МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий

Кафедра Информационных систем и технологий

Специальность 1–40 05 01-03 Информационные системы и технологии (издательско-полиграфический комплекс)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**К КУРСОВОМУ ПРОЕКТУ НА ТЕМУ:**

База данных «Магазин товаров для творчества» с использованием технологии резервного копирования и восстановления

Выполнил студент Бобрик В.С.

(Ф.И.О.)

Руководитель проекта ассистент Колмаков М.В.

(учен. степень, звание, должность, Ф.И.О., подпись)

Заведующий кафедрой к.т.н., доц. Смелов В.В .

(учен. степень, звание, должность, Ф.И.О., подпись)

Курсовой проект защищен с оценкой

Минск 2021

**Реферат**

ФИО

Подпись

Дата

Лист

1

КП

Разраб.

Бобрик В.С.

Провер.

Колмаков М.В.

Н. контр.

Утверд.

Реферат

Лит.

Листов

1

БГТУ

Пояснительная записка курсового проекта содержит 43 страницы пояснительной записки, 45 иллюстраций, 5 источников литературы, 9 приложений.

ORACLE, ORACLE SQL DEVELOPER, ORACLE SQLPLUS.

Цель курсового проекта – разработка базы данных и приложения, которые обеспечат простое и удобное взаимодействие между собой, что позволит пользователю оформить заказ, отменить заказ, а также узнать информацию об интересующих товарах, их категориях, получить полную историю своих заказов.

В первой главе проводится аналитический обзор схожих приложений по тематике курсового проекта.

Вторая глава посвящена процессу проектирования системы и описание технологий, использованных во время выполнения проекта.

В третьей главе описано тестирование, проверка работоспособности и анализ данных системы.

В заключении приведены результаты проделанной работы.

Содержание

[Введение 5](#_Toc70781319)

[1. Обзор технических решений и литературных источников 6](#_Toc70781320)

[2. Проектирование и разработка базы данных 7](#_Toc70781321)

[2.1. Спецификация функциональных требований 7](#_Toc70781322)

[2.2. Разработка модели базы данных 8](#_Toc70781323)

[2.3. Пользователи 10](#_Toc70781324)

[2.4. Хранимые процедуры 11](#_Toc70781325)

[2.4.1. Пакет процедур пользователя 11](#_Toc70781326)

[2.4.2. Пакет администратора для работы с покупателями 11](#_Toc70781327)

[2.4.3. Пакет администратора для работы с заказами 12](#_Toc70781328)

[2.4.4. Пакет администратора для работы с товарами 13](#_Toc70781329)

[2.4.5. Пакет администратора для работы с пользователями 13](#_Toc70781330)

[2.4.6. Пакет для экспорта и импорта таблиц в формат xml 14](#_Toc70781331)

[2.5. Индексы 15](#_Toc70781332)

[2.6. Функции 16](#_Toc70781333)

[2.7. Триггеры 17](#_Toc70781334)

[2.8. Представления 17](#_Toc70781335)

[2.9. Реализация технологий 18](#_Toc70781336)

[2.9.1. Реализация технологии резервного копирования и восстановления данных в БД 18](#_Toc70781337)

[3. Тестирование, проверка работоспособности и анализ данных 23](#_Toc70781338)

[Заключение 29](#_Toc70781339)

[Список литературы 30](#_Toc70781340)

[Приложение А 31](#_Toc70781341)

[Приложение Б 32](#_Toc70781342)

[Приложение В 35](#_Toc70781343)

[Приложение Г 38](#_Toc70781344)

[Приложение Д 39](#_Toc70781345)

[Приложение E 40](#_Toc70781346)

[Приложение Ж 41](#_Toc70781347)

[Приложение З 42](#_Toc70781348)

[Приложение И 43](#_Toc70781349)

ФИО

Подпись

Дата

Лист

1

КП

Разраб.

Бобрик В.С.

Провер.

Колмаков М.В.

Н. контр.

Утверд.

Введение

Лит.

Листов

1

БГТУ

Введение

В данной записке приведено описание базы данных, разработанной в соответствии с заданием на курсовое проектирование по теме «Магазин товаров для творчества» по дисциплине «Администрирование баз данных и приложений».

Задачей данной курсовой работы является разработка базы данных, предназначенной для безотказного последующего использования в приложении «Магазин товаров для творчества». Приложение будет обладать следующим функционалом:

− авторизация и регистрация пользователей;

− разделение пользователя на администратора и пользователя;

− добавление и удаление маршрутов товаров;

− просмотр купленных товаров;

− разграничение товаров по категориям;

− просмотр доступных товаров;

− добавление товаров в корзину;

− просмотр и редактирование товаров в корзине;

− редактирование информации о покупателе.

Система управления базами данных (СУБД) — совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

Для разработки и управления базой данных курсового проекта использовалась система управления реляционными базами данных Oracle.

1. Обзор технических решений и литературных источников

В качестве аналога был выбран сайт [hobbyland.by](http://hobbyland.by), поскольку он максимально приближен к теме проекта и база данных будет проектироваться на основе функций данного сайта.

Итак, целью нашего проектирования будет создание универсальной базы данных для магазина товаров для творчества, которая будет предоставлять сервис аналогичный сайту, главная страница которого представлена на рисунке 1.1.

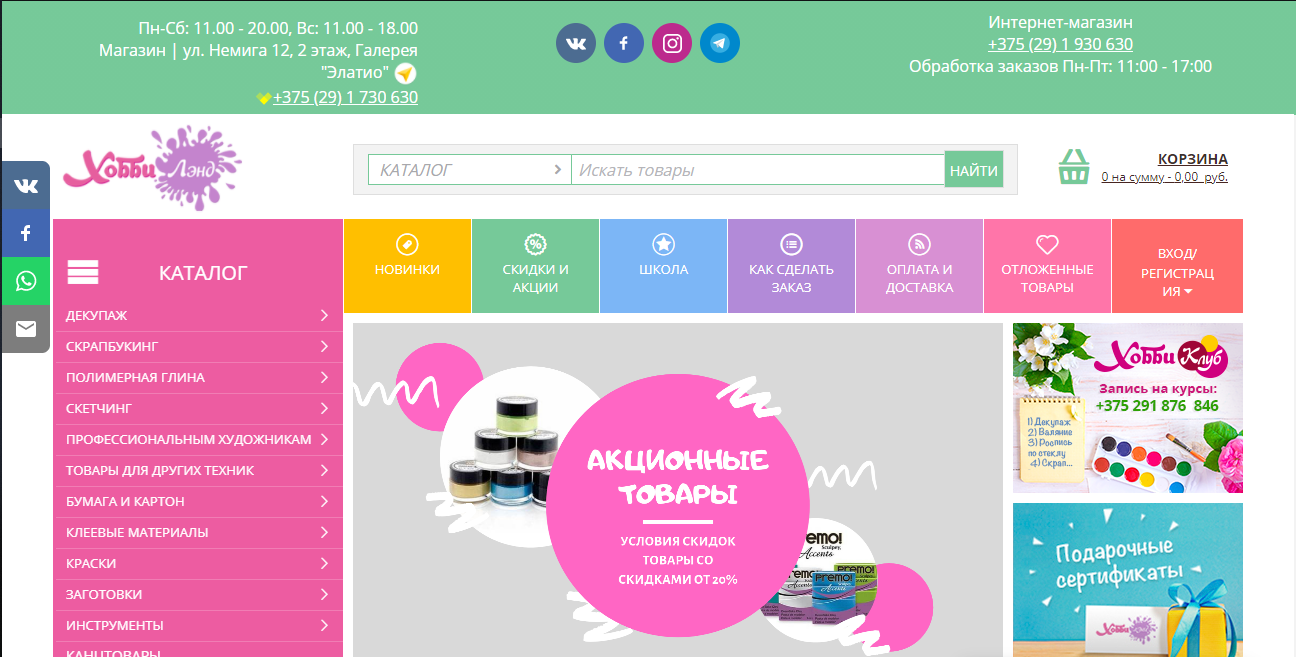


Рисунок 1.1 – Главная страница сайта hobbyland.by

Данный сайт имеет форму поиска товаров, добавления их в корзину и оформления заказов.

Наша задача спроектировать базу данных таким образом, чтобы весь функционал был полностью готов к задачам будущего приложения абсолютно любого магазина.

1. Проектирование и разработка базы данных

[ФИО](https://content.onliner.by/news/2015/09/large/2ef92fb5f20c27cad94594c5162a4dba.png)

[Подпись](https://content.onliner.by/news/2015/09/large/2ef92fb5f20c27cad94594c5162a4dba.png)

[Дата](https://content.onliner.by/news/2015/09/large/2ef92fb5f20c27cad94594c5162a4dba.png)

[Лист](https://content.onliner.by/news/2015/09/large/2ef92fb5f20c27cad94594c5162a4dba.png)

[1](https://content.onliner.by/news/2015/09/large/2ef92fb5f20c27cad94594c5162a4dba.png)

[КП](https://content.onliner.by/news/2015/09/large/2ef92fb5f20c27cad94594c5162a4dba.png)

[Разраб.](https://content.onliner.by/news/2015/09/large/2ef92fb5f20c27cad94594c5162a4dba.png)

[Бобрик В.С.](https://content.onliner.by/news/2015/09/large/2ef92fb5f20c27cad94594c5162a4dba.png)

[Провер.](https://content.onliner.by/news/2015/09/large/2ef92fb5f20c27cad94594c5162a4dba.png)

[Колмаков М.В.](https://content.onliner.by/news/2015/09/large/2ef92fb5f20c27cad94594c5162a4dba.png)

[Н. контр.](https://content.onliner.by/news/2015/09/large/2ef92fb5f20c27cad94594c5162a4dba.png)

[Утверд.](https://content.onliner.by/news/2015/09/large/2ef92fb5f20c27cad94594c5162a4dba.png)

[2.Проектирование и разработка базы данных](https://content.onliner.by/news/2015/09/large/2ef92fb5f20c27cad94594c5162a4dba.png)

[Лит.](https://content.onliner.by/news/2015/09/large/2ef92fb5f20c27cad94594c5162a4dba.png)

[Листов](https://content.onliner.by/news/2015/09/large/2ef92fb5f20c27cad94594c5162a4dba.png)

[1](https://content.onliner.by/news/2015/09/large/2ef92fb5f20c27cad94594c5162a4dba.png)

[БГТУ](https://content.onliner.by/news/2015/09/large/2ef92fb5f20c27cad94594c5162a4dba.png)

При разработке курсового проекта понадобились следующие объекты:

1. Таблицы;

2. Хранимые процедуры;

3. Индексы;

4. Функции;

5. Триггеры;

6. Представления;

7. Пакеты;

2.1. Спецификация функциональных требований

Рассмотрим возможности, которые предоставляет разработанное программное средство. Для наглядности демонстрации использовались UML -диаграммы.

UML – уникальный язык моделирования (Unified Modeling Language) – это система обозначений, которую можно применять для объектно-ориентированного анализа и проектирования. Его можно использовать для визуализации, спецификации, конструирования и документирования программных систем [2].

На рисунке 2.1.1 показана UML-диаграмма, отображающая возможности приложения с точки зрения неавторизованного пользователя.



Рисунок 2.1.1 – UML диаграмма вариантов использования для неавторизованного пользователя

Неавторизованный пользователь имеет только две возможности: авторизоваться и зарегистрироваться.

Диаграмма вариантов использования для администратора и пользователя отображена на рисунке 2.1.2.

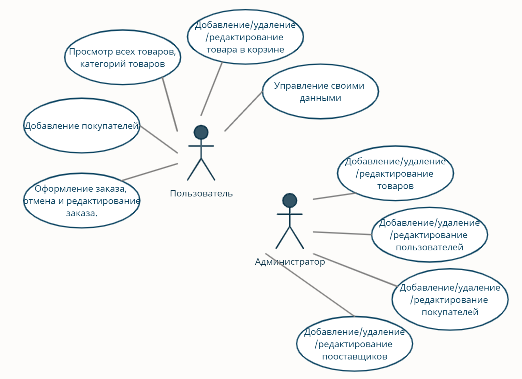


Рисунок 2.1.2 – UML-диаграмма вариантов использования для авторизованного пользователя и администратора

Авторизованный пользователь может управлять своими покупателями, осуществлять поиск товаров и добавление их в корзину, оформлять заказ, просматривать историю заказов, а также управление своими данными.

Также в приложение можно войти как администратор. Тогда открываются следующие возможности: добавление, удаление, изменение и просмотр всех таблиц.

* 1. Разработка модели базы данных

Для реализации базы данных для магазина товаров для творчества было разработано 15 таблиц. Диаграмма базы со структурой связей представлена в приложении А, а в приложении Б представлен скрипт создания БД.

Логически можно вывести 12 основных таблиц: Cart, Cities, Customers, DeliveryMethod, Orders, Order\_Details, PaymentMethod, ProductCategory, Products, Status, Suppliers, Users.

Таблица Cart представляет корзину и состоит из столбцов:

−Id\_Cart – идентификатор корзины, тип integer, первичный ключ;

− Product\_Id – идентификатор товара, тип integer; вторичный ключ;

− Сustomer\_Id – идентификатор покупателя, тип integer; вторичный ключ;

Таблица Cities представляет перечень городов и состоит из столбцов:

−Id – идентификатор города, тип integer, первичный ключ;

− City\_Name – название города, тип varchar2(300);

Таблица Customers представляет информацию о покупателях, состоит из столбцов:

−ID\_customer – идентификатор покупателя, тип integer, первичный ключ;

− First\_Name – имя покупателя, тип varchar(15);

− Middle\_Name – отчество покупателя, тип varchar(15);

− Last\_Name – фамилия покупателя, тип varchar(15);

− Gender – пол, тип varchar(2);

− Phone\_Number – номер телефона, тип number;

− Email – почта, тип varchar(25);

− User\_Id – идентификатора пользователя, тип integer, внешний ключ

− Active – активность покупателя, тип number(1)

Таблица DeliveryMethod представляет информацию о методах доставки, состоит из столбцов:

−Id\_delivery – идентификатор метода доставки, тип integer, первичный ключ;

− Delivery\_Method – метод доставки, тип varchar(15);

− Delivery\_Price – цена доставки, тип number;

Таблица Orders представляет информацию о заказах, состоит из столбцов:

−Id\_order – идентификатор заказа, тип integer, первичный ключ;

− Customer\_Id – идентификатор покупателя, тип integer, внешний ключ;

− DeliveryMethod\_Id – идентификатор доставки, тип integer, внешний ключ;

− PaymentMethod\_Id – идентификатор способа оплаты, тип integer, внешний ключ ключ;

− Address – адрес заказа, тип nvarchar2(40);

− City\_Id – идентификатор города, тип integer, внешний ключ ключ;

− Status\_Id – идентификатор статуса заказа, тип integer, внешний ключ ключ;

Таблица Order\_Details представляет информацию о деталях заказа, состоит из столбцов:

−Id\_order\_details – идентификатор деталей заказа, тип integer, первичный ключ;

− Order\_Id – идентификатор заказа, тип integer, внешний ключ;

− Count\_Of\_Product – количество продуктов, тип int;

− Product\_Id – идентификатор продукта, тип integer, внешний ключ;

− Active – активность заказа, тип number(1);

Таблица PaymentMethod представляет информацию о способах оплаты, состоит из столбцов:

−Id\_payment – идентификатор способа оплаты, тип integer, первичный ключ;

− PaymentMethod\_Name – способ оплаты, тип varchar(25);

Таблица ProductCategory представляет информацию о категориях продуктов, состоит из столбцов:

−ID\_category – идентификатор категории, тип integer, первичный ключ;

− Category\_Name – имя категории, тип nvarchar(25);

− Active – активность категории, тип number(1);

Таблица Products представляет информацию о товаре, состоит из столбцов:

−ID\_product – идентификатор товара, тип integer, первичный ключ;

− Product\_Name – название товара, тип nvarchar2(35);

− Price – цена товара, тип number;

− Count\_of\_Products – количество товара, тип number;

− Short\_Name – краткое название товара, тип varchar(10);

− Description – описание товара, тип varchar(30);

− Category\_Id – идентификатор категории, тип integer, внешний ключ;

− CanOrder – активность категории, тип number(1);

Таблица Status представляет информацию о статусе заказа, состоит из столбцов:

−Id\_status – идентификатор статуса, тип integer, первичный ключ;

− Status\_Name – название статуса, тип nvarchar2(25);

Таблица Suppliers представляет информацию с поставщиках товаров, состоит из столбцов:

−Id\_supplier – идентификатор поставщика, тип integer, первичный ключ;

− Name\_supplier – имя поставщика, тип varchar(30);

− Email – почта, тип varchar(25);

− Phone\_Number – номер телефона, тип number;

− City\_Id – идентификатор города, тип integer, внешний ключ;

− Active – активность поставщиков, тип number(1);

Таблица Users представляет информацию о пользователях, состоит из столбцов:

−Id\_user – идентификатор пользователя, тип integer, первичный ключ;

− Type\_of\_user – тип пользователя, тип varchar(15);

− Login – имя пользователя, тип varchar(15);

− Password – пароль, тип varchar(15);

− Active – активность пользователя, тип number(1);

* 1. Пользователи

Пользователь базы данных – это физическое или юридическое лицо, которое имеет доступ к БД и пользуется услугами информационной системы для получения информации.

При проектировании базы данных были использованы 2 пользователя. Первый пользователь – UserShop – обычный пользователь, имеет доступ для чтения таблиц, связанных с товарами, их категориями, заказами и корзиной, а также к изменению данных о своих покупателях, оформлению заказа и отмена этого заказа.

Второй пользователь – AdminShop– обладает более обширным перечнем прав. Имеет права для чтения, изменения и записи во всех таблицах.

* 1. Хранимые процедуры

Хранимая процедура – объект базы данных, представляющий собой набор SQL-инструкций, который компилируется один раз и хранится на сервере.

При разработке курсового проекта было создано 44 процедуры, которые были объединены в следующие пакеты:

1. Пакет пользователя;

2. Пакет администратора для работы с покупателями;

3. Пакет администратора для работы с заказами;

4. Пакет администратора для работы с деталями заказов;

5. Пакет администратора для работы с товарами;

6. Пакет администратора для работы с категориями товаров;

7. Пакет администратора для работы с поставщиками;

8. Пакет администратора для работы с пользователями;

9. Пакет администратора для работы с методами доставки, способами оплаты и статусами заказов;

10. Пакет администратора для экспорта и импорта таблиц в формат xml;

Весь перечень созданных пакетов с процедурами будет представлен в Приложении В.

* + 1. Пакет процедур пользователя

Для реализации функционала для пользователя были написаны следующие процедуры: Register\_User, LoginUser,AddOrder, RepealOrder, GetAllProducts, GetAllCategory, AddToCart, AddCustomer и другие.

Поскольку вывод у некоторых таблиц размером в 100000 строк занимает довольно много времени, было принято решение создать индексы, что позволило уменьшить время вывода таблицы до минимума.

* + 1. Пакет администратора для работы с покупателями

В данный пакет входят следующие процедуры: AddCustomer, DeleteCustomer, ShowInfoCustomer, GetAllCustomers.

При добавлении нового покупателя с помощью процедуры AddCustomer, ему необходимо заполнить все необходимые поля: FirstName, MiddleName, LastName, Gender, PhoneNumber, Email, UserId.

Пример данной процедуры представлен на рисунке 2.4.1

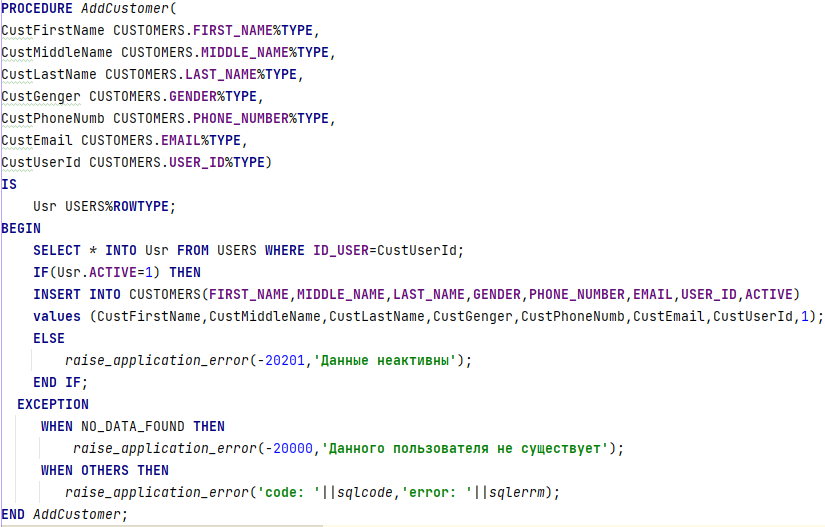


Рисунок 2.4.1 – Пример процедуры AddCustomer

* + 1. Пакет администратора для работы с заказами

В пакет входят следующие процедуры: AddOrder, GetHistoryOrdered, RepealOrder, UpdateStatusOrder.

Для того, чтобы оформить заказ, была разработана процедура AddOrder, которая принимает в себя параметры идентификатор покупателя, идентификатор способа оплаты и метода доставки, города, а также адрес.

Для того, чтобы отменить заказ, была разработана процедура RepealOrder, которая принимает в себя параметр идентификатора заказа и изменяет статус заказа на «Отменен»

Данная процедура представлена на рисунке 2.4.2.

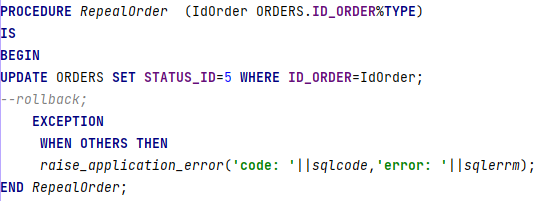


Рисунок 2.4.2 – Пример процедуры RepealOrder

Пакет администратора для работы с деталями заказов аналогичен пакету для работы с заказами.

* + 1. Пакет администратора для работы с товарами

В пакет входят следующие процедуры: AddProduct, DeleteProduct, GetAllProducts, GetAvaliableProducts,GetProductId и другие.

Удаление товара через процедуру DeleteProduct происходит путем смены значения поля CanOrder c 1 на 0.

Данная процедура представлена на рисунке 2.4.3.

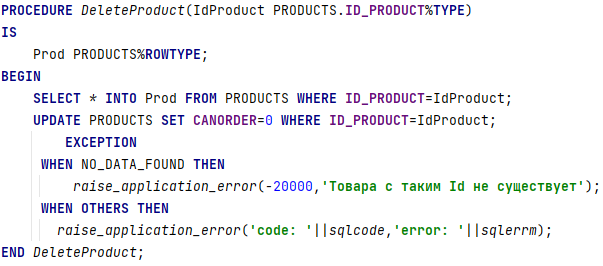


Рисунок 2.4.3 – Пример процедура DeleteProduct

Пакет администратора для работы с категориями товаров, пакет для работы с поставщиками, а также пакет для работы со статусом заказа, методами доставки и способами оплаты аналогичны пакету для работы с товарами.

* + 1. Пакет администратора для работы с пользователями

В пакет входят следующие процедуры: LoginUser, ShowInfoUser, UpdateLogin, UpdatePassword и другие.

Для изменения имени пользователя в процедуре UpdateLogin необходимо ввести идентификатор пользователя, для которого производится замена и новое значение логина.

Данная процедура представлена на рисунке 2.4.4.

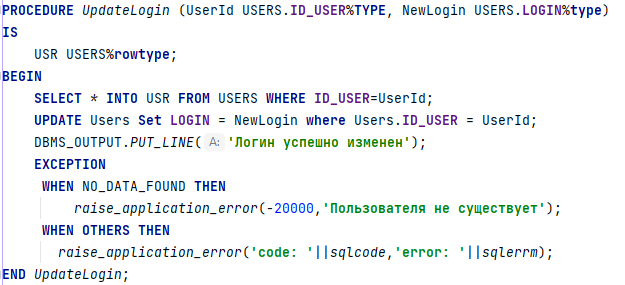


Рисунок 2.4.4 – Пример процедура UpdateLogin

* + 1. Пакет для экспорта и импорта таблиц в формат xml

Для экспорта таблиц в формате xml была разработана процедура EXPORT\_ORDERS\_TO\_XML. Осуществляется экспорт с помощью XMLTYPE, который дает возможность сообщить БД, что заносимый текст – это не просто строка, а строка документа XML

Процедура EXPORT\_ORDERS\_TO\_XML представлена ниже на рисунке 2.4.5. Отрывок из xml представлен в Приложении Г.

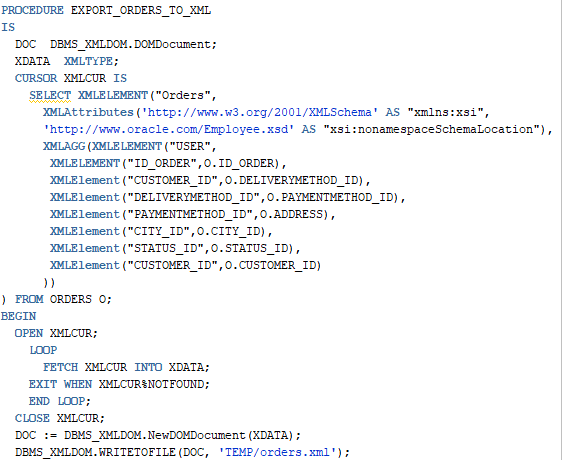


Рисунок 2.4.5 – Пример создания процедуры EXPORT\_ORDERS\_TO\_XML

Для импорта данных в таблицы, из файла формата xml, была разработана процедура ADDCITY которая импортирует данные о городах. Для получения xml файла и последующего разбора со вставкой используется функция XmlTable, которая позволяет перевести объект xml в разделенные поля. Процедура импорта представлена ниже на рисунке 2.4.6.

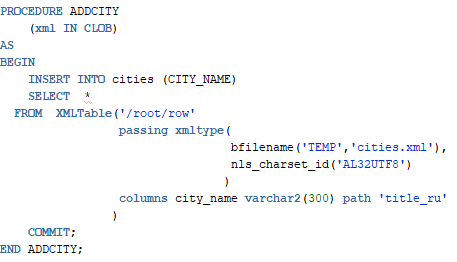


Рисунок 2.4.6 – Пример создания процедуры EXPORT\_XML

* 1. Индексы

Индекс – объект базы данных, создаваемый с целью повышения производительности поиска данных [4]. Таблицы в базе данных могут иметь большое количество строк, которые хранятся в произвольном порядке, и их поиск по заданному критерию путём последовательного просмотра таблицы строка за строкой может занимать много времени. В связи с необходимостью выборки таблиц, заполненных минимум на 100 000 строк, было принято решение разработать 4 индекса для таблиц: Orders, Order\_Details,History\_Orders.

Список индексов, созданных в рамках курсового проекта приведены ниже:

* Индекс index\_orders\_customers – индекс для поиска заказа по идентификатору покупателя;
* Индекс index\_orderDetails – индекс для поиска деталей заказа по идентификатору заказа;
* Индекс index\_orderDetails\_product – индекс для поиска деталей заказа по идентификатору продукта;
* Индекс index\_HistoryOrders\_CreateAt – индекс для поиска истории заказов по дате оформления заказа;

Для наглядной демонстрации повышения производительности поиска данных в таблице, ниже приведены рисунки с необходимой статистикой.

На рисунке 2.5.1 представлено время и число операций чтения необходимые для поиска данных в таблице Orders без индекса.



Рисунок 2.5.1 – Время и число операций чтения необходимые для поиска данных в таблице Orders без индекса

На рисунке 2.5.2 представлено время и число операций чтения необходимые для поиска данных в таблице Orders с индексом.



Рисунок 2.5.2 – Время и число операций чтения необходимые для поиска данных в таблице Orders с индексом

* 1. Функции

Аналогично функциям в языках программирования, определяемые пользователем функции Oracle представляют собой подпрограммы, которые принимают параметры, выполняют действие, например, сложные вычисления, и возвращают результат этого действия в виде значения.[4] Возвращаемое значение может быть либо единичным скалярным значением, либо результирующим набором. Поскольку некоторые процедуры содержат слишком много мало читаемого кода, было решено создать несколько функций. Список функций, созданных в рамках курсового проекта приведены ниже:

– GetAvaliableProducts – функция, предназначенная для подсчета актуального количества товаров

Входные параметры: идентификатор продукта;

Выходные параметры: количество товаров, тип int;

На рисунке 2.6.1 представлен пример реализации функции GetAvaliableProducts.

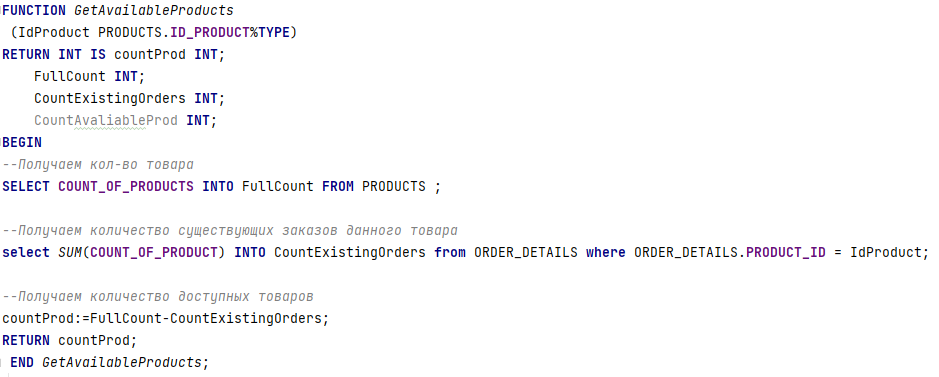


Рисунок 2.6.1 – Функция GetAvaliableProducts

– GetUserId – Функция, получающая идентификатор пользователя по идентификатору покупателя.

Входные параметры: идентификатор покупателя;

Выходные параметры: идентификатор пользователя, тип integer;

– GetProductId – Функция, получающая идентификатор продукта по его имени.

Входные параметры: название продукта;

Выходные параметры: идентификатор продукта, тип integer;

Функции будут представлены в Приложении Е.

* 1. Триггеры

Триггеры представляют специальный тип хранимой процедуры, которая вызывается автоматически при выполнении определенного действия над таблицей или представлением, в частности, при добавлении, изменении или удалении данных, то есть при выполнении команд INSERT, UPDATE, DELETE [4].

В моем случае было создано две таблицы HistoryOrders и HistoryUserAction, для регистрации определенных операций добавления или удаления. Сохранение действий пользователя, с целью ведения общей истории, для мониторинга регистрации пользователей и оформлением заказов играет важную роль. Триггеры будут представлены в Приложении Ж.

Фрагмент after-insert триггера будет представлен на рисунке 2.7.1

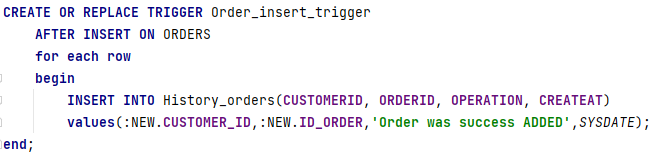


Рисунок 2.7.1 – Пример реализации after-триггера на вставку

* 1. Представления

Представление (VIEW) — объект базы данных, являющийся результатом выполнения запроса к базе данных, определенного с помощью оператора SELECT, в момент обращения к представлению [4]. Потребность в использовании представлений - удобство за счет автоматического выполнения таких действий как доступ к определенной части строк и/или столбцов.

С целью последующего вывода удобных для пользования данных было представление GetProductsOrderedCustomer, которое представлено на рисунке 2.8.1

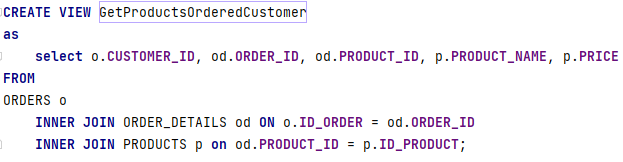


Рисунок 2.8.1 – Реализация представления GetProductsOrderedCustomer

Представление будет также представлено в Приложении З.

* 1. Реализация технологий

В процессе разработки проекта были использована технология: Резервное копирование и восстановление.

* + 1. Реализация технологии резервного копирования и восстановления данных в БД

Резервное копирование в Oracle можно осуществить с помощью встроенной утилиты RMAN.

RMAN поддерживает следующие типы резервного копирования:

* Полная резервная копия содержит все используемые блоки файлов данных.
* Инкрементное резервное копирование уровня 0 эквивалентен полному резервному копированию, который был отмечен как уровень 0.
* Совокупное инкрементное резервное копирование уровня 1 содержит только блоки, измененные начиная с последнего инкрементного резервного копирования уровня 0.
* Дифференциальное инкрементное копирование уровня 1 содержит только блоки, измененные начиная с последнего инкрементного резервного копирования.

Я выбрала для базы данных модели инкрементного резервного копирования 0 и 1 уровня, так как процедуры инкрементного резервного копирования проходят гораздо быстрее резервного копирования всей базы данных, потому что предусматривают копирование только тех блоков данных, которые изменились с момента последнего резервного копирования.

Процедура инкрементного резервного копирования на уровне 0 подразумевает копирование всех блоков данных, точно так же как и полное резервное копирование, и служит основой для последующих процедур инкрементного резервного копирования.

BACKUP INCREMENTAL LEVEL 0 DATABASE

Для выполнения процедуры инкрементного копирования на уровне 1 обязательно нужна исходная инкрементная резервная копия уровня 0.

BACKUP INCREMENTAL LEVEL 1 DATABASE

Код, представленный на рисунке 2.9.1 является реализацией инкрементного бэкапа 0 уровня.

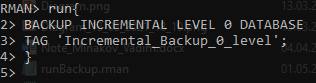


Рисунок 2.9.1 – Пример реализации инкрементного бэкапа 0 уровня

Результат выполнения бэкапа представлен на рисунке 2.9.2.

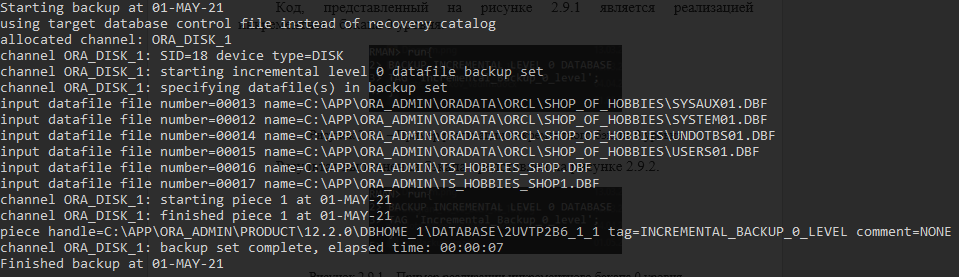


Рисунок 2.9.2 – Результат выполнения бэкапа 0 уровня

Код, представленный на рисунке 2.9.3 является реализацией инкрементного бэкапа 1 уровня.

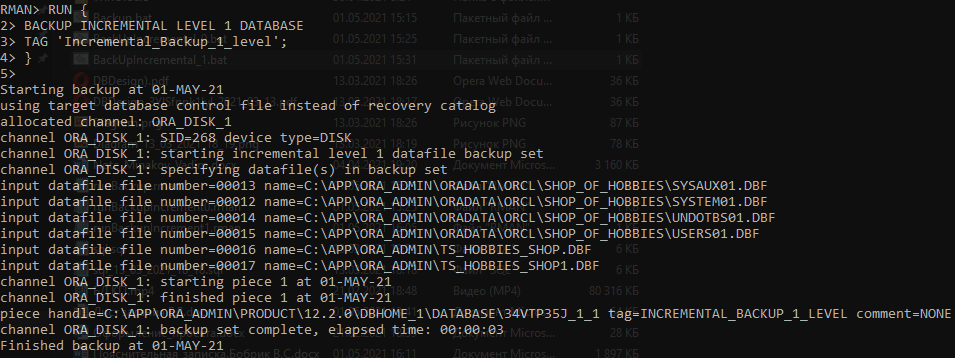


Рисунок 2.9.3 – Пример реализации инкрементного бэкапа 1 уровня

Co временем у клиента разрастается база данных до таких размеров, что создание полной резервной копии может занимать от 8 часов до нескольких дней. После перевода на инкрементное резервное копирование, сокращается время с 8 часов до 2-4 минут. Однако следует раз в неделю делать полную копию БД.

**План выполнения бэкапов.**

На основе вышеописанного составим наш план резервного копирования данных.  
 Мы будем делать следующие резервные копии:

* Инкрементная копия 0 уровня(полная), чаще чем раз в неделю нет необходимости
* Инкрементная копия 1 уровня, каждый день

Для создания расписания я воспользуюсь Планировщиком задач Windows, который позволит создать ряд заданий с составленным мною планом резервного копирования данных.

Создадим первую задачу, которая будет осуществлять инкрементный бэкап 0 уровня раз в неделю, она будет представлена на рисунке 2.9.4.

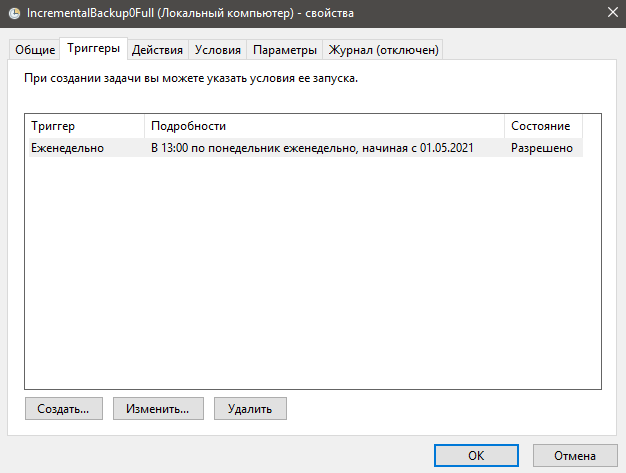


Рисунок 2.9.4 – Задача IncrementalBackup0Full

Она будет осуществлять запуск .bat файла, в котором находится команда запуска бэкапа (Рис.2.9.5).

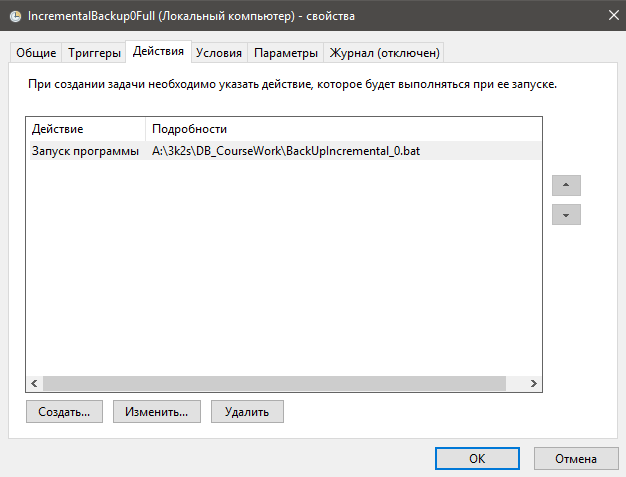


Рисунок 2.9.5 – Действие задачи IncrementalBackup0Full

Теперь приступим к создании второй задачи, которая будет осуществлять инкрементный бэкап 1 уровня каждый день, она будет представлена на рисунке 2.9.6.

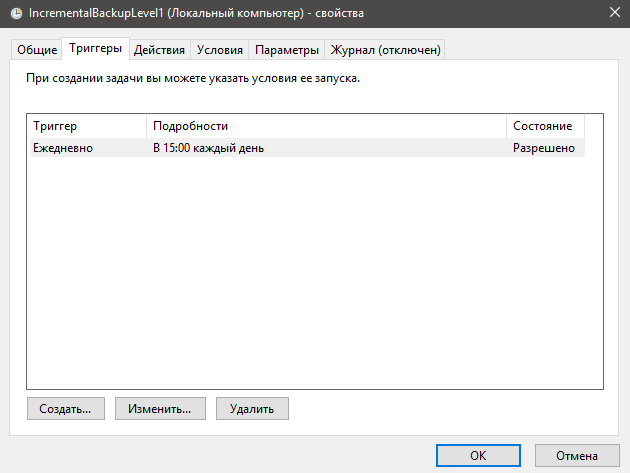


Рисунок 2.9.6 – Задача IncrementalBackup0Full

Она будет осуществлять запуск .bat файла, в котором находится команда запуска соответствующего бэкапа (Рис.2.9.7)

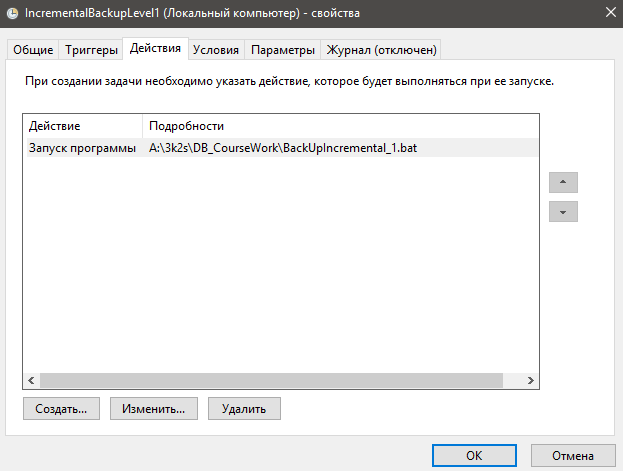


Рисунок 2.9.7 – Задача IncrementalBackup0Full

В результате получили 2 задачи представленные на рисунке 2.9.8



Рисунок 2.9.8 – Созданные задачи планирования бэкапов

Восстановление базы данных реализовано в файле RestoreDatabase.bat.

Код выполнения backup и restore представлен в Приложении И.

1. Тестирование, проверка работоспособности и анализ данных

ФИО

Подпись

Дата

Лист

1

КП

Разраб.

Бобрик В.С.

Провер.

Колмаков М.В.

Н. контр.

Утверд.

3 Тестирование, проверка работоспособности и анализ данных

Лит.

Листов

1

БГТУ

Разработанное программное средство полностью соответствует созданному для него техническому заданию, для проверки корректности работы программного средства, продемонстрируем добавление нового пользователя с помощью процедуры Register\_user на рисунке 3.1.



Рисунок 3.1 – Процедура регистрации нового пользователя

После выполнения процедуры пользователь заносится в систему с ролью обычного пользователя, это представлено на рисунке 3.2.

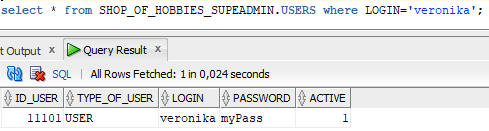


Рисунок 3.2 – Запись только что зарегистрированного пользователя

Также у пользователя, который есть в базе, есть возможность добавить покупателей, пример представлен на рисунке 3.3.



Рисунок 3.3 – Вызов процедуры добавления пользователя

После выполнения процедуры покупатель заносится в систему, это представлено на рисунке 3.4.

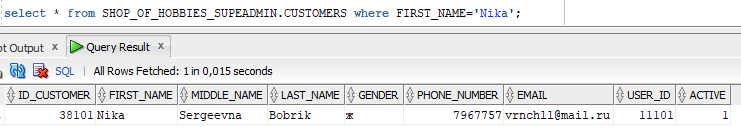


Рисунок 3.4 – Запись только что созданного покупателя

После добавления пользователя, мы можем получить информацию о нем с помощью вызова процедуры ShowInfoCustomer, пример ниже на рисунке 3.5.

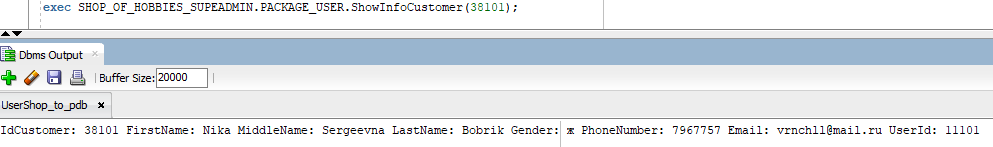


Рисунок 3.5 – Вызов процедуры ShowInfoCustomer

Пользователь может добавлять товар в корзину, пример ниже на рисунке 3.6.



Рисунок 3.6 – Вызов процедуры AddToCart

Результат добавления можно посмотреть с помощью процедуры ShowProductInCart, пример ниже на рисунке 3.7.

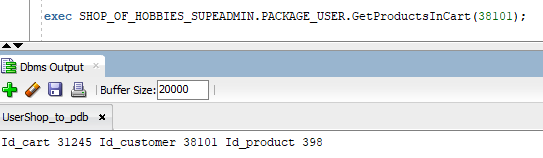


Рисунок 3.7 – Результат процедуры ShowProductInCart

Также пользователь может редактировать товары в корзине, пример ниже на рисунке 3.8.



Рисунок 3.8 – Вызов процедуры UpdateProdInCart

Результат изменения также можно посмотреть с помощью процедуры ShowProductInCart, пример ниже на рисунке 3.9.

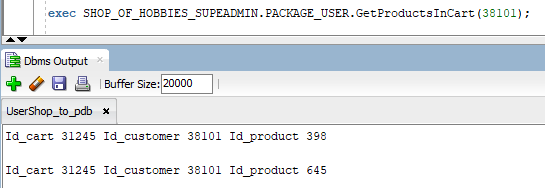


Рисунок 3.9 – Результат процедуры ShowProductInCart

Пользователь может просматривать доступные товары, пример ниже на рисунке 3.10.

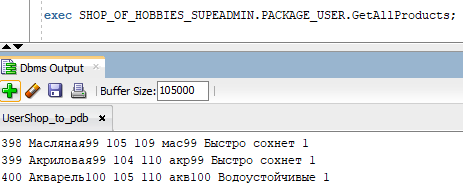


Рисунок 3.10 – Вызов процедуры GetAllProducts

Пользователь может изменять данные своих покупателей, пример ниже на рисунке 3.11.

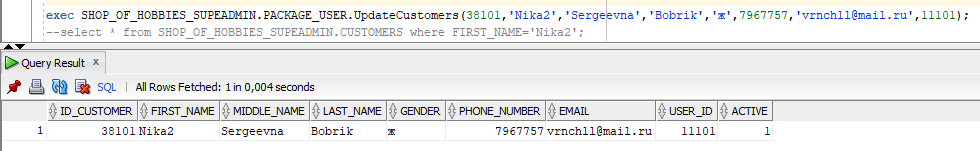


Рисунок 3.11 – Результат процедуры изменения данных у покупателя

Пользователь может оформить заказ с помощью соответствующей процедуры, пример ниже на рисунке 3.12.

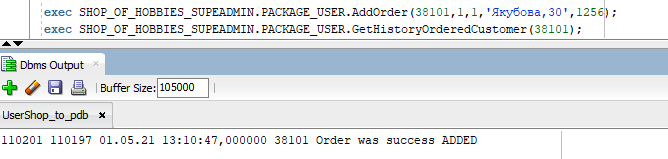


Рисунок 3.12 – Результат процедуры оформления заказа

Пользователь может отменить заказ с помощью соответствующей процедуры, пример ниже на рисунке 3.13.



Рисунок 3.13 – Результат процедуры отмены заказа

Пользователь может узнать идентификатор пользователя по идентификатору покупателя с помощью соответствующей функции, пример ниже на рисунке 3.14.

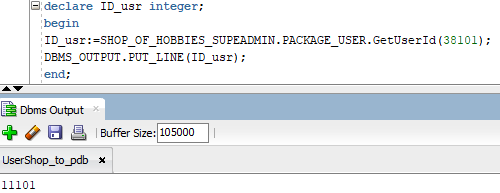


Рисунок 3.14 – Результат функции получения идентификатора пользователя

Что касаемо возможностей администратора, то в его возможности входит добавление и удаление товаров, пример ниже на рисунке 3.15.

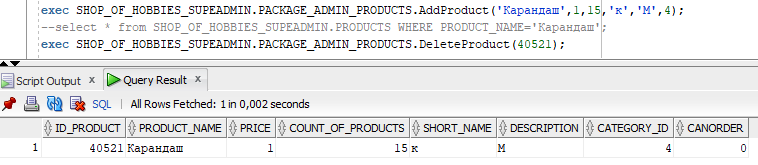


Рисунок 3.15 – Добавление и удаление нового товара

Администратор может получить информацию обо всех пользователях, пример ниже на рисунке 3.16.

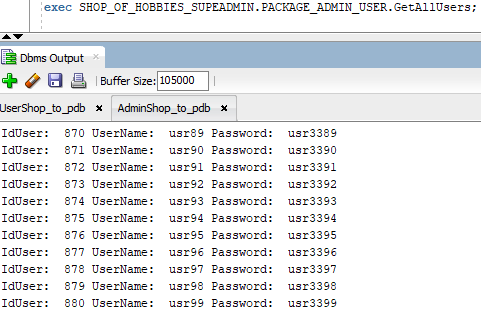


Рисунок 3.16 – Результат процедуры просмотра всех пользователей

Администратор также может получить информацию действий пользователей, пример ниже на рисунке 3.17.

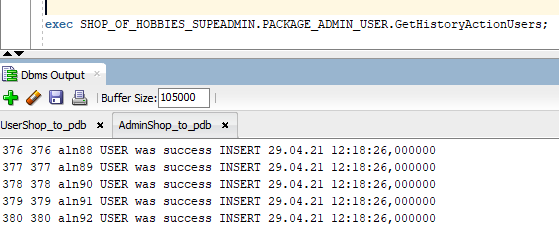


Рисунок 3.17 – Результат процедуры просмотра действий, связанных с пользователями

Администратор имеет возможности импорт файла cities.xml, который представлен на рисунке 3.18.

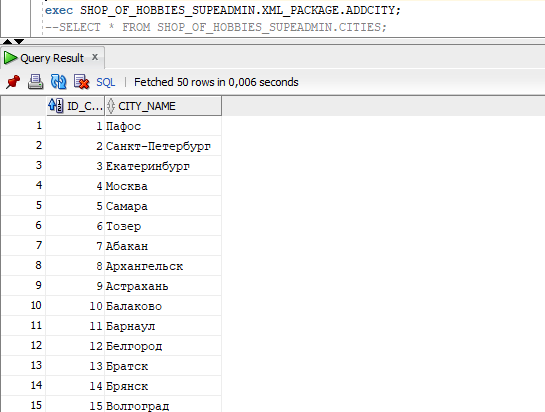


Рисунок 3.18 – Результат импорта файла cities.xml

Результат выполнения экспорта таблицы Orders представлен на рисунке 3.19.

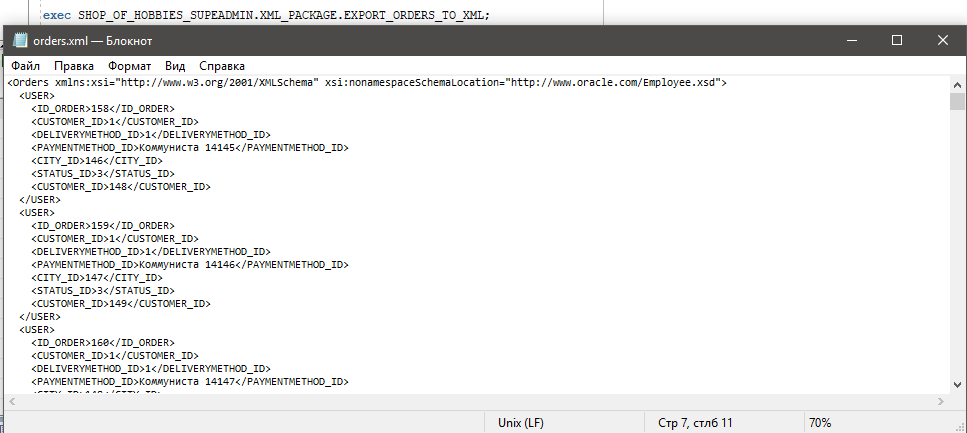


Рисунок 3.18 – Результат импорта файла cities.xml

Заключение

ФИО

Подпись

Дата

Лист

1

КП

Разраб.

Бобрик В.С.

Провер.

Колмаков М.В.

Н. контр.

Утверд.

Заключение

Лит.

Листов

1

БГТУ

В процессе решения задачи была достигнута поставленная цель по созданию базы данных «Магазин товаров для творчества», которая способна предоставить полный перечень сервисов по оформлению заказов, ведение базы товаров, категорий товаров, покупателей, поставщиков, а также городах. Основной целью курсового проекта стало проектирование базы данных для дальнейшей интеграции с приложением, которое помогло облегчить взаимодействие с базой данных посредством программного интерфейса. При разработке выполнены следующие пункты:

* Регистрация и авторизация;
* Разделение на администратора и пользователя;
* Добавление, удаление товаров;
* Просмотр купленных товаров;
* Разграничение товаров по категориям;
* Просмотр доступных товаров;
* Добавление товаров в корзину;
* Просмотр и редактирование товаров в корзине;
* Редактирование информации о покупателе;
* Технология резервного копирования и восстановления с реализацией собственного плана копирования;
* Импорт и экспорт XML.

Приложение прошло тестирование при использовании в БД большого количество данных.

Реализация нашей базы данных будет хорошим решением для любого магазина данного направления, поскольку позволяет охватить необходимый минимум требований для администрирования и использования.

В соответствии с полученным результатом работы программы можно сделать вывод, что разработанная программа работает верно, а требования технического задания выполнены в полном объёме.

Список литературы

ФИО

Подпись

Дата

Лист

1

КП

Разраб.

Бобрик В.С.

Провер.

Колмаков М.В.

Н. контр.

Утверд.

Список литературы

Лит.

Листов

1

БГТУ

1. Сайт hobbyland.by – <http://hobbyland.by/>
2. Интернет ресурс для проектирования и создания UML-диаграмм – <https://creately.com/ru/lp/инструмент-создания-диаграмм-UML-онлайн/>
3. Портал [Электронный ресурс] / Foundation, Inc – <https://oracle-dba.ru>
4. Портал [Электронный ресурс] / Foundation, Inc –

<https://docs.oracle.com/en/database/oracle/oracle-database/>

1. Портал [Электронный ресурс] / Foundation, Inc –

<https://oracle-patches.com/oracle/backup/3057-rman-утилита-recovery-manager-для-создания-backup-oracle>

Приложение А

ФИО

Подпись

Дата

Лист

1

КП

Разраб.

Бобрик В.С.

Провер.

Колмаков М.В.

Н. контр.

Утверд.

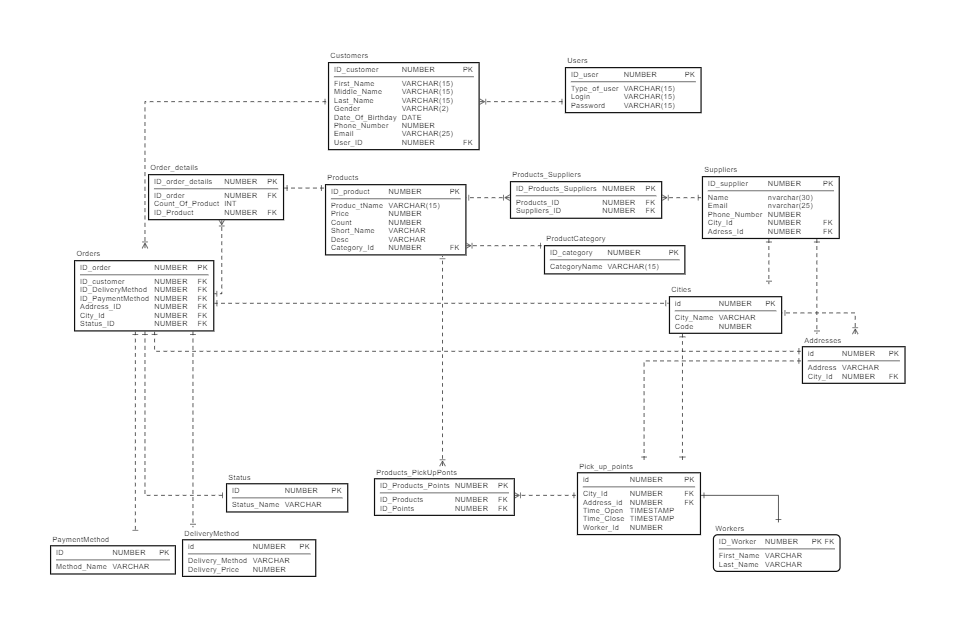
Приложение А

Лит.

Листов

1

БГТУ

****

Приложение Б

ФИО

Подпись

Дата

Лист

1

КП

Разраб.

Бобрик В.С.

Провер.

Колмаков М.В.

Н. контр.

Утверд.

Приложение Б

Лит.

Листов

1

БГТУ

CREATE TABLE Cities (

Id\_city INTEGER GENERATED BY DEFAULT AS IDENTITY,

City\_name VARCHAR2(300),

PRIMARY KEY (Id\_city));

CREATE TABLE DeliveryMethod (

Id\_delivery INTEGER GENERATED BY DEFAULT AS IDENTITY,

Delivery\_Method VARCHAR(15),

Delivery\_Price NUMBER,

PRIMARY KEY (Id\_delivery)

);

CREATE TABLE PaymentMethod (

Id\_payment INTEGER GENERATED BY DEFAULT AS IDENTITY,

PaymentMethod\_Name VARCHAR(25),

PRIMARY KEY (Id\_payment)

);

CREATE TABLE ProductCategory (

ID\_category INTEGER GENERATED BY DEFAULT AS IDENTITY,

Category\_Name NVARCHAR2(25) NOT NULL,

Active number(1),

PRIMARY KEY (ID\_category)

);

CREATE TABLE Status (

Id\_status INTEGER GENERATED BY DEFAULT AS IDENTITY,

Status\_Name NVARCHAR2(25),

PRIMARY KEY (Id\_status)

);

CREATE TABLE Users (

Id\_user INTEGER GENERATED BY DEFAULT AS IDENTITY,

Type\_of\_user VARCHAR(15),

Login VARCHAR(15),

Password VARCHAR(15),

Active number(1),

PRIMARY KEY (Id\_user)

);

CREATE TABLE Customers (

ID\_customer INTEGER GENERATED BY DEFAULT AS IDENTITY,

First\_Name VARCHAR(15),

Middle\_Name VARCHAR(15),

Last\_Name VARCHAR(15),

Gender VARCHAR(2),

Phone\_Number NUMBER,

Email VARCHAR(25),

User\_Id INTEGER,

Active number(1),

PRIMARY KEY (ID\_customer),

CONSTRAINT fk\_customers FOREIGN KEY (User\_Id) REFERENCES Users(Id\_user)

);

CREATE TABLE Products (

ID\_product INTEGER GENERATED BY DEFAULT AS IDENTITY,

Product\_Name NVARCHAR2(35),

Price NUMBER,

Count\_of\_Products NUMBER,

Short\_Name VARCHAR(10),

Description VARCHAR(30),

Category\_Id INTEGER,

CanOrder NUMBER(1),

PRIMARY KEY (ID\_product),

CONSTRAINT fk\_products FOREIGN KEY (Category\_Id) REFERENCES ProductCategory(Id\_category)

);

CREATE TABLE Orders (

Id\_order INTEGER GENERATED BY DEFAULT AS IDENTITY,

Customer\_Id INTEGER,

DeliveryMethod\_Id INTEGER,

PaymentMethod\_Id INTEGER,

Address nvarchar2(40),

City\_Id INTEGER,

Status\_Id INTEGER,

PRIMARY KEY(Id\_order),

CONSTRAINT fk\_orders\_customers FOREIGN KEY (Customer\_Id) REFERENCES Customers(Id\_customer),

CONSTRAINT fk\_orders\_status FOREIGN KEY (Status\_Id) REFERENCES Status(Id\_status),

CONSTRAINT fk\_orders\_deliveryMethod FOREIGN KEY (DeliveryMethod\_Id) REFERENCES DeliveryMethod(Id\_delivery),

CONSTRAINT fk\_orders\_paymentMethod FOREIGN KEY (PaymentMethod\_Id) REFERENCES PaymentMethod(Id\_payment),

CONSTRAINT fk\_orders\_cities FOREIGN KEY (City\_Id) REFERENCES Cities(Id\_city)

);

CREATE TABLE Suppliers (

Id\_supplier INTEGER GENERATED BY DEFAULT AS IDENTITY,

Name\_supplier VARCHAR(30) ,

Email VARCHAR(25),

Phone\_Number NUMBER,

City\_Id INTEGER,

Active number(1),

PRIMARY KEY(Id\_supplier),

CONSTRAINT fk\_suppliers\_cities FOREIGN KEY (City\_Id) REFERENCES Cities (Id\_city)

);

CREATE TABLE Order\_details (

Id\_order\_details INTEGER GENERATED BY DEFAULT AS IDENTITY,

Order\_Id INTEGER,

Count\_Of\_Product INT,

Product\_Id INTEGER,

Active number(1),

PRIMARY KEY (Id\_order\_details),

CONSTRAINT fk\_order\_details\_orders FOREIGN KEY (Order\_Id) REFERENCES Orders(Id\_order),

CONSTRAINT fk\_order\_details\_products FOREIGN KEY (Product\_Id) REFERENCES Products(Id\_product)

);

CREATE TABLE Products\_Suppliers (

ID\_Products\_Suppliers INTEGER GENERATED BY DEFAULT AS IDENTITY,

Products\_ID INTEGER,

Suppliers\_ID INTEGER,

PRIMARY KEY(ID\_Products\_Suppliers),

CONSTRAINT fk\_Products\_Suppliers\_products FOREIGN KEY (Products\_ID) REFERENCES Products(Id\_product),

CONSTRAINT fk\_Products\_Suppliers\_suppliers FOREIGN KEY (Suppliers\_ID) REFERENCES Suppliers(Id\_supplier)

);

CREATE TABLE Cart (

ID\_Cart INTEGER GENERATED BY DEFAULT AS IDENTITY,

ID\_Customer INTEGER,

ProductId INTEGER,

PRIMARY KEY(ID\_Cart),

CONSTRAINT fk\_Cart\_Customer FOREIGN KEY (ID\_Customer) REFERENCES Customers(ID\_customer),

CONSTRAINT fk\_Cart\_Product FOREIGN KEY (ProductId) REFERENCES PRODUCTS(ID\_product)

);

Приложение В

ФИО

Подпись

Дата

Лист

1

КП

Разраб.

Бобрик В.С.

Провер.

Колмаков М.В.

Н. контр.

Утверд.

Приложение В

Лит.

Листов

1

БГТУ

|  |  |  |
| --- | --- | --- |
| Название процедуры | Описание | Принимаемые параметры |
| *Register\_user* | Процедура регистрации пользователя | **USERS**.**LOGIN**%**TYPE**  **USERS**.**PASSWORD**%**TYPE** |
| *Register\_user\_admin* | Процедура регистрации администратора | **USERS**.**LOGIN**%**TYPE**  **USERS**.**PASSWORD**%**TYPE** |
| *LoginUser* | Процедура авторизации | **USERS**.**LOGIN**%**TYPE**;  **USERS**.**PASSWORD**%**TYPE** |
| *UpdatePassword* | Процедура изменения пароля | **USERS**.**ID\_USER**%**TYPE**  **USERS**.**Password**%**type** |
| *UpdateLogin* | Процедура изменения логина | **USERS**.**ID\_USER**%**TYPE**  **USERS**.**LOGIN**%**type** |
| *DeleteUser* | Процедура удаления пользователя | **USERS**.**ID\_USER**%**TYPE** |
| *ShowInfoUser* | Процедура вывода информации о пользователе | **USERS**.**ID\_USER**%**TYPE** |
| *GetAllUsers* | Процедура вывода всех пользователей | - |
| *GetHistoryActionUsers* | Процедура вывода истории пользователей | - |
| *AddOrder* | Процедура добавления заказа | ORDERS.**CUSTOMER\_ID**%**TYPE**  ORDERS.**DELIVERYMETHOD\_ID**%**TYPE**  ORDERS.**PAYMENTMETHOD\_ID**%**TYPE**  ORDERS.**ADDRESS**%**TYPE**  ORDERS.**CITY\_ID**%**TYPE**  ORDERS.**STATUS\_ID**%**TYPE** |
| *RepealOrder* | Процедура отмены заказа | ORDERS.**ID\_ORDER**%**TYPE** |
| *GetHistoryOrdered* | Процедура истории заказов | - |
| *UpdateStatusOrder* | Процедура изменения статуса заказа | ORDERS.**ID\_ORDER**%**TYPE**  ORDERS.**STATUS\_ID**%**type** |
| *AddProduct* | Процедура добавления товара | PRODUCTS.**PRODUCT\_NAME**%**TYPE**  PRODUCTS.**PRICE**%**TYPE**  PRODUCTS.**COUNT\_OF\_PRODUCTS**%**TYPE**  PRODUCTS.**SHORT\_NAME**%**TYPE**  PRODUCTS.**DESCRIPTION**%**TYPE**  PRODUCTS.**CATEGORY\_ID**%**TYPE** |
| *DeleteProduct* | Процедура удаления товара | PRODUCTS.**ID\_PRODUCT**%**TYPE** |
| *UpdateProduct* | Процедура изменения товара | PRODUCTS.**ID\_PRODUCT**%**TYPE**  PRODUCTS.**PRODUCT\_NAME**%**TYPE**  PRODUCTS.**PRICE**%**TYPE**  PRODUCTS.**COUNT\_OF\_PRODUCTS**%**TYPE**  PRODUCTS.**SHORT\_NAME**%**TYPE**  PRODUCTS.**DESCRIPTION**%**TYPE**  PRODUCTS.**CATEGORY\_ID**%**TYPE** |
| *UpdateProductPrice* | Процедура изменения цены товара | PRODUCTS.**ID\_PRODUCT**%**TYPE**  PRODUCTS.**PRICE**%**TYPE** |
| *GetAllProducts* | Процедура получения всех товаров | - |

|  |  |  |
| --- | --- | --- |
| *AddCategory* | Процедура добавления категории товара | PRODUCTCATEGORY.**CATEGORY\_NAME**%**TYPE** |
| *DeleteCategory* | Процедура удаления категории | PRODUCTCATEGORY.**ID\_CATEGORY**%**TYPE** |
| *UpdateCategory* | Процедура изменения категории | PRODUCTCATEGORY.**ID\_CATEGORY**%**TYPE**  PRODUCTCATEGORY.**CATEGORY\_NAME**%**TYPE** |
| *GetAllCategory* | Процедура получения всех категорий | - |
| *AddToCart* | Процедура добавления в корзину | CUSTOMERS.**ID\_CUSTOMER**%**TYPE**  PRODUCTS.**ID\_PRODUCT**%**TYPE** |
| *GetProductsInCart* | Процедура просмотра корзины | CUSTOMERS.**ID\_CUSTOMER**%**TYPE** |
| *UpdateProdInCart* | Процедура изменения корзины | CART.ID\_CART%**TYPE**  CART.PRODUCTID%**TYPE** |
| *DeleteProdFromCart* | Процедура удаления из корзины | CART.PRODUCTID%**TYPE**  CUSTOMERS.**ID\_CUSTOMER**%**TYPE** |
| *AddSupplier* | Процедура добавления поставщика | SUPPLIERS.**NAME\_SUPPLIER**%**TYPE**  SUPPLIERS.**EMAIL**%**TYPE**  SUPPLIERS.**PHONE\_NUMBER**%**TYPE**  SUPPLIERS.**CITY\_ID**%**TYPE** |
| *DeleteSupplier* | Процедура удаления поставщика | SUPPLIERS.**ID\_SUPPLIER**%**TYPE** |
| *AddStatus* | Процедура добавления статуса | STATUS.**STATUS\_NAME**%**TYPE** |
| *GetAllStatus* | Процедура получения всех статусов | **-** |
| *AddPayMethod* | Процедура добавления метода оплаты | PAYMENTMETHOD.PAY**\_NAME**%**TYPE** |
| *GetAllPayMethod* | Процедура получения всех способов оплаты | **-** |
| *AddDeliveryMethod* | Процедура добавления метода доставки | DELIVERYMETHOD.DELIVERY**\_NAME**%**TYPE**  DELIVERYMETHOD.**DELIVERY\_PRICE**%**TYPE** |
| *GetAllDeliveryMethod* | Процедура получения всех методов доставки | - |
| *AddOrderDetails* | Процедура добавления деталей заказа | ORDER\_DETAILS.**ORDER\_ID**%**TYPE**  ORDER\_DETAILS.**COUNT\_OF\_PRODUCT**%**TYPE**  ORDER\_DETAILS.**PRODUCT\_ID**%**TYPE** |
| *DeleteOrderDetails* | Процедура удаления деталей заказа | ORDER\_DETAILS.**ID\_ORDER\_DETAILS**%**TYPE** |
| *AddCustomer* | Процедура добавления покупателя | CUSTOMERS.**FIRST\_NAME**%**TYPE**  CUSTOMERS.**MIDDLE\_NAME**%**TYPE**  CUSTOMERS.**LAST\_NAME**%**TYPE**  CUSTOMERS.**GENDER**%**TYPE**  CUSTOMERS.**PHONE\_NUMBER**%**TYPE**  CUSTOMERS.**EMAIL**%**TYPE**  CUSTOMERS.**USER\_ID**%**TYPE** |
| *DeleteCustomers* | Процедура удаления покупателя | CUSTOMERS.**ID\_CUSTOMER**%**TYPE** |
| *ShowInfoСustomer* | Процедура получения информации о покупателе | CUSTOMERS.**ID\_CUSTOMER**%**TYPE** |
| *GetAllCustomers* | Процедура получения информации обо всех покупателях | - |
| *GetHistoryOrderedCustomer* | Процедура получения истории заказов покупателя | CUSTOMERS.**ID\_CUSTOMER**%**TYPE** |
| *ADDCITY* | Процедура импорта в xml | **xml IN CLOB** |
| *EXPORT\_ORDERS\_TO\_XML* | Процедура экспорта в xml | - |
| *UpdateCustomers* | Процедура обновления информации о покупателе | CUSTOMERS.ID%**TYPE**  CUSTOMERS.**FIRST\_NAME**%**TYPE**  CUSTOMERS.**MIDDLE\_NAME**%**TYPE**  CUSTOMERS.**LAST\_NAME**%**TYPE**  CUSTOMERS.**GENDER**%**TYPE**  CUSTOMERS.**PHONE\_NUMBER**%**TYPE**  CUSTOMERS.**EMAIL**%**TYPE**  CUSTOMERS.**USER\_ID**%**TYPE** |

Приложение Г

ФИО

Подпись

Дата

Лист

1

КП

Разраб.

Бобрик В.С.

Провер.

Колмаков М.В.

Н. контр.

Утверд.

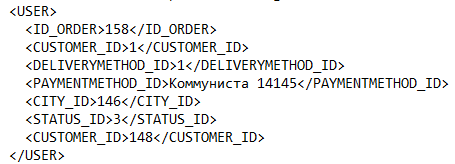
Приложение Г

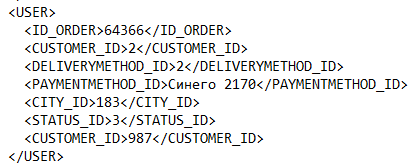
Лит.

Листов

1

БГТУ





Приложение Д

ФИО

Подпись

Дата

Лист

1

КП

Разраб.

Бобрик В.С.

Провер.

Колмаков М.В.

Н. контр.

Утверд.

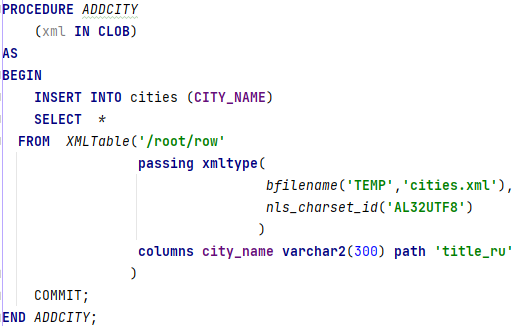
Приложение Д

Лит.

Листов

1

БГТУ



Приложение E

ФИО

Подпись

Дата

Лист

1

КП

Разраб.

Бобрик В.С.

Провер.

Колмаков М.В.

Н. контр.

Утверд.

Приложение Е

Лит.

Листов

1

БГТУ

*--Поиск id пользователя***CREATE OR REPLACE FUNCTION** *GetUserId* (CustId CUSTOMERS.**ID\_CUSTOMER**%**TYPE**)  
**RETURN INTEGER IS Id INTEGER**;  
**BEGIN  
SELECT USER\_ID INTO Id FROM** CUSTOMERS **WHERE** CustId = **ID\_CUSTOMER**;  
**RETURN Id**;  
**EXCEPTION  
 when others then return** -1;  
**END**;  
  
*--Поиск id товара по имени--***CREATE OR REPLACE FUNCTION** *GetProductId* (ProName PRODUCTS.**PRODUCT\_NAME**%**TYPE**)  
**RETURN INTEGER IS Id INTEGER**;  
**BEGIN  
SELECT ID\_PRODUCT INTO Id FROM** PRODUCTS **WHERE PRODUCT\_NAME** = ProName;  
**RETURN Id**;  
**EXCEPTION  
 when others then return** -1;  
**END**;  
  
*-- --Получение количества доступного товара***CREATE OR REPLACE FUNCTION** *GetAvailableProducts* (IdProduct PRODUCTS.**ID\_PRODUCT**%**TYPE**)  
**RETURN INT IS** countProd **INT**;  
 FullCount **INT**;  
 CountExistingOrders **INT**;  
 CountAvaliableProd **INT**;  
**BEGIN***--Получаем кол-во товара***SELECT COUNT\_OF\_PRODUCTS INTO** FullCount **FROM** PRODUCTS **where ID\_PRODUCT**=IdProduct;  
*--Получаем количество существующих заказов данного товара***select** *SUM*(**COUNT\_OF\_PRODUCT**) **INTO** CountExistingOrders **from** ORDER\_DETAILS **where** ORDER\_DETAILS.**PRODUCT\_ID** = IdProduct;  
*--Получаем количество доступных товаров*countProd:=FullCount-CountExistingOrders;  
**RETURN** countProd;  
 **END**;

Приложение Ж

ФИО

Подпись

Дата

Лист

1

КП

Разраб.

Бобрик В.С.

Провер.

Колмаков М.В.

Н. контр.

Утверд.

Приложение Ж

Лит.

Листов

1

БГТУ

*--Триггер на оформление заказа***CREATE OR REPLACE TRIGGER** Order\_insert\_trigger  
 **AFTER INSERT ON** ORDERS  
 **for each row  
 begin  
 INSERT INTO** History\_orders(**CUSTOMERID**, **ORDERID**, **OPERATION**, **CREATEAT**)  
 **values**(:**NEW**.**CUSTOMER\_ID**,:**NEW**.**ID\_ORDER**,**'Order was success ADDED'**,*SYSDATE*);  
**end**;  
 *--Триггер на отмену заказа***CREATE OR REPLACE TRIGGER** Order\_repeat\_trigger  
 **AFTER UPDATE ON** ORDERS  
 **declare** ORD ORDERS%**ROWTYPE**;  
 **CURSOR** trig\_del\_order **is select** *\** **from** ORDERS;  
 **begin  
 OPEN** trig\_del\_order;  
 **FETCH** trig\_del\_order **INTO** ORD;  
 **WHILE** trig\_del\_order%**FOUND  
 LOOP  
 IF**(ORD.**STATUS\_ID**=5) **THEN  
 INSERT INTO** History\_orders(**CUSTOMERID**, **ORDERID**, **OPERATION**, **CREATEAT**)  
 **values**(ORD.**CUSTOMER\_ID**,ORD.**ID\_ORDER**,**'USER was success DELETE'**,*SYSDATE*);  
 **END IF**;  
 **FETCH** trig\_del\_order **INTO** ORD;  
 **END LOOP**;  
**end**;  
*--  
--Триггер на регистрацию пользователя***CREATE OR REPLACE TRIGGER** USER\_REGISTER\_TRIGGER  
 **AFTER INSERT ON USERS  
 for each row  
 begin  
 INSERT INTO** History\_User\_Action(**USERID**, **USERNAME**, **OPERATION**, **CREATEAT**)  
 **values**(:**NEW**.**ID\_USER**,:**NEW**.**LOGIN**,**'USER was success INSERT'**,*SYSDATE*);  
**end**;  
*--Триггер на удаление пользователя***CREATE OR REPLACE TRIGGER** User\_delete\_trigger  
 **AFTER UPDATE ON USERS  
 declare** USR **USERS**%**ROWTYPE**;  
 **CURSOR** trig\_del\_user **is select** *\** **from USERS**;  
 **begin  
 OPEN** trig\_del\_user;  
 **FETCH** trig\_del\_user **INTO** USR;  
 **WHILE** trig\_del\_user%**FOUND  
 LOOP  
 IF**(USR.**ACTIVE**=0) **THEN  
 INSERT INTO** History\_User\_Action(**USERID**, **USERNAME**, **OPERATION**, **CREATEAT**)  
 **values**(USR.**ID\_USER**,USR.**LOGIN**,**'USER was success DELETE'**,*SYSDATE*);  
 **END IF**;  
 **FETCH** trig\_del\_user **INTO** USR;  
 **END LOOP**;  
**end**;

Приложение З

ФИО

Подпись

Дата

Лист

1

КП

Разраб.

Бобрик В.С.

Провер.

Колмаков М.В.

Н. контр.

Утверд.

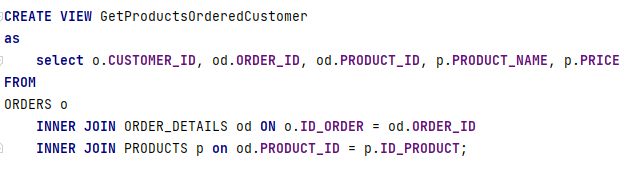
Приложение З

Лит.

Листов

1

БГТУ



Приложение И

ФИО

Подпись

Дата

Лист

1

КП

Разраб.

Бобрик В.С.

Провер.

Колмаков М.В.

Н. контр.

Утверд.

Приложение И

Лит.

Листов

1

БГТУ

run{

restore database;

recover database;

}

run{

BACKUP INCREMENTAL LEVEL 0 DATABASE

TAG 'Incremental\_Backup\_0\_level';

}

RUN {

BACKUP INCREMENTAL LEVEL 1 DATABASE

TAG 'Incremental\_Backup\_1\_level';

}