Оглавление

[Введение 4](#_Toc311625324)

[1. Аналитический раздел 6](#_Toc311625325)

[1.1. Анализ правил делового регламента 6](#_Toc311625326)

[1.2. Построение концептуальной информационно-логической модели системы 14](#_Toc311625327)

[1.3. Требования к проекту. 16](#_Toc311625328)

[2. Конструкторский раздел 18](#_Toc311625329)

[2.1. Создание базы данных 18](#_Toc311625330)

[2.2. Таблицы базы данных 18](#_Toc311625331)

[2.2.1. Таблицы складского учета 20](#_Toc311625332)

[2.2.2. Таблицы поставщиков и получателей 21](#_Toc311625333)

[2.2.3. Таблицы-справочники 22](#_Toc311625334)

[2.3. Ограничения 22](#_Toc311625335)

[2.4. Триггеры 25](#_Toc311625336)

[2.5. Хранимые процедуры 26](#_Toc311625337)

[2.6. Функции 27](#_Toc311625338)

[2.7. Представления 30](#_Toc311625339)

[3. Технологический раздел 33](#_Toc311625340)

[3.1. Выбор языка программирования и среды разработки 33](#_Toc311625341)

[3.2. Технологии доступа к внешним источникам данных 33](#_Toc311625342)

[3.2.1. Создание объектной модели LINQ to SQL 35](#_Toc311625343)

[3.2.2. Примеры обращения с запросами к базе данных с использованием технологии LINQ to SQL 36](#_Toc311625344)

[3.2.3. Изменение базы данных 37](#_Toc311625345)

[3.2.4. Вызов хранимых процедур и определенных пользователем функций 39](#_Toc311625346)

[3.3. Структура приложения 39](#_Toc311625347)

[3.4. Интерфейс приложения 41](#_Toc311625348)

[3.5. Тестирование приложения 47](#_Toc311625349)

[Заключение 48](#_Toc311625350)

[Список использованной литературы 49](#_Toc311625351)

[Приложение А. Первичные формы документов 50](#_Toc311625352)

[Приложение Б. Варианты заполнения таблицы «Документ» 58](#_Toc311625353)

[Приложение В. Сценарии создания таблиц базы данных 59](#_Toc311625354)

[Приложение Г. Триггеры 63](#_Toc311625355)

[Приложение Д. Функции 65](#_Toc311625356)

Введение

С необходимостью создания складских помещений сталкиваются все производственные и торговые структуры, а также предприятия сферы обслуживания. В общем случае функционирование каждого склада состоит из трех основных операций: прием, размещение и выдача товара, каждая из которых сопровождается определенным набором документов (приходные ордера, карточки учета материалов, накладные и т.д.). Соответственно, чем интенсивнее движение товара на складе, тем интенсивнее и документооборот, а значит, затрачивается больше времени и усилий на оформление и обработку документов. Один из вариантов решения данной проблемы – автоматизация управления складом, которая удовлетворяет потребность в полной, своевременной и достоверной информации о товаре и обеспечивает достаточную скорость выполнения складских операций. Чаще всего удовлетворение данных потребностей является достаточным для обеспечения роста эффективности работы склада.

Таким образом, актуальность проблемы автоматизации управления складом на сегодняшний день достаточно высока. Этот факт подтверждает и наличие множества попыток решения данной проблемы. Сегодня существует большое количество программных комплексов, нацеленных на автоматизацию складов: программа управления материалами на складе от 1С, интернет-сервис «Мой Склад», «ARTIS-склад» и многие другие.

Целью данного курсового проекта является разработка автоматизированной системы, позволяющей:

* осуществлять стандартный складской учет и контроль движения товарно-материальных ценностей на складах производственного предприятия (автоматизировать операции по обработке прихода товара, его перемещения внутри организации и отпуска на сторону);
* осуществлять работу с типовыми приходно-расходными и учетными документами.

Для достижения поставленной цели необходимо решить следующие задачи:

Анализ правил делового регламента, действующих в данной предметной области.

Составление перечня требований к проекту.

Разработка и реализация базы данных.

Разработка и реализация пользовательского приложения.

1. Аналитический раздел
   1. Анализ правил делового регламента

Для конкретизации предметной области предполагается, что автоматизированная система складского учета разрабатывается для некоторой гипотетической организации «Умелые ручки», занимающейся производством и продажей товаров для рукоделия. Разработка автоматизированной системы должна вестись на основе правил делового регламента, принятых в данной организации в отношении операций складского учета. Далее представлены эти правила, разбитые для удобства на группы.

Структура организации.

1. В организации может быть несколько подразделений, в том числе и складов разного назначения.

Каждое подразделение может заниматься только одним видом деятельности.

Подразделение может находиться за пределами территории организации (филиал).

Выпускаемая подразделением продукция (или оказываемые услуги) может иметь учетную единицу измерения.

В каждом подразделении может работать несколько сотрудников.

Каждый сотрудник занимает только одну должность.

Товарно-материальные ценности (ТМЦ).

Необходимо иметь следующую информацию о материальных ценностях, находящихся на складе:

* наименование, сорт, размер, марка;
* цена в рублях;
* налог на добавленную стоимость для данного материала;
* учетная единица (определяется по ОКЕИ – Общероссийскому Классификатору Единиц Измерения);
* номер счета, субсчета (для обобщения информации о наличии и движении материалов предназначен счет №10);
* номер паспорта (для изделий, содержащих драгоценные металлы).

В качестве дополнительной информации можно указать:

* срок годности;
* уровень запаса (минимальное количество товара на складе, которое должно быть в наличии);
* код (если в организации применяется система кодирования).

На каждую новую материальную ценность заводится карточка учета материалов. В ней отражаются все приходно-расходные операции по данному товару, а также сведения о поставщике материала и складе, на котором он хранится. Если одно и то же наименование товара хранится на нескольких складах, на каждом из них для него заводится своя учетная карточка. Отдельные карточки заводятся также на материал, поставляемый несколькими поставщиками.

Прием материальных ценностей на склад.

Операция приема товаров на склад фиксируется в приходном ордере (форма М-4), представленном в приложении А.1. Данный документ выписывается на фактически принятое число ценностей и заполняется по следующим правилам:

1. В строке «Структурное подразделение» указывается наименование подразделения, осуществляющего прием товара.
2. Приходный ордер может быть выписан только в день приема товара.
3. В графе «Склад» указывается название склада, на который поступает товар.
4. Поставщиком товара может являться либо одно из подразделений, либо сторонняя организация. Приходный ордер не может содержать сведения о товарах от разных поставщиков.
5. При приеме материалов от сторонней организации необходимо указать номера сопроводительного и платежного документов; при приеме от хозяйств (подразделений) своей организации – только сопроводительного.
6. Приходный ордер выписывается только в том случае, когда фактически принятое количество ценностей совпадает с количеством, указанным в документах поставщика.
7. Сумма без учета НДС = Количество товара \* Цена;
8. Сумма НДС = Цена \* НДС \* Количество товара / 100%;
9. Сумма с учетом НДС = Сумма без учета НДС + Сумма НДС.
10. В графе «Принял» расписывается лицо, принимающее товары и являющееся сотрудником подразделения, указанного в п.1. В графе «Сдал» должна стоять подпись сотрудника подразделения, которое занимается снабжением (в организации «Умелые ручки» за снабжение отвечает отдел поставок), а если поставщиком является хозяйство своей организации – подпись материально-ответственного лица – сотрудника подразделения-поставщика.

Перемещение товарно-материальных ценностей внутри организации.

При выдаче материала подразделению организации, находящемуся в пределах ее территории, выписывается документ формы М-11, утвержденной Госкомстатом России, - требование-накладная. (Приложение А.2) Правила оформления данного документа таковы:

1. Отправителем является подразделение, выдающее товарно-материальные ценности, то есть склад, на котором они хранятся.
2. Получателем ТМЦ является структурное подразделение организации, находящееся в пределах ее территории и не являющееся отправителем.
3. В графе «Учетная единица выпуска продукции (работ, услуг)» указывается (если таковая имеется) учетная единица продукции, выпускаемой подразделением – получателем товара.
4. В графе «Через кого» указывается лицо, непосредственно получающее материалы со склада и являющееся сотрудником подразделения-получателя. В графе «Принял» должна стоять подпись данного лица.
5. В графе «Затребовал» указывается лицо, которое затребовало ценности. Оно также является сотрудником подразделения, указанного в графе «Получатель». Затребовать и получить материалы со склада может один и тот же сотрудник.
6. В графе «Разрешил» указывается сотрудник склада, выдающего товары, который разрешил выдачу указанных в накладной материалов. Этот же сотрудник может и выдавать материал со склада, тогда в строке «Отпустил» будет стоять его подпись.
7. Отпущенное количество товара может не совпадать с затребованным, например, в случае, когда затребованное количество материала превышает имеющийся на складе остаток.

Отпуск ТМЦ на сторону.

Выдача товарно-материальных ценностей на сторону означает их перемещение за пределы территории организации. Получателем может являться как сторонняя организация, так и хозяйство своей организации, расположенное за ее территорией. Данная операция фиксируется в накладной на отпуск материалов на сторону (форма М-15), представленной в приложении А.3. Ниже перечислены требования к заполнению этого документа:

1. Отправителем, как и в случае с требованием-накладной, является склад, с которого производится выдача.
2. Если получателем является сторонняя организация, в графе «Кому» указывается ее наименование, в графах «Структурное подразделение» и «Вид деятельности» для получателя ставится прочерк. Если материалы отпускаются хозяйству своей организации, его наименование и вид деятельности указываются в соответствующих графах, а в строке «Кому» указывается сотрудник подразделения, затребовавший материал.
3. В графу «Ответственный за поставку» заносятся сведения о подразделении, отвечающем за сбыт товара. В организации «Умелые ручки» это отдел поставок. В случае отпуска ТМЦ хозяйству своей организации за поставку может отвечать и подразделение – получатель.
4. В строке «Через кого» указывается получающий товары сотрудник ответственного за поставку подразделения. Он же расписывается в графе «Получил».
5. Накладная на отпуск материалов на сторону может быть выписана на основании договора, распоряжения, приказа и др. документов.
6. Затребованное и отпущенное количество товара могут не совпадать.

Учет движения материалов.

Все операции, связанные с приходом и выдачей материала, отражаются в карточке учета (форма М-17), заведенной для данного материала. В ней фиксируется приход и расход материала, а также текущий остаток. Форма карточки представлена в приложении А.4. В организации «Умелые ручки» принимаются следующие требования к ее заполнению:

1. В графах «Структурное подразделение» и «Склад» указывается наименование склада, на котором находится материал.
2. Указание места хранения материала является необязательным.
3. В графе «Поставщик» указывается фирма, поставляющая товар. Если материал был получен от подразделения своей организации, указывается ее наименование.
4. Так как в данной работе не предполагается наличие в качестве ТМЦ драгоценных металлов, соответствующие графы КУМ не заполняются и не рассматриваются.
5. Графа «От кого получено или кому отпущено» заполняется следующим образом:

* Если отправителем или получателем товара является сторонняя организация, вносится ее наименование;
* Если в роли отправителя (получателя) выступает подразделение своей организации, вносится его наименование, а также должность, фамилия и инициалы материально-ответственного лица, который сдал (получил) материалы.

На рис. 1.1 и 1.2 представлена модель бизнес-процессов предприятия «Умелые ручки» и ее детализация.

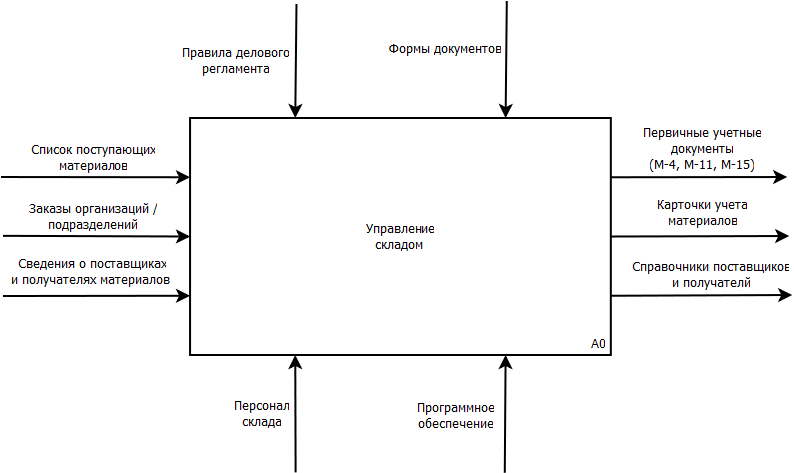


Рис. 1.1. Контекстная диаграмма функционирования организации «Умелые ручки»

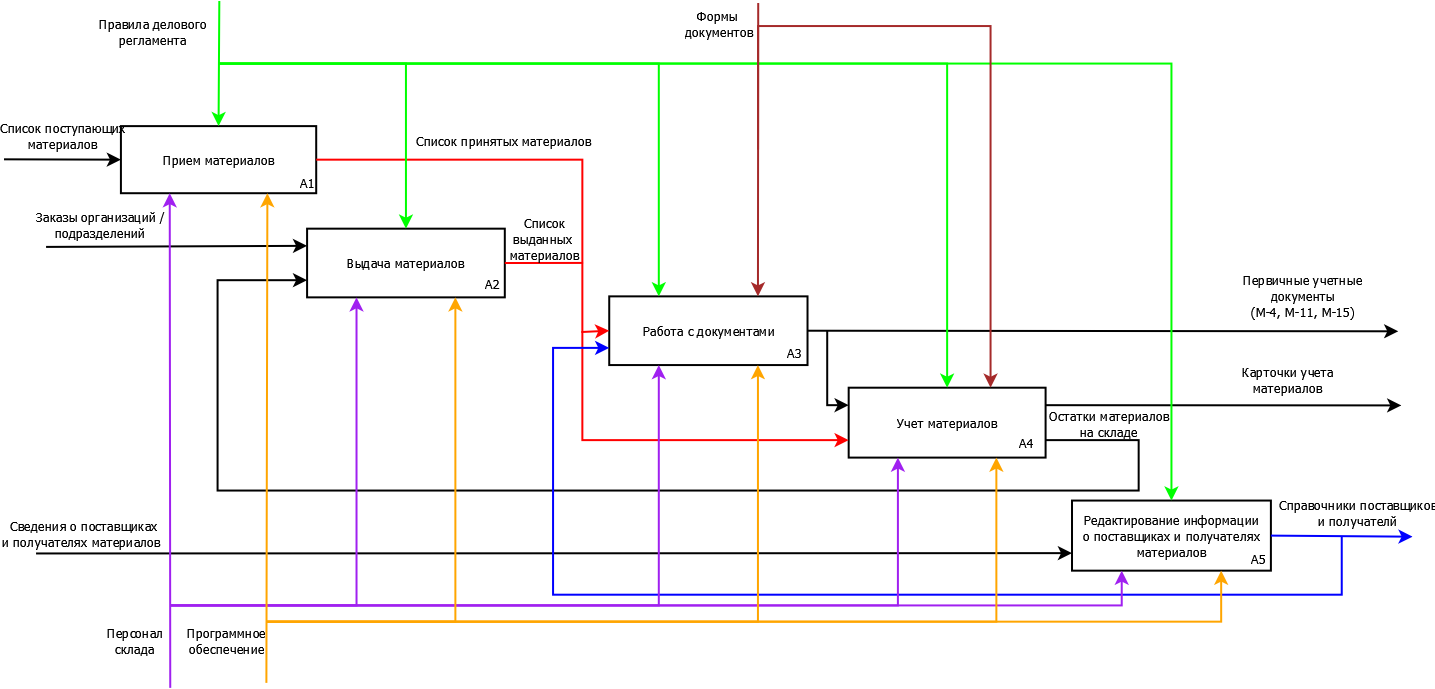


Рис. 1.2. Диаграмма декомпозиции бизнес-процессов организации «Умелые ручки»

* 1. Построение концептуальной информационно-логической модели системы

Исходя из рассмотренных требований и правил делового регламента, в разрабатываемой модели можно выделить следующие основные сущности:

* Материал;
* Документ;
* Сторонняя организация;
* Подразделение;
* Сотрудник (материально-ответственное лицо).

Для снижения информационной избыточности можно также выделить вспомогательные сущности (справочники):

* Единица измерения;
* Учетная единица;
* Тип операции;
* Тип документа;
* Должность;
* Вид деятельности.

Склад является структурным подразделением, однако он занимает отдельное место в системе, поэтому его также можно вынести как отдельную сущность.

На основе описанных выше правил делового регламента можно выделить следующие отношения между приведенными сущностями:

* Экземпляр материала может иметь только одну единицу измерения.
* Экземпляр документа может быть выписан только на одну операцию и имеет определенный тип.
* Сотрудник может занимать только одну должность.
* Подразделение может заниматься только конкретным видом деятельности.
* Карточка учета заводится на конкретном складе на отдельный материал, поступающий от одного поставщика, поэтому ее можно рассматривать как связующее звено между сущностями «Материал» и «Сторонняя организация».

Карточка учета материалов должна содержать учетные записи, сделанные на основании различных документов, вследствие чего целесообразно оформить оборотную ведомость как самостоятельный элемент, связывающий КУМ с документом, на основании которого в ней делается очередная запись.

В одном документе могут содержаться сведения о нескольких материалах, значит, для этих сущностей также необходимо выделить связующий элемент.

Отдельно стоит рассмотреть вопрос о связи экземпляра документа с отправителем и получателем. Исходя из рассмотренных выше форм, в документе должны содержаться следующие сведения:

* отправитель;
* получатель;
* лицо, ответственное за сдачу товара;
* лицо, ответственное за прием товара.

При этом в качестве отправителей и получателей могут выступать как сторонние организации, так и подразделения своей организации. Таким образом, один атрибут является внешним ключом сразу для двух сущностей. Один из возможных путей решения данной проблемы – добавление сущности «Документ» дополнительных атрибутов, которые будут показывать, какой из ключей используется в данном экземпляре («Получатель» - «Сторонняя организация» или «Получатель - Подразделение»). В случае, когда получателем (или отправителем) является хозяйство своей организации, необходимо отразить в документе лицо, которому был выдан (которым был принят) товар. Так как каждое материально-ответственное лицо связано с подразделением идентифицирующей связью, внешние ключи «Получатель» и «Отправитель» сущности «Документ» будут указывать на сущность «Сотрудник».

Созданный в программе документ должен быть сохранен на диске в формате .xls. Чтобы впоследствии этот документ можно было просмотреть средствами создаваемого приложения, необходимо выделить сущность, сопоставляющую документу место его хранения.

ER- диаграмма, описывающая концептуальную модель системы, приведена в приложении 1.3.

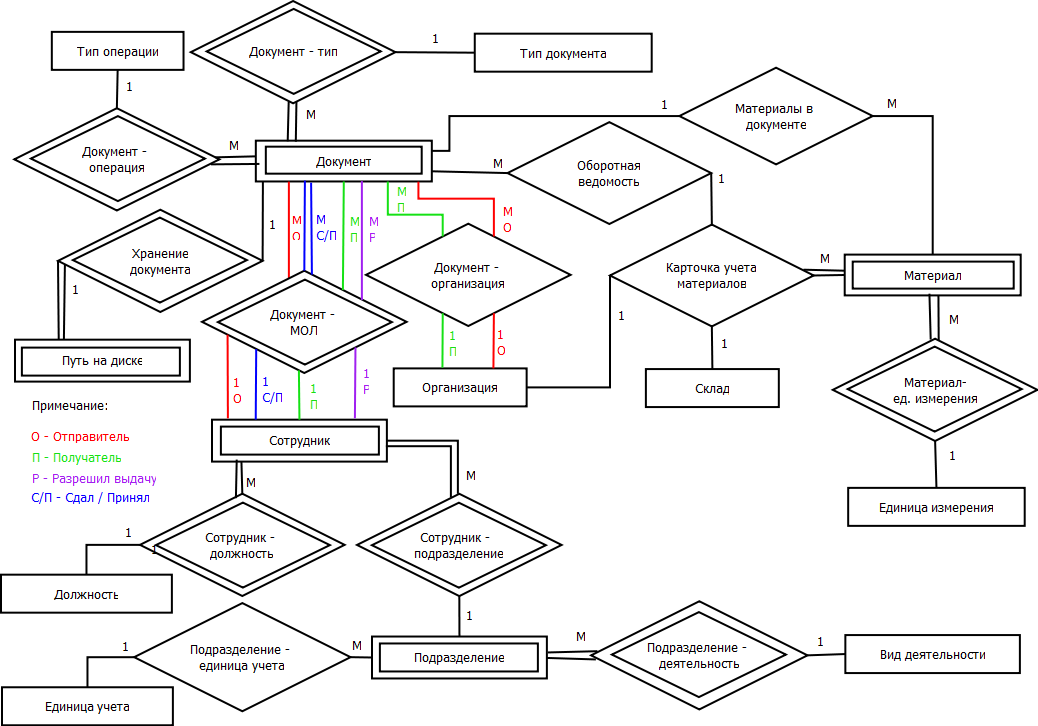


Рис. 1.3. ER-диаграмма

* 1. Требования к проекту.

Необходимо разработать автоматизированную систему учета материала на складе, состоящую из базы данных и пользовательского приложения. С учетом описанных выше правил делового регламента, действующих в сфере складского учета, к составу базы данных и функционалу пользовательского приложения выдвигаются следующие требования.

База данных должна содержать:

* сведения о поступающих товарах (номенклатуру): артикул, вес, цена, количество товара, единица учета, положение на складе и т. д.;
* информацию о сопровождающих документах (номер, тип документа, дата составления и т.д.);
* путь к документу на диске;
* записи балансового учета;
* данные о поставщиках, отправителях и получателях товара;
* справочные данные (принятые в организации единицы учета, виды деятельности подразделений, список должностей и т. д.).

Приложение должно иметь следующий функционал:

* оформление первичных учетных документов (приходный ордер, накладная на отпуск материалов на сторону, требование-накладная);
* сохранение сформированных документов в формате .xls;
* просмотр документов как средствами самого приложения, так и с помощью Microsoft Office Excel;
* редактирование данных о подразделениях, сотрудниках, поставщиках, материалах и т.д.;
* ведение карточек учета материала;
* просмотр записей в КУМ за определенную дату или период.

1. Конструкторский раздел
   1. Создание базы данных

Для разработки базы данных была выбрана СУБД Microsoft SQL Server 2008 Express Edition.

Создать новую базу данных можно либо с помощью команд языка Transact-SQL (T-SQL), либо используя предоставляемые СУБД графические средства. Сценарий создания базы данных для разрабатываемой системы имеет следующий вид:

CREATE DATABASE Warehouse

GO

где Warehouse – имя создаваемой базы данных.

* 1. Таблицы базы данных

Разрабатываемая база данных содержит 16 таблиц, которые можно условно поделить на три логические группы:

1. Складской учет.
2. Поставщики и получатели.
3. Справочники.

Первая группа отвечает за хранение данных о находящихся на складе материалах и карточек учета материалов, включающих в себя сведения о движении товара. К ней же относятся таблицы, содержащие сведения о приходно-расходных документах.

Вторую группу составляют таблицы, хранящие сведения о сторонних организациях, с которыми сотрудничает фирма «Умелые ручки»; о материально-ответственных лицах, выдающих и принимающих товары, а также о подразделениях, сотрудниками которых они являются.

В третью группу вынесены таблицы, содержащие вспомогательную информацию: сведения о единицах измерения и учета, используемых в организации «Умелые ручки», типах документов, кодах операций и др.

Все таблицы представлены в схеме на рис. 2.1.

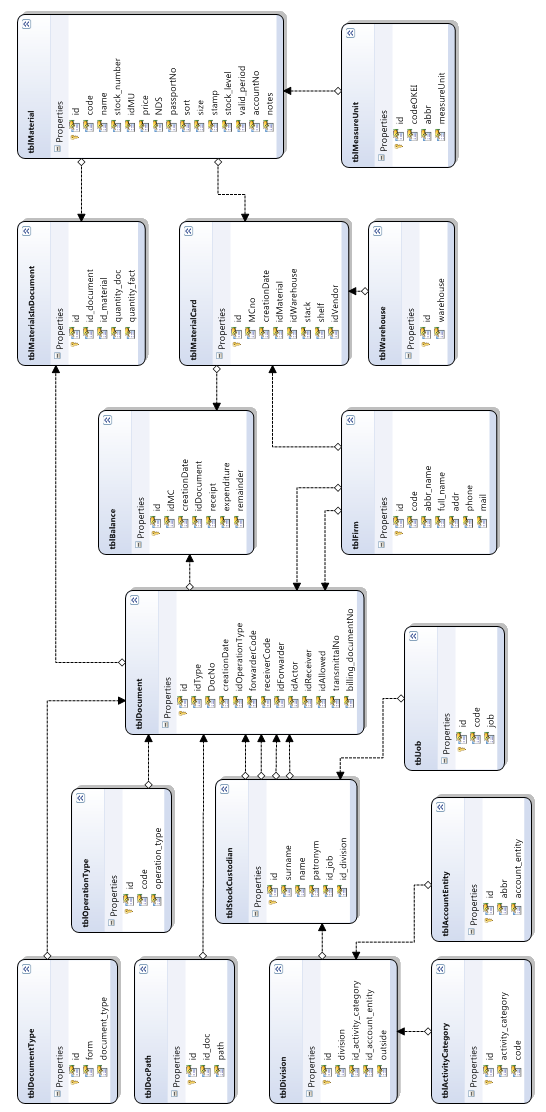


Рис. 2.1. Схема базы данных Warehouse

* + 1. Таблицы складского учета

Данная группа состоит из следующих 6 таблиц:

1. Материал. Содержит сведения об атрибутах товара, необходимых для заполнения первичных учетных документов: наименование, размер, марка, сорт, цена, ставка НДС для данного товара, номенклатурный номер, единица измерения, срок годности и др.
2. Карточка учета материалов. Запись этой таблицы содержит дату создания карточки и ее номер, а также данные о том, кто является поставщиком материала, на каком складе данный материал хранится и где именно он расположен (стеллаж, полка).
3. Оборотная ведомость. Имеет две функции: связь карточки учета материала с документом, на основании которого в ней делается очередная запись, и ведение учета прихода и расхода каждого материала. Формат записи таблицы приближен к формату, используемому в форме М-17 (см. Приложение А.4). На основании информации из данной таблицы формируются записи о движении товара, отражаемые в карточке учета материалов.
4. Документ. Одна из самых важных таблиц базы данных. Она содержит следующую информацию: дата создания документа, его номер и тип; тип операции, сопровождаемой данным документом, и данные о сопроводительных и платежных документах, на основании которых выписывается ордер или накладная. В этой таблице должны также храниться сведения об отправителе и получателе товаров, указанных в документе. В случае приходного ордера (М-4) должно быть указано лицо, сдающее материалы на склад, а в случае накладных (М-11, М-15) – лицо, принимающее товары со склада, и лицо, разрешившее выдачу. Для этого в таблице содержатся поля «Отправитель» (idForwarder), «Получатель» (idReceiver), «Сдал / получил» (idActor), «Разрешил». Поля «код отправителя» и «код получателя» указывают, кто является отправителем и получателем товара: сторонняя организация или материально-ответственное лицо, представляющее подразделение своей организации. В таблице Б.1 приложения Б приведены варианты заполнения указанных полей в зависимости от типа документа. В поле «Разрешил» вне зависимости от типа накладной указывается сотрудник склада, с которого происходит выдача товара.
5. Материалы в документе. Данная таблица играет роль связующего звена между документом и отраженным в нем списком материалов. В ней для каждого созданного документа хранятся указатели на каждое содержащееся в нем наименование с указанием количества принимаемого / выдаваемого материала. На основе данных этой таблицы строится балансовая ведомость.
6. Место хранения документа. Данная таблица сопоставляет каждому документу путь на диске к месту его хранения в формате Excel.

Сценарии создания данных таблиц приведены в Приложении В.1.

* + 1. Таблицы поставщиков и получателей

В данную группу входят следующие таблицы:

1. Фирма. Таблица содержит сведения об организациях, сотрудничающих с ОАО «Умелые ручки»: наименование, адрес, телефон, электронная почта. Первая запись таблицы хранит информацию о своей организации.
2. Подразделение. В данной таблице содержатся сведения о подразделениях организации: название, вид деятельности, учетная единица выпускаемой продукции (если таковая имеется).
3. Склад. Данная таблица содержит список имеющихся в организации складов. Хотя склад является структурным подразделением, во многих формируемых документах он фигурирует как самостоятельный элемент. Поэтому для него создана отдельная таблица, которая представляет собой выборку из таблицы подразделений названий подразделений, видом деятельности которых является хранение.
4. Материально-ответственное лицо. Таблица содержит данные о сотрудниках организации, ответственных за:

* Сдачу товара на склад / получение товара со склада
* Прием и отпуск товаров на складе
* Выдачу разрешения на отпуск материала со склада

Для каждого сотрудника хранятся его фамилия, имя и отчество, а также подразделение, в котором он работает, и должность, которую он занимает.

Сценарии создания этой группы таблиц представлены в приложении В.2.

* + 1. Таблицы-справочники

Данные таблицы содержат атрибутивную информацию, используемую в таблицах первых двух групп:

1. Единица учета. Содержит список принятых в организации единиц учета выпускаемой подразделениями продукции.
2. Единица измерения. Содержит описание единиц, в которых измеряется тот или иной товар, а также их коды по Общероссийскому Классификатору Единиц Измерения.
3. Вид деятельности. Список видов деятельности, которыми может заниматься то или иное подразделение.
4. Тип документа. В таблице хранятся записи о видах обрабатываемых документов: приходный ордер М-4, требование-накладная М-11 и накладная на отпуск материалов на сторону М-15.
5. Должность. Список должностей, которые могут занимать сотрудники организации.
6. Тип операции. Содержит коды и наименования обрабатываемых операций, принятые в организации «Умелые ручки».

В приложении В.3 представлены сценарии создания справочников.

* 1. Ограничения

Ограничения NOT NULL, UNIQUE и DEFAULT заданы при создании таблиц (см. приложение В). Ограничения внешнего ключа, задающие связи между таблицами, представлены в таблице 2.1.

Таблица 2.1. Ограничения внешнего ключа

|  |  |  |
| --- | --- | --- |
| **Имя ограничения** | **Сценарий создания** | **Пояснение** |
| 1 | 2 | 3 |
| Материал | | |
| fk\_material\_measureUnit | ALTER TABLE tblMaterial  ADD CONSTRAINT fk\_material\_measureUnit  FOREIGN KEY (idMU) REFERENCES tblMeasureUnit(id) | Связь материала с единицей измерения. |
| Карточка учета материалов | | |
| fk\_mc\_material | ALTER TABLE tblMaterialCard  ADD CONSTRAINT fk\_mc\_material  FOREIGN KEY (idMaterial)  REFERENCES tblMaterial(id) | Связь КУМ с материалом. |
| fk\_mc\_warehouse | ALTER TABLE tblMaterialCard  ADD CONSTRAINT  FOREIGN KEY (idWarehouse) REFERENCES tblWarehouse(id) | Связь КУМ со складом, на котором хранится материал. |
| fk\_mc\_vendor | ALTER TABLE tblMaterialCard  ADD CONSTRAINT fk\_mc\_vendor  FOREIGN KEY (idVendor)  REFERENCES tblFirm(id) | Связь КУМ с фирмой-поставщиком материала. |
| Балансовая ведомость | | |
| fk\_balance\_materialCard | ALTER TABLE tblBalance  ADD CONSTRAINT fk\_balance\_materialCard  FOREIGN KEY (idMC)  REFERENCES tblMaterialCard (id) | Связь балансовой ведомости с КУМ, для которой ведется учет. |
| fk\_balance\_document | ALTER TABLE tblBalance  ADD CONSTRAINT fk\_balance\_document  FOREIGN KEY (idDocument)  REFERENCES tblDocument(id) | Связь балансовой ведомости с документом, на основании которого вносится очередная запись в ведомость. |
| Документ | | |
| fk\_document\_docType | ALTER TABLE tblDocument  ADD CONSTRAINT fk\_document\_docType  FOREIGN KEY (idType)  REFERENCES tblDocumentType (id) | Связь документа с типом. |
| fk\_document\_opType | ALTER TABLE tblDocument  ADD CONSTRAINT fk\_document\_opType  FOREIGN KEY (idOperationType) REFERENCES tblOperationType (id) | Связь документа с типом операции, которую он сопровождает. |

Таблица 2.1. (продолжение)

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| fk\_document\_forwarderFirm | ALTER TABLE tblDocument  ADD CONSTRAINT fk\_document\_forwarderFirm  FOREIGN KEY (idForwarder) REFERENCES tblFirm(id) | Связь документа с фирмой-отправителем. |
| fk\_document\_forwarderSC | ALTER TABLE tblDocument  ADD CONSTRAINT fk\_document\_forwarderSC  FOREIGN KEY (idForwarder) REFERENCES tblStockCustodian (id) | Связь документа с МОЛ, выступающим в роли отправителя материалов. |
| fk\_document\_receiverFirm | ALTER TABLE tblDocument  ADD CONSTRAINT fk\_document\_receiverFirm  FOREIGN KEY (idReceiver)  REFERENCES tblFirm(id) | Связь документа с фирмой-получателем. |
| fk\_document\_receiverSC | ALTER TABLE tblDocument  ADD CONSTRAINT fk\_document\_receiverSC  FOREIGN KEY (idReceiver)  REFERENCES tblStockCustodian (id) | Связь документа с МОЛ, выступающим в роли получателя материалов. |
| fk\_document\_actor | ALTER TABLE tblDocument  ADD CONSTRAINT fk\_document\_actor  FOREIGN KEY (idActor)  REFERENCES tblStockCustodian (id) | Связь документа с МОЛ, сдающим материалы на склад или получающим материалы со склада. |
| fk\_document\_allowed | ALTER TABLE tblDocument  ADD CONSTRAINT fk\_document\_allowed  FOREIGN KEY (idAllowed)  REFERENCES tblStockCustodian (id) | Связь документа с МОЛ, разрешившим выдачу материалов со склада. |
| Материалы в документе | | |
| fk\_mid\_document | ALTER TABLE  tblMaterialsInDocument  ADD CONSTRAINT fk\_mid\_document  FOREIGN KEY (id\_document) REFERENCES tblDocument(id) | Связь с документом |
| fk\_mid\_material | ALTER TABLE  tblMaterialsInDocument  ADD CONSTRAINT fk\_mid\_material  FOREIGN KEY (id\_material) REFERENCES tblMaterial(id) | Связь с материалом |

Таблица 2.1. (продолжение)

|  |  |  |
| --- | --- | --- |
| Место хранения документа | | |
| fk\_path\_document | ALTER TABLE tblDocPath  ADD CONSTRAINT fk\_path\_document  FOREIGN KEY (id\_doc)  REFERENCES tbl Document(id) | Связь документа с местом его хранения на диске |
| Подразделение | | |
| fk\_division\_account\_entity | ALTER TABLE tblDivision  ADD CONSTRAINT fk\_division\_account\_entity  FOREIGN KEY (id\_account\_entity)  REFERENCES tblAccountEntity (id) | Связь подразделения с учетной единицей выпускаемой им продукции |
| fk\_division\_activityCategory | ALTER TABLE tblDivision  ADD CONSTRAINT fk\_division\_activityCategory  FOREIGN KEY (id\_activity\_category)  REFERENCES tblActivityCategory (id) | Связь подразделения с видом его деятельности. |
| Материально – ответственное лицо | | |
| fk\_sc\_job | ALTER TABLE tblStockCustodian  ADD CONSTRAINT fk\_sc\_job  FOREIGN KEY (id\_job)  REFERENCES tblJob(id\_job) | Связь сотрудника и занимаемой им должности. |
| fk\_sc\_division | ALTER TABLE tblStockCustodian  ADD CONSTRAINT fk\_sc\_division  FOREIGN KEY (id\_division)  REFERENCES tblDivision (id\_division) | Связь сотрудника и подразделения, в котором он работает |

* 1. Триггеры

Для обеспечения целостности базы данных необходимо выполнение следующих правил:

* При добавлении нового подразделения необходимо проверить, является ли оно складом, и, если является, добавить соответствующую запись в таблицу «Склад».
* При редактировании информации о подразделении также необходимо отслеживать изменение информации о складах и вносить соответствующие исправления в таблицу складов. Аналогичную проверку надо производить и при удалении подразделения.
* При добавлении очередной записи в балансовую ведомость необходимо производить перерасчет остатка материала, к которому относится вносимая запись.

Выполнение данных правил обеспечивается с помощью триггеров, представленных в приложении Г в таблице Г.1.

* 1. Хранимые процедуры

База данных содержит 2 хранимые процедуры, позволяющие включать или отключать соответствующие ограничения внешнего ключа для таблицы «Документ» в зависимости от кодов отправителя и получателя. Если код отправителя, передаваемый процедуре в качестве параметра, равен 1, то есть отправителем является сторонняя организация, то для данной записи необходимо обеспечить выполнение ограничения fk\_document\_forwarderFirm, связывающего таблицу «Документ» с таблицей «Фирма» по полю «idForwarder» (cм. таблицу 2.1). При этом необходимо отключить проверку ограничения внешнего ключа fk\_document\_forwarderSC для данного документа, чтобы не нарушить целостность базы данных. Если же код отправителя равен 2, то есть материал поступает от подразделения своей организации, запись в таблице «Документ» должна быть связана с материально-ответственным лицом, представляющим данное подразделение. В этом случае необходимо отключить проверку ограничения fk\_document\_forwarderFirm и включить проверку fk\_document\_forwarderSC. Ниже приведена реализация описанной процедуры.

CREATE PROCEDURE set\_fk\_document\_forwarder(@f\_code INT)

-- @f\_code - код отправителя

AS

IF @f\_code = 1

-- отправитель - сторонняя организация

BEGIN

-- включить проверку внешнего ключа с таблицей "Фирма"

ALTER TABLE tblDocument CHECK CONSTRAINT fk\_document\_forwarderFirm

-- отключить проверку внешнего ключа с таблицей "МОЛ"

ALTER TABLE tblDocument NOCHECK CONSTRAINT fk\_document\_forwarderSC

END

ELSE IF @f\_code = 2

-- отправитель - подразделение

BEGIN

-- включить проверку внешнего ключа с таблицей "МОЛ"

ALTER TABLE tblDocument CHECK CONSTRAINT fk\_document\_forwarderSC

-- отключить проверку внешнего ключа с таблицей "Фирма"

ALTER TABLE tblDocument NOCHECK CONSTRAINT fk\_document\_forwarderFirm

END

Процедура, производящая переключение ограничений, связанных с получателем товара, работает аналогично.

* 1. Функции

Получить информацию из базы данных можно с помощью вызова функций, являющихся ее объектами. Они могут возвращать как табличные, так и скалярные значения. Реализуемая база данных содержит 4 табличных и 16 скалярных функций.

Наибольший интерес среди табличных представляет набор функций, с помощью которых вычисляется значение графы «от кого получено / кому отпущено» карточки учета материалов. Правила заполнения этой графы были описаны в п.1.1. при рассмотрении делового регламента. Они реализованы с помощью трех функций:

* get\_ForwarderOrReceiver(@id INT);
* get\_Forwarder(@id INT);
* get\_Receiver(@id INT).

Функции получают в качестве аргумента идентификатор, являющийся первичным ключом таблицы «Документ». Возвращаемое значение – таблица из двух полей, содержащая непосредственно значение графы «от кого получено / кому отпущено», а также значение, которое будет записано в графе «Учетная единица выпуска продукции» КУМ.

Функция get\_ForwarderOrReceiver(id) определяет тип полученного документа. Если это приходный ордер, то в качестве искомого значения необходимо вернуть либо название фирмы – отправителя, либо название структурного подразделения и сведения о материально ответственном лице, сдающем материал на склад. В противном случае документ является накладной, то есть сопровождает выдачу товара со склада, и в этом случае необходимо указать аналогичный набор данных уже для отправителя. В первом случае вызывается функция get\_Forwarder(id), во втором – get\_Receiver(id):

CREATE FUNCTION get\_ForwarderOrReceiver(@doc\_id INT)

RETURNS @result TABLE (forw\_rec TEXT, acc\_ent VARCHAR(20))

AS

BEGIN

IF ((SELECT doc.idType FROM tblDocument AS doc WHERE doc.id = @doc\_id) = 1)

BEGIN

INSERT INTO @result

SELECT \* FROM dbo.get\_Forwarder(@doc\_id)

END

ELSE

BEGIN

INSERT INTO @result

SELECT \* FROM dbo.get\_Receiver(@doc\_id)

END

RETURN

END

Функция get\_Forwarder(id) проверяет код отправителя для данного документа. Если он равен единице, она возвращает название фирмы, связанной по внешнему ключу с полем idForwarder записи, соответствующей этому документу. Если же код отправителя равен 2, функция вернет необходимый набор данных для сотрудника, ответственного за сдачу товара по данному документу, и учетную единицу продукции, выпускаемой подразделением, в котором он работает, если таковая имеется.

CREATE FUNCTION get\_Forwarder(doc\_id INT)

RETURNS @forwarder TABLE (forwarder TEXT, acc\_ent VARCHAR(20))

AS

BEGIN

IF ((SELECT doc.forwarderCode FROM tblDocument AS doc WHERE doc.id = @doc\_id) = 1)

BEGIN

INSERT INTO @forwarder

SELECT firm.abbr\_name, '' AS ae

FROM tblDocument AS doc

JOIN tblFirm\_list AS firm

ON doc.idForwarder = firm.id

WHERE doc.id = @doc\_id

END

ELSE

BEGIN

INSERT INTO @forwarder

SELECT sc.division + '. ' + sc.job + ' ' +

sc.surname + ' ' + sc.name + ' ' + sc.patronym AS forwarder,

ae.abbr AS acc\_ent

FROM tblDocument AS doc

JOIN stock\_custodian\_view AS sc

ON doc.idForwarder = sc.id\_sc

JOIN tblDivision AS div

ON div.id = sc.id\_div

LEFT JOIN tblAccountEntity AS ae

ON ae.id = div.id\_account\_entity

WHERE doc.id = @doc\_id

END

RETURN

END

Функция get\_Receiver(id) выполняет аналогичные действия, проверяя код получателя и работая с полем idReceiver записи, представляющей документ.

При заполнении каждого из первичных документов, обрабатываемых в системе, необходимо указывать как количество материала по сопровождающему документу, так и фактически принятое или отпущенное. Чтобы избежать обращения к базе данных с двумя одинаковыми запросами, их можно выполнить в одной функции и вернуть сразу оба значения. За это отвечает табличная функция get\_QuantityFromDocument(@doc\_id INT, @mat\_id INT). Она получает в качестве аргументов указатели на документ и на материал в виде их первичных ключей и возвращает таблицу с требуемыми значениями:

CREATE FUNCTION getReceiptFromDocument(@doc\_id int, @mat\_id int)

RETURNS TABLE

AS

RETURN

( SELECT quantity\_doc AS q\_doc, quantity\_fact AS q\_fact

FROM tblMaterialsInDocument AS t

WHERE t.id\_document = @doc\_id AND t.id\_material = @mat\_id

)

Некоторые из используемых скалярных функций перечислены в таблице 2.2.

Таблица 2.2. Скалярные функции

|  |  |
| --- | --- |
| **Название** | **Действие** |
| calc\_remainder\_of\_material (@m\_id INT)   * @m\_id – id материала | Вычислить остаток материала; если записей о материале в балансовой ведомости нет, вернуть NULL. |
| getRemainder(@mc\_id INT)   * @mc\_id - id КУМ | Получить остаток материала; если сведений о материале не поступало, вернуть 0. |
| getLastDocumentID() | Получить id последнего добавленного документа. |
| getNextDocumentNo(@type INT)   * @type – тип документа | Получить порядковый номер следующего документа заданного типа. |
| getSCCountFromDivision(@div\_id INT)   * @div\_id – id подразделения | Получить количество сотрудников, работающих в данном подразделении. |
| getTotalQuantityFromDocument(@doc\_id INT)   * @doc\_id – id документа | Получить общее число позиций материалов в заданном документе. |
| getActivityCategoryByStockCustodian (@sc\_id INT)   * @sc\_id – id сотрудника | Получить вид деятельности подразделения, имея информацию о его сотруднике. |

Сценарии перечисленных скалярных функций представлены в приложении Д.

* 1. Представления

Представление – это тип таблицы, содержание которой выбирается из других таблиц с помощью выполнения запроса всякий раз, когда к нему происходит обращение. Представления целесообразно применять при частом использовании одного и того же запроса.

В решаемой задаче во всех документах требуется указывать ФИО материально ответственных лиц, их должности, а иногда и подразделения, в которых они работают. Чтобы получить такой набор данных, необходимо выполнить следующий запрос:

SELECT surname, name, patronym, job, division

FROM tblStockCustodian AS sc

JOIN tblJob AS j ON j.id = sc.id\_job

JOIN tblDivision AS d ON d.id = sc.id\_division

Чтобы не выполнять этот запрос каждый раз, когда требуется получить данные о том или ином сотруднике, можно создать на его основе представление. Для упрощения связывания с другими таблицами в представление можно также добавить ключевые атрибуты всех трех входящих в него сущностей. Сценарий создания имеет следующий вид:

CREATE VIEW stock\_custodian\_view

AS

SELECT sc.id AS id\_sc, surname, name, patronym, d.id AS id\_div, division, j.id AS id\_job, job

FROM tblStockCustodian AS sc

JOIN tblJob AS j ON j.id = sc.id\_job

JOIN tblDivision AS d ON d.id = sc.id\_division

Выборку данных для отображения записей оборотной ведомости также удобно оформить в виде представления:

CREATE VIEW balance\_view\_m17 AS

SELECT doc.id AS doc\_id, creationDate AS rec\_date,

document\_type + ' № ' + CONVERT(varchar(3), DocNo) AS [document],

receipt, expenditure, remainder, idMC

FROM tblBalanceList AS b

JOIN tblDocument AS doc ON b.idDocument = doc.id

JOIN tblDocumentType AS dt ON doc.idType = dt.id

Для удобства отображения информации о материале, указанном в первичном документе, в заданной форме созданы следующие представления.

Для приходного ордера (форма М-4):

CREATE VIEW material\_view\_m4

AS

SELECT mat.id AS mat\_id, name, sort, size, stamp,

stock\_number, price, NDS, passportNo,

abbr AS mu\_abbr, f.id AS firmID,

mc.id AS mc\_id, MCno, idWarehouse

FROM tblMaterial AS mat

JOIN tblMaterialCard AS mc ON mat.id = mc.idMaterial

JOIN tblMeasureUnit AS mu ON mat.idMU = mu.id

JOIN tblFirm AS f ON mc.idVendor = f.id

Для накладных М-11 и М15:

CREATE VIEW material\_view\_m11\_m15 AS

SELECT mat.id, name, name, sort, size, stamp,

stock\_number, price, NDS, passportNo,

abbr AS mu\_abbr, codeOKEI,

MCno, idWarehouse, dbo.getRemainder(mc.id) AS remainder

FROM tblMaterial AS mat

JOIN tblMaterialCard AS mc ON mat.id = mc.idMaterial

JOIN tblMeasureUnit AS mu ON mat.idMU = mu.id

Для карточки учета материалов М-17:

CREATE VIEW material\_view\_m17 AS

SELECT mat.id AS mat\_id, name, sort, size, stamp,

stock\_number, price, NDS, passportNo,

stock\_level, valid\_period,

codeOKEI, abbr AS mu\_abbr,

mc.id AS id\_mc, MCno, creationDate, stack, shelf,

idWarehouse as wh\_id, f.abbr\_name AS firm

FROM tblMaterial AS mat

JOIN tblMeasureUnit AS mu ON mat.idMU = mu.id

JOIN tblMaterialCard AS mc ON mc.idMaterial = mat.id

JOIN tblFirm AS f ON mc.idVendor = f.id

1. Технологический раздел
   1. Выбор языка программирования и среды разработки

Пользовательское приложение написано на языке C# платформы .NET 4.0. в среде Microsoft Visual Studio 2010. Платформа .NET Framework обладает следующими положительными чертами:

* поддержка множества языков программирования;
* обширная библиотека базовых классов;
* поддержка нескольких технологий доступа к внешним источникам данных;
* возможность взаимодействия со средствами Miscrosoft Office (Excel, Word).

Язык С# был выбран в силу следующих его достоинств:

* автоматическая очистка памяти (сборка мусора);
* упрощенная модель делегат/событие;
* поддержка строго типизированных запросов (LINQ) для взаимодействия с различными источниками данных;
* поддержка ООП технологий на уровне языка.

Microsoft Visual Studio 2010 является современной средой разработки, поддерживающей большой набор интерфейсов программирования приложений: Windows Forms, WPF, ASP.NET и др. Для разработки пользовательского приложения был выбран интерфейс Windows Forms, отвечающий за работу с функциями GUI.

* 1. Технологии доступа к внешним источникам данных

Существует несколько возможностей обратиться к базе данных из программы, написанной на языке платформы .NET Framework:

* интеграция SQL Server и среды CLR платформы .NET Framework;
* LINQ (Language Integrated Query – язык интегрированных запросов);
* технология доступа к данным ADO.NET;
* визуальные средства связывания данных.

В данной работе для получения доступа к внешнему источнику данных была использована технология LINQ. В отличие от традиционных запросов к данным с использованием ADO.NET, которые выражаются в виде строк на SQL без проверки типов при компиляции и поддержки IntelliSence, LINQ позволяет создавать запросы к строго типизированным коллекциям объектов с помощью операторов языка C#.

Все операции запроса LINQ состоят из трех действий: получения источника данных, создания и выполнения запроса. Запросы LINQ можно подавать к различным перечислимым источникам данных, которые реализуют интерфейс IEnumerable или его универсальный вариант IEnumerable<T>. Запрос хранится в переменной запроса и инициализируется выражением запроса. Фактическое выполнение запроса откладывается до выполнения итерации переменной запроса в операторе foreach. Такую концепцию называют отложенным выполнением запроса.

Связь LINQ с высокоуровневыми языками программирования и различными источниками данных приведена на рис. 3.1.

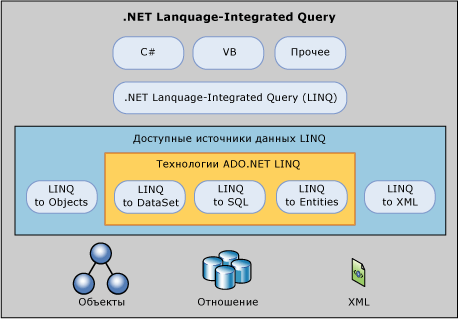


Рис. 3.1. Связь LINQ с источниками данных и языками программирования

Из рисунка видно, что источники данных LINQ делятся на три категории:

* LINQ to Objects;
* LINQ to XML;
* ADO.NET LINQ.

Последняя категория в свою очередь предоставляет три отдельные технологии: LINQ to DataSet, LINQ to Entities, и LINQ to SQL, которая и используется в данной работе.

Технология LINQ to SQL позволяет непосредственно запрашивать схемы баз данных SQL Server. При ее использовании в приложении автоматически генерируются классы платформы .NET Framework, которые напрямую указывают на таблицы базы данных, представления, хранимые процедуры и определяемые пользователем функции.

* + 1. Создание объектной модели LINQ to SQL

В Visual Studio 2010 встроен дизайнер LINQ to SQL, предоставляющий простой способ визуализировать базу данных как объектную модель LINQ to SQL.

При сохранении этой модели Visual Studio создает и сохраняет .NET классы, представляющие сущности и связи моделируемой базы данных. Для каждого добавленного в проект файла дизайнера LINQ создается собственный класс DataContext, являющийся основным модулем обмена данных, используя который можно запрашивать данные из БД, а также добавлять, удалять или модифицировать записи. Свойства данного класса представляют каждую таблицу моделируемой базы данных, а методы - хранимые процедуры или определенные пользователем функции.

В разрабатываемом приложении класс обмена данных имеет имя DataClassesDataContext. Его свойства отражают таблицы и представления базы данных Warehouse, добавленные в объектную модель (рис 3.2 справа), для каждой таблицы автоматически сгенерирован представляющий ее класс (рис. 3.2 слева).

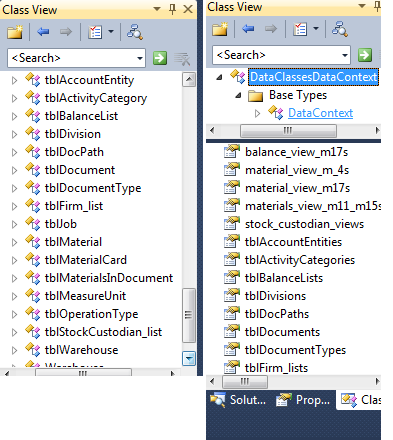


Рис. 3.2. Объектная модель LINQ to SQL

* + 1. Примеры обращения с запросами к базе данных с использованием технологии LINQ to SQL

1. Получить список названий всех организаций, с которыми сотрудничает фирма «Умелые ручки»:

DataClassesDataContext db = new DataClassesDataContext();

List<String> abbrList = new List<String>();

// переменная запроса

var abbrs = (from f in db.tblFirms

orderby f.abbr\_name

select f.abbr\_name);

// выполнение запроса

foreach (String item in abbrs)

abbrList.Add(item);

// первая запись относится к организации «Умелые ручки», поэтому ее надо удалить

abbrList.RemoveAt(0);

1. Получить список всех сотрудников, кроме тех, которые работают в подразделения division:

DataClassesDataContext db = new DataClassesDataContext();

var fio\_list =

(from sc in db.tblStockCustodians

join div in db.tblDivisions on sc.id\_division equals div.id

where !div.division.Contains(division)

orderby sc.surname, sc.name, sc.patronym

// анонимный тип

select new

{

id = sc.id, // id cотрудника

s = sc.surname, // фамилия

n = sc.name, // имя

p = sc.patronym // отчество

});

foreach (var item in fio\_list)

{ // формирование и добавление ФИО в список

fio.Add(StringRedactor.getFIO(item.s, item.n, item.p));

}

1. Получить все материалы, указанные в документе doc:

DataClassesDataContext db = new DataClassesDataContext();

IEnumerable<tblMaterial> materials = (

from m in db.tblMaterials

join md in db.tblMaterialsInDocuments

on m.id equals md.id\_material

where md.id\_document == doc.id

select m);

1. Получить запись о сотруднике, указанном в документе doc как лицо, разрешившее выдачу товара:

DataClassesDataContext db = new DataClassesDataContext();

stock\_custodian\_view allowed =

db.stock\_custodian\_views.Single(sc => sc.id\_sc == doc.idAllowed);

Метод Single() говорит о том, что в результате выполнения запроса должна быть получена только одна запись представления stock\_custodian\_view.

* + 1. Изменение базы данных

Для добавления данных в базу или удаления их из нее с использованием технологии LINQ to SQL можно добавить или удалить данные из соответствующей коллекции класса DataClassesDataContext, а затем вызвать его метод SubmitChanges(). LINQ to SQL сохраняет историю этих изменений и автоматически выполняет соответствующие выражения INSERT или DELETE на языке SQL при вызове метода SubmitChanges().

Например, для добавления информации об организации необходимо создать новый экземпляр класса tblFirm, записать в его свойства значения полей таблицы, а затем добавить его к коллекции tblFirms класса DataClassesDataContext. После вызова SubmitChanges() в таблице tblFirms базы данных Warehouse появится новая строка.

DataClassesDataContext db = new DataClassesDataContext();

// создать новый экземпляр класса tblFirm

firm = new tblFirm();

// заполнить свойства данными

firm.full\_name = fullName;

firm.abbr\_name = abbrName;

firm.addr = address;

firm.code = code;

firm.mail = mail;

firm.phone = phone;

// добавить экземпляр в коллекцию

db.tblFirm\_lists.InsertOnSubmit(firm);

// обновить базу данных

db.SubmitChanges();

Удаление записи из таблицы происходит аналогичным образом:

// удалить экземпляр firm из коллекции

db.tblFirms.DeleteOnSubmit(firm);

// обновить базу данных

db.SubmitChanges();

Вышеперечисленное справедливо и для операции UPDATE:

public void update(String code, String addr, String phone, String mail)

{

firm.code = code;

firm.addr = addr;

firm.phone = phone;

firm.mail = mail;

db.SubmitChanges();

}

При вызове метода SubmitChanges() выполняемые им изменения базы данных всегда оборачиваются в транзакцию, что гарантирует нахождение базы данных в согласованном состоянии при выполнении над ней каких-либо изменений: будут сохранены либо все обновления, примененные к коллекциям DataClassesDataContext, либо ни одного. Объект db автоматически запускает транзакцию, гарантирующую безопасное изменение состояния базы данных.

* + 1. Вызов хранимых процедур и определенных пользователем функций

Хранимые процедуры и пользовательские функции, добавленные в объектную модель LINQ to SQL, представляются методами класса DataClassesDataContex; для их выполнения нужно просто вызвать соответствующий метод:

// вызов хранимых процедур

db.set\_fk\_document\_forwarder(vendor\_code);

db.set\_fk\_document\_receiver(receiver\_code);

// вызов скалярной функции

document.DocNo = (int)db.getNextDocumentNo(document.idType);

act\_cat = db.getActivityCategoryByStockCustodian(receiver.id\_sc);

// вызов табличной функции

var forw\_rec = (from item in db.get\_ForwarderOrReceiver(rec.doc\_id)

select new

{

name = item.forw\_rec,

acc\_ent = item.acc\_ent

}).Single();

* 1. Структура приложения

Все классы приложения можно разбить на три группы: классы, сгенерированные LINQ и отвечающие за доступ к базе данных, классы, представляющие графический интерфейс приложения, и классы, реализующие логику работы приложения. Последняя группа представляет собой набор «оберток» вокруг каждого класса, сгенерированного LINQ to SQL для представления таблицы базы данных. Классы этой группы хранят соответствующие экземпляры автоматически сгенерированных классов, относящихся к первой группе, и реализуют обращение с запросами к базе данных с помощью этих экземпляров. Они наследуются от класса Table, который хранит у себя экземпляр DataClassesDataContext, используемый дочерними классами для связи с базой данных.

Классы интерфейса отвечают за отображение на экране форм приложения и обработку действий пользователя. К ним относятся классы, представляющие формы для создания первичных учетных документов. Они хранят у себя экземпляры классов третьей группы, которые необходимы им для создания своего документа, и вызывают их методы для обращения к базе данных.

Обособленно выделяется класс ExcelAdapter, предоставляющий интерфейс открытия и сохранения документов средствами Microsoft Office Excel.

На рис. 3.3 представлена диаграмма классов приложения. Рис. 3.4 иллюстрирует группировку и назначение используемых классов.

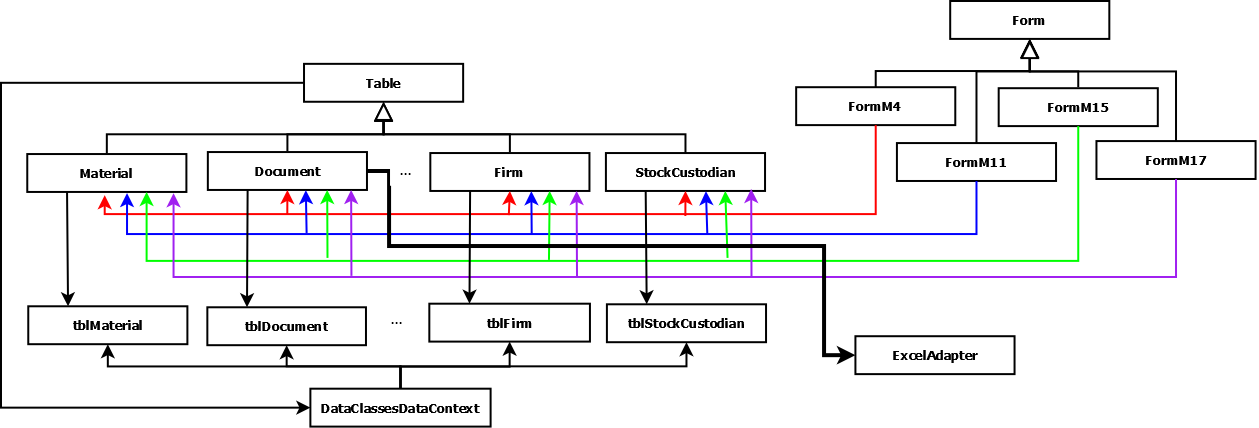


Рис. 3.3. Диаграмма классов приложения

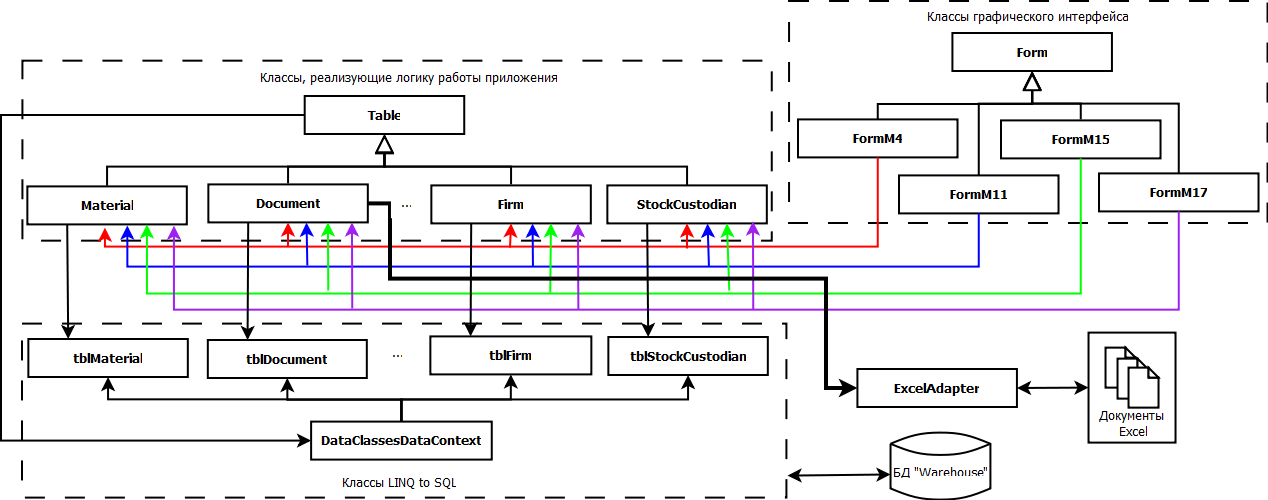


Рис. 3.4. Группировка и назначение классов приложения

* 1. Интерфейс приложения

Главное окно приложения представлено на рис. 3.5. Для начала работы необходимо выбрать из выпадающего списка вверху окна название склада и ФИО сотрудника этого склада, от имени которого будут идти прием и выдача материалов.

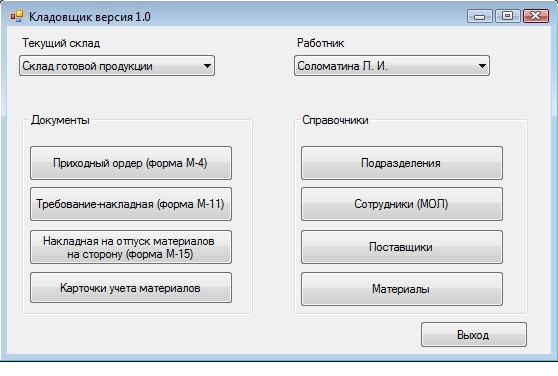


Рис. 3.5. Главное окно приложения

На рис. 3.6, 3.7 и 3.8 представлены формы для формирования приходных ордеров, требований-накладных и накладных на отпуск материалов на сторону соответственно.

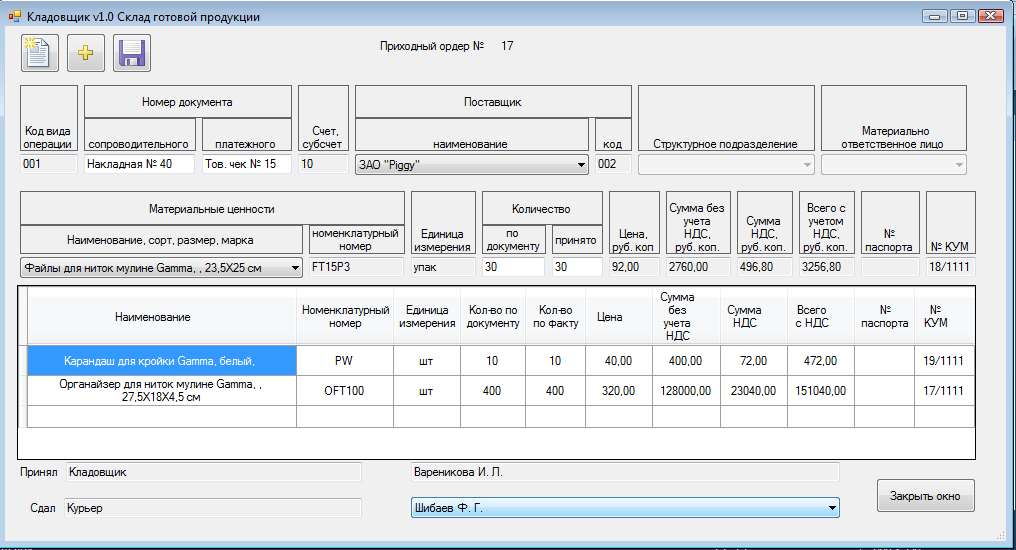


Рис. 3.6. Окно формирования приходного ордера

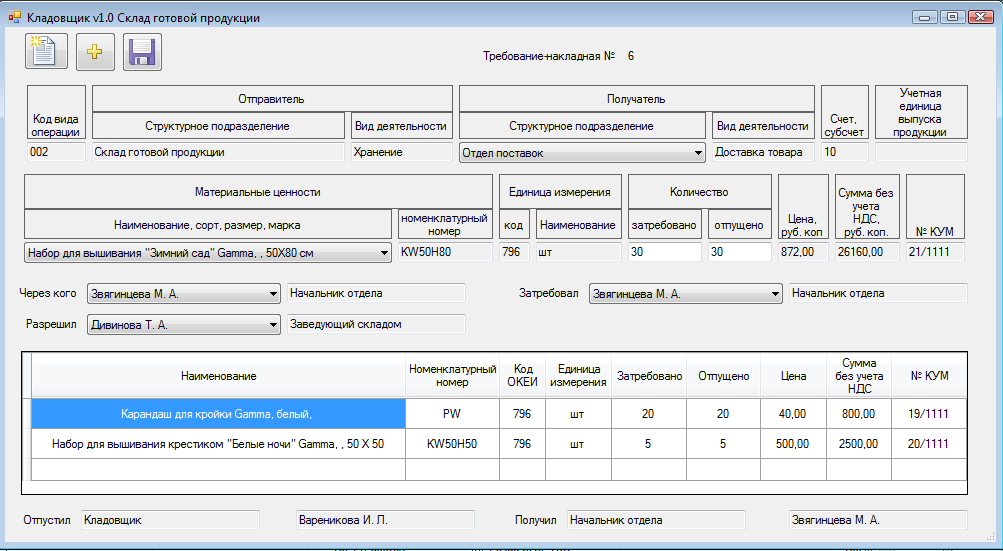


Рис. 3.7. Окно формирования требования-накладной

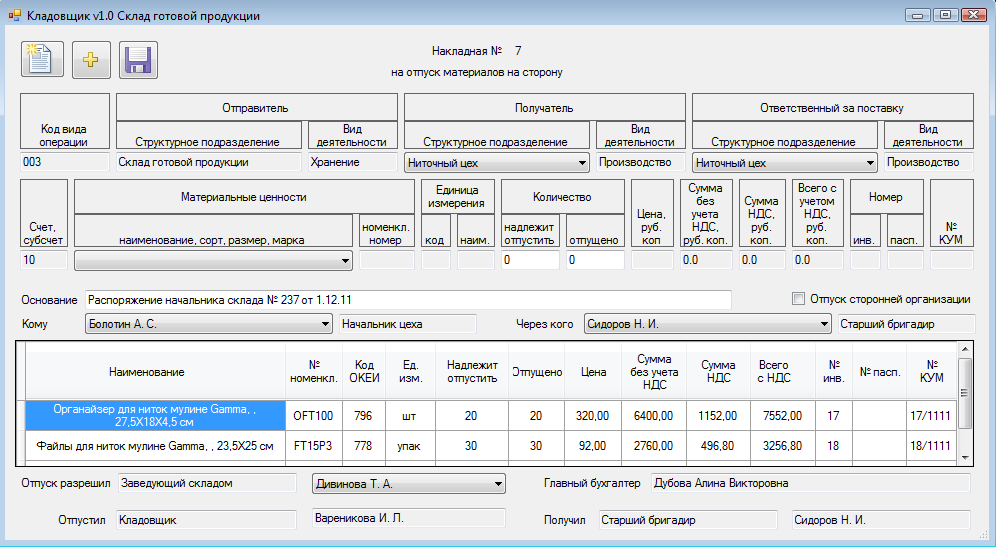


Рис. 3.8. Окно формирования накладной на отпуск материалов на сторону

Верхняя панель в каждом из представленных окон позволяет очищать форму, добавлять очередное наименование в документ и сохранять его в базе данных и на диске.

На рис. 3.9 представлено окно выбора карточек учета материалов. В списке справа перечислены карточки, удовлетворяющие заданным условиям поиска, который может быть произведен по наименованию фирмы-поставщика материала, по дате или периоду создания, либо по обоим критериям сразу. Имеется возможность также сразу выбрать карточку по конкретному материалу из списка.

Для просмотра карточки нужно дважды кликнуть по ней мышкой.

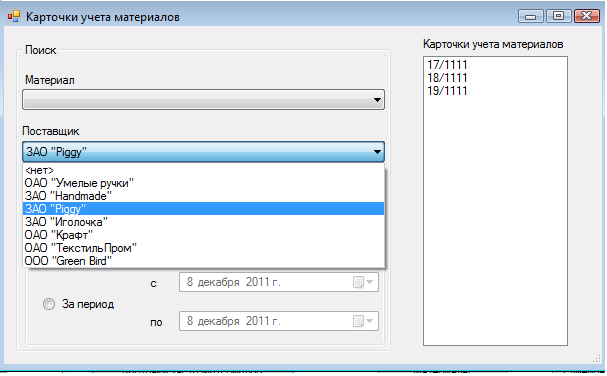


Рис. 3.9. Окно выбора карточек учета материалов

После двойного клика по карточке из списка откроется окно просмотра, представленной на рис. 3.10. При просмотре можно также выводить записи за конкретную дату или за заданный период времени.

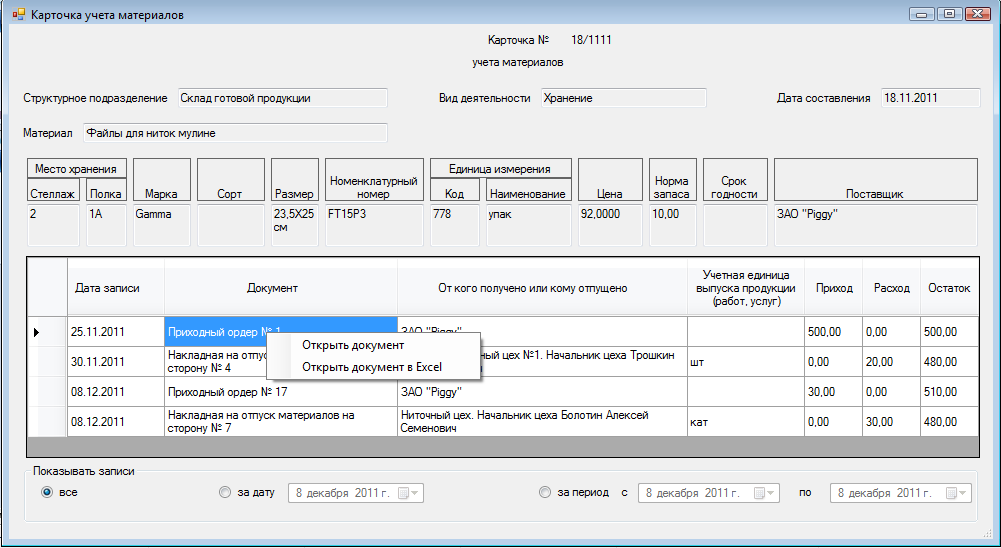


Рис. 3.10. Окно просмотра карточки учета материалов

Чтобы просмотреть документ, на основании которого сделана запись в КУМ, необходимо кликнуть по нему правой клавишей мышки и в появившемся меню выбрать вариант просмотра. При выборе пункта «Открыть в Excel» документ будет открыт в программе Microsoft Office Excel в режиме «только для чтения».

Еще одну часть интерфейса составляют редакторы справочников. В качестве примера на рис. 3.11, 3.12 и 3.13 приведены окна добавления, просмотра и редактирования справочника подразделений соответственно.

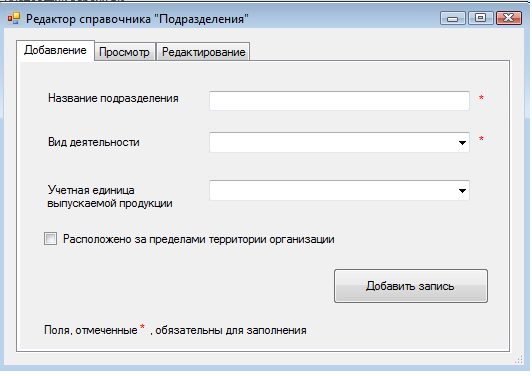


Рис. 3.11. Добавление нового подразделения

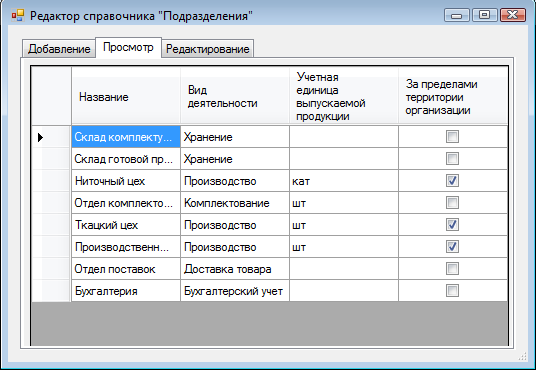


Рис. 3.12. Окно просмотра справочника «Подразделения»

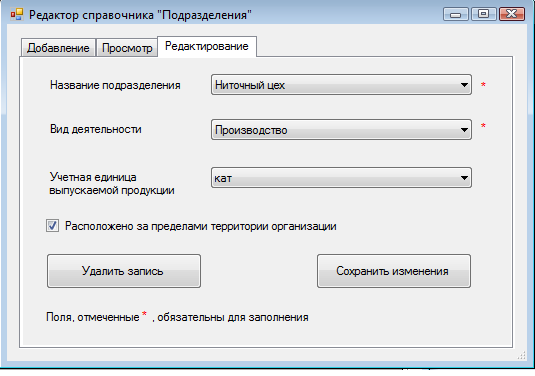


Рис. 3.13. Окно редактирования справочника «Подразделения»

* 1. Тестирование приложения

При тестировании работоспособности приложения было выявлено, что оно отвечает предъявленным требованиям, то есть:

* формирует первичные учетные документы и сохраняет их в формате \*.xls;
* позволяет открывать созданные документы для просмотра как в окне приложения, так и с помощью Microsoft Office Excel;
* позволяет просматривать, редактировать, добавлять и удалять записи справочных таблиц;
* организует поиск и просмотр карточек учета материалов по конкретному поставщику, за определенную дату или период создания; позволяет фильтровать записи в карточке по дате или за период.

Все вышеперечисленные функции работают корректно.

При тестировании также проводилась проверка возможных ошибок пользователя:

* сохранение документа с незаполненными обязательными полями;
* сохранение пустого документа (то есть документа, в котором не указан ни один материал);
* добавление в документ материалов при некорректном указании или без указания их количества;
* выдача большего количества материала, чем осталось на складе;
* удаление записей, которые используются в других таблицах как внешние ключи, из справочников.

Все эти ошибки отслеживаются, и пользователь получает соответствующее уведомление о невозможности выполнения операции в каждом из вышеперечисленных случаев.

# Заключение

В результате данного проекта была разработана система автоматизированного складского учета. Реализована база данных из 16 таблиц, хранящая необходимую информацию о материалах и документах на складе, и пользовательское приложение для работы с данной системой. Взаимодействие приложения и сервера базы данных осуществлено по технологии LINQ to SQL.

Одним из главных недостатков разработанного программного комплекса является отсутствие системы безопасности, поэтому дальнейшие разработки будут направлены на защиту системы. Планируется добавить авторизацию пользователей, хранение информации на сервере базы данных и документов на диске в зашифрованном виде. Также в перспективе стоит добавление обработки новых учетных документов.

Разработанную систему можно использовать как самостоятельный программный продукт, либо как модуль более крупной системы управления, включающей помимо складских операций автоматизацию бухгалтерии, логистики и т.д.

Список использованной литературы

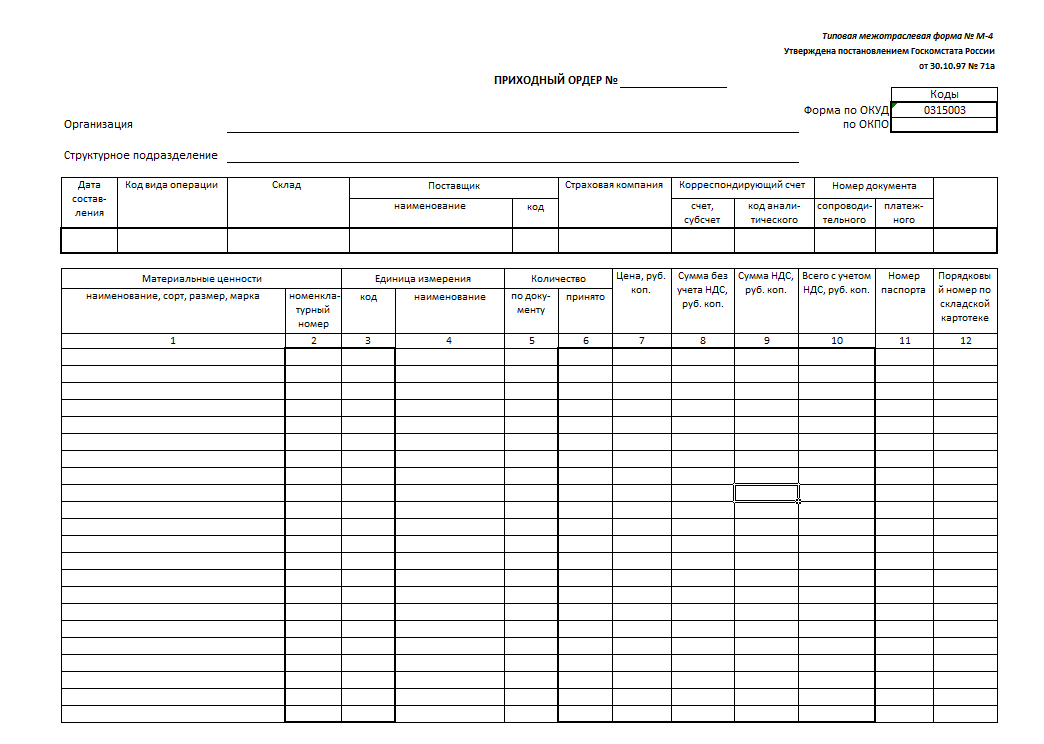
1. Troelsen Andrew Pro C# 2010 and the .NET 4 Platform, Fifth Edition.: (англ.) . – Apress, 2010. – 1712 с.: ил.
2. Руководство по программированию (LINQ to SQL) [электронный ресурс]: MSDN Library. – URL

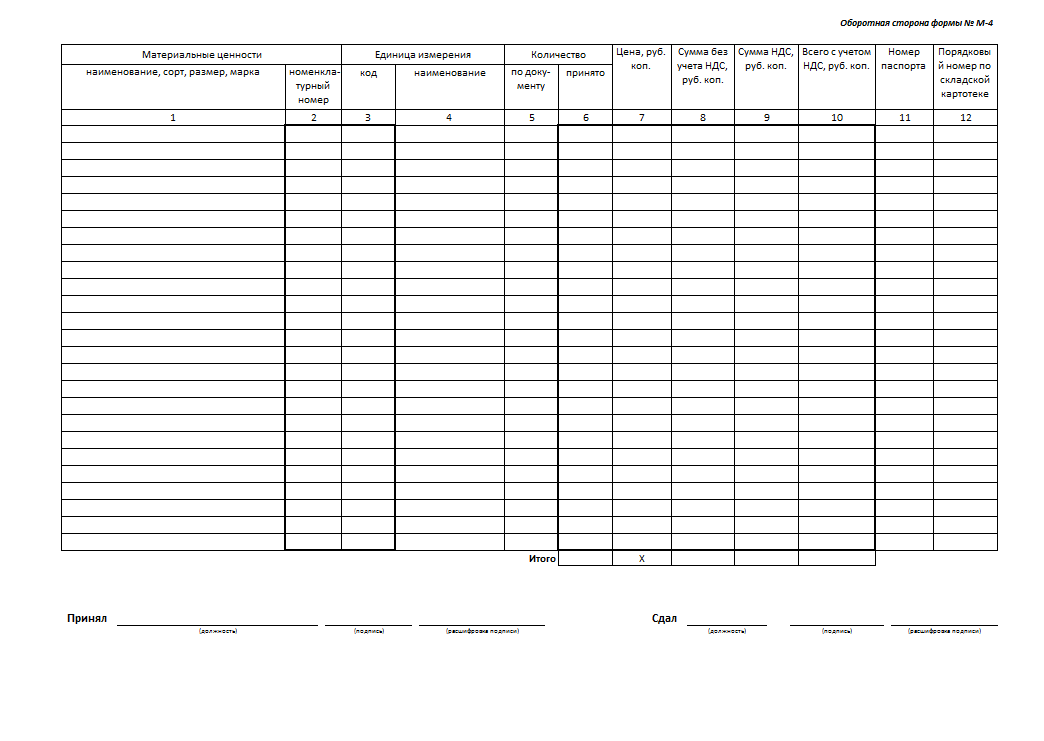
<http://msdn.microsoft.com/ru-ru/library/bb397676(v=VS.90).aspx> , 2011.

1. Учет материалов - типовые проводки, методические рекомендации, бланки форм М, МХ [электронный ресурс]: SysTecs: Справочник бухгалтера. – URL <http://sprbuh.systecs.ru/uchet/uchet_materialov/> , 2011.
2. Дейт К. Дж. Введение в системы баз данных, 8-е издание.: Пер. с англ. — М.: Издательский дом "Вильямс", 2005. — 1328 с.: ил.
3. Просуков Е. А. Методические указания к курсовому проекту по базам данных. –[электронный ресурс].
4. Просуков Е.А. Методические указания к лабораторным работам по базам данных. –[электронный ресурс].

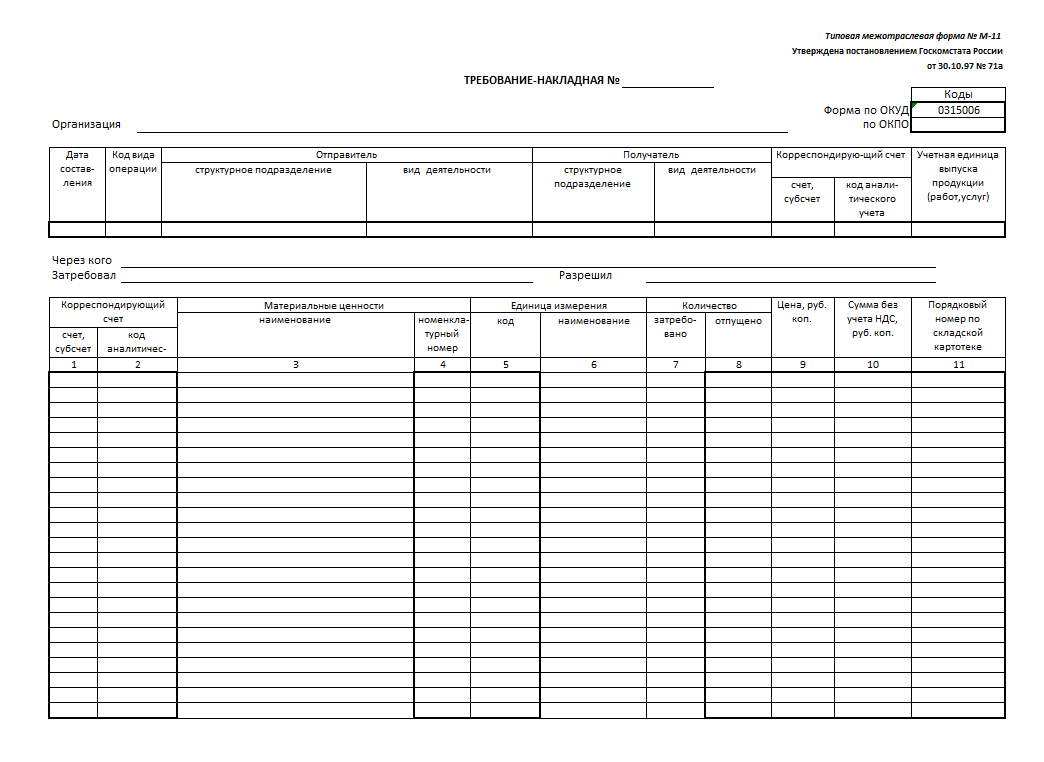
# Приложение А. Первичные формы документов

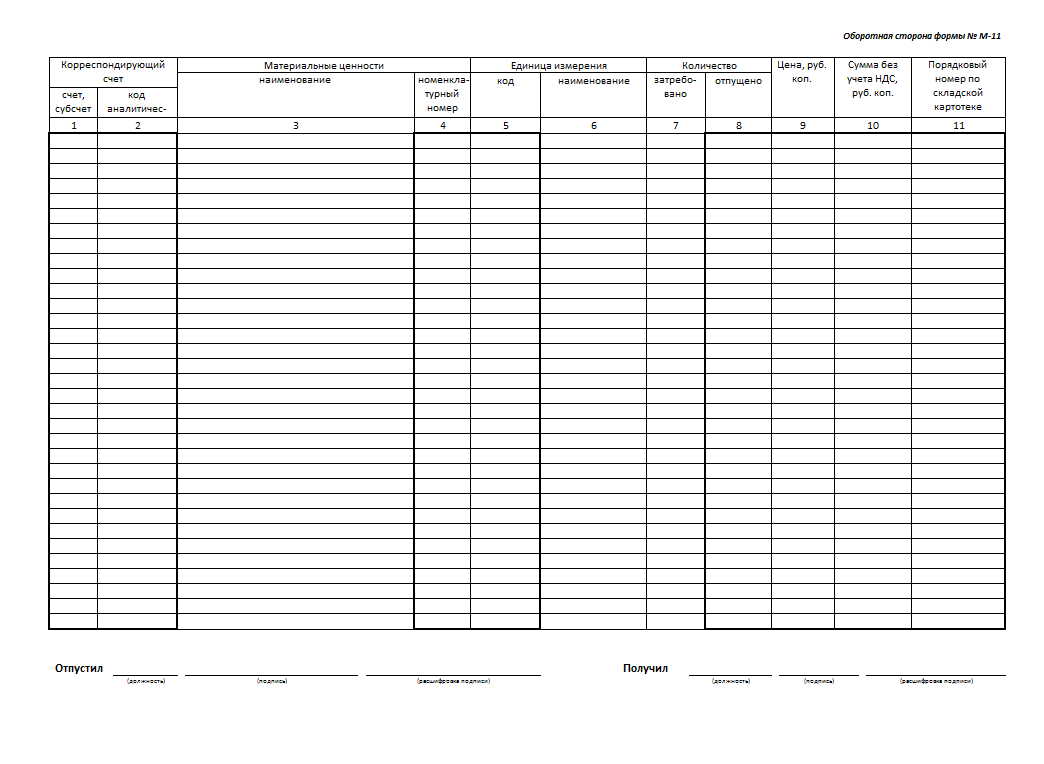
П. А.1. Приходный ордер (форма М-4)



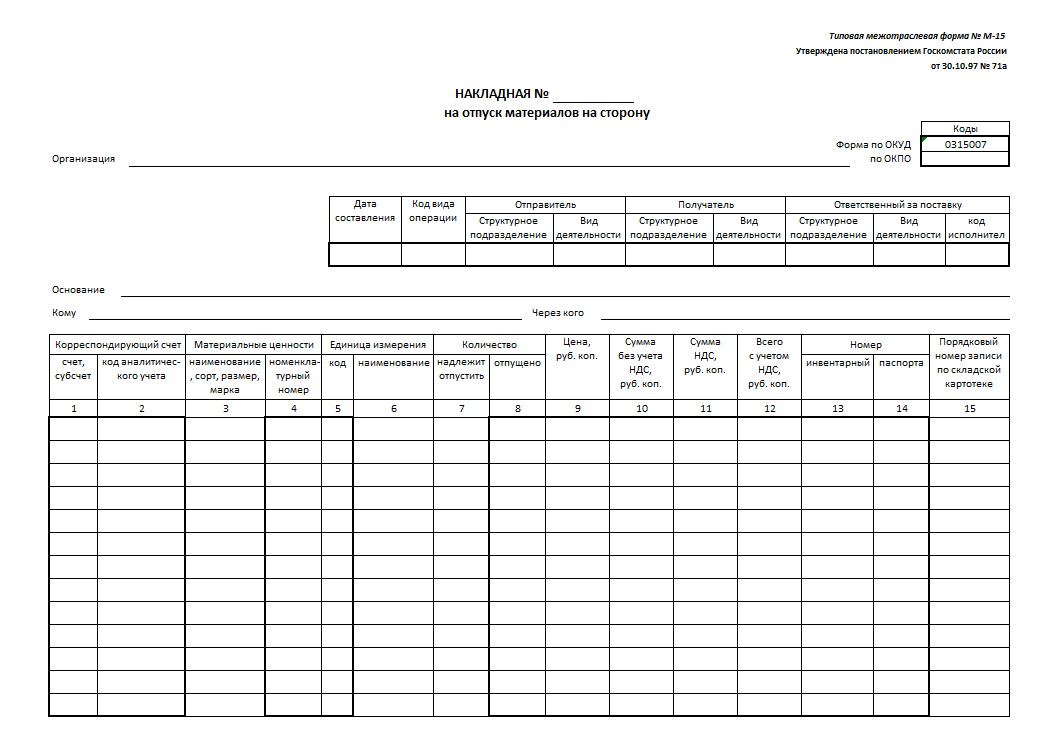


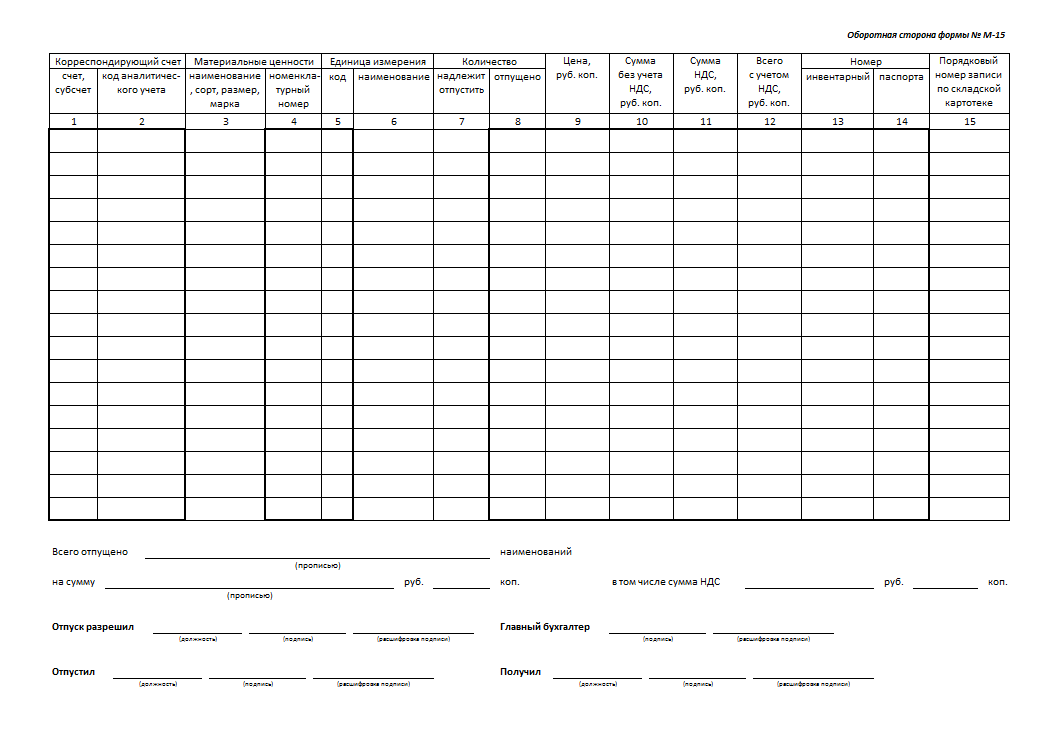
П. А.2. Требование-накладная (форма М-11)



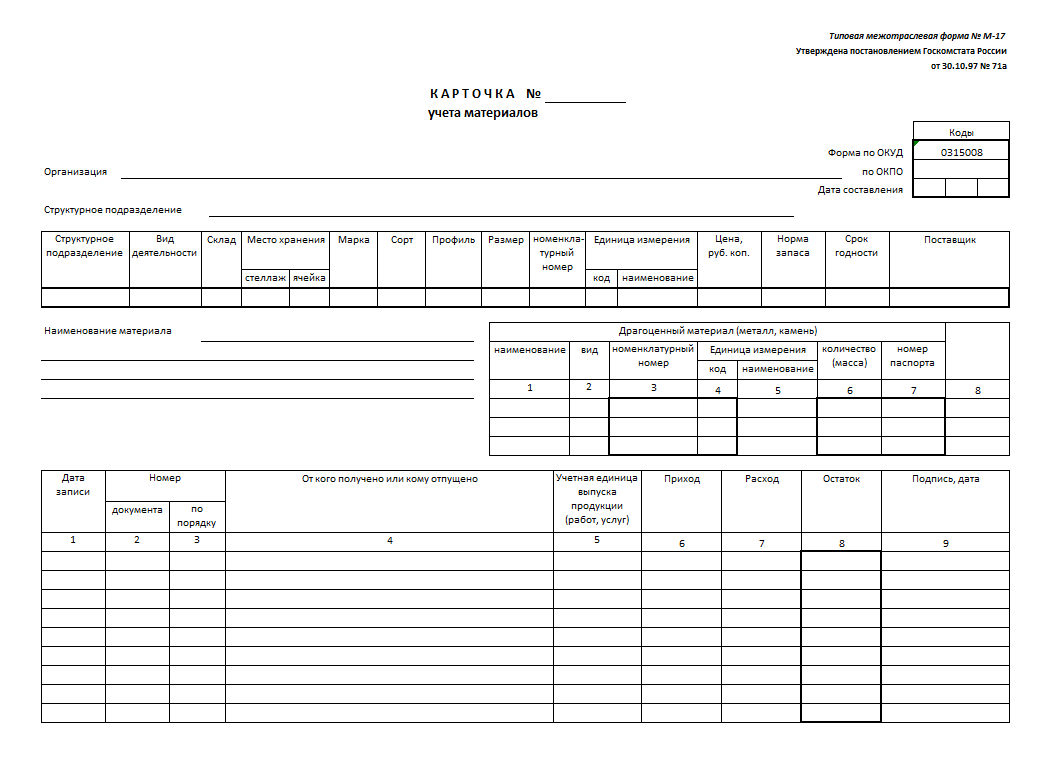


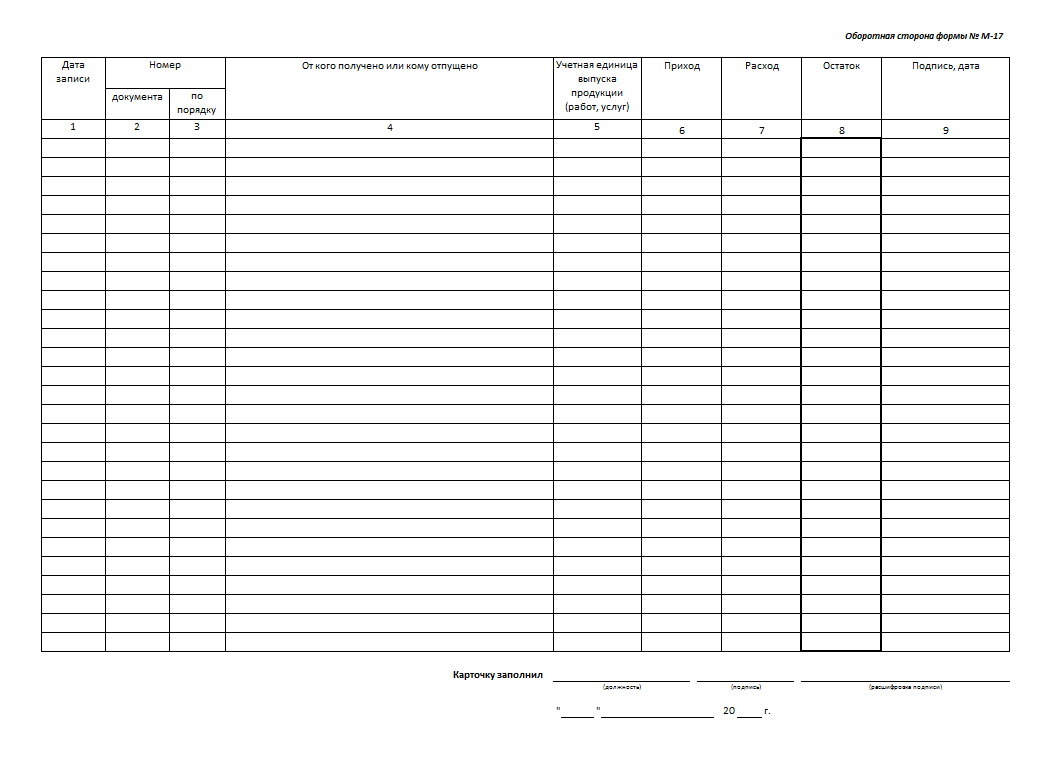
П. А.3. Накладная на отпуск материалов на сторону (форма М-15)





П. А.4. Карточка учета материалов (форма М-17)





# 

# Приложение Б. Варианты заполнения таблицы «Документ»

Таблица Б.1. Заполнение полей документа в зависимости от его типа

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Тип документа** | **Код отправи-теля** | **Код получа-теля** | **Отправитель**  **(idForwarder)** | **Получатель**  **(idReceiver)** | **Сдал / получил**  **(idActor)** |
| Приходный ордер | 1 | 2 | Сторонняя организация | МОЛ – работник склада, принимающий товары | МОЛ – сотрудник отдела поставок, сдающий товары |
| 2 | МОЛ – сотрудник подразделения, передающего товары на склад | МОЛ – сотрудник подразделения, передающего товары на склад |
| Требование-накладная | 2 | 2 | МОЛ – работник склада, выдающий товары | МОЛ – сотрудник подразделения-получателя, затребовавший товары | МОЛ – сотрудник подразделения получателя, принимающий товары со склада |
| Накладная на выдачу материалов на сторону | 2 | 1 | МОЛ – работник склада, выдающий товары | Сторонняя организация | МОЛ – сотрудник отдела поставок, принимающий товары |
| 2 | МОЛ – сотрудник подразделения-получателя, затребовавший товары | МОЛ – сотрудник подразделения-получателя или отдела поставок, принимающий товары |

# Приложение В. Сценарии создания таблиц базы данных

П. В.1. Таблицы складского учета

Материал

CREATE TABLE tblMaterial (

-- материал, товар

id INTEGER NOT NULL PRIMARY KEY, -- id, первичный ключ

code VARCHAR(10), -- код товара

name VARCHAR(255) NOT NULL, -- наименование

stock\_number VARCHAR(20) NOT NULL, -- номенклатурный номер

idMU INTEGER NOT NULL, -- единица измерения

price MONEY NOT NULL, -- цена

NDS NUMERIC(5,2) NOT NULL, -- ставка НДС

passportNo VARCHAR(20), -- № паспорта (для изделий с драг. металлами)

sort VARCHAR(50), -- сорт

size VARCHAR(50), -- размер

stamp VARCHAR(50), -- марка

stock\_level NUMERIC(12,2), -- норма запаса

valid\_period NUMERIC(10,0), -- срок годности

accountNo VARCHAR(50), -- № счета

notes VARCHAR(200) -- примечания

)

Карточка учета материалов

CREATE TABLE tblMaterialCard (

-- карточка учета материала (КУМ)

id INTEGER NOT NULL PRIMARY KEY, -- id, первичный ключ

MCno NUMERIC(10,0) NOT NULL UNIQUE, -- номер КУМ

creationDate DATE NOT NULL DEFAULT (getDate()), -- дата создания

idMaterial INTEGER NOT NULL, -- материал

idWarehouse INTEGER NOT NULL, -- склад

stack VARCHAR(10), -- стеллаж

shelf VARCHAR(10), -- полка

idVendor INTEGER NOT NULL -- поставщик

)

Оборотная ведомость

CREATE TABLE tblBalance (

-- оборотная (балансовая) ведомость

id INTEGER NOT NULL PRIMARY KEY, -- id, первичный ключ

idMC INTEGER NOT NULL, -- КУМ

creationDate DATE NOT NULL DEFAULT (getDate()), -- дата записи

idDocument INTEGER NOT NULL, -- документ, на основании которого

-- сделана запись

receipt NUMERIC(18,2) NOT NULL, -- приход

expenditure NUMERIC(18,2) NOT NULL, -- расход

remainder NUMERIC(18,2) NOT NULL -- остаток

)

Документ

CREATE TABLE tblDocument(

-- документ (приходный ордер (М-4) / требование-накладная (М-11) /

-- накладная на отпуск ТМЦ на сторону (М15) )

id INTEGER NOT NULL PRIMARY KEY, -- id документа, первичный ключ

idType INTEGER NOT NULL, -- тип документа

DocNo INTEGER NOT NULL, -- номер документа

creationDate DATE NOT NULL DEFAULT (getDate()), -- дата создания

idOperationType INTEGER NOT NULL, -- тип операции

-- - приход

-- - перемещение внутри организации

-- - отпуск на сторону

forwarderCode NUMERIC(2,0) NOT NULL, -- код отправителя (1 - фирма, 2 - МОЛ)

receiverCode NUMERIC(2,0) NOT NULL, -- код получателя (1 - фирма, 2 - МОЛ)

idForwarder INTEGER NOT NULL, -- отправитель

idReceiver INTEGER NOT NULL, -- получатель

idActor INTEGER NOT NULL, -- для М-4: МОЛ, сдающее материалы

-- для М-11, М-15: МОЛ, принимающее материалы

idAllowed INTEGER, -- М-11, М-15: МОЛ, разрешившее выдачу материалов

-- М-4: NULL

transmittalNo VARCHAR(20) UNIQUE, -- № сопроводительного документа

billing\_documentNo VARCHAR(20) UNIQUE -- № платежного документа

)

Материалы в документе

CREATE TABLE tblMaterialsInDocument (

-- связь "Документ - материалы"

id INTEGER NOT NULL PRIMARY KEY, -- id, первичный ключ

id\_document INTEGER NOT NULL, -- документ

id\_material INTEGER NOT NULL, -- материал

quantity\_doc DECIMAL(18, 2) NOT NULL, -- количество материала по сопроводительным

-- документам

quantity\_fact DECIMAL (18, 2) -- фактическое количество материала

)

Место хранения документа

CREATE TABLE tblDocPath (

-- связь документа с местом хранения на диске

id INT NOT NULL PRIMARY KEY, -- id, первичный ключ

id\_doc INT NOT NULL, -- документ

path VARCHAR(50) -- путь

)

П. В.2. Таблицы поставщиков и получателей

Место хранения документа

CREATE TABLE tblFirm

-- организация (фирма)

(

id INTEGER NOT NULL PRIMARY KEY, -- id, первичный ключ

code VARCHAR(10), -- код фирмы

abbr\_name VARCHAR(255), -- сокращенное наименование

full\_name VARCHAR(255) NOT NULL, -- полное наименование

addr VARCHAR(100) NOT NULL, -- адрес

phone VARCHAR(20), -- телефон

mail VARCHAR(30) -- e-mail

)

Подразделение

CREATE TABLE tblDivision(

-- структурное подразделение

id INTEGER NOT NULL PRIMARY KEY, -- id, первичный ключ

division VARCHAR(100) NOT NULL, -- название подразделения

id\_activity\_category INT NOT NULL, -- вид деятельности

id\_account\_entity INT, -- учетная единица выпускаемой продукции

outside INT NOT NULL -- расположение:

-- 1 - за пределами территории организации

-- 0 - на территории организации

)

Склад

CREATE TABLE tblWarehouse (

-- склад

id INTEGER NOT NULL PRIMARY KEY, -- id, первичный ключ

warehouse VARCHAR(100) NOT NULL -- название склада

)

Материально-ответственное лицо

CREATE TABLE tblStockCustodian (

-- Материально-ответственное лицо (МОЛ)

id INTEGER NOT NULL PRIMARY KEY, -- id, первичный ключ

surname VARCHAR(50) NOT NULL, -- фамилия

name VARCHAR(50) NOT NULL, -- имя

patronym VARCHAR(50), -- отчество

id\_job INTEGER NOT NULL, -- должность

id\_division INTEGER NOT NULL -- подразделение

)

П. В.3. Таблицы – справочники

Единица учета

CREATE TABLE tblAccountEntity (

-- справочник "Учетные единицы выпуска продукции"

id INTEGER NOT NULL PRIMARY KEY, -- id, первичный ключ

abbr VARCHAR (20), -- сокращенное название

account\_entity VARCHAR(30) NOT NULL -- название

)

Единица измерения

CREATE TABLE tblMeasureUnit (

-- справочник "Единицы измерения материалов"

id INTEGER NOT NULL PRIMARY KEY, -- id, первичный ключ

codeOKEI VARCHAR(3) NOT NULL UNIQUE, -- код ОКЕИ

abbr VARCHAR(20) NOT NULL, -- сокращенное название по ОКЕИ

measureUnit VARCHAR(30) NOT NULL -- название

)

Вид деятельности

CREATE TABLE tblActivityCategory (

-- справочник "Виды деятельности подразделений"

id INTEGER NOT NULL PRIMARY KEY, -- id, первичный ключ

code VARCHAR(10), -- код вида деятельности

activity\_category VARCHAR(50) NOT NULL -- вид деятельности

)

Тип документа

CREATE TABLE tblDocumentType (

-- справочник "Типы документов"

id INTEGER NOT NULL PRIMARY KEY, -- id, первичный ключ

form VARCHAR(10), -- форма документа (М-4, М-11, М-15)

document\_type VARCHAR(30) NOT NULL -- тип документа

)

Должность

CREATE TABLE tblJob (

-- справочник "Должности"

id INTEGER NOT NULL PRIMARY KEY, -- id, первичный ключ

code VARCHAR(10), -- код должности

job VARCHAR(50) NOT NULL -- должность

)

Тип операции

CREATE TABLE tblOperationType (

-- справочник "Типы операций"

id INTEGER NOT NULL PRIMARY KEY, -- id, первичный ключ

code VARCHAR(10), -- код типа операции

operation\_type VARCHAR(50) NOT NULL -- тип операции

)

# Приложение Г. Триггеры

Таблица Г.1. Триггеры

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Условие срабатывания** | **Сценарий создания** | **Принцип действия** |
| 1 | 2 | 3 | 4 |
| division\_inserted | Добавление подразделения | CREATE TRIGGER division\_inserted ON dbo.tblDivision  AFTER INSERT  AS  INSERT INTO tblWarehouse (warehouse)  (  SELECT division FROM inserted  JOIN tblActivityCategory AS ac  ON inserted.id\_activity\_category=ac.id  WHERE ac.activity\_category = 'Хранение'  ) | Триггер срабатывает после добавления данных в таблицу  «Подразделения». Он проверяет добавленные записи,  выбирает те из них, которые относятся к складам, и добавляет новый склад в соответствующую таблицу. |
| division\_updated | Редактирование подразделения | CREATE TRIGGER division\_updated ON tblDivision  AFTER UPDATE  AS  UPDATE tblWarehouse  SET warehouse =  (  SELECT inserted.division FROM inserted  JOIN tblActivityCategory AS ac  ON ac.id = inserted.id\_activity\_category  WHERE activity\_category = 'Хранение'  )  WHERE warehouse IN  (SELECT division FROM deleted) | Триггер срабатывает после изменения записи в таблице «Подразделения». Он проверяет, было ли изменено название какого-либо склада, и, если было, изменяет соответствующую запись в таблице складов. |

Таблица Г.1. (продолжение)

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| division\_deleted | Удаление подразделения | CREATE TRIGGER division\_deleted  ON dbo.tblDivision  AFTER DELETE/\*  AS  DELETE tblWarehouse  WHERE warehouse IN  (SELECT division FROM deleted) | Триггер срабатывает после удаления подразделения и удаляет запись о соответствующем складе, если это необходимо. |
| balance\_inserted | Добавление или обновление  записи в балансовой ведомости | CREATE TRIGGER balance\_inserted ON dbo.tblBalance  AFTER INSERT, UPDATE  AS  DECLARE @mat\_id INT  SET @mat\_id =  (SELECT IdMaterial  FROM tblMaterialCard AS mc  WHERE mc.id =  (SELECT idMC FROM inserted)  )  UPDATE tblBalance  SET remainder = dbo.calc\_remainder\_of\_material(@mat\_id)  WHERE id = (SELECT id FROM inserted) | Триггер срабатывает после каждого изменения таблицы балансовой ведомости. Он находит id материала, к которому относится добавленная или измененная запись, после чего пересчитывает остаток данного материала на складе. |

# Приложение Д. Функции

* Вычислить остаток материала; если записей о материале в балансовой ведомости нет, вернуть NULL.

CREATE FUNCTION calc\_remainder\_of\_material(@m\_id INT)

RETURNS DECIMAL

AS

BEGIN

RETURN

(

SELECT SUM(receipt) - SUM(expenditure)

FROM tblBalanceList AS bl

JOIN tblDocument AS doc

ON bl.idDocument = doc.id

WHERE doc.idMaterial = @m\_id

)

END

* Получить остаток материала; если сведений о материале не поступало, вернуть 0.

CREATE FUNCTION getRemainder(@mc\_id INT)

RETURNS DECIMAL

AS

BEGIN

IF EXISTS (

SELECT remainder FROM tblBalance

WHERE idMC = @mc\_id

)

BEGIN

RETURN (

SELECT TOP 1 remainder

FROM tblBalance

WHERE idMC = @mc\_id

ORDER BY id DESC

)

END

RETURN 0.0

END

* Получить id последнего добавленного документа.

CREATE FUNCTION getLastDocumentID ()

RETURNS INT

AS

BEGIN

RETURN (

SELECT TOP 1 id

FROM tblDocument

ORDER BY tblDocument.id DESC

)

END

* Получить порядковый номер следующего документа заданного типа.

CREATE FUNCTION getNextDocumentNo(@type INT)

RETURNS INT

AS

BEGIN

IF EXISTS (

SELECT \* FROM tblDocument

WHERE idType = @type

)

BEGIN

-- в базе уже есть документы типа @type

RETURN (

SELECT TOP 1 docNo+1

FROM tblDocument

WHERE idType = @type

ORDER BY id DESC

)

END

-- это первый документ такого типа

RETURN (1)

END

* Получить количество сотрудников, работающих в данном подразделении.

CREATE FUNCTION getSCCountFromDivision(@id\_div INT)

RETURNS INT

AS

BEGIN

RETURN (

SELECT COUNT(\*) FROM tblStockCustodian\_list AS sc

WHERE sc.id\_division = @id\_div

)

END

* Получить число наименований материалов в заданном документе.

CREATE FUNCTION getTotalQuantityFromDocument(@id\_doc INT)

RETURNS INT

AS

BEGIN

RETURN (

SELECT Count(mid.id\_document)

FROM tblMaterialsInDocument AS mid

WHERE mid.id\_document = @id\_doc

)

END

* Получить вид деятельности подразделения, имея информацию об его сотруднике.

CREATE FUNCTION getActivityCategoryByStockCustodian (@sc\_id int)

RETURNS varchar(50)

AS

BEGIN

RETURN (

SELECT ac.activity\_category FROM tblActivityCategory AS ac

JOIN tblDivision AS div ON div.id\_activity\_category = ac.id

JOIN stock\_custodian\_view AS sc ON sc.id\_div = div.id

WHERE sc.id\_sc = @sc\_id

)

END