|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство образования и науки Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ «ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ»

КАФЕДРА «КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ»

ДИСЦИПЛИНА «БАЗЫ ДАННЫХ»

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К КУРСОВОЙ РАБОТЕ***

***НА ТЕМУ:***

База данных "Снабжение"

Студент ИУ6-42 **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Бурлаков А.С.**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель курсового проекта **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Скворцова М.А.**

(Подпись, дата) (И.О.Фамилия)

*2018 г.*

**Министерство образования и науки Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технический университет имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

**(МГТУ им. Н.Э. Баумана)**

УТВЕРЖДАЮ

Заведующий кафедрой ИУ6

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.В. Пролетарский

« \_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2017 г.

**ЗАДАНИЕ**

**на выполнение курсовой работы**

по дисциплине ***Базы данных***

Студент группы ИУ6-42

Бурлаков Александр Сергеевич

(Фамилия, имя, отчество)

Тема курсовой работы

База Данных «Снабжение»

Направленность КР ***учебная***

Источник тематики ***кафедра ИУ6***

График выполнения КР: 25% к 3 нед., 50% к 7 нед., 75% к 10 нед., 100% к 14 нед.

Техническое задание Разработать базу данных “Снабжение”, состоящую минимум из 7 таблиц, имеющую сложные запросы и отчеты. Разработать удобный и понятный пользовательский интерфейс. В качестве СУБД для разработки использовать Oracle Database Express Edition 11g, Oracle Database SQL Developer. Для проектирования схемы данных использовать Oracle SQL Developer Data Modeler. Для проектирования пользовательского интерфейса необходимо использовать Oracle APEX версии 5.1.

***Оформление курсовой работы:***

Расчетно-пояснительная записка на не менее 25 листах формата А4.

Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.)

Дата выдачи задания «05» февраля 2018 г.

**Руководитель курсовой работы**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ М.А. Скворцова

(Подпись, дата) (И.О.Фамилия)

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** А.С.Бурлаков

(Подпись, дата) (И.О.Фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

**РЕФЕРАТ**

Расчетно-пояснительная записка 27 с., 9 рис., 10 табл., 4 источника

Объектом разработки является база данных для снабжения склада и пользовательский интерфейс для неё.

Цель работы — создание удобного и доступного интерфейса для базы данных снабжения склада.

Поставленная цель достигается за счет средств Oracle APEX версии 5.1 и Oracle SQL Developer Data Modeler. В качестве СУБД для разработки используется Oracle Database Express Edition 11g, Oracle Database SQL Developer. Разрабатываемая база данных содержит 13 таблиц, взаимодействующих между собой при помощи связей. Пользовательский интерфейс реализует взаимодействие сотрудников склада, поставщиков и заказчиков за счет формирования заказов, просмотра информации по предыдущим заказам и формированию накладных

Ключевые слова - Oracle APEX, база данных, СУБД, Oracle SQL Developer Data Modeler, таблицы, триггеры, записи, поля.

**СОДЕРЖАНИЕ**

ВВЕДЕНИЕ 6

1 Анализ и проектирование базы данных 7

1.1 Исследование предметной области 7

1.2 Бизнес процессы пользователей информационной системы 7

1.3 Создание модели в Oracle SQL Developer Data Modeler 8

2 Формирование базы данных 12

2.1 Создание пользователя для базы данных 12

2.2 Перенос базы данных в Oracle Database SQL Developer 12

2.3 Добавление данных в таблицы 12

3 Создание интерфейса в среде Oracle APEX версии 5.1 14

3.1 Создание страницы авторизации 14

3.2 Создание страницы Администратора 15

3.3 Страница ввода 16

ЗАКЛЮЧЕНИЕ 18

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ 19

Приложение А 20

Приложение Б 24

Приложение В 26

**ВВЕДЕНИЕ**

* данной работе производится создание базы данных для снабжения склада и пользовательского интерфейса для неё. Она предназначена для формирования заказов. Интерфейс предоставляет возможность работнику склада просматривать информацию о заказе, создавать заказы как для заказчиков, так и для дозакупки товаров, удалять и изменять заказы.

Разработка такой базы данных является актуальной, поскольку даёт возможность администрации склада контролировать клиентскую базу и грамотно распределять рабочие ресурсы. Данная база данных упрощает работу, выполняемую работниками склада.

Интерфейс разрабатывался в соответствии со следующими требованиями:

* наличие пользователя-администратора и пользователя-работника;
* простота и понятность обычному пользователю;



* возможность просмотра списка всех пользователей администратором;
* лаконичность интерфейса.

На данный момент существует множество аналогов такой базы данных. Можно найти множество различных интерфейсов для учета товаров на складе в интернете.

При разработке интерфейса для базы данных "Снабжение" учтены некоторые недостатки имеющихся аналогов, что позволило повысить универсальность разрабатываемого продукта.

**1 Анализ и проектирование базы данных**

**1.1 Исследование предметной области**

Предметной областью разрабатываемого программного продукта является формирование заказа для клиентов и формирование заказа для склада.

База данных «Снабжение» предназначена для упрощения обработки информации о клиентах, которые заказывают товары со склада.

Эта база данных предназначена для менеджеров (работников склада), которые заносят информацию в базу, регистрируя и работая с клиентами. Но также может использоваться людьми, имеющими непосредственное отношение к работе БД, такими как вышестоящее руководство.

База данных выполняет следующие функции:

1. Хранение информации о сотрудниках
2. Хранение информации о клиентах
3. Хранение информации о поставщиках
4. Хранение информации о заказах

Данные требования к разрабатываемой базе данных учтены при создании модели, а также доработаны в соответствии с возникающими проблемами при проектировании.

После рассмотрения предметной области было решено создать следующие страницы таблиц:

1. «Сеть магазинов», в которой должна содержаться информация о торговых сетях («Пятёрочка») ;
2. «Магазин», в которой должна содержаться информация об адресе и названии магазина, а так же об принадлежности его сети магазинов;
3. «Заказ», в которой должна находиться информация о складе, с которого заказывают товар, текущем статусе заказа;
4. «Заказ склада», в которой должна находиться информация о складе, на который заказывают товар, текущем статусе этого заказа;
5. «Продукт», в которой должен содержаться полный перечень названий и цен продуктов;
6. «Продукт на складе», в которой должна находиться информация о количестве продукта, находящемся на определенном складе.
7. «Продукт заказа», в которой должна находиться информация о количестве продуктов определенного заказа;
8. «Продукт заказа склада», в которой должна находиться информация о количестве продуктов определенного заказа склада;
9. «Пользователь», в которой должна находиться информация о логине, пароле, роли и ФИО пользователя;
10. «Склад», в которой должна содержаться информация о названии и балансе склада

Схема связи таблиц представлена на рисунке 1.

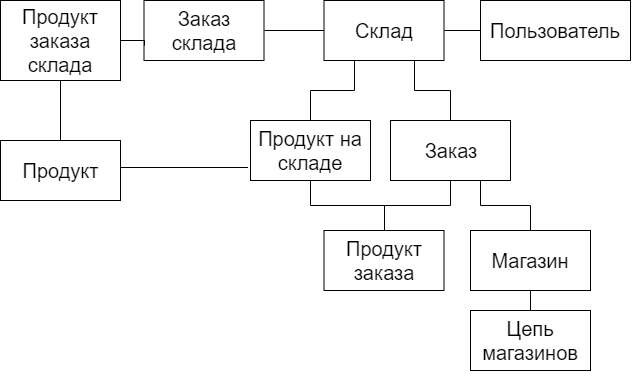


Рисунок 1 - Схема связи таблиц

**1.2 Создание модели в Oracle SQL Developer Data Modeler**

Для хранения баланса на складе, а так же учета сотрудников склада создана таблица «Склад». Она содержит информацию о балансе склада, администраторе склада, полученным из таблицы «Пользователь» и его названии.

Так как необходимо учитывать работников склада создана таблица «Пользователь». Она содержит логин и пароль пользователя, с помощью которых он будет заходить в систему. Помимо этого таблица содержит данные о пользователе – его ФИО и текущий статус (работник склада «Р» или администратор «А»).

Основное назначение базы данных – формирование заказов, поэтому создана таблица «Заказ». В таблице содержится информация о магазине, информация о котором получена из таблицы «Магазин», складе, с которого заказ будет приниматься, текущий статус заказа (Закрыт «A», Открыт «Y»). Помимо этого учитывается дата создания и закрытия заказа.

Для заказа так же необходимо учитывать продукты, которые в него входят. Для этого создана таблица «Продукт заказа», в которой содержится информация о продукте со склада, полученным из таблицы «Продукт склада», количестве заказанных продуктов, а так же номеру заказа, полученному из таблицы «Заказ», который необходим для реализации связи один-много.

Из предыдущей таблицы появляется необходимость создания следующей: таблицы «Продукт склада», которая содержит информацию о названии продукта, его количестве на складе, цене, а так же идентификатор склада, о продукте которого ведется описание. Идентификатор берется из таблицы «Склад».

Так как заказчиков у склада может быть много, решено создать таблицу «Цепь магазинов» которая объединяет некоторые магазины. В этой таблице хранится информация об названии цепи, например «Пятерочка».

Для учета заказчиков создана таблица «Магазин», в которой хранится информация о цепи магазинов, информация о которых берется из таблицы «Цепь магазинов», к которой магазин принадлежит, а так же его название и адрес.

Все следующие таблицы созданы для хранения информации о поставках на склад.

Таблица «Заказ склада» имеет информацию о складе, информация о котором берется из таблицы «Склад», на который завозят товары, текущем статусе заказа (Закрыт «A» и Открыт «Y»), дате создания и закрытия заказа.

Таблица «Продукт заказа склада» содержит информацию о заказываемом продукте, информация о котором берется из таблицы «Продукт», количестве этого продукта, а так же информация о заказе, которая берется из таблицы «Заказ склада»

Таблица «Продукт» содержит информацию о названии и цене продукта.

Модель разработанной базы данных "Снабжение" представлена на рисунке 5

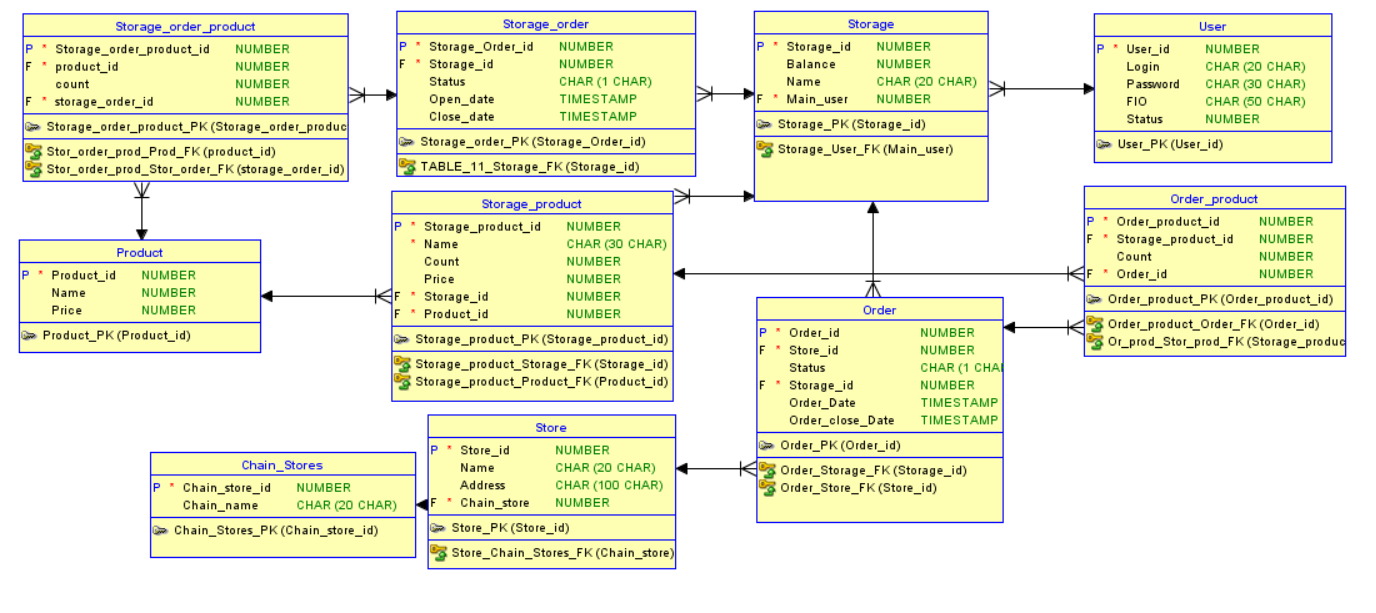


Рисунок 5 - Модель разработанной базы данных

Таблицей называется элемент, который представлен на рисунке 6.

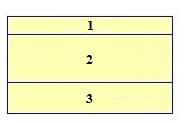


Рисунок 6 - Таблица

В данной таблице присутствуют следующие обозначения:

1. Название таблицы;
2. Описание столбцов таблицы (название и тип данных);
3. Описание ключей, связывающих данную таблицу с другими таблицами.

Основными таблицами являются таблицы “Storage\_order” и “Order”, так как основное назначение базы данных – создание заказов и дозакупка товаров.

Таблица “Product” содержит информацию о товарах. Эта таблица состоит из следующих полей:

1. Product\_id – первичный ключ;
2. Name – название товара;
3. Price – цена товара.

Таблица “Storage\_product” содержит информацию о товарах склада. Эта таблица состоит из следующих полей:

1. Storage\_product\_id – первичный ключ;
2. Name – название товара (так как возможно его переименовали);
3. Count – количество такого товара на складе;
4. Price – цена товара;
5. Storage\_id – склад. Вторичный ключ, содержащий первичный ключ таблицы “Storage”;
6. Product\_id – товар. Вторичный ключ, содержащий первичный ключ таблицы “Product”.

Таблица “Storage\_order\_product” содержит информацию о товарах заказа товаров допоставки на склад. Эта таблица состоит из следующих полей:

1. Storage\_order\_product\_id – первичный ключ;
2. product\_id – продукт, вторичный ключ, содержащий первичный ключ таблицы “Product”;
3. count – количество товара в заказе;
4. storage\_order\_id – заказ допоставки, вторичный ключ, содержащий первичный ключ таблицы “Storage\_order”.

Таблица “Storage\_order” содержит информацию о заказах допоставки на склад. Эта таблица состоит из следующих полей:

1. Storage\_order\_id – первичный ключ;
2. Storage\_id – склад, вторичный ключ, содержащий первичный ключ таблицы “Storage”;
3. Status – текущий статус заказа. Хранит значение ‘A’, если заказ закрыт и значение ‘Y’, если заказ активен;
4. Open\_date – дата открытия заказа;
5. Close\_date – дата закрытия заказа.

Таблица “Storage” содержит информацию о складах. Эта таблица состоит из следующих полей:

1. Storage\_id – первичный ключ;
2. Balance – текущий баланс склада;
3. Name – название склада;
4. Main\_user – пользователь-администратор, вторичный ключ, содержащий первичный ключ таблицы “User”.

Таблица “User” содержит информацию о пользователях программы. Эта таблица состоит из следующих полей:

1. User\_id – первичный ключ;
2. Login – никнейм пользователя;
3. Password – пароль для входа в систему;
4. FIO – фамилия, имя, отчество пользователя;
5. Status – роль, данная пользователю. Хранит значение ‘A’, если пользователь является администратором и значение ‘P’, если пользователь не имеет таких прав.

Таблица “Chain\_Stores” содержит информацию о торговых цепях. Эта таблица состоит из следующих полей:

1. Chain\_store\_id – первичный ключ;
2. Chain\_name – название торговой цепи.

Таблица “Store” содержит информацию о магазинах. Эта таблица состоит из следующих полей:

1. Store\_id – первичный ключ;
2. Name – название магазина;
3. Address – адрес магазина;
4. Chain\_store – торговая цепь, в которую входит магазин, вторичный ключ, содержащий первичный ключ таблицы “Chain\_Stores”.

Таблица “Order” содержит информацию о закзах склада. Эта таблица состоит из следующих полей:

1. Order\_id – первичный ключ;
2. Store\_id – магазин заказчик, вторичный ключ, содержащий первичный ключ таблицы “Store”;
3. Status – текущий статус заказа. Хранит значение ‘A’, если заказ закрыт и ‘Y’, если заказ активен.
4. Storage\_id – склад, вторичный ключ, содержащий первичный ключ таблицы “Storage”.
5. Order\_Date – дата создания заказа;
6. Order\_close\_date – дата закрытия заказа.

Таблица “Order\_product” содержит информацию о товарах заказа склада. Эта таблица состоит из следующих полей:

1. Order\_product\_id – первичный ключ;
2. Storage\_product\_id – продукт склада, вторичный ключ, содержащий первичный ключ таблицы “Storage\_product”;
3. Count – количество заказанного товара;
4. Order\_id – заказ, вторичный ключ, содержащий первичный ключ таблицы “Order”.

**1.2 Бизнес процессы пользователей информационной системы**

В данной системе предусмотрены процессы создания заказа, а так же отметка о том, что заказ отдан. Следовательно, можно выделить два бизнес процесса:

1. Создание заказа;
2. Отметка об отдаче заказа;

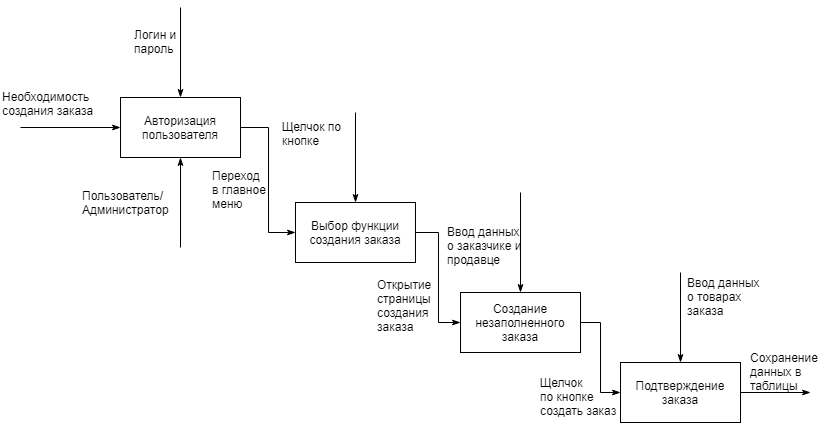


Рисунок 1 – Бизнес-процесс создания заказа

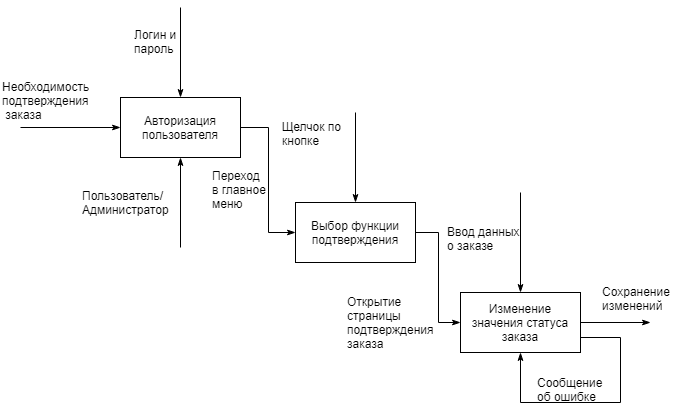


Рисунок 2 – Бизнес-процесс подтверждения заказа

2 **Формирование базы данных**

**2.1 Создание пользователя для базы данных**

Создание пользователя происходило с помощью графического интерфейса SQL Developer, т.к. данный способ создания позволяет видеть все права, которыми может обладать пользователь, и выбирать их из списка.

Принято решение наделить пользователя следующими правами доступа:

* + возможность создания, изменения и удаления таблиц;
  + возможность создания, изменения и удаления процедур;
  + возможность создания, изменения и удаления триггеров;
  + возможность создания представлений;
  + возможность создания, изменения и удаления счетчиков;
  + возможность создания сессий;
  + возможность изменения пользователя.

**2.2 Перенос базы данных в Oracle Database SQL Developer**

Формирование базы данных началось с переноса схемы базы данных в Oracle Database SQL Developer. Для этого было необходимо получить схему базы данных в виде SQL кода и выполнить его в схеме пользователя, созданного предварительно в Oracle Database Express Edition 11g.

# Листинг SQL-код, сгенерированный SQL Data Modeler

-- Generated by Oracle SQL Developer Data Modeler 17.2.0.188.1059

-- at: 2018-05-19 16:25:34 MSK

-- site: Oracle Database 11g

-- type: Oracle Database 11g

CREATE TABLE chain\_stores (

chain\_store\_id NUMBER NOT NULL,

chain\_name CHAR(20 CHAR)

);

ALTER TABLE chain\_stores ADD CONSTRAINT chain\_stores\_pk PRIMARY KEY ( chain\_store\_id );

CREATE TABLE "Order" (

order\_id NUMBER NOT NULL,

store\_id NUMBER NOT NULL,

status CHAR(1 CHAR),

storage\_id NUMBER NOT NULL,

order\_date TIMESTAMP,

order\_close\_date TIMESTAMP

);

ALTER TABLE "Order" ADD CONSTRAINT order\_pk PRIMARY KEY ( order\_id );

CREATE TABLE order\_product (

order\_product\_id NUMBER NOT NULL,

storage\_product\_id NUMBER NOT NULL,

count NUMBER,

order\_id NUMBER NOT NULL

);

ALTER TABLE order\_product ADD CONSTRAINT order\_product\_pk PRIMARY KEY ( order\_product\_id );

CREATE TABLE product (

product\_id NUMBER NOT NULL,

name NUMBER,

price NUMBER

);

ALTER TABLE product ADD CONSTRAINT product\_pk PRIMARY KEY ( product\_id );

CREATE TABLE storage (

storage\_id NUMBER NOT NULL,

balance NUMBER,

name CHAR(20 CHAR),

main\_user NUMBER NOT NULL

);

ALTER TABLE storage ADD CONSTRAINT storage\_pk PRIMARY KEY ( storage\_id );

CREATE TABLE storage\_order (

storage\_order\_id NUMBER NOT NULL,

storage\_id NUMBER NOT NULL,

status CHAR(1 CHAR),

open\_date TIMESTAMP,

close\_date TIMESTAMP

);

ALTER TABLE storage\_order ADD CONSTRAINT storage\_order\_pk PRIMARY KEY ( storage\_order\_id );

CREATE TABLE storage\_order\_product (

storage\_order\_product\_id NUMBER NOT NULL,

product\_id NUMBER NOT NULL,

count NUMBER,

storage\_order\_id NUMBER NOT NULL

);

ALTER TABLE storage\_order\_product ADD CONSTRAINT storage\_order\_product\_pk PRIMARY KEY ( storage\_order\_product\_id );

CREATE TABLE storage\_product (

storage\_product\_id NUMBER NOT NULL,

name CHAR(30 CHAR) NOT NULL,

count NUMBER,

price NUMBER,

storage\_id NUMBER NOT NULL

);

ALTER TABLE storage\_product ADD CONSTRAINT storage\_product\_pk PRIMARY KEY ( storage\_product\_id );

CREATE TABLE store (

store\_id NUMBER NOT NULL,

name CHAR(20 CHAR),

address CHAR(100 CHAR),

chain\_store NUMBER NOT NULL

);

ALTER TABLE store ADD CONSTRAINT store\_pk PRIMARY KEY ( store\_id );

CREATE TABLE "User" (

user\_id NUMBER NOT NULL,

login CHAR(20 CHAR),

password CHAR(30 CHAR),

fio CHAR(50 CHAR),

status NUMBER

);

ALTER TABLE "User" ADD CONSTRAINT user\_pk PRIMARY KEY ( user\_id );

ALTER TABLE order\_product

ADD CONSTRAINT or\_prod\_stor\_prod\_fk FOREIGN KEY ( storage\_product\_id )

REFERENCES storage\_product ( storage\_product\_id );

ALTER TABLE order\_product

ADD CONSTRAINT order\_product\_order\_fk FOREIGN KEY ( order\_id )

REFERENCES "Order" ( order\_id );

ALTER TABLE "Order"

ADD CONSTRAINT order\_storage\_fk FOREIGN KEY ( storage\_id )

REFERENCES storage ( storage\_id );

ALTER TABLE "Order"

ADD CONSTRAINT order\_store\_fk FOREIGN KEY ( store\_id )

REFERENCES store ( store\_id );

ALTER TABLE storage\_order\_product

ADD CONSTRAINT stor\_order\_prod\_prod\_fk FOREIGN KEY ( product\_id )

REFERENCES product ( product\_id );

ALTER TABLE storage\_order\_product

ADD CONSTRAINT stor\_order\_prod\_stor\_order\_fk FOREIGN KEY ( storage\_order\_id )

REFERENCES storage\_order ( storage\_order\_id );

ALTER TABLE storage\_product

ADD CONSTRAINT storage\_product\_storage\_fk FOREIGN KEY ( storage\_id )

REFERENCES storage ( storage\_id );

ALTER TABLE storage

ADD CONSTRAINT storage\_user\_fk FOREIGN KEY ( main\_user )

REFERENCES "User" ( user\_id );

ALTER TABLE store

ADD CONSTRAINT store\_chain\_stores\_fk FOREIGN KEY ( chain\_store )

REFERENCES chain\_stores ( chain\_store\_id );

ALTER TABLE storage\_order

ADD CONSTRAINT table\_11\_storage\_fk FOREIGN KEY ( storage\_id )

REFERENCES storage ( storage\_id );

-- Oracle SQL Developer Data Modeler Summary Report:

--

-- CREATE TABLE 10

-- CREATE INDEX 0

-- ALTER TABLE 20

-- CREATE VIEW 0

-- ALTER VIEW 0

-- CREATE PACKAGE 0

-- CREATE PACKAGE BODY 0

-- CREATE PROCEDURE 0

-- CREATE FUNCTION 0

-- CREATE TRIGGER 0

-- ALTER TRIGGER 0

-- CREATE COLLECTION TYPE 0

-- CREATE STRUCTURED TYPE 0

-- CREATE STRUCTURED TYPE BODY 0

-- CREATE CLUSTER 0

-- CREATE CONTEXT 0

-- CREATE DATABASE 0

-- CREATE DIMENSION 0

-- CREATE DIRECTORY 0

-- CREATE DISK GROUP 0

-- CREATE ROLE 0

-- CREATE ROLLBACK SEGMENT 0

-- CREATE SEQUENCE 0

-- CREATE MATERIALIZED VIEW 0

-- CREATE SYNONYM 0

-- CREATE TABLESPACE 0

-- CREATE USER 0

--

-- DROP TABLESPACE 0

-- DROP DATABASE 0

--

-- REDACTION POLICY 0

--

-- ORDS DROP SCHEMA 0

-- ORDS ENABLE SCHEMA 0

-- ORDS ENABLE OBJECT 0

--

-- ERRORS 0

-- WARNINGS 0

**2.3 Добавление данных в таблицы**

Было решено первыми заполнить таблицы, которые не зависят от данных

* других таблицах. Сначала были заполнены таблицы «Product»,

«User », «Chain\_stores», «store», «Storage».

Для добавления данных в таблицы было решено использовать SQL.[6] В схеме пользователя был написан и отлажен оператор INSERT, который создаёт записи в таблице. В листинге 1 представлено добавление записи в таблицу "STORAGE\_PRODUCT" при помощи оператора INSERT:

# Листинг 1 - Добавление записи в таблицу «*STORAGE\_PRODUCT*»

INSERT INTO STORAGE\_PRODUCT (NAME, PRICE, COUNT, STORAGE\_ID)

VALUES ('Продукт 1', 35, 10, 1);

Команда INSERT за один вызов заносит в таблицу лишь одну строку, поэтому, чтобы заполнить нашу таблица нам понадобится вызвать команду столько раз, сколько строк мы планируем заполнить.

Для формирования идентификаторов у каждой таблицы необходимо создать последовательности. Для каждой таблицы были созданы последовательности, номер которых начинался с 0.

# Листинг 2 – Добавление последовательности в таблицу order

CREATE SEQUENCE order\_SEQ START WITH 0 INCREMENT BY 1 MINVALUE 0;

Для автоматического заполнения идентификаторов таблиц были созданы триггеры, использующие последовательности для заполнения номеров.

# Листинг 3 – Добавление триггера для таблицы order

CREATE OR REPLACE TRIGGER order\_TRG BEFORE

INSERT ON "Order" FOR EACH ROW

BEGIN

:NEW.order\_id := order\_SEQ.NEXTVAL;

END;

**3 Создание интерфейса в среде Oracle APEX версии 5.1**

**3.1 Создание страницы авторизации**

Было решено создать страницу авторизации, на которой пользователю предлагается ввести свои логин и пароль.

Для осуществления авторизации используется таблица “User”.

Для осуществления авторизации нового постояльца был создан пакет Pkg\_Security. Его спецификация и тело представлено в Приложении В.

Спецификация описывает прототип функции Authenticate\_User, результатом выполнения которого является логическое значение, зависящее от наличия пользователя в таблице «USER\_LOGIN» (возвращает TRUE, если пользователь найден, FALSE - в противном случае), а так же передача в текущую сессию Apex значение логина текущего пользователя и его роль (Администратор или пользователь).

* + теле функции осуществляется сравнение введенного имени пользователя в поле «Login» с полем «p\_Login» и введенного пароля
* поле «Password» с полем «p\_Password».

Спецификация также описывает прототип функции Process\_Login, которая производит переход на домашнюю страницу приложения и вызывает предыдущую функцию.

Для авторизации необходимо было создать страницу, в которой присутствуют поля для ввода логина и пароля и кнопка «Войти». Интерфейс страницы авторизации представлен на рисунке 7.

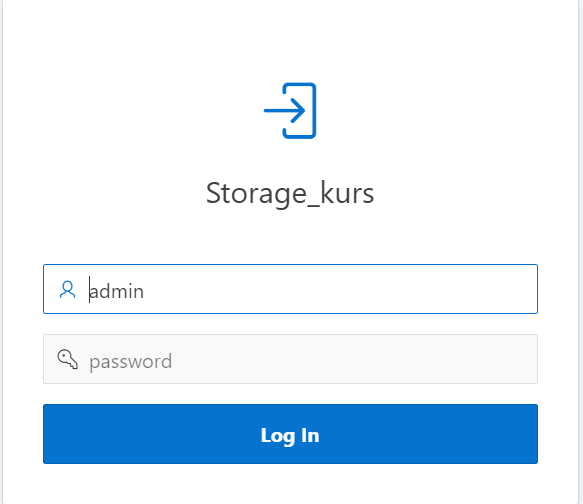
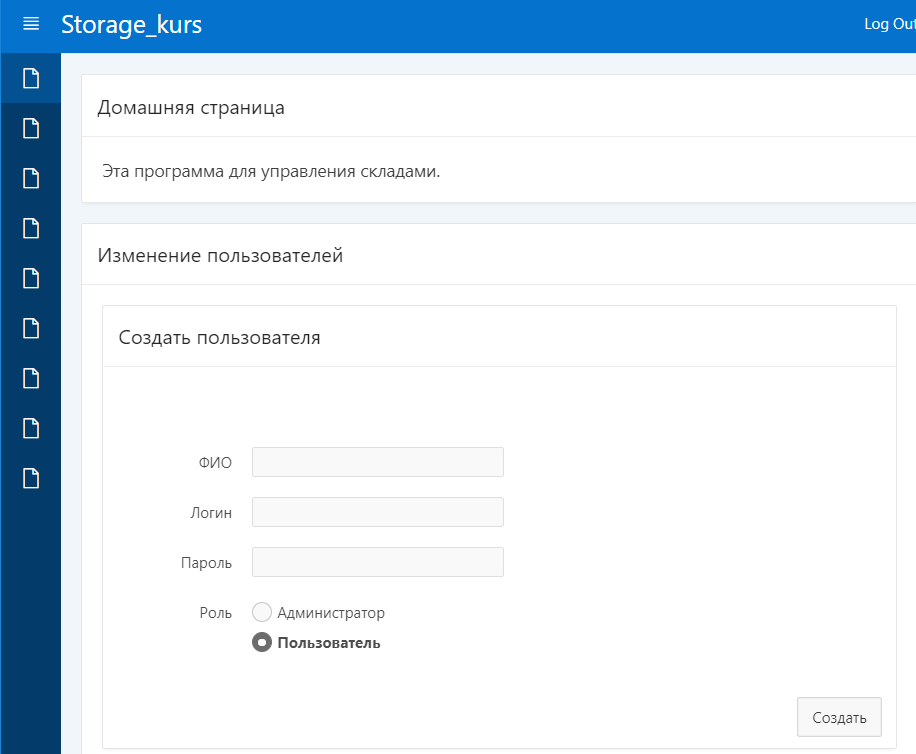


Рисунок 7 - Страница авторизации

**3.2. Создание страницы Администратора**

Для администратора системы предусмотрена страница управления пользователями, которая включает в себя функцию создания, изменения и удаления пользователей и просмотра всех пользователей системы. Создание, удаление и изменение пользователей происходит при нажатии на соответствующие кнопки (см. Рисунок 8).



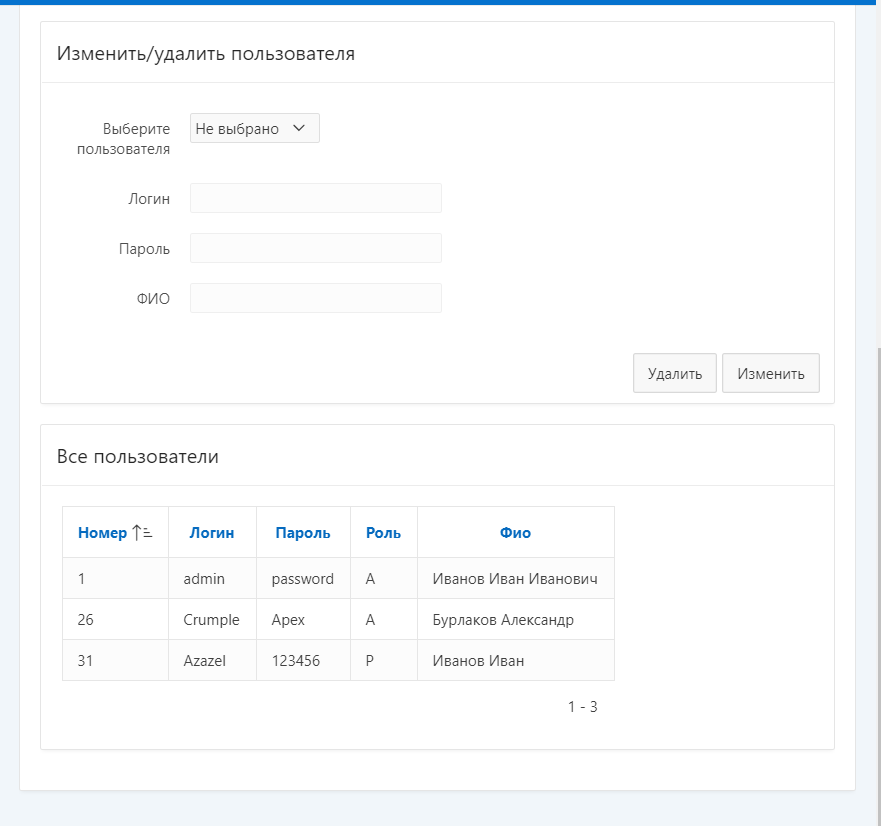


Рисунок 8 - Страница управления пользователями

**3.3. Страница входа**

Страница входа в систему (Рисунок 9) состоит из формы для ввода имени учётной записи и пароля.

На этом этапе разница между пользователем и администратором отсутствует.

Также страница содержит кнопку входа “Log In”, запускающая проверку введённых данных на существование в таблице пользователей. Если пользователя, соответствующего введённым данным, не существует, то вход не будет произведён. Если проверка пройдена, то пользователь получает доступ к приложению в рамках своих прав.

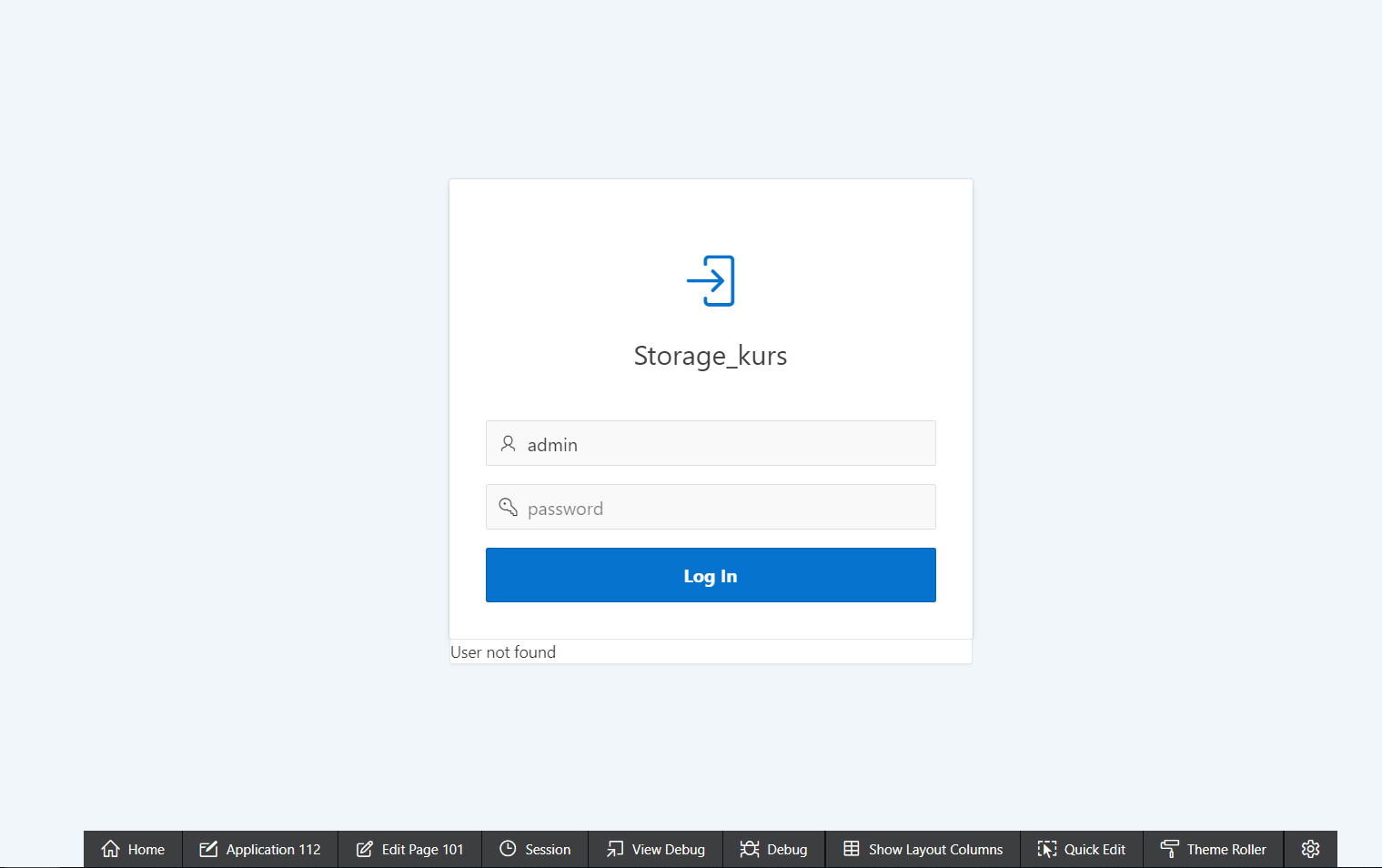
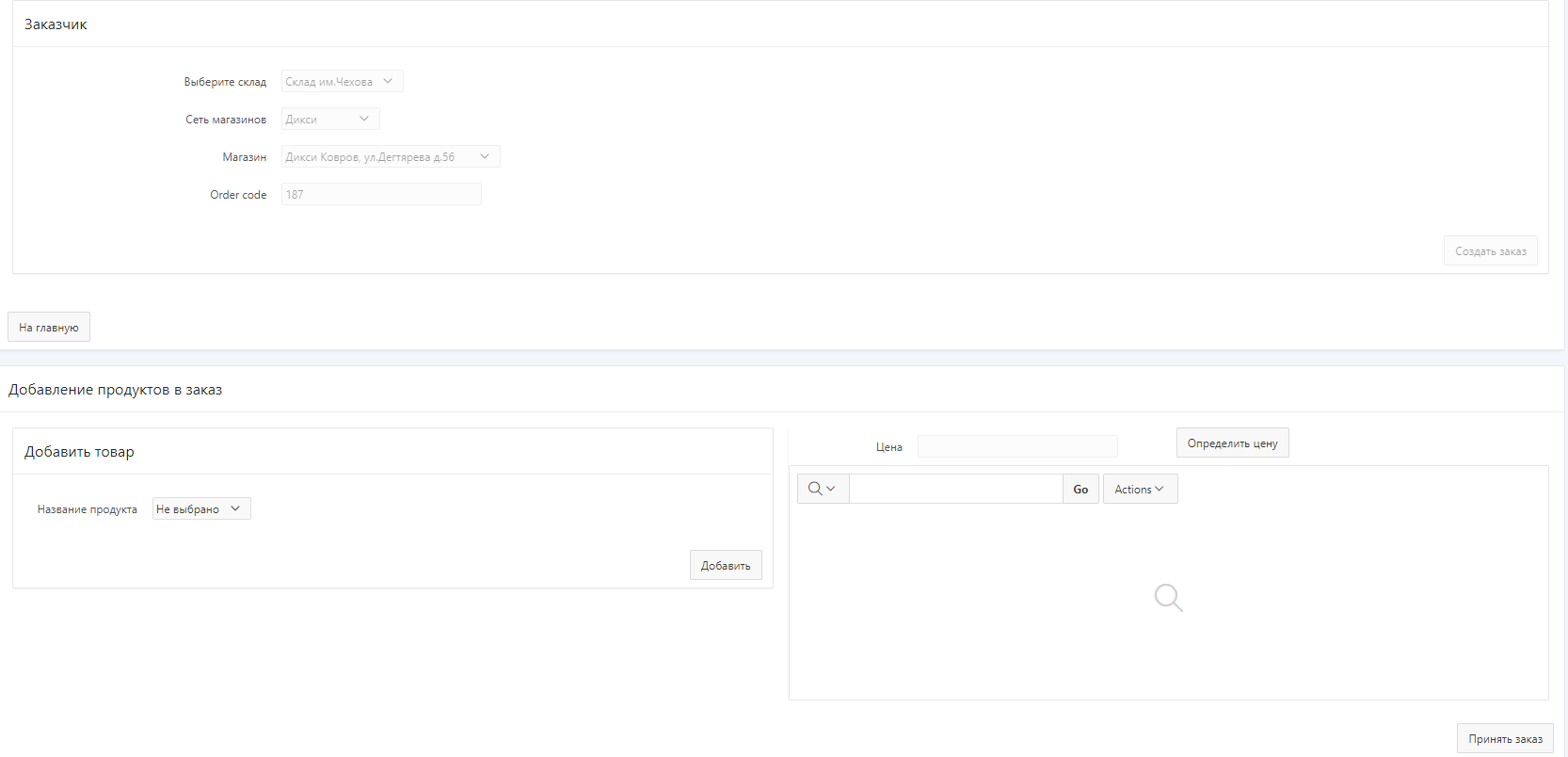


Рисунок 9 - Страница входа в систему

**3.4 Страница создания заказа**

Рисунок \* - страница создания заказа

При создании страницы были прописаны SQL-коды для генерации значений Select List у полей P2\_STORAGE, P2\_CHAIN, P2\_STORE, P2\_PRODUCT\_ID, которые представлены ниже:

SQL-код генерации значений у P2\_STORAGE:

SELECT

name, storage\_id

FROM

"STORAGE";

SQL-код генерации значений у P2\_CHAIN:

SELECT

chain\_name, chain\_store\_id

FROM

chain\_stores;

SQL-код генерации значений у P2\_STORE:

SELECT

name||' '||address, store\_id

FROM

store

WHERE

chain\_store = :P2\_CHAIN;

SQL-код генерации значений у P2\_PRODUCT\_ID:

select

name, storage\_product\_id

from

storage\_product

where

:P2\_STORAGE = storage\_id;

SQL-код заполнения таблицы «Текущие продукты в заказе»:

select

s.name as "Название товара",

p.count as "Количество",

s.price as "Цена за штуку"

from

storage\_product s,

order\_product p

where

p.order\_code = :P2\_ORDER\_CODE and

p.storage\_product\_id = s.storage\_product\_id;

При нажатии на кнопку «Создать» создается неактивный заказ. SQL-код создания такого заказа:

IF (:P2\_STORE IS NOT NULL) AND

(:P2\_STORAGE IS NOT NULL)

THEN

INSERT INTO "Order"

(store\_id, status, storage\_id, order\_date)

VALUES

(:P2\_STORE, 'N', :P2\_STORAGE, CURRENT\_DATE);

END IF;

:P2\_ORDER\_CODE := ORDER\_SEQ.CURRVAL;

При нажатии на кнопку «Добавить продукт» создается новая запись в таблице «Order\_product». SQL-код добавления товара:

add\_order\_product(:P2\_PRODUCT\_ID, :P2\_PRODUCT\_COUNT, :P2\_ORDER\_CODE);

При нажатии на кнопку «Определить цену» в поле P2\_PRICE выводится общая цена заказа. SQL-код определения цены:

DECLARE

C INT := 0;

BEGIN

FOR PRODUCT IN (SELECT \* FROM ORDER\_PRODUCT WHERE ORDER\_CODE = :P2\_ORDER\_CODE) LOOP

FOR STOR\_PROD IN (SELECT \* FROM STORAGE\_PRODUCT

WHERE PRODUCT.STORAGE\_PRODUCT\_ID = STORAGE\_PRODUCT\_ID) LOOP

C := C + PRODUCT.COUNT \* STOR\_PROD.PRICE;

END loop;

END loop;

:P2\_PRICE := C;

end;

При нажатии на кнопку «Принять заказ» заказ становится активным и его можно увидеть во всех остальных страницах приложения. SQL-код активации заказа:

UPDATE "Order" SET status='Y' where order\_id = :P2\_ORDER\_CODE;

**ЗАКЛЮЧЕНИЕ**

* результате выполнения курсового проекта была создана система работы со складом.

Данная система основана на базе данных, реализованной при помощи программных средств СУБД MySQL, и имеет пользовательский интерфейс, созданный на основе APEX Oracle.

В процессе реализации проекта были выполнены:

— Исследование предметной области, ознакомление с существующими аналогами и разработка концепции системы;

— Построение схемы объектов базы данных при помощи Oracle Database

— Генерация SQL-кодов по созданной модели, исправление ошибок генерации и необходимая доработка в соответствии с особенностями СУБД

— Заполнение базы данных;

— Реализация пользовательского интерфейса с помощью APEX Oracle;

— Проверка функционирования всех форм интерфейса.

**СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ**

1. Райордан Р, Основы реляционных баз данных [Текст] / Ребекка Райордан: Основы реляционных баз данных - Microsoft Press Русская редакция, 2012, 390с. - ISBN 5-7502-0150-3
2. Грофф Джеймс Р., SQL Полное руководство [Текст] / Джеймс Р. Грофф, SQL Полное руководство - Издательский дом "Вильямс", 2015, 390с. - ISBN 978-5-8459-1654-9
3. Иванова Г.С., Ничушкина Т.Н. Оформление текстовых документов. Методические указания по оформлению расчетно-пояснительных записок дипломных и квалификационных работ. - М.: Издательство МГТУ им.Н.Э. Баумана, 2004. - 10 с.
4. Фейерштейн С., Прибыл Б. Oracle PL/SQL. Для профессионалов. 6-е изд. — СПб.: Питер, 2015. — 1024 с.

**ПРИЛОЖЕНИЯ**

**Приложение А**

**Код создания таблиц в Oracle SQL Database**

-- Generated by Oracle SQL Developer Data Modeler 17.2.0.188.1059

-- at: 2018-05-19 16:25:34 MSK

-- site: Oracle Database 11g

-- type: Oracle Database 11g

CREATE TABLE chain\_stores (

chain\_store\_id NUMBER NOT NULL,

chain\_name CHAR(20 CHAR)

);

ALTER TABLE chain\_stores ADD CONSTRAINT chain\_stores\_pk PRIMARY KEY ( chain\_store\_id );

CREATE TABLE "Order" (

order\_id NUMBER NOT NULL,

store\_id NUMBER NOT NULL,

status CHAR(1 CHAR),

storage\_id NUMBER NOT NULL,

order\_date TIMESTAMP,

order\_close\_date TIMESTAMP

);

ALTER TABLE "Order" ADD CONSTRAINT order\_pk PRIMARY KEY ( order\_id );

CREATE TABLE order\_product (

order\_product\_id NUMBER NOT NULL,

storage\_product\_id NUMBER NOT NULL,

count NUMBER,

order\_id NUMBER NOT NULL

);

ALTER TABLE order\_product ADD CONSTRAINT order\_product\_pk PRIMARY KEY ( order\_product\_id );

CREATE TABLE product (

product\_id NUMBER NOT NULL,

name NUMBER,

price NUMBER

);

ALTER TABLE product ADD CONSTRAINT product\_pk PRIMARY KEY ( product\_id );

CREATE TABLE storage (

storage\_id NUMBER NOT NULL,

balance NUMBER,

name CHAR(20 CHAR),

main\_user NUMBER NOT NULL

);

ALTER TABLE storage ADD CONSTRAINT storage\_pk PRIMARY KEY ( storage\_id );

CREATE TABLE storage\_order (

storage\_order\_id NUMBER NOT NULL,

storage\_id NUMBER NOT NULL,

status CHAR(1 CHAR),

open\_date TIMESTAMP,

close\_date TIMESTAMP

);

ALTER TABLE storage\_order ADD CONSTRAINT storage\_order\_pk PRIMARY KEY ( storage\_order\_id );

CREATE TABLE storage\_order\_product (

storage\_order\_product\_id NUMBER NOT NULL,

product\_id NUMBER NOT NULL,

count NUMBER,

storage\_order\_id NUMBER NOT NULL

);

ALTER TABLE storage\_order\_product ADD CONSTRAINT storage\_order\_product\_pk PRIMARY KEY ( storage\_order\_product\_id );

CREATE TABLE storage\_product (

storage\_product\_id NUMBER NOT NULL,

name CHAR(30 CHAR) NOT NULL,

count NUMBER,

price NUMBER,

storage\_id NUMBER NOT NULL

);

ALTER TABLE storage\_product ADD CONSTRAINT storage\_product\_pk PRIMARY KEY ( storage\_product\_id );

CREATE TABLE store (

store\_id NUMBER NOT NULL,

name CHAR(20 CHAR),

address CHAR(100 CHAR),

chain\_store NUMBER NOT NULL

);

ALTER TABLE store ADD CONSTRAINT store\_pk PRIMARY KEY ( store\_id );

CREATE TABLE "User" (

user\_id NUMBER NOT NULL,

login CHAR(20 CHAR),

password CHAR(30 CHAR),

fio CHAR(50 CHAR),

status NUMBER

);

ALTER TABLE "User" ADD CONSTRAINT user\_pk PRIMARY KEY ( user\_id );

ALTER TABLE order\_product

ADD CONSTRAINT or\_prod\_stor\_prod\_fk FOREIGN KEY ( storage\_product\_id )

REFERENCES storage\_product ( storage\_product\_id );

ALTER TABLE order\_product

ADD CONSTRAINT order\_product\_order\_fk FOREIGN KEY ( order\_id )

REFERENCES "Order" ( order\_id );

ALTER TABLE "Order"

ADD CONSTRAINT order\_storage\_fk FOREIGN KEY ( storage\_id )

REFERENCES storage ( storage\_id );

ALTER TABLE "Order"

ADD CONSTRAINT order\_store\_fk FOREIGN KEY ( store\_id )

REFERENCES store ( store\_id );

ALTER TABLE storage\_order\_product

ADD CONSTRAINT stor\_order\_prod\_prod\_fk FOREIGN KEY ( product\_id )

REFERENCES product ( product\_id );

ALTER TABLE storage\_order\_product

ADD CONSTRAINT stor\_order\_prod\_stor\_order\_fk FOREIGN KEY ( storage\_order\_id )

REFERENCES storage\_order ( storage\_order\_id );

ALTER TABLE storage\_product

ADD CONSTRAINT storage\_product\_storage\_fk FOREIGN KEY ( storage\_id )

REFERENCES storage ( storage\_id );

ALTER TABLE storage

ADD CONSTRAINT storage\_user\_fk FOREIGN KEY ( main\_user )

REFERENCES "User" ( user\_id );

ALTER TABLE store

ADD CONSTRAINT store\_chain\_stores\_fk FOREIGN KEY ( chain\_store )

REFERENCES chain\_stores ( chain\_store\_id );

ALTER TABLE storage\_order

ADD CONSTRAINT table\_11\_storage\_fk FOREIGN KEY ( storage\_id )

REFERENCES storage ( storage\_id );

-- Oracle SQL Developer Data Modeler Summary Report:

--

-- CREATE TABLE 10

-- CREATE INDEX 0

-- ALTER TABLE 20

-- CREATE VIEW 0

-- ALTER VIEW 0

-- CREATE PACKAGE 0

-- CREATE PACKAGE BODY 0

-- CREATE PROCEDURE 0

-- CREATE FUNCTION 0

-- CREATE TRIGGER 0

-- ALTER TRIGGER 0

-- CREATE COLLECTION TYPE 0

-- CREATE STRUCTURED TYPE 0

-- CREATE STRUCTURED TYPE BODY 0

-- CREATE CLUSTER 0

-- CREATE CONTEXT 0

-- CREATE DATABASE 0

-- CREATE DIMENSION 0

-- CREATE DIRECTORY 0

-- CREATE DISK GROUP 0

-- CREATE ROLE 0

-- CREATE ROLLBACK SEGMENT 0

-- CREATE SEQUENCE 0

-- CREATE MATERIALIZED VIEW 0

-- CREATE SYNONYM 0

-- CREATE TABLESPACE 0

-- CREATE USER 0

--

-- DROP TABLESPACE 0

-- DROP DATABASE 0

--

-- REDACTION POLICY 0

--

-- ORDS DROP SCHEMA 0

-- ORDS ENABLE SCHEMA 0

-- ORDS ENABLE OBJECT 0

--

-- ERRORS 0

-- WARNINGS 0

**ПРИЛОЖЕНИЕ Б**

**Создание триггеров и последовательностей**

CREATE SEQUENCE chain\_stores\_SEQ START WITH 0 INCREMENT BY 1 MINVALUE 0;

CREATE SEQUENCE order\_SEQ START WITH 0 INCREMENT BY 1 MINVALUE 0;

CREATE SEQUENCE order\_product\_SEQ START WITH 0 INCREMENT BY 1 MINVALUE 0;

CREATE SEQUENCE product\_SEQ START WITH 0 INCREMENT BY 1 MINVALUE 0;

CREATE SEQUENCE storage\_SEQ START WITH 0 INCREMENT BY 1 MINVALUE 0;

CREATE SEQUENCE storage\_order\_SEQ START WITH 0 INCREMENT BY 1 MINVALUE 0;

CREATE SEQUENCE stor\_order\_prod\_SEQ START WITH 0 INCREMENT BY 1 MINVALUE 0;

CREATE SEQUENCE storage\_product\_SEQ START WITH 0 INCREMENT BY 1 MINVALUE 0;

CREATE SEQUENCE store\_SEQ START WITH 0 INCREMENT BY 1 MINVALUE 0;

CREATE SEQUENCE user\_SEQ START WITH 0 INCREMENT BY 1 MINVALUE 0;

CREATE OR REPLACE TRIGGER chain\_stores\_TRG BEFORE

INSERT ON chain\_stores FOR EACH ROW

BEGIN

:NEW.chain\_store\_id := chain\_stores\_SEQ.NEXTVAL;

END;

CREATE OR REPLACE TRIGGER order\_TRG BEFORE

INSERT ON "Order" FOR EACH ROW

BEGIN

:NEW.order\_id := order\_SEQ.NEXTVAL;

END;

CREATE OR REPLACE TRIGGER order\_product\_TRG BEFORE

INSERT ON Order\_product FOR EACH ROW

BEGIN

:NEW.order\_product\_id := order\_product\_SEQ.NEXTVAL;

END;

CREATE OR REPLACE TRIGGER product\_TRG BEFORE

INSERT ON product FOR EACH ROW

BEGIN

:NEW.product\_id := product\_SEQ.NEXTVAL;

END;

CREATE OR REPLACE TRIGGER storage\_TRG BEFORE

INSERT ON "STORAGE" FOR EACH ROW

BEGIN

:NEW.storage\_id := storage\_SEQ.NEXTVAL;

END;

CREATE OR REPLACE TRIGGER storage\_order\_TRG BEFORE

INSERT ON "STORAGE\_ORDER" FOR EACH ROW

BEGIN

:NEW.storage\_order\_id := storage\_order\_SEQ.NEXTVAL;

END;

CREATE OR REPLACE TRIGGER storage\_order\_product\_TRG BEFORE

INSERT ON "STORAGE\_ORDER\_PRODUCT" FOR EACH ROW

BEGIN

:NEW.storage\_order\_product\_id := stor\_order\_prod\_SEQ.NEXTVAL;

END;

CREATE OR REPLACE TRIGGER storage\_product\_TRG BEFORE

INSERT ON "STORAGE\_PRODUCT" FOR EACH ROW

BEGIN

:NEW.storage\_product\_id := STORAGE\_PRODUCT\_SEQ.NEXTVAL;

END;

CREATE OR REPLACE TRIGGER store\_TRG BEFORE

INSERT ON "STORE" FOR EACH ROW

BEGIN

:NEW.store\_id := STORE\_SEQ.NEXTVAL;

END;

CREATE OR REPLACE TRIGGER user\_TRG BEFORE

INSERT ON "User" FOR EACH ROW

BEGIN

:NEW.user\_id := user\_SEQ.NEXTVAL;

END;

**ПРИЛОЖЕНИЕ В**

**Спецификация и тело пакета PKG\_Security**

create or replace Package Pkg\_Security Is

--- Заголовок авторизации

Function Authenticate\_User(p\_Login Char

,p\_Password Char) Return Boolean;

-----

Procedure Process\_Login(p\_Login Char

,p\_Password Char

,p\_App\_Id Number);

End Pkg\_Security;

create or replace Package Body Pkg\_Security Is

--- Авторизация

Function Authenticate\_User(p\_Login Char

,p\_Password Char) Return Boolean As

v\_Password "User".Password%Type;

v\_Status "User".Status%Type;

Begin

If p\_Login Is Null Or p\_Password Is Null Then

-- Write to Session, Notification must enter a username and password

Apex\_Util.Set\_Session\_State('LOGIN\_MESSAGE'

,'Please enter Username and password.');

Return False;

End If;

----

Begin

Select u.Password,

u.Status

Into v\_Password

,v\_Status

From "User" u

Where u.Login = p\_Login;

Exception

When No\_Data\_Found Then

-- Write to Session, User not found.

Apex\_Util.Set\_Session\_State('LOGIN\_MESSAGE'

,'User not found');

Return False;

End;

If v\_Password <> p\_Password Then

-- Write to Session, Password incorrect.

Apex\_Util.Set\_Session\_State('LOGIN\_MESSAGE'

,'Password incorrect');

Return False;

End If;

---

-- Write user information to Session.

--

Apex\_Util.Set\_Session\_State('SESSION\_USER\_NAME'

,p\_Login);

Apex\_Util.Set\_Session\_State('SESSION\_USER\_STATUS'

,v\_Status);

---

---

Return True;

End;

--------------------------------------

Procedure Process\_Login(p\_Login char

,p\_Password char

,p\_App\_Id Number) As

v\_Result Boolean := False;

Begin

v\_Result := Authenticate\_User(p\_Login

,p\_Password);

If v\_Result = True Then

-- Redirect to Page 1 (Home Page).

Wwv\_Flow\_Custom\_Auth\_Std.Post\_Login(p\_Login -- p\_User\_Name

,p\_Password -- p\_Password

,v('APP\_SESSION') -- p\_Session\_Id

,p\_App\_Id || ':1' -- p\_Flow\_page

);

Else

-- Login Failure, redirect to page 101 (Login Page).

Owa\_Util.Redirect\_Url('f?p=112:101:');

End If;

End;

End Pkg\_Security;