер КАК НомерСтроки,
| ТоварыПроектовПоставки.Номенклатура,
| ТоварыПроектовПоставки.Характеристика,
| ТоварыПроектовПоставки.Количество,
| ТоварыПроектовПоставки.РекомендованнаяЦена,
| ТоварыПроектовПоставки.ЗакупочнаяЦена,
| ТоварыПроектовПоставки.РозничнаяЦена,
| ТоварыПроектовПоставки.СуммаЗакупочная,
| ТоварыПроектовПоставки.СуммаРозничная,
| ТоварыПроектовПоставки.СуммаРекомендованная,
| ТоварыПроектовПоставки.СкидкаЗЦОтРРЦ,
| ТоварыПроектовПоставки.СкидкаРЦОтРРЦ,
| ТоварыПроектовПоставки.Наценка,
| ТоварыПроектовПоставки.СтавкаНДС,
| ТоварыПроектовПоставки.СуммаНДС,
| ТоварыПроектовПоставки.ПроцентНаДХ,
| ТоварыПроектовПоставки.ДатаВыдачиСДХ,
| ТоварыПроектовПоставки.НомерКороба,
| ТоварыПроектовПоставки.ПризнакТОРГ12,
| ТоварыПроектовПоставки.АртикулИсходный,

//|	ТоварыПроектовПоставки.Артикул,
| ТоварыПроектовПоставки.Номенклатура.Артикул КАК Артикул,
| ТоварыПроектовПоставки.Наименование,
| ТоварыПроектовПоставки.Пол,
| ТоварыПроектовПоставки.Цвет,
| ТоварыПроектовПоставки.РазмерПоставщика,
| ТоварыПроектовПоставки.Поставщик,
| ТоварыПроектовПоставки.ТоварнаяКатегория,
| ТоварыПроектовПоставки.Марка,
| ТоварыПроектовПоставки.ТипБренда,
| ТоварыПроектовПоставки.СезонПоставщика,
//|	ТоварыПроектовПоставки.Сезон,

//|	ТоварыПроектовПоставки.Качество,
| ВЫБОР
|	КОГДА ТоварыПроектовПоставки.Качество = ЗНАЧЕНИЕ(Перечисление.ГрадацииКачества.Новый)
|		ТОГДА ""Регулярный товар""
|	ИНАЧЕ ""Брак""
| КОНЕЦ КАК Качество,

| ТоварыПроектовПоставки.Состав,
| ТоварыПроектовПоставки.СтранаПроисхождения,
| ТоварыПроектовПоставки.Штрихкод,
| ТоварыПроектовПоставки.ДополнительныйПризнак,
| ТоварыПроектовПоставки.КоллекцияНоменклатуры,
| ТоварыПроектовПоставки.ПризнакМаркировки,

| ТоварыПроектовПоставки.Номенклатура.Производитель.Наименование как Производитель,
| ТоварыПроектовПоставки.Номенклатура.Производитель.Адрес как АдресПроизводителя
//|	ТоварыПроектовПоставки.КиЗГИСМGTIN как GTIN

|ИЗ
| РегистрСведений.ТоварыПроектовПоставки КАК ТоварыПроектовПоставки
|ГДЕ
| ТоварыПроектовПоставки.ПроектПоставки В(&МассивОбъектов)
|
|УПОРЯДОЧИТЬ ПО
| НомерСтроки
|ИТОГИ ПО
| ПроектПоставки";

Запрос.УстановитьПараметр("МассивОбъектов", МассивОбъектов);

Макет = УправлениеПечатью.МакетПечатнойФормы("Документ.ПроектПоставки.ШаблонЗагрузкиПроекта");

ПервыйДокумент = Истина;

ВыборкаПоДокументам = Запрос.Выполнить().Выбрать(ОбходРезультатаЗапроса.ПоГруппировкам);
Пока ВыборкаПоДокументам.Следующий() Цикл

  Если Не ПервыйДокумент Тогда
	ТабличныйДокумент.ВывестиГоризонтальныйРазделительСтраниц();
  КонецЕсли;
```

```
                ПервыйДокумент = Ложь;

                Область = Макет.ПолучитьОбласть("Заголовок");
                ТабличныйДокумент.Вывести(Область);

                ВыборкаПоТоварам = ВыборкаПоДокументам.Выбрать();
                Пока ВыборкаПоТоварам.Следующий() Цикл

                        Область = Макет.ПолучитьОбласть("СтрокаТаблицы");
                        Область.Параметры.Заполнить(ВыборкаПоТоварам);

                        Область.Параметры.ДатаВыдачиСДХ = Формат(ВыборкаПоТоварам.ДатаВыдачиСДХ, "ДФ=dd.MM.yyyy");
                        ТекСвойство = ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.ДистрибьютерИмпортер;
                        СтрокаСоСвойством =
ВыборкаПоТоварам.Характеристика.ДополнительныеРеквизиты.Найти(ТекСвойство,"Свойство");
                        Если СтрокаСоСвойством <> Неопределено и ЗначениеЗаполнено(СтрокаСоСвойством.Значение) Тогда

                                Область.Параметры.ДистрибьютерИмпортер = СтрокаСоСвойством.Значение.Наименование;
                                Область.Параметры.АдресДистрибьютераИмпортера = СтрокаСоСвойством.Значение.Адрес;
                        КонецЕсли;

                        ТабличныйДокумент.Вывести(Область);

                КонецЦикла;

                КонецЦикла;

                Если ПривилегированныйРежим() Тогда
                  УстановитьПривилегированныйРежим(Ложь);
                КонецЕсли;

                Возврат ТабличныйДокумент;

КонецФункции

Функция СформироватьПечатнуюФормуСогласованиеРасхожденийСПоставщиком(МассивОбъектов, ОбъектыПечати)

                УстановитьПривилегированныйРежим(Истина);

                ТабличныйДокумент = Новый ТабличныйДокумент;
                ТабличныйДокумент.АвтоМасштаб = Истина;
                ТабличныйДокумент.ИмяПараметровПечати = "ПАРАМЕТРЫ_ПЕЧАТИ_ПроектПоставки_СогласованиеРасхождений";

                ЗапросОрдСхема = Новый Запрос;
                ЗапросОрдСхема.Текст =
                "ВЫБРАТЬ
                | ПроектПоставки.Ссылка КАК Ссылка
                |ИЗ
                | Документ.ПроектПоставки КАК ПроектПоставки
                |ГДЕ
                | ПроектПоставки.Склад.ИспользоватьОрдернуюСхемуПриПоступлении = ИСТИНА
                | И ПроектПоставки.Ссылка В(&МассивОбъектов)";

                ЗапросОрдСхема.УстановитьПараметр("МассивОбъектов", МассивОбъектов);
                Результат = ЗапросОрдСхема.Выполнить();

                Если Результат.Пустой()=Ложь Тогда

                  ВыборкаПроект = Результат.Выбрать();
                  ВыборкаПроект.Следующий();
                  ТекПроект = ВыборкаПроект.Ссылка;

                  ЗапросПО = Новый Запрос;
                  ЗапросПО.Текст =
                  "ВЫБРАТЬ РАЗРЕШЕННЫЕ
                  |       ПриходныйОрдерНаТовары.Ссылка КАК ПриходныйОрдер
                  |ИЗ
                  |       Документ.ПриходныйОрдерНаТовары КАК ПриходныйОрдерНаТовары
                  |ГДЕ
                  |       ПриходныйОрдерНаТовары.Распоряжение В (&МассивОбъектов)
                  |       И ПриходныйОрдерНаТовары.Проведен = ИСТИНА";

                  ЗапросПО.УстановитьПараметр("МассивОбъектов", МассивОбъектов);
                  РезультатПО = ЗапросПО.Выполнить();

                  Если РезультатПО.Пустой() Тогда
```

```
                ЗапросПО.Текст =
                "ВЫБРАТЬ РАЗРЕШЕННЫЕ
                |       ПриходныйОрдерНаТовары.Ссылка КАК ПриходныйОрдер
                |ИЗ
                |       Документ.ПриходныйОрдерНаТовары КАК ПриходныйОрдерНаТовары
                |ГДЕ
                |       ПриходныйОрдерНаТовары.Распоряжение В
                |               (ВЫБРАТЬ
                |                       ПроектПоставкиДокументыПоступлений.ПриобретениеТоваровУслуг КАК ПТУ
                |               ИЗ
                |                       Документ.ПроектПоставки.ДокументыПоступлений КАК
ПроектПоставкиДокументыПоступлений
                |               ГДЕ
                |                       ПроектПоставкиДокументыПоступлений.Ссылка В (&МассивОбъектов))
                |       И ПриходныйОрдерНаТовары.Проведен = ИСТИНА";

                ЗапросПО.УстановитьПараметр("МассивОбъектов", МассивОбъектов);
                РезультатПО = ЗапросПО.Выполнить();
                МассивПО = РезультатПО.Выгрузить().ВыгрузитьКолонку("ПриходныйОрдер");

                Если РезультатПО.Пустой() Тогда

                        МассивПО = Новый Массив;

                КонецЕсли;

        иначе

                МассивПО = РезультатПО.Выгрузить().ВыгрузитьКолонку("ПриходныйОрдер");

        КонецЕсли;

        Запрос = Новый Запрос;
        Запрос.Текст =
        "ВЫБРАТЬ
        |       ПроектПоставкиДокументыПоступлений.ПриобретениеТоваровУслуг КАК Поступление,
        |       ПроектПоставкиДокументыПоступлений.Ссылка КАК ПроектПоставки,
        |       ПроектПоставкиДокументыПоступлений.ПризнакТОРГ12 КАК ПризнакТОРГ12
        |ПОМЕСТИТЬ ПТУ
        |ИЗ
        |       Документ.ПроектПоставки.ДокументыПоступлений КАК ПроектПоставкиДокументыПоступлений
        |ГДЕ
        |       ПроектПоставкиДокументыПоступлений.Ссылка В(&МассивОбъектов)
        |;
        |
        |////////////////////////////////////////////////////////////////////////////
        |ВЫБРАТЬ
        |       ППТовары.ПроектПоставки КАК ПроектПоставки,
        |       ППТовары.Номенклатура КАК Номенклатура,
        |       ППТовары.Характеристика КАК Характеристика,
        |       МАКСИМУМ(ППТовары.Штрихкод) КАК ШтрихКод,
        |       МАКСИМУМ(ППТовары.РазмерПоставщика) КАК Размер,
        |       МАКСИМУМ(ППТовары.Цвет) КАК Цвет,
        |       МАКСИМУМ(ППТовары.Наименование) КАК Наименование
        |ПОМЕСТИТЬ ТоварыПроектов
        |ИЗ
        |       РегистрСведений.ТоварыПроектовПоставки КАК ППТовары
        |ГДЕ
        |       ППТовары.ПроектПоставки В(&МассивОбъектов)
        |
        |СГРУППИРОВАТЬ ПО
        |       ППТовары.ПроектПоставки,
        |       ППТовары.Номенклатура,
        |       ППТовары.Характеристика
        |;
        |
        |////////////////////////////////////////////////////////////////////////////
        |ВЫБРАТЬ
        |       ПТУ.ПроектПоставки КАК ПроектПоставки,
        |       МАКСИМУМ(ПТУ.ПризнакТОРГ12) КАК ТОРГ12,
        |       ПриобретениеТоваровУслугТовары.Номенклатура КАК Номенклатура,
        |       ПриобретениеТоваровУслугТовары.Характеристика КАК Характеристика,
        |       СУММА(ПриобретениеТоваровУслугТовары.Количество) КАК Количество,
        |       МАКСИМУМ(ПриобретениеТоваровУслугТовары.Цена) КАК Цена,
        |       СУММА(ПриобретениеТоваровУслугТовары.Сумма) КАК Сумма,
        |       МАКСИМУМ(ПриобретениеТоваровУслугТовары.Номенклатура.Артикул) КАК Артикул,
```

```
|	СУММА(ПриобретениеТоваровУслугТовары.СуммаСНДС) КАК СуммаСНДС
|ПОМЕСТИТЬ ПЛАН
|ИЗ
|	Документ.ПриобретениеТоваровУслуг.Товары КАК ПриобретениеТоваровУслугТовары
|		ЛЕВОЕ СОЕДИНЕНИЕ ПТУ КАК ПТУ
|		ПО ПриобретениеТоваровУслугТовары.Ссылка = ПТУ.Поступление
|ГДЕ
|	ПриобретениеТоваровУслугТовары.Ссылка В
|			(ВЫБРАТЬ
|				Поступления.ПриобретениеТоваровУслуг
|			ИЗ
|				Документ.ПроектПоставки.ДокументыПоступлений КАК Поступления
|			ГДЕ
|				Поступления.Ссылка В (&МассивОбъектов))
|
|СГРУППИРОВАТЬ ПО
|	ПТУ.ПроектПоставки,
|	ПриобретениеТоваровУслугТовары.Номенклатура,
|	ПриобретениеТоваровУслугТовары.Характеристика
|;
|
|////////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|	&ПроектПоставки КАК ПроектПоставки,
|	ПОТовары.Номенклатура КАК Номенклатура,
|	ПОТовары.Характеристика КАК Характеристика,
|	СУММА(ПОТовары.Количество) КАК Количество,
|	МАКСИМУМ(ПОТовары.Номенклатура.Артикул) КАК Артикул
|ПОМЕСТИТЬ ФАКТ
|ИЗ
|	Документ.ПриходныйОрдерНаТовары.Товары КАК ПОТовары
|ГДЕ
|	ПОТовары.Ссылка В(&МассивПО)
|
|СГРУППИРОВАТЬ ПО
|	&ПроектПоставки,
|	ПОТовары.Номенклатура,
|	ПОТовары.Характеристика
|;
|
|////////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|	&ПроектПоставки КАК ПроектПоставки,
|	ПОТовары.Номенклатура КАК Номенклатура,
|	ПОТовары.Характеристика КАК Характеристика,
|	МАКСИМУМ(ХарактеристикиЦвет.Значение) КАК Цвет,
|	МАКСИМУМ(ХарактеристикиРазмер.Значение) КАК Размер
|ПОМЕСТИТЬ ФАКТЦВЕТАРАЗМЕРЫ
|ИЗ
|	Документ.ПриходныйОрдерНаТовары.Товары КАК ПОТовары
|		ЛЕВОЕ СОЕДИНЕНИЕ Справочник.ХарактеристикиНоменклатуры.ДополнительныеРеквизиты КАК
ХарактеристикиЦвет
|		ПО ПОТовары.Характеристика = ХарактеристикиЦвет.Ссылка
|			И (ЗНАЧЕНИЕ(ПланВидовХарактеристик.ДополнительныеРеквизитыИСведения.Цвет) =
ХарактеристикиЦвет.Свойство)
|		ЛЕВОЕ СОЕДИНЕНИЕ Справочник.ХарактеристикиНоменклатуры.ДополнительныеРеквизиты КАК
ХарактеристикиРазмер
|		ПО ПОТовары.Характеристика = ХарактеристикиРазмер.Ссылка
|			И (ЗНАЧЕНИЕ(ПланВидовХарактеристик.ДополнительныеРеквизитыИСведения.Размер) =
ХарактеристикиРазмер.Свойство)
|ГДЕ
|	ПОТовары.Ссылка В(&МассивПО)
|
|СГРУППИРОВАТЬ ПО
|	&ПроектПоставки,
|	ПОТовары.Номенклатура,
|	ПОТовары.Характеристика
|;
|
|////////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|	ВложенныйЗапрос.ПроектПоставки КАК ПроектПоставки,
|	МАКСИМУМ(ТоварыПроектов.ШтрихКод) КАК ШтрихКод,
|	МАКСИМУМ(ВложенныйЗапрос.Артикул) КАК Артикул,
|	МАКСИМУМ(ВложенныйЗапрос.Номенклатура) КАК Наименование,
|	ЕСТЬNULL(МАКСИМУМ(ТоварыПроектов.Цвет), МАКСИМУМ(ФактЦветаРазмеры.Цвет)) КАК Цвет,
```

```
|	ЕСТЬNULL(МАКСИМУМ(ТоварыПроектов.Размер), МАКСИМУМ(ФактЦветаРазмеры.Размер)) КАК Размер,
|	МАКСИМУМ(ВложенныйЗапрос.Торг12) КАК Торг12,
|	МАКСИМУМ(ВложенныйЗапрос.ЗЦ) КАК ЗЦена,
|	СУММА(ВложенныйЗапрос.КоличествоПлан) КАК КолвоПлан,
|	СУММА(ВложенныйЗапрос.ЗЦ * ВложенныйЗапрос.КоличествоПлан) КАК СуммаЗЦПлан,
|	СУММА(ВложенныйЗапрос.КоличествоФакт) КАК КолвоФакт,
|	СУММА(ВложенныйЗапрос.КоличествоФакт - ВложенныйЗапрос.КоличествоПлан) КАК РасхожденияШт,
|	СУММА(ВложенныйЗапрос.СуммаСНДСПлан) КАК СуммаСНДСПлан
|ИЗ
|	(ВЫБРАТЬ
|		План.ПроектПоставки КАК ПроектПоставки,
|		План.Артикул КАК Артикул,
|		План.ТОРГ12 КАК Торг12,
|		План.Номенклатура КАК Номенклатура,
|		План.Характеристика КАК Характеристика,
|		План.Количество КАК КоличествоПлан,
|		План.Цена КАК ЗЦ,
|		0 КАК КоличествоФакт,
|		План.СуммаСНДС КАК СуммаСНДСПлан
|	ИЗ
|		ПЛАН КАК План
|
|	ОБЪЕДИНИТЬ ВСЕ
|
|	ВЫБРАТЬ
|		Факт.ПроектПоставки,
|		Факт.Артикул,
|		0,
|		Факт.Номенклатура,
|		Факт.Характеристика,
|		0,
|		0,
|		Факт.Количество,
|		0
|	ИЗ
|		ФАКТ КАК Факт) КАК ВложенныйЗапрос
|		ЛЕВОЕ СОЕДИНЕНИЕ ТоварыПроектов КАК ТоварыПроектов
|		ПО ВложенныйЗапрос.ПроектПоставки = ТоварыПроектов.ПроектПоставки
|			И ВложенныйЗапрос.Номенклатура = ТоварыПроектов.Номенклатура
|			И ВложенныйЗапрос.Характеристика = ТоварыПроектов.Характеристика
|		ЛЕВОЕ СОЕДИНЕНИЕ ФАКТЦВЕТАРАЗМЕРЫ КАК ФактЦветаРазмеры
|		ПО ВложенныйЗапрос.ПроектПоставки = ФактЦветаРазмеры.ПроектПоставки
|			И ВложенныйЗапрос.Номенклатура = ФактЦветаРазмеры.Номенклатура
|			И ВложенныйЗапрос.Характеристика = ФактЦветаРазмеры.Характеристика
|
|СГРУППИРОВАТЬ ПО
|	ВложенныйЗапрос.ПроектПоставки,
|	ВложенныйЗапрос.Номенклатура,
|	ВложенныйЗапрос.Характеристика
|ИТОГИ
|	СУММА(КолвоПлан),
|	СУММА(СуммаЗЦПлан),
|	СУММА(КолвоФакт),
|	СУММА(РасхожденияШт)
|ПО
|	ПроектПоставки";

Запрос.УстановитьПараметр("ПроектПоставки", ТекПроект);
Запрос.УстановитьПараметр("МассивПО", МассивПО);
Запрос.УстановитьПараметр("МассивОбъектов", МассивОбъектов);

Иначе


//Источником являются документы Приобретение товаров и услуг
Запрос = Новый Запрос;
Запрос.Текст =
"ВЫБРАТЬ
|	ПроектПоставкиДокументыПоступлений.ПриобретениеТоваровУслуг КАК Поступление,
|	ПроектПоставкиДокументыПоступлений.Ссылка КАК ПроектПоставки,
|	ПроектПоставкиДокументыПоступлений.ПризнакТОРГ12 КАК ПризнакТОРГ12
|ПОМЕСТИТЬ ПТУ
|ИЗ
|	Документ.ПроектПоставки.ДокументыПоступлений КАК ПроектПоставкиДокументыПоступлений
|ГДЕ
|	ПроектПоставкиДокументыПоступлений.Ссылка В(&МассивОбъектов)
```

```
|;
|
////////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|	ППТовары.ПроектПоставки КАК ПроектПоставки,
|	ППТовары.Номенклатура КАК Номенклатура,
|	ППТовары.Характеристика КАК Характеристика,
|	МАКСИМУМ(ППТовары.Штрихкод) КАК ШтрихКод,
|	МАКСИМУМ(ППТовары.РазмерПоставщика) КАК Размер,
|	МАКСИМУМ(ППТовары.Цвет) КАК Цвет,
|	МАКСИМУМ(ППТовары.Наименование) КАК Наименование
|ПОМЕСТИТЬ ТоварыПроектов
|ИЗ
|	РегистрСведений.ТоварыПроектовПоставки КАК ППТовары
|ГДЕ
|	ППТовары.ПроектПоставки В(&МассивОбъектов)
|
|СГРУППИРОВАТЬ ПО
|	ППТовары.ПроектПоставки,
|	ППТовары.Номенклатура,
|	ППТовары.Характеристика
|;
|
////////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|	ПТУ.ПроектПоставки КАК ПроектПоставки,
|	МАКСИМУМ(ПТУ.ПризнакТОРГ12) КАК ТОРГ12,
|	ПриобретениеТоваровУслугТовары.Номенклатура КАК Номенклатура,
|	ПриобретениеТоваровУслугТовары.Характеристика КАК Характеристика,
|	СУММА(ПриобретениеТоваровУслугТовары.Количество) КАК Количество,
|	МАКСИМУМ(ПриобретениеТоваровУслугТовары.Цена) КАК Цена,
|	СУММА(ПриобретениеТоваровУслугТовары.Сумма) КАК Сумма,
|	СУММА(ПриобретениеТоваровУслугТовары.СуммаСНДС) КАК СуммаСНДС,
|	МАКСИМУМ(ПриобретениеТоваровУслугТовары.Номенклатура.Артикул) КАК Артикул
|ПОМЕСТИТЬ ПЛАН
|ИЗ
|	Документ.ПриобретениеТоваровУслуг.Товары КАК ПриобретениеТоваровУслугТовары
|		ЛЕВОЕ СОЕДИНЕНИЕ ПТУ КАК ПТУ
|		ПО ПриобретениеТоваровУслугТовары.Ссылка = ПТУ.Поступление
|ГДЕ
|	ПриобретениеТоваровУслугТовары.Ссылка В
|			(ВЫБРАТЬ
|				Поступления.ПриобретениеТоваровУслуг
|			ИЗ
|				Документ.ПроектПоставки.ДокументыПоступлений КАК Поступления
|			ГДЕ
|				Поступления.Ссылка В (&МассивОбъектов))
|
|СГРУППИРОВАТЬ ПО
|	ПТУ.ПроектПоставки,
|	ПриобретениеТоваровУслугТовары.Номенклатура,
|	ПриобретениеТоваровУслугТовары.Характеристика
|;
|
////////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|	ПринятыеТовары.ПроектПоставки КАК ПроектПоставки,
|	ПринятыеТовары.Номенклатура КАК Номенклатура,
|	ПринятыеТовары.Характеристика КАК Характеристика,
|	СУММА(ПринятыеТовары.Количество) КАК Количество,
|   МАКСИМУМ(ПринятыеТовары.Номенклатура.Артикул) КАК Артикул
|ПОМЕСТИТЬ Факт
|ИЗ
|	РегистрСведений.ПринятыеТоварыПроектовПоставки КАК ПринятыеТовары
|ГДЕ
|	ПринятыеТовары.ПроектПоставки В(&МассивОбъектов)
|
|СГРУППИРОВАТЬ ПО
|	ПринятыеТовары.ПроектПоставки,
|	ПринятыеТовары.Номенклатура,
|	ПринятыеТовары.Характеристика
|;
|
////////////////////////////////////////////////////////////////////////////////
|ВЫБРАТЬ
|	ВложенныйЗапрос.ПроектПоставки КАК ПроектПоставки,
```

```
|	МАКСИМУМ(ТоварыПроектов.ШтрихКод) КАК ШтрихКод,
|	МАКСИМУМ(ВложенныйЗапрос.Артикул) КАК Артикул,
|	МАКСИМУМ(ТоварыПроектов.Наименование) КАК Наименование,
|	МАКСИМУМ(ТоварыПроектов.Цвет) КАК Цвет,
|	МАКСИМУМ(ТоварыПроектов.Размер) КАК Размер,
|	МАКСИМУМ(ВложенныйЗапрос.Торг12) КАК Торг12,
|	МАКСИМУМ(ВложенныйЗапрос.ЗЦ) КАК ЗЦена,
|	СУММА(ВложенныйЗапрос.КоличествоПлан) КАК КолвоПлан,
|	СУММА(ВложенныйЗапрос.Сумма) КАК СуммаЗЦПлан,
|	СУММА(ВложенныйЗапрос.СуммаСНДС) КАК СуммаСНДСПлан,
|	СУММА(ВложенныйЗапрос.КоличествоФакт) КАК КолвоФакт,
|	СУММА(ВложенныйЗапрос.КоличествоФакт - ВложенныйЗапрос.КоличествоПлан) КАК РасхожденияШт
ИЗ
|	(ВЫБРАТЬ
|		План.ПроектПоставки КАК ПроектПоставки,
|		План.Артикул КАК Артикул,
|		План.ТОРГ12 КАК Торг12,
|		План.Номенклатура КАК Номенклатура,
|		План.Характеристика КАК Характеристика,
|		План.Количество КАК КоличествоПлан,
|		План.Цена КАК ЗЦ,
|	План.Сумма КАК Сумма,
|	План.СуммаСНДС КАК СуммаСНДС,
|		0 КАК КоличествоФакт
|	ИЗ
|		ПЛАН КАК План
|
|	ОБЪЕДИНИТЬ ВСЕ
|
|	ВЫБРАТЬ
|		Факт.ПроектПоставки,
|		Факт.Артикул,
|		0,
|		Факт.Номенклатура,
|		Факт.Характеристика,
|		0,
|		0,
|	0,
|	0,
|		Факт.Количество
|	ИЗ
|		ФАКТ КАК Факт) КАК ВложенныйЗапрос
|		ЛЕВОЕ СОЕДИНЕНИЕ ТоварыПроектов КАК ТоварыПроектов
|		ПО ВложенныйЗапрос.ПроектПоставки = ТоварыПроектов.ПроектПоставки
|			И ВложенныйЗапрос.Номенклатура = ТоварыПроектов.Номенклатура
|			И ВложенныйЗапрос.Характеристика = ТоварыПроектов.Характеристика
|
СГРУППИРОВАТЬ ПО
|	ВложенныйЗапрос.ПроектПоставки,
|	ВложенныйЗапрос.Номенклатура,
|	ВложенныйЗапрос.Характеристика
ИТОГИ
|	СУММА(КолвоПлан),
|	СУММА(СуммаЗЦПлан),
|	СУММА(КолвоФакт),
|	СУММА(РасхожденияШт)
ПО
|	ПроектПоставки";

 Запрос.УстановитьПараметр("МассивОбъектов", МассивОбъектов);

КонецЕсли;

//	Запрос.УстановитьПараметр("МассивОбъектов", МассивОбъектов);

Макет =
УправлениеПечатью.МакетПечатнойФормы("Документ.ПроектПоставки.ПФ_MXL_СогласованиеРасхождений СПоставщиком");
ПервыйДокумент = Истина;

РезультатЗапроса = Запрос.Выполнить();
ВыборкаДок = РезультатЗапроса.Выбрать(ОбходРезультатаЗапроса.ПоГруппировкам);

Пока ВыборкаДок.Следующий() Цикл

 Если Не ПервыйДокумент Тогда
	ТабличныйДокумент.ВывестиГоризонтальныйРазделительСтраниц();
```

```
КонецЕсли;

ПервыйДокумент = Ложь;

Область = Макет.ПолучитьОбласть("Шапка");
ТабличныйДокумент.Вывести(Область);

ИтогКолвоПлан = ВыборкаДок.КолвоПлан;
ИтогКолвоФакт = 0;
ИтогСуммаПлан = ВыборкаДок.СуммаЗЦПлан;
ИтогСуммаФакт = 0;
ИтогРасхожденияШт = ВыборкаДок.РасхожденияШт;
ИтогРасхожденияРуб = 0;
ИтогСуммаСНДСПлан = 0;
ИтогСуммаСНДСФакт = 0;

ВыборкаПоТоварам = ВыборкаДок.Выбрать();
Пока ВыборкаПоТоварам.Следующий() Цикл

        Область = Макет.ПолучитьОбласть("Строка");
        Область.Параметры.Заполнить(ВыборкаПоТоварам);

        Если ВыборкаПоТоварам.КолвоПлан>0 Тогда
                //Без НДС
                ЗакупЦена = ВыборкаПоТоварам.СуммаЗЦПлан/ВыборкаПоТоварам.КолвоПлан;
                СуммаЗЦФакт = ВыборкаПоТоварам.КолвоФакт*ЗакупЦена;
                Область.Параметры.СуммаЗЦФакт = СуммаЗЦФакт;
        иначе
                СуммаЗЦФакт=0;
        КонецЕсли;

        Если ВыборкаПоТоварам.КолвоПлан>0 Тогда
                //С НДС
                ЗакупЦенаНДС = ВыборкаПоТоварам.СуммаСНДСПлан/ВыборкаПоТоварам.КолвоПлан;
                СуммаСНДСФакт = ВыборкаПоТоварам.КолвоФакт*ЗакупЦенаНДС;
                Область.Параметры.СуммаСНДСФакт = СуммаСНДСФакт;
                Область.Параметры.ЗЦена = ЗакупЦенаНДС;
        иначе
                СуммаСНДСФакт = 0;
        КонецЕсли;

        //РазностьРуб = СуммаЗЦФакт - ВыборкаПоТоварам.СуммаЗЦПлан;

        РазностьРуб = СуммаСНДСФакт - ВыборкаПоТоварам.СуммаСНДСПлан;

        Область.Параметры.РасхожденияРуб = РазностьРуб;
        ТабличныйДокумент.Вывести(Область);

        ИтогКолвоФакт = ИтогКолвоФакт + ВыборкаПоТоварам.КолвоФакт;
        ИтогСуммаФакт = ИтогСуммаФакт + СуммаЗЦФакт;
        ИтогСуммаСНДСПлан = ИтогСуммаСНДСПлан + ВыборкаПоТоварам.СуммаСНДСПлан;
        ИтогСуммаСНДСФакт = ИтогСуммаСНДСФакт + СуммаСНДСФакт;
        ИтогРасхожденияРуб = ИтогРасхожденияРуб + РазностьРуб;

КонецЦикла;

Область = Макет.ПолучитьОбласть("Подвал");
Область.Параметры.ИтогКолвоПлан = ИтогКолвоПлан;
Область.Параметры.ИтогКолвоФакт = ИтогКолвоФакт;
Область.Параметры.ИтогСуммаПлан = ИтогСуммаПлан;
Область.Параметры.ИтогСуммаФакт = ИтогСуммаФакт;
Область.Параметры.ИтогСуммаСНДСПлан = ИтогСуммаСНДСПлан;
Область.Параметры.ИтогСуммаСНДСФакт = ИтогСуммаСНДСФакт;
Область.Параметры.ИтогРасхожденияШт = ИтогРасхожденияШт;
Область.Параметры.ИтогРасхожденияРуб = ИтогРасхожденияРуб;
ТабличныйДокумент.Вывести(Область);

КонецЦикла;

////Начало нового кода
//Макет =
УправлениеПечатью.МакетПечатнойФормы("Документ.ПроектПоставки.ПФ_MXL_СогласованиеРасхождений СПоставщиком");
//ПервыйДокумент = Истина;
//
//Для Каждого ПроектПоставки Из МассивОбъектов Цикл
//
```

```
		//	ТаблицаФактаПриемки = РасхожденияСервер.ПолучитьФактПриемкиПроектаПоставки(ПроектПоставки);
		//
		//	Запрос = Новый Запрос;
		//	Запрос.Текст =
		//	"ВЫБРАТЬ
		//	|	ПроектПоставкиДокументыПоступлений.ПриобретениеТоваровУслуг КАК ДокументПриобретения,
		//	|	ТоварыПроектовПоставки.СтрокаНомер КАК НомерСтроки,
		//	|	ТоварыПроектовПоставки.Номенклатура КАК Номенклатура,
		//	|	ТоварыПроектовПоставки.Характеристика КАК Характеристика,
		//	|	ТоварыПроектовПоставки.Штрихкод КАК Штрихкод,
		//	|	ТоварыПроектовПоставки.Номенклатура.Артикул КАК Артикул,
		//	|	ПРЕДСТАВЛЕНИЕ(ТоварыПроектовПоставки.Номенклатура) КАК Наименование,
		//	|	ТоварыПроектовПоставки.Цвет КАК Цвет,
		//	|	ТоварыПроектовПоставки.РазмерПоставщика КАК Размер,
		//	|	ПроектПоставкиДокументыПоступлений.ПризнакТОРГ12 КАК Торг12,
		//	|	ТоварыПроектовПоставки.Количество КАК КоличествоПлан,
		//	|	0 КАК КоличествоФакт,
		//	|	ТоварыПроектовПоставки.ЗакупочнаяЦена КАК Цена,
		//	|	ТоварыПроектовПоставки.СуммаЗакупочная - ТоварыПроектовПоставки.СуммаНДС КАК
СуммаБезНДС,
		//	|	ТоварыПроектовПоставки.СтавкаНДС КАК СтавкаНДС,
		//	|	ТоварыПроектовПоставки.СуммаЗакупочная КАК СуммаСНДС
		//	|ИЗ
		//	|	Документ.ПроектПоставки.ДокументыПоступлений КАК ПроектПоставкиДокументыПоступлений
		//	|		ВНУТРЕННЕЕ СОЕДИНЕНИЕ РегистрСведений.ТоварыПроектовПоставки КАК
ТоварыПроектовПоставки
		//	|		ПО ПроектПоставкиДокументыПоступлений.Ссылка =
ТоварыПроектовПоставки.ПроектПоставки
		//	|			И ПроектПоставкиДокументыПоступлений.ПризнакТОРГ12 =
ТоварыПроектовПоставки.ПризнакТОРГ12
		//	|ГДЕ
		//	|	ПроектПоставкиДокументыПоступлений.Ссылка = &Ссылка
		//	|
		//	|УПОРЯДОЧИТЬ ПО
		//	|	ПроектПоставкиДокументыПоступлений.ПриобретениеТоваровУслуг.Дата
		//	|ИТОГИ ПО
		//	|	ДокументПриобретения";
		//
		//	Запрос.УстановитьПараметр("ПроектПоставки", ПроектПоставки);
		//
		//	ДеревоПриобретений = Запрос.Выполнить().Выгрузить(ОбходРезультатаЗапроса.ПоГруппировкам);
		//
		//	//По идее вся информация о товаре который отсутствует в РС ТоварыПроектовПоставки должна быть в РС
ПринятыеТоварыПроектовПоставки, но в настоящий момент это не так
		//	ПоследняяСтрокаПриобретения = ДеревоПриобретений.Строки[ДеревоПриобретений.Строки.Количество() - 1];
		//
		//	ПризнакТОРГ12 = Неопределено;
		//	Для Каждого СтрокаТовара Из ПоследняяСтрокаПриобретения.Строки Цикл
		//
		//		Если ПризнакТОРГ12 = Неопределено Тогда
		//			ПризнакТОРГ12 = СтрокаТовара.Торг12;
		//		КонецЕсли;
		//
		//		Если СтрокаТовара.НомерСтроки = 0 Тогда
		//
		//			СтрокаТовара.Торг12 = ПризнакТОРГ12;
		//
		//			РеквизитыНоменклатуры =
ОбщегоНазначения.ЗначенияРеквизитовОбъекта(СтрокаТовара.Номенклатура, "Артикул, Наименование, СтавкаНДС");

		//			СтрокаТовара.Артикул = РеквизитыНоменклатуры.Артикул;
		//			СтрокаТовара.Наименование = РеквизитыНоменклатуры.Наименование;
		//			СтрокаТовара.СтавкаНДС = РеквизитыНоменклатуры.СтавкаНДС;
		//
		//			ТаблицаЗначенийСвойств =
УправлениеСвойствами.ПолучитьЗначенияСвойств(СтрокаТовара.ХарактеристикаНоменклатуры, Истина, Ложь);
		//
		//			//Не доделано...
		//
		//		КонецЕсли;
		//
		//	КонецЦикла;
		//
		//	РасхожденияСервер.РаспределитьФактПриемкиПоПриобретениям(ДеревоПриобретений,
ТаблицаФактаПриемки);
		//
```

```
//	ИтогоКоличествоПлан = 0;
//	ИтогоКоличествоФакт = 0;
//	ИтогоСуммаПлан = 0;
//	ИтогоСуммаФакт = 0;
//	ИтогоСуммаСНДСПлан = 0;
//	ИтогоСуммаСНДСФакт = 0;
//	ИтогоКоличествоРасхождения = 0;
//	ИтогоСуммаРасхождения = 0;
//
//	Если Не ПервыйДокумент Тогда
//		ТабличныйДокумент.ВывестиГоризонтальныйРазделительСтраниц();
//	КонецЕсли;
//
//	ПервыйДокумент = Ложь;
//
//	ОбластьШапка = Макет.ПолучитьОбласть("Шапка");
//	ТабличныйДокумент.Вывести(ОбластьШапка);
//
//	Для Каждого СтрокаПриобретения Из ДеревоПриобретений.Строки Цикл
//
//		Для Каждого СтрокаТовара Из СтрокаПриобретения.Строки Цикл
//
//			ОбластьСтрока = Макет.ПолучитьОбласть("Строка");
//			ЗаполнитьЗначенияСвойств(ОбластьСтрока, СтрокаТовара, "Штрихкод, Артикул,
Наименование, Цвет, Размер, Торг12");
//
//			ОбластьСтрока.ЗЦена = СтрокаТовара.Цена;
//			ОбластьСтрока.КолвоПлан = СтрокаТовара.КоличествоПлан;
//			ОбластьСтрока.КолвоФакт = СтрокаТовара.КоличествоФакт;
//			ОбластьСтрока.СуммаЗЦПлан = СтрокаТовара.СуммаБезНДС;
//			ОбластьСтрока.СуммаСНДСПлан	= СтрокаТовара.СуммаСНДС;
//
//			Если СтрокаТовара.КоличествоПлан <> 0 Тогда
//
//				СуммаСНДСФакт	= СтрокаТовара.СуммаСНДС / СтрокаТовара.КоличествоПлан *
СтрокаТовара.КоличествоФакт;
//				СуммаБезНДСФакт = СтрокаТовара.СуммаБезНДС / СтрокаТовара.КоличествоПлан *
СтрокаТовара.КоличествоФакт;
//
//			Иначе
//
//				СуммаСНДСФакт = СтрокаТовара.Цена * СтрокаТовара.КоличествоФакт;
//				СуммаБезНДСФакт = СуммаСНДСФакт -
ЦенообразованиеКлиентСервер.РассчитатьСуммуНДС(СуммаСНДСФакт, СтрокаТовара.СтавкаНДС, Истина);
//
//			КонецЕсли;
//
//			ОбластьСтрока.СуммаЗЦФакт = СуммаБезНДСФакт;
//			ОбластьСтрока.СуммаСНДСФакт = СуммаСНДСФакт;
//
//			ОбластьСтрока.РасхожденияШт = СтрокаТовара.КоличествоФакт -
СтрокаТовара.КоличествоПлан;
//			ОбластьСтрока.РасхожденияРуб = СуммаСНДСФакт - СтрокаТовара.СуммаСНДС;
//
//			ИтогоКоличествоПлан = ИтогоКоличествоПлан + СтрокаТовара.КоличествоПлан;
//			ИтогоКоличествоФакт = ИтогоКоличествоФакт + СтрокаТовара.КоличествоФакт;
//			ИтогоСуммаПлан = ИтогоСуммаПлан + СтрокаТовара.СуммаБезНДС;
//			ИтогоСуммаФакт = ИтогоСуммаФакт + СуммаБезНДСФакт;
//			ИтогоСуммаСНДСПлан = ИтогоСуммаСНДСПлан + СтрокаТовара.СуммаСНДС;
//			ИтогоСуммаСНДСФакт = ИтогоСуммаСНДСФакт + СуммаСНДСФакт;
//			ИтогоКоличествоРасхождения = ИтогоКоличествоРасхождения +
СтрокаТовара.КоличествоФакт - СтрокаТовара.КоличествоПлан;
//			ИтогоСуммаРасхождения = ИтогоСуммаРасхождения + СуммаСНДСФакт -
СтрокаТовара.СуммаСНДС;
//
//		КонецЦикла;
//
//	КонецЦикла;
//
//	ОбластьПодвал = Макет.ПолучитьОбласть("Подвал");
//
//	ОбластьПодвал.Параметры.ИтогКолвоПлан = ИтогоКоличествоПлан;
//	ОбластьПодвал.Параметры.ИтогКолвоФакт = ИтогоКоличествоФакт;
//	ОбластьПодвал.Параметры.ИтогСуммаПлан = ИтогоСуммаПлан;
//	ОбластьПодвал.Параметры.ИтогСуммаФакт = ИтогоСуммаФакт;
//	ОбластьПодвал.Параметры.ИтогСуммаСНДСПлан = ИтогоСуммаСНДСПлан;
```

```
                //          ОбластьПодвал.Параметры.ИтогСуммаСНДСФакт = ИтогоСуммаСНДСФакт;
                //          ОбластьПодвал.Параметры.ИтогРасхожденияШт = ИтогоКоличествоРасхождения;
                //          ОбластьПодвал.Параметры.ИтогРасхожденияРуб = ИтогоСуммаРасхождения;
                //
                //          ТабличныйДокумент.Вывести(ОбластьПодвал);
                //
                //КонецЦикла;

                Если ПривилегированныйРежим() Тогда
                  УстановитьПривилегированныйРежим(Ложь);
                КонецЕсли;

                Возврат ТабличныйДокумент;

КонецФункции

#КонецОбласти

#Область ЗагрузкаПроектаПоставки

Функция СтуктураКешируемойИнформацииПоНоменклатуре()

                ИнформацияПоНоменклатуре = Новый Структура;

                ИнформацияПоНоменклатуре.Вставить("Номенклатура");
                ИнформацияПоНоменклатуре.Вставить("Артикул");
                ИнформацияПоНоменклатуре.Вставить("Наименование");
                ИнформацияПоНоменклатуре.Вставить("ГруппаНоменклатуры");
                ИнформацияПоНоменклатуре.Вставить("Поставщик");
                ИнформацияПоНоменклатуре.Вставить("СтавкаНДС");
                ИнформацияПоНоменклатуре.Вставить("ВидНоменклатуры");
                ИнформацияПоНоменклатуре.Вставить("Марка");
                ИнформацияПоНоменклатуре.Вставить("ТоварнаяКатегория");
                ИнформацияПоНоменклатуре.Вставить("КоллекцияНоменклатуры");
                ИнформацияПоНоменклатуре.Вставить("АртикулИсходный");
                ИнформацияПоНоменклатуре.Вставить("Состав");
                ИнформацияПоНоменклатуре.Вставить("СтранаПроисхождения");
                ИнформацияПоНоменклатуре.Вставить("Пол");
                ИнформацияПоНоменклатуре.Вставить("Цвет");
                ИнформацияПоНоменклатуре.Вставить("РоссийскийРазмер");
                ИнформацияПоНоменклатуре.Вставить("СезонПоставщика");
                ИнформацияПоНоменклатуре.Вставить("ДополнительныйПризнак");
                ИнформацияПоНоменклатуре.Вставить("ЕдиницаИзмерения");
                ИнформацияПоНоменклатуре.Вставить("КодТНВЭД");
                ИнформацияПоНоменклатуре.Вставить("ВестиУчетПоГТД");
                ИнформацияПоНоменклатуре.Вставить("Вес");
                ИнформацияПоНоменклатуре.Вставить("НоваяНоменклатура"    , Истина);
                ИнформацияПоНоменклатуре.Вставить("НоменклатураНеИзКэша" , Неопределено);
                ИнформацияПоНоменклатуре.Вставить("ВыполненныеДействия"  , Новый Структура("Добавлено, Изменено", Ложь,
Ложь));

                Возврат ИнформацияПоНоменклатуре;

КонецФункции

#Область СозданиеНоменклатурыХарактеристик

Процедура СоздатьНоменклатуруХарактеристики(ТаблицаТоваров, Ошибки, КэшированныеЗначения, КодПоставщика) Экспорт

                ОтборСтрок = Новый Структура("ИдентификаторТовара");

                МассивТоваров = ОбщегоНазначения.ВыгрузитьКолонку(ТаблицаТоваров, "ИдентификаторТовара", Истина);
                Для Каждого ИдентификаторТовара Из МассивТоваров Цикл

                  ОтборСтрок.ИдентификаторТовара = ИдентификаторТовара;

                  //Номенклатура
                  НайденныеСтрокиНоменклатуры = ТаблицаТоваров.НайтиСтроки(ОтборСтрок);

                  ЭталоннаяСтрокаНоменклатуры = НайденныеСтрокиНоменклатуры[0];
                  Если ЗначениеЗаполнено(ЭталоннаяСтрокаНоменклатуры.Номенклатура) Тогда

                        //ОбновитьНоменклатуру(ЭталоннаяСтрокаНоменклатуры.Номенклатура,
ЭталоннаяСтрокаНоменклатуры.ЗначенияРеквизитовНоменклатуры, КэшированныеЗначения);
                        ОбновитьНоменклатуру(ЭталоннаяСтрокаНоменклатуры.Номенклатура, ЭталоннаяСтрокаНоменклатуры,
КэшированныеЗначения);
```

```
                Иначе

                    СоздатьЗаполнитьНоменклатуру(ЭталоннаяСтрокаНоменклатуры, Ошибки, КодПоставщика);

                    Для Каждого СтрокаНоменклатуры Из НайденныеСтрокиНоменклатуры Цикл

                        СтрокаНоменклатуры.Номенклатура = ЭталоннаяСтрокаНоменклатуры.Номенклатура;
                        СтрокаНоменклатуры.НовыйТовар   = ЭталоннаяСтрокаНоменклатуры.НовыйТовар;
                        СтрокаНоменклатуры.Артикул = ЭталоннаяСтрокаНоменклатуры.Артикул

                    КонецЦикла;

                КонецЕсли;

                //Характеристика
                Если ЭталоннаяСтрокаНоменклатуры.ПараметрыУчета.ИспользоватьХарактеристики Тогда

                    СтруктураПоиска = Новый Структура("Номенклатура, Характеристика",
ЭталоннаяСтрокаНоменклатуры.Номенклатура, Справочники.ХарактеристикиНоменклатуры.ПустаяСсылка());

                    Для Каждого СтрокаНоменклатуры Из НайденныеСтрокиНоменклатуры Цикл

                        Если ЗначениеЗаполнено(СтрокаНоменклатуры.Характеристика) Тогда
                            продолжить;
                        КонецЕсли;

                        ЗначенияРеквизитовХарактеристики = СтрокаНоменклатуры.ЗначенияРеквизитовХарактеристики;
                        Для Каждого РеквизитХарактеристики Из ЗначенияРеквизитовХарактеристики Цикл
                            СтруктураПоиска.Вставить(РеквизитХарактеристики.Ключ,
РеквизитХарактеристики.Значение);
                        КонецЦикла;

                        НайденныеСтрокиХарактеристики = ТаблицаТоваров.НайтиСтроки(СтруктураПоиска);

                        Если НайденныеСтрокиХарактеристики.Количество() > 0 Тогда

                            ЭталоннаяСтрокаХарактеристики = НайденныеСтрокиХарактеристики[0];

                            НайтиСоздатьЗаполнитьХарактеристику(ЭталоннаяСтрокаХарактеристики, Ошибки, Не
ЭталоннаяСтрокаХарактеристики.НовыйТовар);

                            Для Каждого СтрокаХарактеристики Из НайденныеСтрокиХарактеристики Цикл
                                СтрокаХарактеристики.Характеристика =
ЭталоннаяСтрокаХарактеристики.Характеристика;
                                СтрокаХарактеристики.НовыйТовар     = ЭталоннаяСтрокаХарактеристики.НовыйТовар;
                            КонецЦикла;

                        КонецЕсли;

                    КонецЦикла;

                КонецЕсли;

            КонецЦикла;

КонецПроцедуры



Функция ПолучитьНастройкиУчетаНоменклатуры() Экспорт

                Запрос = Новый Запрос;
                Запрос.Текст =
                "ВЫБРАТЬ
                | ТоварныеКатегории.Ссылка КАК ТоварнаяКатегория,
                | ТоварныеКатегории.Владелец КАК ВидНоменклатуры,
                | ТоварныеКатегории.Владелец.ИспользоватьХарактеристики КАК ИспользоватьХарактеристики,
                | ТоварныеКатегории.Владелец.МаркируетсяШтрихкодомПоставщика КАК МаркируетсяШтрихкодомПоставщика,
                | ТоварныеКатегории.Владелец.ИспользоватьСоответствияРоссийскимРазмерам КАК
ИспользоватьСоответствияРоссийскимРазмерам,
                | ТоварныеКатегории.СтавкаНДС КАК СтавкаНДС,

                | ТоварныеКатегории.Владелец.ШаблонРабочегоНаименованияХарактеристики КАК
ШаблонРабочегоНаименованияХарактеристики,
                | ТоварныеКатегории.Владелец.ШаблонНаименованияДляПечатиХарактеристики КАК
ШаблонНаименованияДляПечатиХарактеристики
```

```
                |ИЗ
                | Справочник.ТоварныеКатегории КАК ТоварныеКатегории
                |
                |ОБЪЕДИНИТЬ ВСЕ
                |
                |ВЫБРАТЬ
                | ТоварныеКатегории.Ссылка,
                | ВидыНоменклатуры.Ссылка,
                | ВидыНоменклатуры.ИспользоватьХарактеристики,
                | ВидыНоменклатуры.МаркируетсяШтрихкодомПоставщика,
                | ВидыНоменклатуры.ИспользоватьСоответствияРоссийскимРазмерам,
                | ТоварныеКатегории.СтавкаНДС,
                | ВидыНоменклатуры.ШаблонРабочегоНаименованияХарактеристики,
                | ВидыНоменклатуры.ШаблонНаименованияДляПечатиХарактеристики
                |ИЗ
                | Справочник.ТоварныеКатегории КАК ТоварныеКатегории
                |       ВНУТРЕННЕЕ СОЕДИНЕНИЕ Справочник.ВидыНоменклатуры КАК ВидыНоменклатуры
                |       ПО ТоварныеКатегории.Владелец = ВидыНоменклатуры.ВладелецТоварныхКатегорий";

                возврат Запрос.Выполнить().Выгрузить();

КонецФункции

//Функция ПолучитьПараметрыУчетаТовара(СтрокаТовара, КэшированныеЗначения, ТаблицаТоварныхКатегорий,
РеквизитыХарактеристикВидаНоменклатуры)
Функция ПолучитьПараметрыУчетаТовара(СтрокаТовара, НастройкиУчетаНоменклатуры, КэшированныеЗначения) Экспорт

                ПараметрыУчетаТовара = Новый Структура("ВидНоменклатуры, ИспользоватьХарактеристики,
МаркируетсяШтрихкодомПоставщика, ИспользоватьСоответствияРоссийскимРазмерам, СтавкаНДС");

                ПараметрыУчетаТовара.Вставить("ШаблонРабочегоНаименованияХарактеристики");
                ПараметрыУчетаТовара.Вставить("ШаблонНаименованияДляПечатиХарактеристики");

                ПараметрыУчетаТовара.Вставить("СобственнаяМаркировка", СтрокаТовара.ПризнакМаркировки);

                РеквизитыНоменклатуры = Новый Структура("ГруппаНоменклатуры, АртикулИсходный, Наименование, Марка, Цвет,
РоссийскийРазмер, Пол, ТоварнаяКатегория, СезонПоставщика,
                                                                |КоллекцияНоменклатуры, Состав,
СтранаПроисхождения, ДополнительныйПризнак, Поставщик, СтавкаНДС");

                РеквизитыНоменклатуры.Вставить("Качество");
                РеквизитыНоменклатуры.Вставить("ЕдиницаИзмерения");
                РеквизитыНоменклатуры.Вставить("КодТНВЭД");
                РеквизитыНоменклатуры.Вставить("Вес");
                РеквизитыНоменклатуры.Вставить("ВестиУчетПоГТД");
                РеквизитыНоменклатуры.Вставить("ТипБренда");
                РеквизитыНоменклатуры.Вставить("Производитель");
                РеквизитыНоменклатуры.Вставить("КиЗГИСМGTIN");
                РеквизитыХарактеристик = Новый Структура;

                Если ЗначениеЗаполнено(СтрокаТовара.Номенклатура) Тогда

                  ДанныеКэша = КэшированныеЗначения.Номенклатура[СтрокаТовара.Номенклатура];

                  //Данные вида номенклатуры
                  ОтборВидаНоменклатуры = Новый Структура("ТоварнаяКатегория, ВидНоменклатуры",
СтрокаТовара.ТоварнаяКатегория, ДанныеКэша.ВидНоменклатуры);

                  НайденныеСтроки = НастройкиУчетаНоменклатуры.НайтиСтроки(ОтборВидаНоменклатуры);

                  Если НайденныеСтроки.Количество() > 0 Тогда

                        //Все ок
                        ЗаполнитьЗначенияСвойств(ПараметрыУчетаТовара, НайденныеСтроки[0], "ВидНоменклатуры,
ИспользоватьХарактеристики, МаркируетсяШтрихкодомПоставщика, ИспользоватьСоответствияРоссийскимРазмерам");

                        ПараметрыУчетаТовара.ШаблонРабочегоНаименованияХарактеристики =
НайденныеСтроки[0].ШаблонРабочегоНаименованияХарактеристики;
                        ПараметрыУчетаТовара.ШаблонНаименованияДляПечатиХарактеристики =
НайденныеСтроки[0].ШаблонНаименованияДляПечатиХарактеристики;

                        //По логике системы ставка НДС по умолчанию это реквизит вида номенклатуры, но у нас немного иначе все
                        Если СтрокаТовара.СтавкаНДС = Перечисления.СтавкиНДС.БезНДС Тогда
                              ПараметрыУчетаТовара.СтавкаНДС = НайденныеСтроки[0].СтавкаНДС;
                        Иначе //Продаем по той же ставке, что и покупаем
                              ПараметрыУчетаТовара.СтавкаНДС = СтрокаТовара.СтавкаНДС;
```

```
                КонецЕсли;

            КонецЕсли;

        Иначе

            //Данные вида номенклатуры
            ОтборВидаНоменклатуры = Новый Структура("ТоварнаяКатегория, ИспользоватьХарактеристики,
МаркируетсяШтрихкодомПоставщика");
            ОтборВидаНоменклатуры.ТоварнаяКатегория              = СтрокаТовара.ТоварнаяКатегория;
            ОтборВидаНоменклатуры.ИспользоватьХарактеристики     = Истина;
            ОтборВидаНоменклатуры.МаркируетсяШтрихкодомПоставщика = Не
ПараметрыУчетаТовара.СобственнаяМаркировка;

            //НайденныеСтроки = ТаблицаТоварныхКатегорий.НайтиСтроки(ОтборВидаНоменклатуры);
            НайденныеСтроки = НастройкиУчетаНоменклатуры.НайтиСтроки(ОтборВидаНоменклатуры);

            Если НайденныеСтроки.Количество() > 0 Тогда

                //Все ок
                ЗаполнитьЗначенияСвойств(ПараметрыУчетаТовара, НайденныеСтроки[0], "ВидНоменклатуры,
ИспользоватьХарактеристики, МаркируетсяШтрихкодомПоставщика, ИспользоватьСоответствияРоссийскимРазмерам");

                ПараметрыУчетаТовара.ШаблонРабочегоНаименованияХарактеристики =
НайденныеСтроки[0].ШаблонРабочегоНаименованияХарактеристики;
                ПараметрыУчетаТовара.ШаблонНаименованияДляПечатиХарактеристики =
НайденныеСтроки[0].ШаблонНаименованияДляПечатиХарактеристики;

                //По логике системы ставка НДС по умолчанию это реквизит вида номенклатуры, но у нас немного иначе все
                Если СтрокаТовара.СтавкаНДС = Перечисления.СтавкиНДС.БезНДС Тогда
                    ПараметрыУчетаТовара.СтавкаНДС = НайденныеСтроки[0].СтавкаНДС;
                Иначе //Продаем по той же ставке, что и покупаем
                    ПараметрыУчетаТовара.СтавкаНДС = СтрокаТовара.СтавкаНДС;
                КонецЕсли;

            КонецЕсли;

        КонецЕсли;

        Если ЗначениеЗаполнено(ПараметрыУчетаТовара.ВидНоменклатуры) И
ПараметрыУчетаТовара.ИспользоватьХарактеристики Тогда

            //Получаем массив с используемыми для данного вида номенклатуры набора свойств дополнительных реквизитов

            //СписокРеквизитов = ПолучитьСписокРеквизитовХарактеристики(ПараметрыУчетаТовара.ВидНоменклатуры,
РеквизитыХарактеристикВидаНоменклатуры);
            СписокРеквизитов = ПолучитьСписокРеквизитовХарактеристики(ПараметрыУчетаТовара.ВидНоменклатуры,
КэшированныеЗначения.РеквизитыХарактеристик);

            РеквизитыХарактеристик.Вставить("КиЗГИСМGTIN");

            Свойство = ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.ПроектПоставки;
            Если СписокРеквизитов.Найти(Свойство) <> Неопределено Тогда
                РеквизитыХарактеристик.Вставить("ПроектПоставки");
            КонецЕсли;

            Свойство = ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.Сезон;
            Если СписокРеквизитов.Найти(Свойство) <> Неопределено Тогда
                РеквизитыХарактеристик.Вставить("КоллекцияНоменклатуры");
            КонецЕсли;

            Свойство = ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.Цвет;
            Если СписокРеквизитов.Найти(Свойство) <> Неопределено Тогда
                РеквизитыХарактеристик.Вставить("Цвет");
            КонецЕсли;

            Свойство = ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.Размер;
            Если СписокРеквизитов.Найти(Свойство) <> Неопределено Тогда
                РеквизитыХарактеристик.Вставить("РоссийскийРазмер");
            КонецЕсли;

        КонецЕсли;

        Для Каждого Реквизит Из РеквизитыХарактеристик Цикл

            Если РеквизитыНоменклатуры.Свойство(Реквизит.Ключ) Тогда
```

```
                    РеквизитыНоменклатуры.Удалить(Реквизит.Ключ);
                КонецЕсли;

            КонецЦикла;

            ПараметрыУчетаТовара.Вставить("РеквизитыНоменклатуры"  , РеквизитыНоменклатуры);
            ПараметрыУчетаТовара.Вставить("РеквизитыХарактеристик" , РеквизитыХарактеристик);

            возврат ПараметрыУчетаТовара;

КонецФункции

Функция ПолучитьТаблицуСвойствХарактеристик(РеквизитыХарактеристики)

            //Подготовим таблицу значений используемых для данного вида номенклатуры свойств
            ТаблицаСвойствИЗначений = Новый ТаблицаЗначений;
            ТаблицаСвойствИЗначений.Колонки.Добавить("Свойство", Новый
ОписаниеТипов("ПланВидовХарактеристикСсылка.ДополнительныеРеквизитыИСведения"));
            ТаблицаСвойствИЗначений.Колонки.Добавить("Значение");

            Если РеквизитыХарактеристики.Свойство("ПроектПоставки") Тогда
              НоваяСтрока = ТаблицаСвойствИЗначений.Добавить();
              НоваяСтрока.Свойство = ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.ПроектПоставки;;
              НоваяСтрока.Значение = РеквизитыХарактеристики.ПроектПоставки;
            КонецЕсли;

            Если РеквизитыХарактеристики.Свойство("КоллекцияНоменклатуры") Тогда
              НоваяСтрока = ТаблицаСвойствИЗначений.Добавить();
              НоваяСтрока.Свойство = ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.Сезон;
              НоваяСтрока.Значение = РеквизитыХарактеристики.КоллекцияНоменклатуры;
            КонецЕсли;

            Если РеквизитыХарактеристики.Свойство("Цвет") Тогда
              НоваяСтрока = ТаблицаСвойствИЗначений.Добавить();
              НоваяСтрока.Свойство = ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.Цвет;
              НоваяСтрока.Значение = РеквизитыХарактеристики.Цвет;
            КонецЕсли;

            Если РеквизитыХарактеристики.Свойство("РоссийскийРазмер") Тогда
              НоваяСтрока = ТаблицаСвойствИЗначений.Добавить();
              НоваяСтрока.Свойство = ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.Размер;
              НоваяСтрока.Значение = РеквизитыХарактеристики.РоссийскийРазмер;
            КонецЕсли;

            возврат ТаблицаСвойствИЗначений;

КонецФункции

Функция ИзмененыРеквизитыНоменклатуры(Номенклатура, КэшированныеЗначения, НовыеЗначенияРеквизитов)

            ДанныеКэша = КэшированныеЗначения.Номенклатура[Номенклатура];

            Для Каждого КлючЗначение Из НовыеЗначенияРеквизитов Цикл

              Если КлючЗначение.Значение <> ДанныеКэша[КлючЗначение.Ключ] Тогда
                    возврат Истина;
              КонецЕсли;

            КонецЦикла;

            возврат Ложь;

КонецФункции

Функция НайтиПодходящиеХарактеристики(Номенклатура, ТаблицаДополнительныхРеквизитов)

            Запрос = Новый Запрос;
            Запрос.УстановитьПараметр("Владелец", Номенклатура);

            ИмяТаблицы = "ТаблицаХарактеристик";

            ТекстЗапроса =
            "ВЫБРАТЬ РАЗЛИЧНЫЕ
            | " + ИмяТаблицы + ".Ссылка КАК ХарактеристикаНоменклатуры

            |," + ИмяТаблицы + ".Ссылка.КизГИСMGTIN КАК КизГИСMGTIN
```

```
                |," + ИмяТаблицы + ".Ссылка.ПометкаУдаления КАК ПометкаУдаления

                |ИЗ
                | Справочник.ХарактеристикиНоменклатуры.ДополнительныеРеквизиты КАК " + ИмяТаблицы;

                ТекстУсловия = "
                |ГДЕ
                | " + ИмяТаблицы + ".Ссылка.Владелец = &Владелец";

                СчетчикУсловий = 1;

                // Генерируем текст условия запроса по дополнительным реквизитам, устанавливаем параметры
                Для Каждого СтрокаУсловия Из ТаблицаДополнительныхРеквизитов Цикл

                  ИмяТаблицыСУсловием = ИмяТаблицы + "СУсловиемПоДопРеквизиту_" + Формат(СчетчикУсловий,"ЧГ=0");

                  СтрокаНомераУсловия = Формат(СчетчикУсловий, "ЧГ=");

                  ТекстУсловия = ТекстУсловия +
                  "
                  | И " + ИмяТаблицыСУсловием + ".Свойство = &Свойство_" + СтрокаНомераУсловия +
                  "
                  | И " + ИмяТаблицыСУсловием + ".Значение = &Значение_" + СтрокаНомераУсловия +
                  "
                  | И " + ИмяТаблицыСУсловием + ".Ссылка.Владелец = &Владелец";

                  Запрос.УстановитьПараметр("Свойство_" + СтрокаНомераУсловия, СтрокаУсловия.Свойство);
                  Запрос.УстановитьПараметр("Значение_" + СтрокаНомераУсловия, СтрокаУсловия.Значение);

                  ТекстСоединения = "
                  | ВНУТРЕННЕЕ СОЕДИНЕНИЕ Справочник.ХарактеристикиНоменклатуры.ДополнительныеРеквизиты КАК " +
ИмяТаблицыСУсловием + "
                  | ПО " + ИмяТаблицы + ".Ссылка = " + ИмяТаблицыСУсловием + ".Ссылка";

                  ТекстЗапроса = ТекстЗапроса + ТекстСоединения;
                  СчетчикУсловий = СчетчикУсловий + 1;

                КонецЦикла;

                Запрос.Текст = ТекстЗапроса + ТекстУсловия;

                возврат Запрос.Выполнить().Выгрузить();

КонецФункции

Функция ПолучитьСписокРеквизитовХарактеристики(ВидНоменклатуры, СпискиРеквизитовПоВидуНоменклатуры)

                СписокРеквизитов = СпискиРеквизитовПоВидуНоменклатуры.Получить(ВидНоменклатуры);

                Если СписокРеквизитов <> Неопределено Тогда
                  возврат СписокРеквизитов;
                КонецЕсли;

                Запрос = Новый Запрос;
                Запрос.Текст =
                "ВЫБРАТЬ
                | НаборыДополнительныхРеквизитовИСведенийДополнительныеРеквизиты.Свойство
                |ИЗ
                | Справочник.ВидыНоменклатуры КАК ВидыНоменклатуры
                |             ВНУТРЕННЕЕ СОЕДИНЕНИЕ
Справочник.НаборыДополнительныхРеквизитовИСведений.ДополнительныеРеквизиты КАК
НаборыДополнительныхРеквизитовИСведенийДополнительныеРеквизиты
                |             ПО ВидыНоменклатуры.НаборСвойствХарактеристик =
НаборыДополнительныхРеквизитовИСведенийДополнительныеРеквизиты.Ссылка
                |ГДЕ
                | НЕ НаборыДополнительныхРеквизитовИСведенийДополнительныеРеквизиты.ПометкаУдаления
                | И ВидыНоменклатуры.Ссылка = &ВидНоменклатуры";

                Запрос.Параметры.Вставить("ВидНоменклатуры", ВидНоменклатуры);

                Результат = Запрос.Выполнить().Выгрузить().ВыгрузитьКолонку("Свойство");

                СпискиРеквизитовПоВидуНоменклатуры.Вставить(ВидНоменклатуры, Результат);

                Возврат Результат;
КонецФункции
```

```
Процедура ЗаполнитьЗначенияРеквизитовТовара(СтрокаТовара) Экспорт

		ПараметрыУчетаТовара = СтрокаТовара.ПараметрыУчета;

		ЗначенияРеквизитовНоменклатуры =
ОбщегоНазначенияКлиентСервер.СкопироватьСтруктуру(ПараметрыУчетаТовара.РеквизитыНоменклатуры);

		//Номенклатура
		ЗаполнитьЗначенияСвойств(ЗначенияРеквизитовНоменклатуры, СтрокаТовара);

		ЗначенияРеквизитовНоменклатуры.Вставить("ВидНоменклатуры", ПараметрыУчетаТовара.ВидНоменклатуры);

		//Переопределим ставку НДС покупки на ставку НДС продажи
		ЗначенияРеквизитовНоменклатуры.СтавкаНДС = ПараметрыУчетаТовара.СтавкаНДС;

		Если ЗначенияРеквизитовНоменклатуры.Свойство("РоссийскийРазмер") И
ЗначениеЗаполнено(ПараметрыУчетаТовара.ВидНоменклатуры) Тогда

		  Если ПараметрыУчетаТовара.ИспользоватьСоответствияРоссийскимРазмерам Тогда
			ЗначенияРеквизитовНоменклатуры.РоссийскийРазмер = СтрокаТовара.РоссийскийРазмер;
		  Иначе
			ЗначенияРеквизитовНоменклатуры.РоссийскийРазмер = СтрокаТовара.РазмерПоставщика;
		  КонецЕсли;

		КонецЕсли;

		ЗначенияРеквизитовНоменклатуры.Вставить("ПометкаУдаления", Ложь);

		ПараметрыУчетаТовара.Удалить("РеквизитыНоменклатуры");

		СтрокаТовара.ЗначенияРеквизитовНоменклатуры = ЗначенияРеквизитовНоменклатуры;

		//Характеристика
		ЗначенияРеквизитовХарактеристики =
ОбщегоНазначенияКлиентСервер.СкопироватьСтруктуру(ПараметрыУчетаТовара.РеквизитыХарактеристик);

		ЗаполнитьЗначенияСвойств(ЗначенияРеквизитовХарактеристики, СтрокаТовара);

		Если ЗначенияРеквизитовХарактеристики.Свойство("РоссийскийРазмер") И
ЗначениеЗаполнено(ПараметрыУчетаТовара.ВидНоменклатуры) Тогда

		  Если ПараметрыУчетаТовара.ИспользоватьСоответствияРоссийскимРазмерам Тогда
			ЗначенияРеквизитовХарактеристики.РоссийскийРазмер = СтрокаТовара.РоссийскийРазмер;
		  Иначе
			ЗначенияРеквизитовХарактеристики.РоссийскийРазмер = СтрокаТовара.РазмерПоставщика;
		  КонецЕсли;

		КонецЕсли;

		ПараметрыУчетаТовара.Удалить("РеквизитыХарактеристик");

		СтрокаТовара.ЗначенияРеквизитовХарактеристики = ЗначенияРеквизитовХарактеристики;

КонецПроцедуры

Функция ПолучитьСоставТрехКолонок(НоменклатураОбъект, СтрокаНоменклатуры)

		НоменклатураОбъект.СоставТовара.Очистить();

		ТекстСостав = "";
		Если НЕ ПустаяСтрока(СтрокаНоменклатуры.СоставВерх) Тогда

		  НовСостав = НоменклатураОбъект.СоставТовара.Добавить();
		  НовСостав.ВидСостава = Перечисления.ВидыСоставаТоваров.МатериалВерха;
		  НовСостав.ОписаниеМатериала = СтрокаНоменклатуры.СоставВерх;
		  ТекстСостав = ТекстСостав+""+НовСостав.ВидСостава+": "+СокрЛП(НовСостав.ОписаниеМатериала)+Символы.ПС;

		КонецЕсли;
		Если НЕ ПустаяСтрока(СтрокаНоменклатуры.СоставПодкладка) Тогда

		  НовСостав = НоменклатураОбъект.СоставТовара.Добавить();
		  НовСостав.ВидСостава = Перечисления.ВидыСоставаТоваров.МатериалПодкладки;
		  НовСостав.ОписаниеМатериала = СтрокаНоменклатуры.СоставПодкладка;
		  ТекстСостав = ТекстСостав+""+НовСостав.ВидСостава+": "+СокрЛП(НовСостав.ОписаниеМатериала)+Символы.ПС;

		КонецЕсли;
```

```
                Если НЕ ПустаяСтрока(СтрокаНоменклатуры.СоставПодошва) Тогда

                  НовСостав = НоменклатураОбъект.СоставТовара.Добавить();
                  НовСостав.ВидСостава = Перечисления.ВидыСоставаТоваров.МатериалПодошвы;
                  НовСостав.ОписаниеМатериала = СтрокаНоменклатуры.СоставПодошва;
                  ТекстСостав = ТекстСостав+""+НовСостав.ВидСостава+": "+СокрЛП(НовСостав.ОписаниеМатериала)+Символы.ПС;

                КонецЕсли;

                Возврат ТекстСостав;

КонецФункции

Процедура СоздатьЗаполнитьНоменклатуру(СтрокаНоменклатуры, Ошибки, КодПоставщика)

                ЗначенияРеквизитов = СтрокаНоменклатуры.ЗначенияРеквизитовНоменклатуры;

                //Создание номенклатуры
                НоменклатураОбъект = Справочники.Номенклатура.СоздатьЭлемент();
                ЗаполнитьРеквизитыНоменклатуры(НоменклатураОбъект, ЗначенияРеквизитов);

                НоменклатураОбъект.УстановитьНовыйКод();

                Отказ = Ложь;
                Справочники.Номенклатура.ЗаполнитьРеквизитыПоВидуНоменклатуры(НоменклатураОбъект, Истина, Отказ);

                Если Отказ Тогда
                  ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Возникли ошибки при заполнении по виду номенклатуры.",
СтрокаНоменклатуры.НомерСтроки);
                  возврат;
                КонецЕсли;


                Если СтрокаНоменклатуры.ВестиУчетПоГТД Тогда

                  НоменклатураОбъект.Артикул = КодПоставщика + "-" +
ПрефиксацияОбъектовКлиентСервер.НомерНаПечать(НоменклатураОбъект.Код, Истина, Истина);
                  НоменклатураОбъект.ВестиУчетПоГТД = СтрокаНоменклатуры.ВестиУчетПоГТД;
                  НоменклатураОбъект.ЕдиницаИзмерения = СтрокаНоменклатуры.ЕдиницаИзмерения;

                  НоменклатураОбъект.ВесЧислитель = СтрокаНоменклатуры.Вес;
                  НоменклатураОбъект.ВесИспользовать = ЗначениеЗаполнено(СтрокаНоменклатуры.Вес);
                  Если НоменклатураОбъект.ВесИспользовать Тогда
                            НоменклатураОбъект.ВесЗнаменатель = 1;
                            НоменклатураОбъект.ВесЕдиницаИзмерения = НоменклатураСервер.ЕдиницаИзмеренияПоУмолчанию("Вес")
                  КонецЕсли;

                Иначе
                  НоменклатураОбъект.Артикул = КодПоставщика + СтрокаНоменклатуры.АртикулИсходный;

                КонецЕсли;

                //Переопределим ставку НДС
                НоменклатураОбъект.СтавкаНДС = ЗначенияРеквизитов.СтавкаНДС;

                //Шаблоны могли изменить наименование
                НоменклатураОбъект.Наименование       = ЗначенияРеквизитов.Наименование;
                НоменклатураОбъект.НаименованиеПолное = ЗначенияРеквизитов.Наименование;

                ТекстСостав = ПолучитьСоставТрехКолонок(НоменклатураОбъект, СтрокаНоменклатуры);
                Если НЕ ПустаяСтрока(ТекстСостав) Тогда

                  НоменклатураОбъект.Состав = ТекстСостав;

                КонецЕсли;
                Если ЗначениеЗаполнено(НоменклатураОбъект.ТоварнаяКатегория)
                   И ЗначениеЗаполнено(НоменклатураОбъект.ТоварнаяКатегория.ПризнакСертификации) Тогда
                   НоменклатураОбъект.ПризнакСертификации = НоменклатураОбъект.ТоварнаяКатегория.ПризнакСертификации;
                КонецЕсли;


                НоменклатураОбъект.ОбменДанными.Загрузка = Истина;
                НоменклатураОбъект.Записать();

                СтрокаНоменклатуры.Номенклатура = НоменклатураОбъект.Ссылка;
                СтрокаНоменклатуры.НовыйТовар = Истина;
```

```
            СтрокаНоменклатуры.Артикул = НоменклатураОбъект.Артикул;

КонецПроцедуры

Процедура ОбновитьНоменклатуру(Номенклатура, СтрокаНоменклатуры, КэшированныеЗначения)

            ОбновляемыеРеквизиты = Новый Структура("ГруппаНоменклатуры, АртикулИсходный, Наименование, Марка, Цвет,
РоссийскийРазмер, Пол, ТоварнаяКатегория,
                                                                                    |СезонПоставщика, КоллекцияНоменклатуры,
Состав, ДополнительныйПризнак, Вес");


            Если НЕ Номенклатура.ВестиУчетПоГТД Тогда
              ОбновляемыеРеквизиты.Вставить("СтранаПроисхождения");
            КонецЕсли;

            ОбновляемыеРеквизиты.Вставить("ПометкаУдаления");

            ОбновляемыеРеквизиты.Вставить("ТипБренда");

            ОбновляемыеРеквизиты.Вставить("Производитель");

            ОбновляемыеРеквизиты.Вставить("КиЗГИСМGTIN");

            ЗначенияРеквизитов = СтрокаНоменклатуры.ЗначенияРеквизитовНоменклатуры;

            Для Каждого Реквизит Из ОбновляемыеРеквизиты Цикл

              Если Не ЗначенияРеквизитов.Свойство(Реквизит.Ключ) Тогда
                    ОбновляемыеРеквизиты.Удалить(Реквизит.Ключ);
              КонецЕсли;

            КонецЦикла;

            ЗаполнитьЗначенияСвойств(ОбновляемыеРеквизиты, ЗначенияРеквизитов);

            Если ИзмененыРеквизитыНоменклатуры(Номенклатура, КэшированныеЗначения, ОбновляемыеРеквизиты) Тогда

              //Обновление номенклатуры
              НоменклатураОбъект = Номенклатура.ПолучитьОбъект();

              ЗаблокироватьДанныеДляРедактирования(Номенклатура);

              ЗаполнитьРеквизитыНоменклатуры(НоменклатураОбъект, ОбновляемыеРеквизиты);


              Если  ОбновляемыеРеквизиты.Свойство("ТоварнаяКатегория")
                    И ЗначениеЗаполнено(ОбновляемыеРеквизиты.ТоварнаяКатегория.ПризнакСертификации)
              Тогда
                 НоменклатураОбъект.ПризнакСертификации = ОбновляемыеРеквизиты.ТоварнаяКатегория.ПризнакСертификации;
              КонецЕсли;

              ТекстСостав = ПолучитьСоставТрехКолонок(НоменклатураОбъект, СтрокаНоменклатуры);
              Если НЕ ПустаяСтрока(ТекстСостав) Тогда

                    НоменклатураОбъект.Состав = ТекстСостав;

              КонецЕсли;



              НоменклатураОбъект.ОбменДанными.Загрузка = Истина;

              НоменклатураОбъект.Записать();

            КонецЕсли;

КонецПроцедуры

Процедура НайтиСоздатьЗаполнитьХарактеристику(СтрокаХарактеристики, Ошибки, ИскатьХарактеристику = Истина)

            РеквизитыХарактеристики = СтрокаХарактеристики.ЗначенияРеквизитовХарактеристики;

            ТаблицаСвойствИЗначений = ПолучитьТаблицуСвойствХарактеристик(РеквизитыХарактеристики);

            Если ТаблицаСвойствИЗначений.Количество() = 0 Тогда
```

```
                ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Попытка создания характеристики с пустым набором свойств.",
СтрокаХарактеристики.НомерСтроки);
                возврат;

            КонецЕсли;

            //Попробуем найти характеристику
            Если ИскатьХарактеристику Тогда

                ТаблицаПодходящихХарактеристик = НайтиПодходящиеХарактеристики(СтрокаХарактеристики.Номенклатура,
ТаблицаСвойствИЗначений);

                Для Каждого СтрокаПодходящей Из ТаблицаПодходящихХарактеристик Цикл

                    Если  РеквизитыХарактеристики.Свойство("КиЗГИСMGTIN")
                        И ЗначениеЗаполнено(РеквизитыХарактеристики.КиЗГИСMGTIN)
                        И НЕ РеквизитыХарактеристики.КиЗГИСMGTIN = СтрокаПодходящей.КизГИСMGTIN
                    Тогда
                        Продолжить;
                    КонецЕсли;

                    ТаблицаСвойствИЗначенийПодходящей =
УправлениеСвойствами.ПолучитьЗначенияСвойств(СтрокаПодходящей.ХарактеристикаНоменклатуры, Истина, Ложь);

                    ИсключитьСтрокуСоСвойством =
ТаблицаСвойствИЗначенийПодходящей.Найти(ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.ДистрибьютерИмпо
ртер, "Свойство");
                    Если ИсключитьСтрокуСоСвойством <> Неопределено Тогда
                        ТаблицаСвойствИЗначенийПодходящей.Удалить(ИсключитьСтрокуСоСвойством);
                    КонецЕсли;

                    Если ОбщегоНазначения.КоллекцииИдентичны(ТаблицаСвойствИЗначений,
ТаблицаСвойствИЗначенийПодходящей) Тогда

                        Если СтрокаПодходящей.ПометкаУдаления Тогда

                            ХарактеристикаОбъект = СтрокаПодходящей.ХарактеристикаНоменклатуры.ПолучитьОбъект();

                            ЗаблокироватьДанныеДляРедактирования(СтрокаПодходящей.ХарактеристикаНоменклатуры);

                            ХарактеристикаОбъект.ПометкаУдаления = Ложь;
                            ХарактеристикаОбъект.ОбменДанными.Загрузка = Истина;
                            ХарактеристикаОбъект.Записать();

                        КонецЕсли;


                        Если ЗначениеЗаполнено(СтрокаХарактеристики.ДистрибьютерИмпортер) Тогда

                            ХарактеристикаОбъект = СтрокаПодходящей.ХарактеристикаНоменклатуры.ПолучитьОбъект();
                            ТекСвойство =
ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.ДистрибьютерИмпортер;
                            СтрокаСоСвойством =
ХарактеристикаОбъект.ДополнительныеРеквизиты.Найти(ТекСвойство,"Свойство");
                            Если СтрокаСоСвойством <> Неопределено и СтрокаСоСвойством.Значение =
СтрокаХарактеристики.ДистрибьютерИмпортер Тогда
                                // все есть
                            Иначе
                                Если СтрокаСоСвойством = Неопределено Тогда
                                    СтрокаСоСвойством =
ХарактеристикаОбъект.ДополнительныеРеквизиты.Добавить();
                                    СтрокаСоСвойством.Свойство =
ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.ДистрибьютерИмпортер;
                                КонецЕсли;
                                СтрокаСоСвойством.Значение = СтрокаХарактеристики.ДистрибьютерИмпортер;
                                ХарактеристикаОбъект.Записать();
                            КонецЕсли;
                        КонецЕсли;

                        СтрокаХарактеристики.Характеристика = СтрокаПодходящей.ХарактеристикаНоменклатуры;

                        возврат;

                    КонецЕсли;

                КонецЦикла;
```

```
КонецЕсли;

//Создание характеристик
ХарактеристикаОбъект = Справочники.ХарактеристикиНоменклатуры.СоздатьЭлемент();

ХарактеристикаОбъект.Владелец = СтрокаХарактеристики.Номенклатура;

Для Каждого СтрокаСвойства Из ТаблицаСвойствИЗначений Цикл

  НоваяСтрока = ХарактеристикаОбъект.ДополнительныеРеквизиты.Добавить();
  НоваяСтрока.Свойство = СтрокаСвойства.Свойство;
  НоваяСтрока.Значение = СтрокаСвойства.Значение;

КонецЦикла;

ХарактеристикаОбъект.Наименование =
НоменклатураСервер.НаименованиеПоШаблону(СтрокаХарактеристики.ПараметрыУчета.ШаблонРабочегоНаименованияХарактеристики, ХарактеристикаОбъект);

ХарактеристикаОбъект.НаименованиеПолное =
НоменклатураСервер.НаименованиеПоШаблону(СтрокаХарактеристики.ПараметрыУчета.ШаблонНаименованияДляПечатиХарактеристики, ХарактеристикаОбъект);


ХарактеристикаОбъект.ОбменДанными.Загрузка = Истина;

Если  РеквизитыХарактеристики.Свойство("КиЗГИСMGTIN")
  И ЗначениеЗаполнено(РеквизитыХарактеристики.КиЗГИСMGTIN)
Тогда
  ХарактеристикаОбъект.КиЗГИСMGTIN = РеквизитыХарактеристики.КиЗГИСMGTIN;
КонецЕсли;

Если ЗначениеЗаполнено(СтрокаХарактеристики.ДистрибьютерИмпортер) Тогда
  НужнаяСтрока = ХарактеристикаОбъект.ДополнительныеРеквизиты.Добавить();
  НужнаяСтрока.Свойство = ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.ДистрибьютерИмпортер;
  НужнаяСтрока.Значение = СтрокаХарактеристики.ДистрибьютерИмпортер;
КонецЕсли;

ХарактеристикаОбъект.Записать();

СтрокаХарактеристики.Характеристика = ХарактеристикаОбъект.Ссылка;
СтрокаХарактеристики.НовыйТовар = Истина;

КонецПроцедуры

Процедура ПолучитьДанныеПоНоменклатуре(МассивНоменклатуры, КэшированныеЗначения)

Запрос = Новый Запрос;
Запрос.Текст =
"ВЫБРАТЬ
| Номенклатура.Ссылка КАК Номенклатура,
| Номенклатура.Артикул КАК Артикул,
| Номенклатура.Наименование КАК Наименование,
| Номенклатура.Родитель КАК ГруппаНоменклатуры,
| Номенклатура.Поставщик КАК Поставщик,
| Номенклатура.СтавкаНДС КАК СтавкаНДС,
| Номенклатура.ВидНоменклатуры КАК ВидНоменклатуры,
| Номенклатура.Марка КАК Марка,
| Номенклатура.ТоварнаяКатегория КАК ТоварнаяКатегория,
| Номенклатура.КоллекцияНоменклатуры КАК КоллекцияНоменклатуры,
| Номенклатура.АртикулИсходный КАК АртикулИсходный,
| Номенклатура.Состав КАК Состав,
| Номенклатура.СтранаПроисхождения КАК СтранаПроисхождения,
| Номенклатура.ор_Пол КАК Пол,
| Номенклатура.Цвет КАК Цвет,
| Номенклатура.Размер КАК РоссийскийРазмер,
| Номенклатура.ЕдиницаИзмерения КАК ЕдиницаИзмерения,
| Номенклатура.КодТНВЭД КАК КодТНВЭД,
| Номенклатура.ВестиУчетПоГТД КАК ВестиУчетПоГТД,
| ЕСТЬNULL(ДополнительныйРеквизитСезонПоставщика.Значение,
ЗНАЧЕНИЕ(Справочник.ЗначенияСвойствОбъектов.ПустаяСсылка)) КАК СезонПоставщика,
| ЕСТЬNULL(ДополнительныйРеквизитДопПризнак.Значение, """") КАК ДополнительныйПризнак
|ИЗ
| Справочник.Номенклатура КАК Номенклатура
|           ЛЕВОЕ СОЕДИНЕНИЕ Справочник.Номенклатура.ДополнительныеРеквизиты КАК
ДополнительныйРеквизитСезонПоставщика
```

```
                |            ПО (ДополнительныйРеквизитСезонПоставщика.Ссылка = Номенклатура.Ссылка)
                |                  И (ДополнительныйРеквизитСезонПоставщика.Свойство = &СвойствоСезонПоставщика)
                |            ЛЕВОЕ СОЕДИНЕНИЕ Справочник.Номенклатура.ДополнительныеРеквизиты КАК
ДополнительныйРеквизитДопПризнак
                |            ПО (ДополнительныйРеквизитДопПризнак.Ссылка = Номенклатура.Ссылка)
                |                  И (ДополнительныйРеквизитДопПризнак.Свойство = &СвойствоДопПризнак)
                |ГДЕ
                | Номенклатура.Ссылка В(&МассивНоменклатуры)";

                //Обливаюсь кровавыми слезами:
                Запрос.УстановитьПараметр("СвойствоСезонПоставщика",
ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.НайтиПоНаименованию("Сезон поставщика", Истина));
                Запрос.УстановитьПараметр("СвойствоДопПризнак",
ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.НайтиПоНаименованию("Доп. признак", Истина));
                Запрос.УстановитьПараметр("МассивНоменклатуры", МассивНоменклатуры);

                КэшированныеЗначения.Вставить("Номенклатура", Новый Соответствие);

                Выборка = Запрос.Выполнить().Выбрать();
                Пока Выборка.Следующий() Цикл

                  ИнформацияПоНоменклатуре = СтуктураКешируемойИнформацииПоНоменклатуре();
                  ЗаполнитьЗначенияСвойств(ИнформацияПоНоменклатуре, Выборка);

                  КэшированныеЗначения.Номенклатура.Вставить(Выборка.Номенклатура, ИнформацияПоНоменклатуре);

                КонецЦикла;

КонецПроцедуры

Процедура ЗаполнитьРеквизитыНоменклатуры(НоменклатураОбъект, ЗначенияРеквизитов)

                ЗаполнитьЗначенияСвойств(НоменклатураОбъект, ЗначенияРеквизитов);

                Если ЗначенияРеквизитов.Свойство("РоссийскийРазмер") Тогда
                  НоменклатураОбъект.Размер = ЗначенияРеквизитов.РоссийскийРазмер;
                КонецЕсли;

                НоменклатураОбъект.ор_Пол = ЗначенияРеквизитов.Пол;
                НоменклатураОбъект.Родитель = ЗначенияРеквизитов.ГруппаНоменклатуры;

                НоменклатураОбъект.ДополнительныеРеквизиты.Очистить();

                Если ЗначенияРеквизитов.Свойство("СезонПоставщика") И ЗначениеЗаполнено(ЗначенияРеквизитов.СезонПоставщика)
Тогда
                  НоваяСтрока = НоменклатураОбъект.ДополнительныеРеквизиты.Добавить();
                  НоваяСтрока.Свойство = ОбщегоНазначения.ЗначениеРеквизитаОбъекта(ЗначенияРеквизитов.СезонПоставщика,
"Владелец");
                  НоваяСтрока.Значение = ЗначенияРеквизитов.СезонПоставщика;
                КонецЕсли;

                Если ЗначенияРеквизитов.Свойство("ДополнительныйПризнак") И
ЗначениеЗаполнено(ЗначенияРеквизитов.ДополнительныйПризнак) Тогда
                  НоваяСтрока = НоменклатураОбъект.ДополнительныеРеквизиты.Добавить();
                  НоваяСтрока.Свойство =
ПланыВидовХарактеристик.ДополнительныеРеквизитыИСведения.НайтиПоНаименованию("Доп. признак", Истина);;
                  НоваяСтрока.Значение = ЗначенияРеквизитов.ДополнительныйПризнак;
                КонецЕсли;

                Если ЗначенияРеквизитов.Свойство("Вес") Тогда
                  НоменклатураОбъект.ВесИспользовать = ЗначениеЗаполнено(ЗначенияРеквизитов.Вес);
                  Если НоменклатураОбъект.ВесИспользовать Тогда
                        НоменклатураОбъект.ВесЗнаменатель = 1;
                        НоменклатураОбъект.ВесЕдиницаИзмерения = НоменклатураСервер.ЕдиницаИзмеренияПоУмолчанию("Вес")
                  КонецЕсли
                КонецЕсли;

КонецПроцедуры

Процедура ПометитьНаУдалениеНеиспользуемуюНоменклатуру()
КонецПроцедуры

#КонецОбласти

#Область ПроверкиТаблицыТоваров
```

Продолжение Приложения В

```
Процедура ПроверитьТаблицуТоваров(ТаблицаДанных, Ошибки, КэшированныеЗначения, ПараметрыОбработки) Экспорт

	Для Каждого СтрокаДанных Из ТаблицаДанных Цикл

	//Не допускается пустое значение даты выдачи с ДХ если задан процент на ДХ
	Если ЗначениеЗаполнено(СтрокаДанных.ПроцентНаДХ) И НЕ ЗначениеЗаполнено(СтрокаДанных.ДатаВыдачиСДХ)
Тогда
		ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Не указана дата выдачи с ДХ.", СтрокаДанных.НомерСтроки);
	КонецЕсли;

	//Дата выдачи с ДХ должна быть не менее даты планового поступления

	Если ЗначениеЗаполнено(СтрокаДанных.ДатаВыдачиСДХ) И СтрокаДанных.ДатаВыдачиСДХ <
ПараметрыОбработки.ПлановаяДатаПоставки Тогда

		ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Дата выдачи с ДХ не может быть меньше даты планового
поступления.", СтрокаДанных.НомерСтроки);
	КонецЕсли;

	Если ПараметрыОбработки.СоздаватьДокументыУстановкиЦен Тогда

		Если Не ЗначениеЗаполнено(СтрокаДанных.РекомендованнаяЦена) Тогда
			ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Не заполнена рекомендованная цена.",
СтрокаДанных.НомерСтроки);
		КонецЕсли;

		Если Не ЗначениеЗаполнено(СтрокаДанных.ЗакупочнаяЦена) Тогда
			ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Не заполнена закупочная цена.",
СтрокаДанных.НомерСтроки);
		КонецЕсли;

		Если Не ЗначениеЗаполнено(СтрокаДанных.РозничнаяЦена) Тогда
			ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Не заполнена розничная цена.",
СтрокаДанных.НомерСтроки);
		КонецЕсли;

		Если СтрокаДанных.РозничнаяЦена > СтрокаДанных.РекомендованнаяЦена Тогда
			ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Розничная цена больше рекомендованной.",
СтрокаДанных.НомерСтроки);
		КонецЕсли;

	КонецЕсли;

	ПараметрыУчетаТовара = СтрокаДанных.ПараметрыУчета;
	//РеквизитыНоменклатуры = СтрокаДанных.ЗначенияРеквизитовНоменклатуры;

	Если ЗначениеЗаполнено(СтрокаДанных.ТоварнаяКатегория) Тогда //Проверки только если товарная категория
заполнена

		//Проверка определения вида номенклатуры
		Если Не ЗначениеЗаполнено(ПараметрыУчетаТовара.ВидНоменклатуры) Тогда

			Если ЗначениеЗаполнено(СтрокаДанных.Номенклатура) Тогда

				ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Вид существующей номенклатуры не связан с
товарной категорией.", СтрокаДанных.НомерСтроки);

			Иначе

				ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Отсутствует подходящий вид номенклатуры.",
СтрокаДанных.НомерСтроки);

			КонецЕсли;

		Иначе //Проверки только если вид номенклатуры заполнен

			//Признаки маркировки
			Если ПараметрыУчетаТовара.СобственнаяМаркировка =
ПараметрыУчетаТовара.МаркируетсяШтрихкодомПоставщика И ЗначениеЗаполнено(СтрокаДанных.Номенклатура) Тогда

				ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Отличается признак маркировки для
существующей номенклатуры", СтрокаДанных.НомерСтроки);

			КонецЕсли;

			//Ставка НДС продажи
```

```
                                Если Не ЗначениеЗаполнено(ПараметрыУчетаТовара.СтавкаНДС) И
ЗначениеЗаполнено(СтрокаДанных.СтавкаНДС) Тогда
                                        ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Не заполнена ставка НДС продажи.",
СтрокаДанных.НомерСтроки);
                                КонецЕсли;

                                Если ПараметрыУчетаТовара.ИспользоватьХарактеристики И
СтрокаДанных.ЗначенияРеквизитовХарактеристики.Количество() = 0 Тогда
                                        ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Отсутсвуют свойства для характеристик
номенклатуры.", СтрокаДанных.НомерСтроки);
                                КонецЕсли;

                        КонецЕсли;

                        // Здесь надо проверить:
                        // Заполнение GTIN для обуви
                        Если НЕ(СтрНайти(СтрокаДанных.ТоварнаяКатегория.Наименование, "1.1") +
                                        СтрНайти(СтрокаДанных.ТоварнаяКатегория.Наименование, "1.2")) = 0
                                И ПустаяСтрока(СтрокаДанных.КиЗГИСМGTIN)
                        Тогда
                                //ВРЕМЕННО ОТКЛЮЧАЕМ
                                //ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Не заполнен GTIN.", СтрокаДанных.НомерСтроки);
                        КонецЕсли;

                        // Заполнение Производителя для детских товаров
                        Если НЕ(СтрНайти(СтрокаДанных.ТоварнаяКатегория.Наименование, "5.1") +
                                        СтрНайти(СтрокаДанных.ТоварнаяКатегория.Наименование, "5.2") +
                                        СтрНайти(СтрокаДанных.ТоварнаяКатегория.Наименование, "5.2")) = 0
                                И НЕ ЗначениеЗаполнено(СтрокаДанных.Производитель)
                        Тогда
                                ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Не заполнен Производитель номенклатуры.",
СтрокаДанных.НомерСтроки);
                        КонецЕсли;

                КонецЕсли;

                //Неизменяемые реквизиты номенклатуры
                Если ЗначениеЗаполнено(СтрокаДанных.Номенклатура) Тогда

                        ДанныеКэша = КэшированныеЗначения.Номенклатура[СтрокаДанных.Номенклатура];

                        Если ПараметрыОбработки.Поставщик <> ДанныеКэша.Поставщик Тогда

                                ТекстОшибки = "Артикул уже используется поставщиком """ + Строка(ДанныеКэша.Поставщик) +
""".";
                                ДобавитьОшибкуЗагрузкиТоваров(Ошибки, ТекстОшибки, СтрокаДанных.НомерСтроки);

                        КонецЕсли;

                        Если ЗначениеЗаполнено(СтрокаДанных.СтавкаНДС) И
ЗначениеЗаполнено(ПараметрыУчетаТовара.ВидНоменклатуры) Тогда

  //Надо сравнивать ставку НДС не в загружаемой строке проекта поставки, а в параметрах учета товара (товарные категории), в
//строке данных может быть
                                //БЕЗНДС от поставщика.
                                //Если СтрокаДанных.СтавкаНДС <> ДанныеКэша.СтавкаНДС Тогда
                                //      ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "В существующей номенклатуре иная ставка НДС.
"+ДанныеКэша.СтавкаНДС+ ", в проекте "+ПараметрыУчетаТовара.СтавкаНДС, СтрокаДанных.НомерСтроки);

                                //КонецЕсли;
                                Если ПараметрыУчетаТовара.СтавкаНДС <> ДанныеКэша.СтавкаНДС Тогда
                                        ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "В существующей номенклатуре иная ставка НДС.
"+ДанныеКэша.СтавкаНДС+ ", в товарной категории "+ПараметрыУчетаТовара.СтавкаНДС, СтрокаДанных.НомерСтроки);

                                КонецЕсли;

                        КонецЕсли;



                        Если СтрокаДанных.ВестиУчетПоГТД Тогда

                                Если СтрокаДанных.ЕдиницаИзмерения <> ДанныеКэша.ЕдиницаИзмерения Тогда
                                        ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "В существующей номенклатуре иная единица
измерения.", СтрокаДанных.НомерСтроки);
                                КонецЕсли;
```

```
                КонецЕсли

                //ИначеЕсли ПроверятьНоменклатуру Тогда
            ИначеЕсли Не ПараметрыОбработки.СоздаватьДокументыПриобретения И Не
ПараметрыОбработки.СозданиеНоменклатуры Тогда

                ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Номенклатуры не существует.", СтрокаДанных.НомерСтроки);

            КонецЕсли;

        КонецЦикла;

        ////Не допускается указание в проекте поставки различных типов брендов для одной марки
        //ПроверитьТипыБрендов(ТаблицаДанных, Ошибки);

        ПроверитьДатыВыдачиДХ(ТаблицаДанных, Ошибки);

        ПроверитьСоответствияРазмеров(ТаблицаДанных, Ошибки);

        ПроверитьРазличияРеквизитовНоменклатуры(ТаблицаДанных, Ошибки);

        //Не допускается указание в проекте поставки различных розничных цен для совокупности номенклатуры и
характеристики

КонецПроцедуры

Процедура ПроверитьДатыВыдачиДХ(ТаблицаДанных, Ошибки)


        //Не более 5 дат выдачи на проект поставки
        ТаблицаДатВыдачи = ОбщегоНазначенияУТ.СвернутаяКопияТаблицы(ТаблицаДанных, "ДатаВыдачиСДХ");

        //Пустые значения даты выдачи с ДХ
        ПустыеСтроки = ТаблицаДатВыдачи.НайтиСтроки(Новый Структура("ДатаВыдачиСДХ", '00010101'));

        Для Каждого ПустаяСтрока Из ПустыеСтроки Цикл
          ТаблицаДатВыдачи.Удалить(ПустаяСтрока);
        КонецЦикла;

        Если ТаблицаДатВыдачи.Количество() > 5 Тогда

          ТекстОшибки = СтроковыеФункцииКлиентСервер.ПодставитьПараметрыВСтроку(
                                                                    "В загружаемом файле %1 дат выдачи с
ДХ. Допускается не более 5 различных дат выдачи с ДХ на один проект.",
                                                                    ТаблицаДатВыдачи.Количество());

          ДобавитьОшибкуЗагрузкиТоваров(Ошибки, ТекстОшибки);

        КонецЕсли;

КонецПроцедуры

Процедура ПроверитьСоответствияРазмеров(ТаблицаДанных, Ошибки)

        Для Каждого СтрокаДанных Из ТаблицаДанных Цикл

          Если ЗначениеЗаполнено(СтрокаДанных.РазмерПоставщика) И Не
ЗначениеЗаполнено(СтрокаДанных.РоссийскийРазмер) Тогда
            ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Необходимо загрузить соответствия размеров поставщика.");
            возврат;
          КонецЕсли;

        КонецЦикла;

КонецПроцедуры

Процедура ПроверитьРазличияРеквизитовНоменклатуры(ТаблицаДанных, Ошибки)

        ОтборСтрок = Новый Структура("ИдентификаторТовара");

        МассивТоваров = ОбщегоНазначения.ВыгрузитьКолонку(ТаблицаДанных, "ИдентификаторТовара", Истина);

        Для Каждого ИдентификаторТовара Из МассивТоваров Цикл

          ОтборСтрок.ИдентификаторТовара = ИдентификаторТовара;
```

```
                НайденныеСтроки = ТаблицаДанных.НайтиСтроки(ОтборСтрок);

                Если НайденныеСтроки.Количество() > 1 Тогда

                        ИсходныеРеквизиты = НайденныеСтроки[0].ЗначенияРеквизитовНоменклатуры;

                        Для А = 1 По НайденныеСтроки.Количество() - 1 Цикл

                                ОписаниеОтличий = СравнитьКоллекции(ИсходныеРеквизиты,
НайденныеСтроки[А].ЗначенияРеквизитовНоменклатуры);
                                Если Не ПустаяСтрока(ОписаниеОтличий) Тогда
                                        ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Различаются реквизиты номенклатуры: " +
ОписаниеОтличий, НайденныеСтроки[0].НомерСтроки);
                                КонецЕсли;

                        КонецЦикла;

                КонецЕсли;

                КонецЦикла;

КонецПроцедуры

//сравнивает коллекции и возвращает отличия
//предполагается, что на входе структуры с одинаковыми ключами
Функция СравнитьКоллекции(КоллекцияСтрок1, КоллекцияСтрок2)

                МассивОтличий = Новый Массив;
                ОписаниеОтличий = "";

                Для Каждого ЭлементКоллекции ИЗ КоллекцияСтрок1 Цикл

                КлючПоиска = Неопределено;
                Значение1 = Неопределено;
                Значение2 = Неопределено;

                КлючПоиска = ЭлементКоллекции.Ключ;
                Значение1 = ЭлементКоллекции.Значение;

                Если КоллекцияСтрок2.Свойство(КлючПоиска, Значение2) Тогда
                        Если Значение1 <> Значение2 Тогда
                                МассивОтличий.Добавить(КлючПоиска);
                        КонецЕсли;
                КонецЕсли;

                КонецЦикла;

                Если МассивОтличий.Количество() > 0 Тогда

                ОписаниеОтличий = СтрСоединить(МассивОтличий, ",");

                КонецЕсли;

                Возврат ОписаниеОтличий;

КонецФункции

Процедура ПроверитьШтрихкодаПоставщиков(СтрокиТоваровМаркируемыхПоставщиком, Ошибки, КэшированныеЗначения)

                Для Каждого СтрокаТовара Из СтрокиТоваровМаркируемыхПоставщиком Цикл

                Если ПустаяСтрока(СтрокаТовара.Штрихкод) Тогда

                        ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Не заполнен штрихкод поставщика.",
СтрокаТовара.НомерСтроки);

                Иначе

                        //Штрихкода поставщиков используемые в качестве внутренних штрихкодов
                        ДанныеКэша = КэшированныеЗначения.Штрихкоды[СтрокаТовара.Штрихкод];

                        Если Не ДанныеКэша.НеизвестныйШтрихкод И ДанныеКэша.Номенклатура <> СтрокаТовара.Номенклатура
Тогда
                                ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Штрихкод уже используется для другой номенклатуры.
"+ДанныеКэша.Номенклатура+" "+ДанныеКэша.Номенклатура.Артикул, СтрокаТовара.НомерСтроки);
```

```
                    ИначеЕсли Не ДанныеКэша.НеизвестныйШтрихкод И ДанныеКэша.Характеристика <>
СтрокаТовара.Характеристика Тогда
                        ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Штрихкод уже используется для другой характеристики.
"+ДанныеКэша.Характеристика, СтрокаТовара.НомерСтроки);

                    КонецЕсли;

                КонецЕсли;

            КонецЦикла;

КонецПроцедуры

#КонецОбласти

#Область СозданиеШтрихкодов

Процедура ПолучитьДанныеПоВнутреннимШтрихкодам(Штрихкоды, КэшированныеЗначения)

            УстановитьПривилегированныйРежим(Истина);

            Запрос = Новый Запрос;
            Запрос.Текст =
            "ВЫБРАТЬ
            | ШтрихкодыНоменклатуры.Штрихкод КАК Штрихкод,
            | ШтрихкодыНоменклатуры.Номенклатура КАК Номенклатура,
            | ШтрихкодыНоменклатуры.Характеристика КАК Характеристика
            |ИЗ
            | РегистрСведений.ШтрихкодыНоменклатуры КАК ШтрихкодыНоменклатуры
            |ГДЕ
            | ШтрихкодыНоменклатуры.Штрихкод В(&МассивШтрихкодов)";

            МассивШтрихкодов = Новый Массив;

            Для Каждого ТекШтрихкод Из Штрихкоды Цикл

              Штрихкод = ВРег(СокрЛП(ТекШтрихкод));
              МассивШтрихкодов.Добавить(Штрихкод);

              ИнформацияПоШтрихкодам = Новый Структура("Штрихкод, Номенклатура, Характеристика,
НеизвестныйШтрихкод");
              ИнформацияПоШтрихкодам.Штрихкод = Штрихкод;
              ИнформацияПоШтрихкодам.НеизвестныйШтрихкод = Истина;

              КэшированныеЗначения.Штрихкоды.Вставить(Штрихкод, ИнформацияПоШтрихкодам);

            КонецЦикла;

            Запрос.УстановитьПараметр("МассивШтрихкодов", МассивШтрихкодов);

            Выборка = Запрос.Выполнить().Выбрать();
            Пока Выборка.Следующий() Цикл
              ТекДанные = КэшированныеЗначения.Штрихкоды[ВРег(СокрЛП(Выборка.Штрихкод))];
              ТекДанные.Номенклатура       = Выборка.Номенклатура;
              ТекДанные.Характеристика      = Выборка.Характеристика;
              ТекДанные.НеизвестныйШтрихкод = Ложь;
            КонецЦикла;

КонецПроцедуры

Функция ДанныеШтрихкодовПоставщиков(Штрихкоды)

            Запрос = Новый Запрос;
            Запрос.Текст =
            "ВЫБРАТЬ
            | ШтрихкодыПоставщиковНоменклатуры.Штрихкод,
            | ШтрихкодыПоставщиковНоменклатуры.Номенклатура,
            | ШтрихкодыПоставщиковНоменклатуры.Характеристика
            |ИЗ
            | РегистрСведений.ШтрихкодыПоставщиковНоменклатуры КАК ШтрихкодыПоставщиковНоменклатуры
            |ГДЕ
            | ШтрихкодыПоставщиковНоменклатуры.Штрихкод В(&Штрихкоды)";

            Запрос.УстановитьПараметр("Штрихкоды", Штрихкоды);

            возврат Запрос.Выполнить().Выгрузить();
```

```
КонецФункции

Процедура СоздатьШтрихкода(ТаблицаТоваров, Ошибки, КэшированныеЗначения)

		//Внутренние штрихкода
		ТаблицаТоваровДляВнутреннихШтрихкодов = ТаблицаТоваров.Скопировать(Новый
Структура("МаркируетсяШтрихкодомПоставщика", Ложь), "Номенклатура, Характеристика, НовыйТовар");
		ТаблицаТоваровДляВнутреннихШтрихкодов.Свернуть("Номенклатура, Характеристика, НовыйТовар");

		Для Каждого СтрокаТовара Из ТаблицаТоваровДляВнутреннихШтрихкодов Цикл

		 Если СтрокаТовара.НовыйТовар Тогда

			ПроверитьСоздатьВнутреннийШтрихкод(СтрокаТовара, Ошибки, КэшированныеЗначения);

		 Иначе

			ВнутреннииеШтрихкоды =
РегистрыСведений.ШтрихкодыНоменклатуры.ШтрихкодыНоменклатуры(СтрокаТовара.Номенклатура, СтрокаТовара.Характеристика,
Справочники.УпаковкиЕдиницыИзмерения.ПустаяСсылка());

			Если ВнутреннииеШтрихкоды.Количество() = 0 Тогда
				ПроверитьСоздатьВнутреннийШтрихкод(СтрокаТовара, Ошибки, КэшированныеЗначения);
			КонецЕсли;

		 КонецЕсли;

		КонецЦикла;

		//Штрихкода поставщиков
		МассивШтрихкодовПоставщика = ОбщегоНазначения.ВыгрузитьКолонку(ТаблицаТоваров, "Штрихкод", Истина);
		ТаблицаДанныхШтрихкодовПоставщиков = ДанныеШтрихкодовПоставщиков(МассивШтрихкодовПоставщика);

		Для Каждого СтрокаТовара Из ТаблицаТоваров Цикл

		 Если ПустаяСтрока(СтрокаТовара.Штрихкод) Тогда
			продолжить;
		 КонецЕсли;

		 //Штрихкод поставщика
		 ПроверитьСоздатьШтрихкодПоставщика(СтрокаТовара.Номенклатура, СтрокаТовара.Характеристика,
СтрокаТовара.Штрихкод, ТаблицаДанныхШтрихкодовПоставщиков);

		 //Внутренний штрихкод по штрихкоду поставщика
		 Если СтрокаТовара.МаркируетсяШтрихкодомПоставщика Тогда
			ПроверитьСоздатьВнутреннийШтрихкод(СтрокаТовара, Ошибки, КэшированныеЗначения,
СтрокаТовара.Штрихкод);
		 КонецЕсли;

		 //Иногда нам предоставляют штрихкода длиной 12 символов.
		 //Это происходить по двум причинам:
		 // - это EAN-13 без контрольного символа
		 // - это UPC-A отличающийся от EAN-13 тем, что в нем первым символом всегда идет "0" и этот символ опускается

		 Если СтрДлина(СтрокаТовара.Штрихкод) = 12 И
СтроковыеФункцииКлиентСервер.ТолькоЦифрыВСтроке(СтрокаТовара.Штрихкод) Тогда

			КодUPC_A = "0" + Лев(СтрокаТовара.Штрихкод, 11);
			КонтрольныйСимволUPC_A = Прав(СтрокаТовара.Штрихкод, 1);

			Если КонтрольныйСимволUPC_A =
РегистрыСведений.ШтрихкодыНоменклатуры.КонтрольныйСимволEAN(КодUPC_A, 13) Тогда
				//UPC-A
				ДополнительныйШтрихкод = КодUPC_A + КонтрольныйСимволUPC_A;
			Иначе
				//EAN-13 без контрольного символа
				КонтрольныйСимволEAN13 =
РегистрыСведений.ШтрихкодыНоменклатуры.КонтрольныйСимволEAN(СтрокаТовара.Штрихкод, 13);
				ДополнительныйШтрихкод = СтрокаТовара.Штрихкод + КонтрольныйСимволEAN13;
			КонецЕсли;

			ПроверитьСоздатьШтрихкодПоставщика(СтрокаТовара.Номенклатура, СтрокаТовара.Характеристика,
ДополнительныйШтрихкод, ТаблицаДанныхШтрихкодовПоставщиков);

			//Внутренний штрихкод по штрихкоду поставщика
			Если СтрокаТовара.МаркируетсяШтрихкодомПоставщика Тогда
```

```
                                    ПроверитьСоздатьВнутреннийШтрихкод(СтрокаТовара, Ошибки, КэшированныеЗначения,
ДополнительныйШтрихкод);

                        КонецЕсли;

                КонецЕсли;

            КонецЦикла;

КонецПроцедуры

Процедура ПроверитьСоздатьШтрихкодПоставщика(Номенклатура, Характеристика, Штрихкод,
ТаблицаДанныхШтрихкодовПоставщиков)

            ДобавитьШтрихкодПоставщика = Истина;

            НайденныеШтрихкода = ТаблицаДанныхШтрихкодовПоставщиков.НайтиСтроки(Новый Структура("Штрихкод",
Штрихкод));

            Для Каждого ДанныеШтрихкода Из НайденныеШтрихкода Цикл

                Если ДанныеШтрихкода.Номенклатура = Номенклатура И ДанныеШтрихкода.Характеристика = Характеристика Тогда
                    ДобавитьШтрихкодПоставщика = Ложь;
                    прервать;
                КонецЕсли;

            КонецЦикла;

            Если ДобавитьШтрихкодПоставщика Тогда

                НаборЗаписей = РегистрыСведений.ШтрихкодыПоставщиковНоменклатуры.СоздатьНаборЗаписей();
                НаборЗаписей.Отбор.Штрихкод.Установить(Штрихкод);
                НаборЗаписей.Отбор.Номенклатура.Установить(Номенклатура);
                НаборЗаписей.Отбор.Характеристика.Установить(Характеристика);

                НаборЗаписей.Отбор.Владелец.Установить(ПредопределенноеЗначение("Справочник.Партнеры.ПустаяСсылка"));

                Запись = НаборЗаписей.Добавить();
                Запись.Штрихкод       = Штрихкод;
                Запись.Номенклатура   = Номенклатура;
                Запись.Характеристика = Характеристика;

                Запись.Владелец = ПредопределенноеЗначение("Справочник.Партнеры.ПустаяСсылка");

                НаборЗаписей.ОбменДанными.Загрузка = Истина;
                НаборЗаписей.Записать();
                НоваяСтрока = ТаблицаДанныхШтрихкодовПоставщиков.Добавить();
                НоваяСтрока.Номенклатура   = Номенклатура;
                НоваяСтрока.Характеристика = Характеристика;
                НоваяСтрока.Штрихкод       = Штрихкод;

            КонецЕсли;

КонецПроцедуры

Процедура ПроверитьСоздатьВнутреннийШтрихкод(СтрокаТовара, Ошибки, КэшированныеЗначения, Штрихкод = "")

            //Одномоментно можно записывать только один штрихкод
            Если ПустаяСтрока(Штрихкод) Тогда

                Штрихкод = КэшированныеЗначения.СвободныеШтрихкода[0];

            КонецЕсли;

            ДанныеКэша = КэшированныеЗначения.Штрихкоды[Штрихкод];

            Если ДанныеКэша = Неопределено ИЛИ ДанныеКэша.НеизвестныйШтрихкод Тогда

                НаборЗаписей = РегистрыСведений.ШтрихкодыНоменклатуры.СоздатьНаборЗаписей();
                НаборЗаписей.Отбор.Штрихкод.Установить(Штрихкод);

                Запись = НаборЗаписей.Добавить();
                Запись.Штрихкод       = Штрихкод;
                Запись.Номенклатура   = СтрокаТовара.Номенклатура;
                Запись.Характеристика = СтрокаТовара.Характеристика;
```

```
                НаборЗаписей.ОбменДанными.Загрузка = Истина;
                НаборЗаписей.Записать();

                Если КэшированныеЗначения.СвободныеШтрихкода.Количество() Тогда
                        КэшированныеЗначения.СвободныеШтрихкода.Удалить(0);
                КонецЕсли;

                ПолучитьДанныеПоВнутреннимШтрихкодам(ОбщегоНазначенияКлиентСервер.ЗначениеВМассиве(Штрихкод),
КэшированныеЗначения);


            ИначеЕсли ДанныеКэша.Номенклатура <> СтрокаТовара.Номенклатура И ДанныеКэша.Характеристика <>
СтрокаТовара.Характеристика Тогда

                ДобавитьОшибкуЗагрузкиТоваров(Ошибки, "Попытка изменения внутреннего штрихкода номенклатуры.",
СтрокаТовара.НомерСтроки);

            КонецЕсли;

КонецПроцедуры

#КонецОбласти

#КонецОбласти

#КонецОбласти

#КонецЕсли
```

Приложение Г

Результат опроса сотрудников предприятия

Таблица Г.1 - Результат опроса сотрудников предприятия

№	X	Y	№	X	Y
1	1	1	21	1	1
2	1	1	22	1	1
3	1	1	23	1	1
4	0	0	24	1	1
5	1	0	25	0	0
6	1	1	26	1	0
7	0	1	27	0	1
8	0	0	28	1	1
9	1	0	29	1	0
10	1	1	30	1	1
11	0	0	31	0	0
12	1	1	32	1	1
13	0	0	33	1	1
14	1	1	34	1	0
15	0	0	35	1	1
16	1	1	36	1	0
17	0	0	37	1	1
18	1	1	38	1	1
19	1	0	39	1	1
20	0	1	40	1	1