



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 "Туристическое агентство"

Студент ИУ6-41
(Группа)

(Подпись, дата)

Гринберг Х.Э.
(И.О.Фамилия)

Студент ИУ6-41
(Группа)

(Подпись, дата)

Иванов Д.В.
(И.О.Фамилия)

Руководитель курсового проекта

(Подпись, дата)

Скворцова М.А
(И.О.Фамилия)

2017 г.

УТВЕРЖДАЮ

Заведующий кафедрой ИУ6

А.В. Пролетарский

« ____ » _____ 2017 г.

ЗАДАНИЕ на выполнение курсовой работы

по дисциплине **Базы данных**

Студент группы ИУ6-41

Гринберг Ханна Эдуардовна, Иванов Дмитрий Вячеславович
(Фамилия, имя, отчество)

Тема курсовой работы

База Данных «Туристическое агентство»

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 “Туристическое агентство”, состоящую минимум из 5 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 «10 » февраля 2017 г.

Руководитель курсовой работы

(Подпись, дата)

М.А. Скворцова

(И.О.Фамилия)

Студент

(Подпись, дата)

Х.Э. Гринберг

(И.О.Фамилия)

Студент

(Подпись, дата)

Д.В. Иванов

(И.О.Фамилия)

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

РЕФЕРАТ

Расчетно-пояснительная записка 38 с., 16 рис., 8 табл., 7 источников.

Объектом разработки является база данных для туристического агентства и пользовательский интерфейс для неё.

Цель работы — создание удобного и доступного интерфейса для базы данных туристического агентства.

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 клиентов и администрации туристической фирмы за счет формирования заказа, просмотра информации по рейсам.

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

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	5
1 Анализ и проектирование базы данных.....	6
1.1 Исследование предметной области.....	6
1.2 Создание модели в Oracle SQL Developer Data Modeler.....	7
2 Формирование базы данных.....	13
2.1 Создание пользователя для базы данных.....	13
2.2 Перенос базы данных в Oracle Database SQL Developer.....	13
2.3 Добавление данных в таблицы.....	14
3 Создание интерфейса в среде Oracle APEX версии 5.1.....	15
3.1 Создание страницы авторизации.....	15
3.2 Создание страницы регистрации.....	19
3.3 Создание страницы бронирования.....	21
3.4 Создание отчета для просмотра личного бронирования для пользователя.....	23
3.5 Создание отчета для просмотра данных постояльцев администратором.....	25
ЗАКЛЮЧЕНИЕ.....	29
СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ.....	30
Приложение А.....	31

ВВЕДЕНИЕ

В данной работе производится создание базы данных для туристического агентства и пользовательского интерфейса для неё. Она предназначена для формирования заказов. Интерфейс предоставляет возможность пользователю просмотра информации о заказе,

планирование маршрута, а также заполнения личных данных для покупки тура, а также заполнения необходимых документов.

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

Интерфейс разрабатывался в соответствии со следующими требованиями:

- наличие пользователя-администратора и пользователя-клиента;
- простота и понятность обычному пользователю;
- возможность просмотра личных данных пользователем-клиентом;
- возможность просмотра списка всех клиентов пользователем-

администратором;

- лаконичность интерфейса.

На данный момент существует множество аналогов такой базы данных. Можно найти множество различных интерфейсов для заказа туров и дополнительных услуг в интернете.

При разработке интерфейса для базы данных "Туристическое агентство" мы постарались учесть 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.

База данных «Туристическое агентство» предназначена для упрощения обработки информации о клиентах, которые хотят поехать отдыхать, отелях, городах и турах, о сотрудниках, которые предоставляют услуги турагентства и заключают договор.

Эта база данных предназначена для менеджеров (сотрудников фирмы), которые вносят 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 маршрутах
4. Хранение информации о предлагаемых отелях

Также было принято решение о необходимости наличия стоимости каждого типа номера и типа питания. Такой учёт должен облегчить процесс формирования общей стоимости тура для каждого клиента.

Данные требования к разрабатываемой базе данных были учтены в дальнейшем при создании модели, а также доработаны в соответствии с возникающими проблемами при проектировании.[7]

После рассмотрения предметной области было решено создать тринадцать таблиц: «Заказ», «Клиент», «Документ», «Скидка», «Рейс», «Авиакомпания», «Город», «Аэропорт», «Отель», «Питание», «Размещение», «Оператор», «Социальный статус».

Схема связи таблиц представлена на рисунке 1.



Рисунок 1 - Схема связи таблиц

1.2 Создание модели в Oracle SQL Developer Data Modeler

После исследования предметной области мы приступили к созданию модели в Oracle SQL Developer Data Modeler. [5][7]

Т.к. в базе данных необходимо формирование тура, была создана таблица «Заказ». Она будет содержать информацию о выбранной гостинице, типе питания, типе размещения, полученные из таблиц «Отель», «Питание», «Размещение», а также данные клиента и оператора, которые будут получены из таблиц «Клиент» и «Оператор». Также данная таблица будет содержать информацию о выбранном рейсе, данные которое получены из таблицы «Рейс».

Для хранения информации о клиентах необходимо наличие таблицы «Клиент», которая будет содержать контактную информацию пользователя, его ФИО и паспортные данные, социальный статус, дату рождения, телефон и электронную почту. Для полей тип документа и социальный статус используются данные из таблиц «Документы» и «Социальный статус».

Аналогично таблице «Клиент» была создана таблица «Оператор», где хранится все информация об операторе, которые составляет заказ. В данной таблице рассматриваются такие данные как ФИО, паспортные данные, регистрационный номер, а также адрес и электронная почта. Тип документа выбирается из таблицы «Документы».

Для таблиц «Клиент» и «Оператор» используется таблица «Документы», в который указывается тип документа, который возможно использовать при формировании заказа.

Для определения скидки была создана таблица «Скидка». Сумма скидки зависит от социального статуса, поэтому информацию о социальном статусе мы берем из таблицы «Социальный статус».

Для формирования заказа нам также необходима информация о рейсах, которыми мы можем воспользоваться. Все информация расположена в таблице «Рейс». Данная таблица предоставляет данные о номере рейса, городе вылета и городе прилета, дате полета и авиакомпании, которая осуществляет перевозку. Все необходимые данные по авиакомпании и аэропортах берутся из таблицы «Аэропорт» и «Авиакомпания».

Таблица «Аэропорт» содержит данные о названии аэропорта, его сокращенной аббревиатуре IATA, а также о городе, в котором расположен аэропорт или самом ближайшем городе. Информация о городе получается из таблицы «Город».

По нашему мнению, таблица «Город» необходима в нашей базе данных, так как в ней хранится информации в какой стране расположен город.

Для формирования таблицы «Рейс» используются данные из таблицы «Авиакомпания», где содержится список авиакомпаний, которыми мы можем воспользоваться.

Для получение более подробной информации о заказе нам необходимо знать информацию об отелях, в которых клиент может разместиться. Данные об отеле: название и тип отеля, расположены в таблице «Отель».

В таблице «Размещение» предоставлена информация о возможных вариантах номера, в который гости при желании могут въехать. В данной таблице находятся данные о типе номера, максимальном количестве человек, которые могут в ней проживать, а также о ее цене.

Наша фирма также предоставляет возможность выбрать тип питания. Информацию мы решили хранить в таблице «Питание». Также в данной таблице есть информация о цене на каждый из перечисленных типов питания.

Для предоставления скидка в зависимости от социально статуса было решено создать 2 таблицы: «Социальный статус», в который занесены все возможные социальные статусы, и «Скидка», где предоставлена вся информация по возможным скидкам в зависимости от социального статуса

клиента.

Модель разработанной базы данных "Туристическое агентство" представлена на рисунке

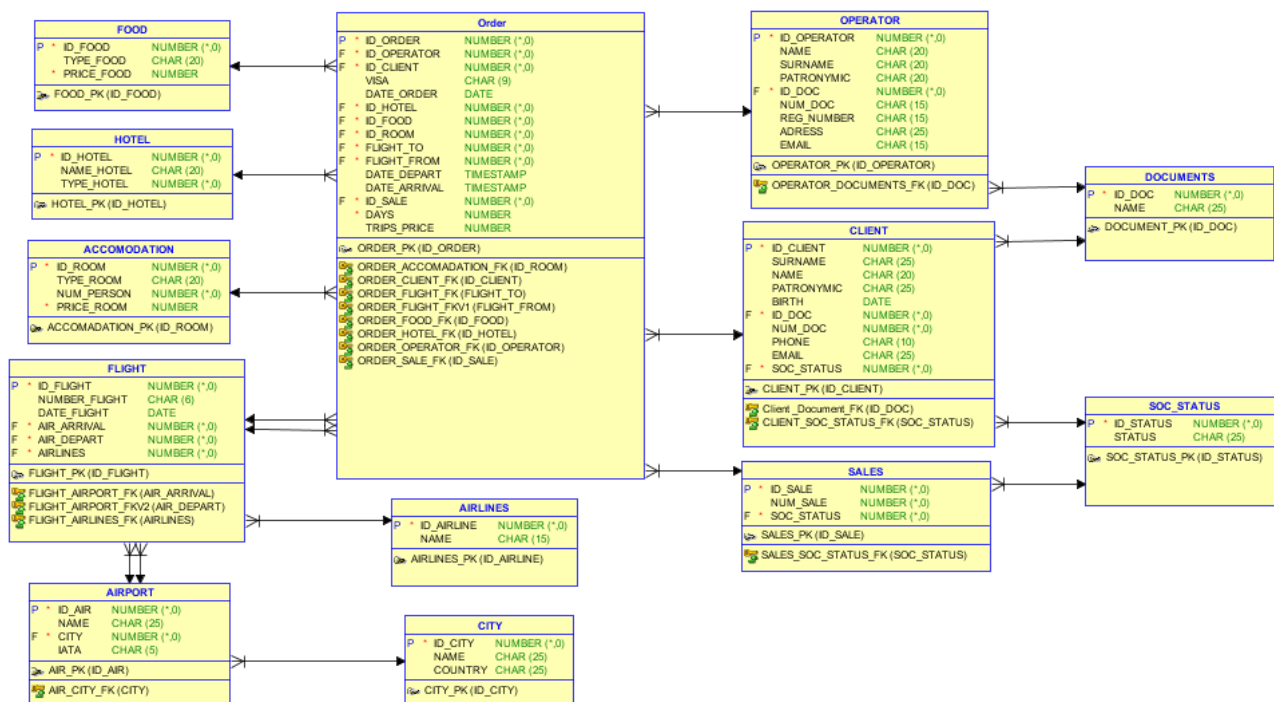


Рисунок 2 - Модель разработанной базы данных

Таблицей называется элемент, который представлен на рисунке 3.

1
2
3

Рисунок 3 - 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.

На данном рисунке присутствует множество стрелок, соединяющих таблицы.

Стрелка, соединяющая таблицы «Заказ» и «Клиент», является идентификацией первичного ключа «ID_CLIENT» таблицы «Клиент» и внешнего ключа «ID_CLIENT» таблицы «Заказ».

Стрелка, соединяющая таблицы «Заказ» и «Оператор», является идентификацией первичного ключа «ID_OPERATOR» таблицы «Оператор» и внешнего ключа «ID_OPERATOR» таблицы «Заказ».

Стрелка, соединяющая таблицы «Заказ» и «Питание», является идентификацией первичного ключа «ID_FOOD» таблицы «Питание» и внешнего ключа «ID_FOOD» таблицы «Заказ».

Стрелка, соединяющая таблицы «Заказ» и «Размещение», является идентификацией первичного ключа «ID_ROOM» таблицы «Размещение» и внешнего ключа «ID_ROOM» таблицы «Заказ».

Стрелка, соединяющая таблицы «Заказ» и «Отель», является идентификацией первичного ключа «ID_HOTEL» таблицы «Отель» и внешнего ключа «ID_HOTEL» таблицы «Заказ».

Стрелка, соединяющая таблицы «Заказ» и «Рейс», является идентификацией первичного ключа «ID_FLIGHT» таблицы «Рейс» и внешнего ключа «FLIGHT_TO» таблицы «Заказ», а также идентификацией первичного ключа «ID_FLIGHT» таблицы «Рейс» и внешнего ключа «FLIGHT_FROM» таблицы «Заказ»,

Стрелка, соединяющая таблицы «Заказ» и «Скидка», является идентификацией первичного ключа «ID_SALE» таблицы «Скидка» и внешнего ключа «ID_SALE» таблицы «Заказ».

Стрелка, соединяющая таблицы «Клиент» и «Документы», является идентификацией первичного ключа «ID_DOC» таблицы «Документы» и внешнего ключа «ID_DOC» таблицы «Клиент».

Стрелка, соединяющая таблицы «Оператор» и «Документы», является

идентификацией первичного ключа «ID_DOC» таблицы «Документы» и внешнего ключа «ID_DOC» таблицы «Оператор».

Стрелка, соединяющая таблицы «Рейс» и «Авиакомпания», является идентификацией первичного ключа «ID_AIRLINE» таблицы «Авиакомпания» и внешнего ключа «AIRLINES» таблицы «Рейс».

Стрелка, соединяющая таблицы «Рейс» и «Аэропорт», является идентификацией первичного ключа «ID_AIR» таблицы «Аэропорт» и внешнего ключа «AIR_ARRIVAL» таблицы «Рейс», а также идентификацией первичного ключа «ID_AIR» таблицы «Аэропорт» и внешнего ключа «AIR_DEPARTURE» таблицы «Рейс».

Стрелка, соединяющая таблицы «Аэропорт» и «Город», является идентификацией первичного ключа «ID_CITY» таблицы «Город» и внешнего ключа «CITY» таблицы «Аэропорт».

Стрелка, соединяющая таблицы «Клиент» и «Социальный статус», является идентификацией первичного ключа «ID_STATUS» таблицы «Социальный статус» и внешнего ключа «SOC_STATUS» таблицы «Клиент».

Стрелка, соединяющая таблицы «Скидка» и «Социальный статус», является идентификацией первичного ключа «ID_STATUS» таблицы «Социальный статус» и внешнего ключа «SOC_STATUS» таблицы «Скидка».

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.

Было принято решение наделить пользователя следующими правами доступа:

- возможность создания таблиц;
- возможность создания процедур;
- возможность создания триггеров;
- возможность создания представлений;
- возможность создания счетчиков;

А также пользователь имеет возможность удалять и изменять процедуры, триггеры и таблицы.

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.[1][6] SQL-код, сгенерированный SQL Data Modeler, и результат переноса представлены в приложении А.

При переносе возникли ошибки из-за превышения допустимого количества символов (допустимо - 30) в названии триггеров и последовательностей, поскольку они создавались автоматически, исходя из названия полей и таблиц.

Изменив имена триггеров и последовательностей на более короткие (таким образом, чтобы название последовательностей и триггеров не теряло смыслового значения), их создание прошло безошибочно.

2.3 Добавление данных в таблицы

Было решено первыми заполнить таблицы, которые не зависят от данных в других таблицах. Сначала были заполнены таблицы «City», «Soc_Status», «Documents», «Accommodation», «Food», «Hotel» и «Airlines».

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 "City" при помощи оператора INSERT:

Листинг 1 - Добавление записи в таблицу «City»

```
insert into CITY (ID_CITY, NAME, COUNTRY) values (1,  
'Moscow', 'Russia');
```

В данном примере команда INSERT добавляет записи в таблицу "Rooms".

В поля « ID_CITY », «Name», « Country»

заносятся данные 1, 'Moscow', 'Russia'.

Команда INSERT за один вызов заносит в таблицу лишь одну строку, поэтому, чтобы заполнить нашу таблицу нам понадобится вызвать команду столько раз, сколько строк мы планируем заполнить.[3]

После того, как были добавлены данные, перечисленных выше таблиц, был написан SQL-код для заполнения остальных таблиц: «Client»,

«Operator», «Flight», «Airport», «Sales».

Последней было решено заполнить таблицу "Order", поскольку она содержит самое большое количество полей и данных, взятых из других таблиц.

В результате проделанной работы на данном этапе была получена база данных «Туристическое агентство», состоящая из 13 таблиц: «Заказ», «Клиент», «Документ», «Скидка», «Рейс», «Авиакомпания», «Город», «Аэропорт», «ОТЕЛЬ», «Питание», «Размещение», «Оператор», «Социальный статус».

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 была создана дополнительная таблица.

Её структура представлена на рисунке 4.

Column Name	Data Type	Nullable	Default	Primary Key
ID_USER	NUMBER	No	"TOURSHOP_IU"."ISEQ\$\$_63516215".nextval	1
U_NAME	VARCHAR2(20)	Yes	-	-
U_PASSWORD	VARCHAR2(20)	Yes	-	-
U_ACTIVATED	NUMBER	Yes	-	-

Рисуно

к 4 - Дополнительная таблица для создания авторизации

Для осуществления авторизации нового постоянного пользователя был создан пакет AUT. Его спецификация и тело представлено в листинге 2.

Листинг 2 - Авторизация нового постоянного пользователя

```
create or replace package AUT as
function user_aut (
    U_name in varchar2
    ,U_password in varchar2 )
return boolean;
end;

create or replace package body "AUT" is
function USER_AUT(
    U_NAME IN VARCHAR2
    ,U_PASSWORD IN VARCHAR2
) return BOOLEAN as
    i_count integer;
begin
    Select Count(*) Into i_count
    From USER_LOGIN
```



```
Where Upper(Trim(USER_NAME)) = Upper(Trim(U_NAME)) and  
      Trim(USERPASS) = Trim(U_PASSWORD);  
  
      If i_count > 0 Then return True; Else return False;  
End If;  
  
end USER_AUT;  
end "AUT";
```

Спецификация описывает прототип функции user_aut, результатом выполнения которого является логическое значение, зависящее от наличия пользователя в таблице «USER_LOGIN» (возвращает TRUE, если пользователь найден, FALSE - в противном случае).

В теле функции осуществляется сравнение введенного имени пользователя в поле «Username» с полем «USER_NAME» и введенного пароля в поле «Password» с полем «USERPASS».

Для авторизации необходимо было создать страницу, в которой присутствуют поля для ввода логина и пароля и кнопка «Login». Интерфейс страницы авторизации представлен на рисунке 5.



The image shows a web form for user authentication. It consists of two input fields: 'Username' and 'Password'. The 'Username' field contains the text 'Иванов И.И.'. The 'Password' field contains six dots, indicating masked text. To the right of the password field are two buttons: 'Login' and 'Registration'.

Рисунок 5 - Страница авторизации

Была создана схема авторизации на основе таблицы «USER_LOGIN» отдельно для пользователя и для администратора. Это связано с тем, что администратор обладает следующими правами: возможность вносить коррективы в таблицы, исправлять данные и пр. Страница Main_H имеет два интерфейса для пользователя и администратора.

3.2. Создание страницы Администратора

Для администратора системы предусмотрена страница управления пользователями (рис. 2), которая включает в себя выбор прав, достаточных для входа в приложение, отчет со списком пользователей, которые имеют привилегию на вход в приложение, функцию создания таких пользователей,

кнопки для добавления новых пользователей, для удаления пользователей и для сохранения изменений. При нажатии кнопки сохранения изменений новые данные заносятся в базу.

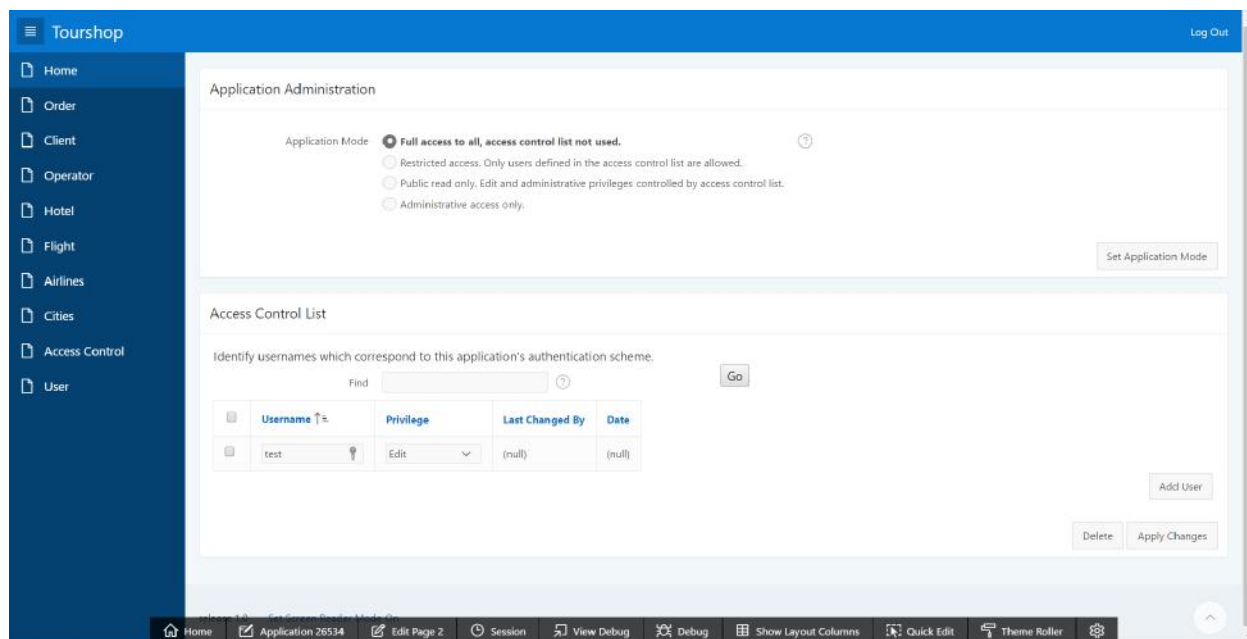



Рисунок 5 - Страница управления пользователями

3.3. Страница входа

Страница входа в систему (рис.3) состоит из формы для ввода имени учётной записи и пароля и выпадающего списка, позволяющего выбрать тип учётной записи, администратор или пользователь с ограниченными правами.

При выборе типа учётной записи срабатывает динамическое действие, при котором заполняются элементы  и заносятся в текущую сессию. Это позволяет обеспечить заполнение страниц приложения в соответствии с правами пользователя.

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.

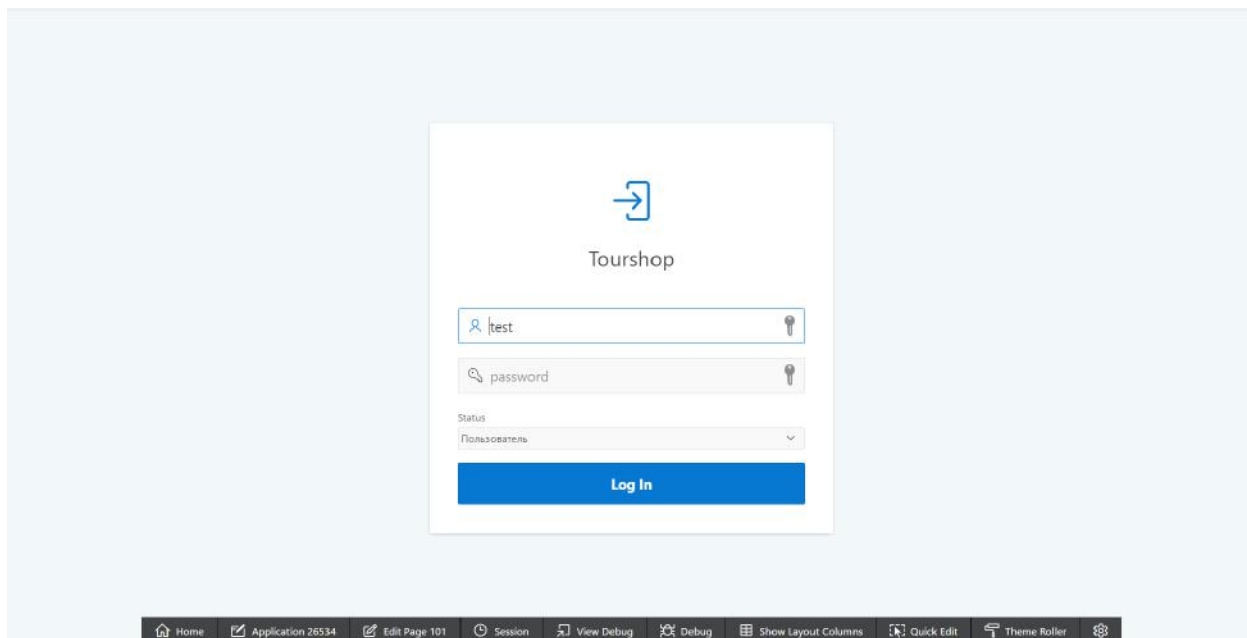


Рисунок 7 - Страница входа

ЗАКЛЮЧЕНИЕ

В результате выполнения курсового проекта была создана система заказа туров.

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 Data
- 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 Дейт К. Дж., Введение в системы баз данных [Текст] / К. Дж. Дейт, Введение в системы баз данных, 8-е издание - Издательский дом "Вильямс", 2005, 1316с. - ISBN 5-8459-0788-8
- 4 MySQL. Руководство администратора. - М.: Вильямс, 2005. - 621 с.
- 5 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.
- 6 Молинаро Э., SQL. Сборник рецептов [Текст] / Энтони Молинаро, SQL. Сборник рецептов - СПб: Символ-Плюс, 2009, 666с. - ISBN 5-93286-125-8
- 7 Борисов В.Ю. Методические рекомендации к выполнению курсовой работы / В.Ю. Борисов. - М.: Нобель Пресс, 2013. - 988 с

SQL-код, сгенерированный SQL Data Modeler

```

CREATE TABLE "SOC STATUS"
( "ID_STATUS" NUMBER(*,0) NOT NULL ENABLE,
  "STATUS" CHAR(25),
  CONSTRAINT "SOC_STATUS_PK" PRIMARY KEY ("ID_STATUS")
  USING INDEX ENABLE
)
/
CREATE TABLE "DOCUMENTS"
( "ID_DOC" NUMBER(*,0) NOT NULL ENABLE,
  "NAME" CHAR(25),
  CONSTRAINT "DOCUMENT_PK" PRIMARY KEY ("ID_DOC")
  USING INDEX ENABLE
)
/
CREATE TABLE "CLIENT"
( "ID_CLIENT" NUMBER(*,0) NOT NULL ENABLE,
  "SURNAME" CHAR(25),
  "NAME" CHAR(20),
  "PATRONYMIC" CHAR(25),
  "BIRTH" DATE,
  "ID_DOC" NUMBER(*,0) NOT NULL ENABLE,
  "NUM_DOC" NUMBER(*,0),
  "PHONE" CHAR(10),
  "EMAIL" CHAR(25),
  "SOC_STATUS" NUMBER(*,0) NOT NULL ENABLE,
  CONSTRAINT "CLIENT_PK" PRIMARY KEY ("ID_CLIENT")
  USING INDEX ENABLE
)
/
CREATE TABLE "OPERATOR"
( "ID_OPERATOR" NUMBER(*,0) NOT NULL ENABLE,
  "NAME" CHAR(20),
  "SURNAME" CHAR(20),
  "PATRONYMIC" CHAR(20),
  "ID_DOC" NUMBER(*,0) NOT NULL ENABLE,
  "NUM_DOC" CHAR(15),
  "REG_NUMBER" CHAR(15),
  "ADDRESS" CHAR(25),
  "EMAIL" CHAR(15),
  CONSTRAINT "OPERATOR_PK" PRIMARY KEY ("ID_OPERATOR")
  USING INDEX ENABLE
)
/
CREATE TABLE "AIRLINES"
( "ID_AIRLINE" NUMBER(*,0) NOT NULL ENABLE,
  "NAME" CHAR(15),
  CONSTRAINT "AIRLINES_PK" PRIMARY KEY ("ID_AIRLINE")
  USING INDEX ENABLE
)
/
CREATE TABLE "FLIGHT"
( "ID_FLIGHT" NUMBER(*,0) NOT NULL ENABLE,
  "NUMBER_FLIGHT" CHAR(6),
  "DATE_FLIGHT" DATE,
  "AIR_ARRIVAL" NUMBER(*,0) NOT NULL ENABLE,
  "AIR_DEPART" NUMBER(*,0) NOT NULL ENABLE,
  "AIRLINES" NUMBER(*,0) NOT NULL ENABLE,
  CONSTRAINT "FLIGHT_PK" PRIMARY KEY ("ID_FLIGHT")
  USING INDEX ENABLE
)
/
CREATE TABLE "SALES"
( "ID_SALE" NUMBER(*,0) NOT NULL ENABLE,
  "NUM_SALE" NUMBER(*,0),

```

```

        "SOC_STATUS" NUMBER(*,0) NOT NULL ENABLE,
        CONSTRAINT "SALES_PK" PRIMARY KEY ("ID_SALE")
    USING INDEX ENABLE
)
/
CREATE TABLE "FOOD"
( "ID_FOOD" NUMBER(*,0) NOT NULL ENABLE,
  "TYPE_FOOD" CHAR(20),
  "PRICE_FOOD" NUMBER NOT NULL ENABLE,
  CONSTRAINT "FOOD_PK" PRIMARY KEY ("ID_FOOD")
  USING INDEX ENABLE
)
/
CREATE TABLE "HOTEL"
( "ID_HOTEL" NUMBER(*,0) NOT NULL ENABLE,
  "NAME_HOTEL" CHAR(20),
  "TYPE_HOTEL" NUMBER(*,0),
  CONSTRAINT "HOTEL_PK" PRIMARY KEY ("ID_HOTEL")
  USING INDEX ENABLE
)
/
CREATE TABLE "ACCOMODATION"
( "ID_ROOM" NUMBER(*,0) NOT NULL ENABLE,
  "TYPE_ROOM" CHAR(20),
  "NUM_PERSON" NUMBER(*,0),
  "PRICE_ROOM" NUMBER NOT NULL ENABLE,
  CONSTRAINT "ACCOMADATION_PK" PRIMARY KEY ("ID_ROOM")
  USING INDEX ENABLE
)
/
CREATE TABLE "Order"
( "ID_ORDER" NUMBER(*,0) NOT NULL ENABLE,
  "ID_OPERATOR" NUMBER(*,0) NOT NULL ENABLE,
  "ID_CLIENT" NUMBER(*,0) NOT NULL ENABLE,
  "VISA" CHAR(9),
  "DATE_ORDER" DATE,
  "ID_HOTEL" NUMBER(*,0) NOT NULL ENABLE,
  "ID_FOOD" NUMBER(*,0) NOT NULL ENABLE,
  "ID_ROOM" NUMBER(*,0) NOT NULL ENABLE,
  "FLIGHT_TO" NUMBER(*,0) NOT NULL ENABLE,
  "FLIGHT_FROM" NUMBER(*,0) NOT NULL ENABLE,
  "DATE_DEPART" TIMESTAMP(6),
  "DATE_ARRIVAL" TIMESTAMP(6),
  "ID_SALE" NUMBER(*,0) NOT NULL ENABLE,
  "DAYS" NUMBER NOT NULL ENABLE,
  "TRIPS_PRICE" NUMBER,
  CONSTRAINT "ORDER_PK" PRIMARY KEY ("ID_ORDER")
  USING INDEX ENABLE
)
/
CREATE TABLE "CITY"
( "ID_CITY" NUMBER(*,0) NOT NULL ENABLE,
  "NAME" CHAR(25),
  "COUNTRY" CHAR(25),
  CONSTRAINT "CITY_PK" PRIMARY KEY ("ID_CITY")
  USING INDEX ENABLE
)
/
CREATE TABLE "AIRPORT"
( "ID_AIR" NUMBER(*,0) NOT NULL ENABLE,
  "NAME" CHAR(25),
  "CITY" NUMBER(*,0) NOT NULL ENABLE,
  "IATA" CHAR(5),
  CONSTRAINT "AIR_PK" PRIMARY KEY ("ID_AIR")
  USING INDEX ENABLE
)
/
CREATE TABLE "APEX_ACCESS_SETUP"
( "ID" NUMBER,

```



```

        "APPLICATION_MODE" VARCHAR2(255),
        "APPLICATION_ID" NUMBER,
        CONSTRAINT "APEX_ACCESS_SETUP_PK" PRIMARY KEY ("ID")
    USING INDEX ENABLE,
        CONSTRAINT "APEX_ACCESS_SETUP_UK" UNIQUE
    ("APPLICATION_ID")
    USING INDEX ENABLE
)
/
CREATE TABLE "APEX_ACCESS_CONTROL"
(
    "ID" NUMBER,
    "ADMIN_USERNAME" VARCHAR2(255),
    "ADMIN_PRIVILEGES" VARCHAR2(255),
    "SETUP_ID" NUMBER,
    "CREATED_BY" VARCHAR2(255),
    "CREATED_ON" DATE,
    "UPDATED_ON" DATE,
    "UPDATED_BY" VARCHAR2(255),
    CONSTRAINT "APEX_ACCESS_CONTROL_PK" PRIMARY KEY
    ("ID")
    USING INDEX ENABLE,
    CONSTRAINT "APEX_ACCESS_CONTROL_UK" UNIQUE
    ("ADMIN_USERNAME", "SETUP_ID")
    USING INDEX ENABLE
)
/
CREATE TABLE "HTMLDB_PLAN_TABLE"
(
    "STATEMENT_ID" VARCHAR2(30),
    "PLAN_ID" NUMBER,
    "TIMESTAMP" DATE,
    "REMARKS" VARCHAR2(4000),
    "OPERATION" VARCHAR2(30),
    "OPTIONS" VARCHAR2(255),
    "OBJECT_NODE" VARCHAR2(128),
    "OBJECT_OWNER" VARCHAR2(128),
    "OBJECT_NAME" VARCHAR2(128),
    "OBJECT_ALIAS" VARCHAR2(261),
    "OBJECT_INSTANCE" NUMBER(*,0),
    "OBJECT_TYPE" VARCHAR2(128),
    "OPTIMIZER" VARCHAR2(255),
    "SEARCH_COLUMNS" NUMBER,
    "ID" NUMBER(*,0),
    "PARENT_ID" NUMBER(*,0),
    "DEPTH" NUMBER(*,0),
    "POSITION" NUMBER(*,0),
    "COST" NUMBER(*,0),
    "CARDINALITY" NUMBER(*,0),
    "BYTES" NUMBER(*,0),
    "OTHER_TAG" VARCHAR2(255),
    "PARTITION_START" VARCHAR2(255),
    "PARTITION_STOP" VARCHAR2(255),
    "PARTITION_ID" NUMBER(*,0),
    "OTHER" LONG,
    "DISTRIBUTION" VARCHAR2(30),
    "CPU_COST" NUMBER(*,0),
    "IO_COST" NUMBER(*,0),
    "TEMP_SPACE" NUMBER(*,0),
    "ACCESS_PREDICATES" VARCHAR2(4000),
    "FILTER_PREDICATES" VARCHAR2(4000),
    "PROJECTION" VARCHAR2(4000),
    "TIME" NUMBER(*,0),
    "QBLOCK_NAME" VARCHAR2(128)
)
/
CREATE TABLE "USERS"
(
    "ID_USER" NUMBER GENERATED BY DEFAULT AS IDENTITY
    MINVALUE 1 MAXVALUE 99999999999999999999999999999999
    INCREMENT BY 1 START WITH 1 CACHE 20 NOORDER NOCYCLE
    NOT NULL ENABLE,

```

```

        "U_NAME" VARCHAR2(20),
        "U_PASSWORD" VARCHAR2(20),
        "U_ACTIVATED" NUMBER,
        CONSTRAINT "USERS_PK" PRIMARY KEY ("ID_USER")
    USING INDEX ENABLE
)
/
ALTER TABLE "AIRPORT" ADD CONSTRAINT "AIR_CITY_FK"
FOREIGN KEY ("CITY")
    REFERENCES "CITY" ("ID_CITY") ENABLE
/
ALTER TABLE "APEX_ACCESS_CONTROL" ADD CONSTRAINT
"APEX_ACCESS_CONTROL_FK" FOREIGN KEY ("SETUP_ID")
    REFERENCES "APEX_ACCESS_SETUP" ("ID") ENABLE
/
ALTER TABLE "CLIENT" ADD CONSTRAINT
"CLIENT_SOC_STATUS_FK" FOREIGN KEY ("SOC_STATUS")
    REFERENCES "SOC_STATUS" ("ID_STATUS") ENABLE
/
ALTER TABLE "CLIENT" ADD CONSTRAINT "Client
_Document_FK" FOREIGN KEY ("ID_DOC")
    REFERENCES "DOCUMENTS" ("ID_DOC") ENABLE
/
ALTER TABLE "FLIGHT" ADD CONSTRAINT "FLIGHT_AIRPORT_FK"
FOREIGN KEY ("AIR_ARRIVAL")
    REFERENCES "AIRPORT" ("ID_AIR") ENABLE
/
ALTER TABLE "FLIGHT" ADD CONSTRAINT
"FLIGHT_AIRPORT_FKV2" FOREIGN KEY ("AIR_DEPART")
    REFERENCES "AIRPORT" ("ID_AIR") ENABLE
/
ALTER TABLE "FLIGHT" ADD CONSTRAINT "FLIGHT_AIRLINES_FK"
FOREIGN KEY ("AIRLINES")
    REFERENCES "AIRLINES" ("ID_AIRLINE") ENABLE
/
ALTER TABLE "OPERATOR" ADD CONSTRAINT
"OPERATOR_DOCUMENTS_FK" FOREIGN KEY ("ID_DOC")
    REFERENCES "DOCUMENTS" ("ID_DOC") ENABLE
/
ALTER TABLE "Order" ADD CONSTRAINT
"ORDER_ACCOMADATION_FK" FOREIGN KEY ("ID_ROOM")
    REFERENCES "ACCOMODATION" ("ID_ROOM") ENABLE
/
ALTER TABLE "Order" ADD CONSTRAINT "ORDER_CLIENT_FK"
FOREIGN KEY ("ID_CLIENT")
    REFERENCES "CLIENT" ("ID_CLIENT") ENABLE
/
ALTER TABLE "Order" ADD CONSTRAINT "ORDER_FLIGHT_FK"
FOREIGN KEY ("FLIGHT_TO")
    REFERENCES "FLIGHT" ("ID_FLIGHT") ENABLE
/
ALTER TABLE "Order" ADD CONSTRAINT "ORDER_FLIGHT_FKV1"
FOREIGN KEY ("FLIGHT_FROM")
    REFERENCES "FLIGHT" ("ID_FLIGHT") ENABLE
/
ALTER TABLE "Order" ADD CONSTRAINT "ORDER_FOOD_FK"
FOREIGN KEY ("ID_FOOD")
    REFERENCES "FOOD" ("ID_FOOD") ENABLE
/
ALTER TABLE "Order" ADD CONSTRAINT "ORDER_HOTEL_FK"
FOREIGN KEY ("ID_HOTEL")
    REFERENCES "HOTEL" ("ID_HOTEL") ENABLE
/
ALTER TABLE "Order" ADD CONSTRAINT "ORDER_OPERATOR_FK"
FOREIGN KEY ("ID_OPERATOR")
    REFERENCES "OPERATOR" ("ID_OPERATOR") ENABLE
/
ALTER TABLE "Order" ADD CONSTRAINT "ORDER_SALE_FK"
FOREIGN KEY ("ID_SALE")

```

```

REFERENCES "SALES" ("ID_SALE") ENABLE
/
ALTER TABLE "SALES" ADD CONSTRAINT "SALES_SOC_STATUS_FK"
FOREIGN KEY ("SOC_STATUS")
REFERENCES "SOC_STATUS" ("ID_STATUS") ENABLE
/
CREATE SEQUENCE "ACCOMADATION_ID_ROOM_SEQ" MINVALUE 1
MAXVALUE 99999999999999999999999999999999 INCREMENT BY 1
START WITH 1 NOCACHE ORDER NOCYCLE NOPARTITION
/
CREATE SEQUENCE "ACCOMODATION_ID_ROOM_SEQ" MINVALUE 1
MAXVALUE 99999999999999999999999999999999 INCREMENT BY 1
START WITH 1 NOCACHE ORDER NOCYCLE NOPARTITION
/
CREATE SEQUENCE "AIRLINES_ID_AIRLINE_SEQ" MINVALUE 1
MAXVALUE 99999999999999999999999999999999 INCREMENT BY 1
START WITH 6 NOCACHE ORDER NOCYCLE NOPARTITION
/
CREATE SEQUENCE "AIRPORT_ID_AIR_SEQ" MINVALUE 1
MAXVALUE 99999999999999999999999999999999 INCREMENT BY 1
START WITH 1 NOCACHE ORDER NOCYCLE NOPARTITION
/
CREATE SEQUENCE "APEX_ACCESS_CONTROL_SEQ" MINVALUE 1
MAXVALUE 99999999999999999999999999999999 INCREMENT BY 1
START WITH 21 CACHE 20 NOORDER NOCYCLE NOPARTITION
/
CREATE SEQUENCE "CITY_ID_CITY_SEQ" MINVALUE 1
MAXVALUE 99999999999999999999999999999999 INCREMENT BY 1
START WITH 8 NOCACHE ORDER NOCYCLE NOPARTITION
/
CREATE SEQUENCE "CLIENT_ID_CLIENT_SEQ" MINVALUE 1
MAXVALUE 99999999999999999999999999999999 INCREMENT BY 1
START WITH 11 NOCACHE ORDER NOCYCLE NOPARTITION
/
CREATE SEQUENCE "DEMO_ORDER_ITEMS_SEQ" MINVALUE 1
MAXVALUE 99999999999999999999999999999999 INCREMENT BY 1
START WITH 160 CACHE 20 NOORDER NOCYCLE NOPARTITION
/
CREATE SEQUENCE "DEMO_ORD_SEQ" MINVALUE 1 MAXVALUE
99999999999999999999999999999999 INCREMENT BY 1 START WITH 11
CACHE 20 NOORDER NOCYCLE NOPARTITION
/
CREATE SEQUENCE "APEX$WS_SEQ" MINVALUE 100 MAXVALUE
99999999999999999999999999999999 INCREMENT BY 1 START WITH 100
CACHE 20 NOORDER NOCYCLE NOPARTITION
/
CREATE SEQUENCE "DEMO_CUST_SEQ" MINVALUE 1 MAXVALUE
99999999999999999999999999999999 INCREMENT BY 1 START WITH
100 CACHE 20 NOORDER NOCYCLE NOPARTITION
/
CREATE SEQUENCE "DEMO_PROD_SEQ" MINVALUE 1 MAXVALUE
99999999999999999999999999999999 INCREMENT BY 1 START WITH
100 CACHE 20 NOORDER NOCYCLE NOPARTITION
/
CREATE SEQUENCE "DEPT_SEQ" MINVALUE 1 MAXVALUE
99999999999999999999999999999999 INCREMENT BY 1 START WITH 50
CACHE 20 NOORDER NOCYCLE NOPARTITION
/
CREATE SEQUENCE "DOCUMENTS_ID_DOC_SEQ" MINVALUE 1
MAXVALUE 99999999999999999999999999999999 INCREMENT BY 1
START WITH 1 NOCACHE ORDER NOCYCLE NOPARTITION
/
CREATE SEQUENCE "EMP_SEQ" MINVALUE 1 MAXVALUE
99999999999999999999999999999999 INCREMENT BY 1 START WITH
8000 CACHE 20 NOORDER NOCYCLE NOPARTITION
/
CREATE SEQUENCE "FLIGHT_ID_FLIGHT_SEQ" MINVALUE 1
MAXVALUE 99999999999999999999999999999999 INCREMENT BY 1
START WITH 12 NOCACHE ORDER NOCYCLE NOPARTITION

```



```

    INSERT ON Client FOR EACH ROW  WHEN (NEW.ID_Client IS
NULL) BEGIN :NEW.ID_Client :=
Client_ID_Client_SEQ.NEXTVAL;
END;

/
ALTER TRIGGER "CLIENT_ID_CLIENT_TRG" ENABLE
/
CREATE OR REPLACE EDITIONABLE TRIGGER
"DOCUMENTS_ID_DOC_TRG" BEFORE
    INSERT ON Documents FOR EACH ROW  WHEN (NEW.ID_Doc IS
NULL) BEGIN :NEW.ID_Doc := Documents_ID_Doc_SEQ.NEXTVAL;
END;

/
ALTER TRIGGER "DOCUMENTS_ID_DOC_TRG" ENABLE
/
CREATE OR REPLACE EDITIONABLE TRIGGER
"FLIGHT_ID_FLIGHT_TRG" BEFORE
    INSERT ON Flight FOR EACH ROW  WHEN (NEW.ID_Flight IS
NULL) BEGIN :NEW.ID_Flight :=
Flight_ID_Flight_SEQ.NEXTVAL;
END;

/
ALTER TRIGGER "FLIGHT_ID_FLIGHT_TRG" ENABLE
/
CREATE OR REPLACE EDITIONABLE TRIGGER "FOOD_ID_FOOD_TRG"
BEFORE
    INSERT ON Food FOR EACH ROW  WHEN (NEW.ID_Food IS NULL)
BEGIN :NEW.ID_Food := Food_ID_Food_SEQ.NEXTVAL;
END;

/
ALTER TRIGGER "FOOD_ID_FOOD_TRG" ENABLE
/
CREATE OR REPLACE EDITIONABLE TRIGGER
"HOTEL_ID_HOTEL_TRG" BEFORE
    INSERT ON Hotel FOR EACH ROW  WHEN (NEW.ID_Hotel IS
NULL) BEGIN :NEW.ID_Hotel := Hotel_ID_Hotel_SEQ.NEXTVAL;
END;

/
ALTER TRIGGER "HOTEL_ID_HOTEL_TRG" ENABLE
/
CREATE OR REPLACE EDITIONABLE TRIGGER
"OPERATOR_ID_OPERATOR_TRG" BEFORE
    INSERT ON Operator FOR EACH ROW  WHEN (NEW.ID_Operator
IS NULL) BEGIN :NEW.ID_Operator :=
Operator_ID_Operator_SEQ.NEXTVAL;
END;

/
ALTER TRIGGER "OPERATOR_ID_OPERATOR_TRG" ENABLE
/
CREATE OR REPLACE EDITIONABLE TRIGGER
"ORDER_ID_ORDER_TRG" BEFORE
    INSERT ON "Order" FOR EACH ROW  WHEN (NEW.ID_Order IS
NULL) BEGIN :NEW.ID_Order := Order_ID_Order_SEQ.NEXTVAL;
END;

/
ALTER TRIGGER "ORDER_ID_ORDER_TRG" ENABLE
/
CREATE OR REPLACE EDITIONABLE TRIGGER
"SALES_ID_SALE_TRG" BEFORE
    INSERT ON Sales FOR EACH ROW  WHEN (NEW.ID_Sale IS
NULL) BEGIN :NEW.ID_Sale := Sales_ID_Sale_SEQ.NEXTVAL;
END;

```

```

/
ALTER TRIGGER "SALES_ID_SALE_TRG" ENABLE
/
CREATE OR REPLACE EDITIONABLE TRIGGER
"SOC_STATUS_ID_STATUS_TRG" BEFORE
  INSERT ON Soc_Status FOR EACH ROW WHEN (NEW.ID_Status
IS NULL) BEGIN :NEW.ID_Status :=
Soc_Status_ID_Status_SEQ.NEXTVAL;
END;

/
ALTER TRIGGER "SOC_STATUS_ID_STATUS_TRG" ENABLE
/
CREATE OR REPLACE EDITIONABLE TRIGGER
"BI_APEX_ACCESS_CONTROL"
  before insert or update on apex_access_control
  for each row
begin
  if inserting and :new.id is null then
    select apex_access_control_seq.nextval
into :new.id from sys.dual;
  end if;
  if inserting then
    :new.created_by := v('USER');
    :new.created_on := sysdate;
  end if;
  if updating then
    :new.updated_by := v('USER');
    :new.updated_on := sysdate;
  end if;
end;

/
ALTER TRIGGER "BI_APEX_ACCESS_CONTROL" ENABLE
/
CREATE OR REPLACE EDITIONABLE TRIGGER
"BI_APEX_ACCESS_SETUP"
  before insert or update on apex_access_setup
  for each row
begin
  if inserting and :new.id is null then
    select apex_access_control_seq.nextval into :new.id from
sys.dual;
  end if;
  if :new.application_id is null then
    :new.application_id := v('APP_ID');
  end if;
end;

/
ALTER TRIGGER "BI_APEX_ACCESS_SETUP" ENABLE
/
CREATE OR REPLACE EDITIONABLE TRIGGER "BI_USERS"
  before insert on "USERS"
  for each row
begin
  if :NEW."ID_USER" is null then
    select "USERS_SEQ".nextval into :NEW."ID_USER" from
sys.dual;
  end if;
end;

/
ALTER TRIGGER "BI_USERS" ENABLE
/

```