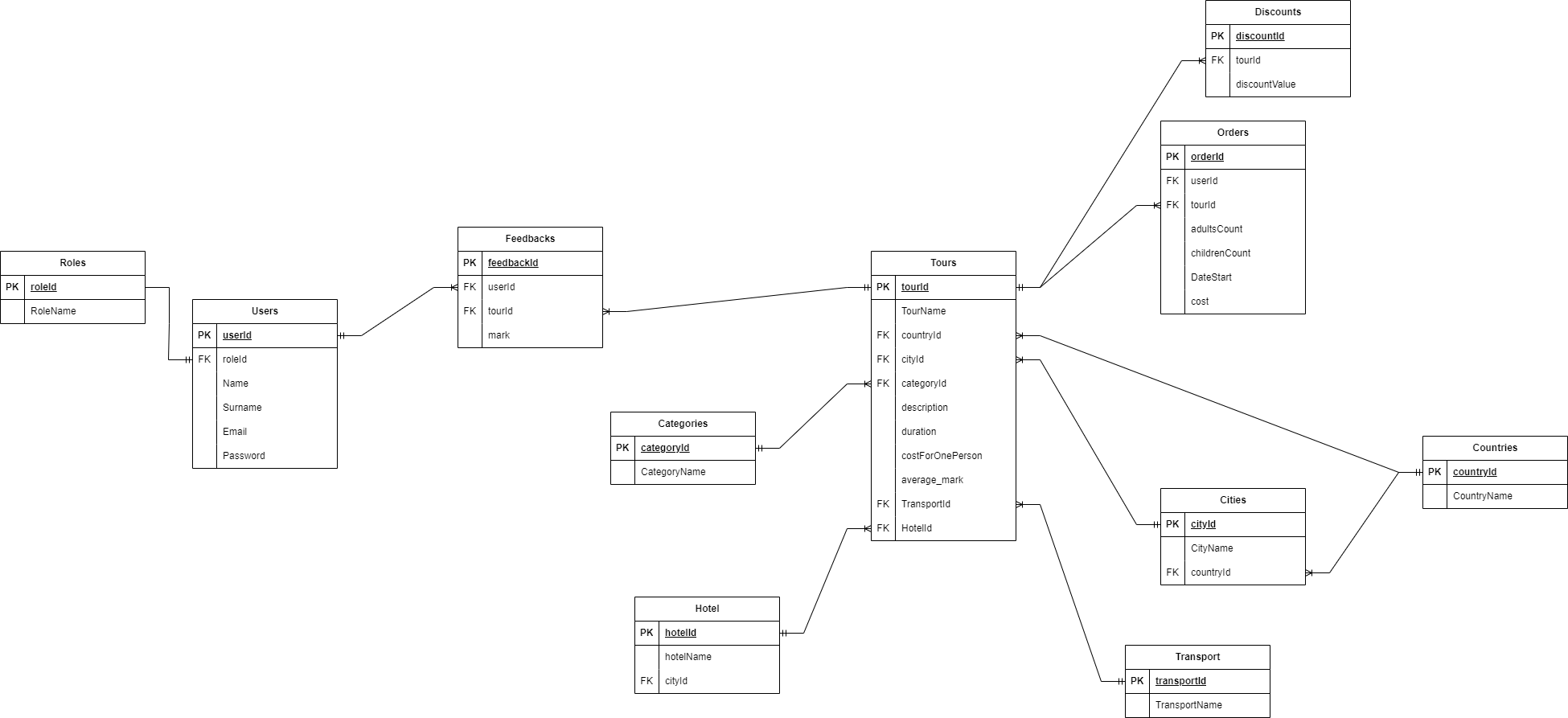
**Турфирма. Лабораторная работа №2**

Схема базы данных:



# MSSQL

## Таблицы

### Роли

|  |
| --- |
| create table ROLES(  roleId uniqueidentifier primary key not null,  roleName varchar(20) not null  ); |

### Пользователи

|  |
| --- |
| create table USERS(  userId uniqueidentifier primary key not null,  name varchar(50) not null,  surname varchar(50) not null,  email varchar(50) not null unique,  roleId uniqueidentifier foreign key references ROLES(roleId) not null,  password varchar(32) not null  ); |

### Категории

|  |
| --- |
| create table CATEGORIES(  categoryId uniqueidentifier primary key not null,  categoryName varchar(100) not null  ); |

### Страны

|  |
| --- |
| create table COUNTRIES(  countryId char(2) primary key not null,  countryName varchar(30) not null  ); |

### Города

|  |
| --- |
| create table CITIES(  cityId char(3) primary key not null,  cityName varchar(30) not null,  countryId char(2) foreign key references COUNTRIES(countryId) not null,  ); |

### Отели

|  |
| --- |
| create table HOTELS(  hotelId uniqueidentifier primary key not null,  hotelName varchar(40) not null,  cityId char(3) foreign key(cityId) references CITIES(cityId)  ); |

### Транспорт

|  |
| --- |
| create table TRANSPORTS(  transportId uniqueidentifier primary key not null,  transportName varchar(20) not null  ); |

### Туры

|  |
| --- |
| create table TOURS(  tourId uniqueidentifier primary key not null,  tourName varchar(100) not null,  countryId char(2) foreign key references COUNTRIES(countryId) not null,  cityId char(3) foreign key references CITIES(cityId) not null,  categoryId uniqueidentifier foreign key references CATEGORIES(categoryId) not null,  description varchar(1000) not null,  duration int not null,  costForOnePerson float not null check(costForOnePerson>0),  average\_mark float not null,  transportId uniqueidentifier foreign key references TRANSPORTS(transportId) not null,  hotelId uniqueidentifier foreign key references HOTELS(hotelId) not null  ); |

### Отзывы

|  |
| --- |
| create table FEEDBACKS(  feedbackId uniqueidentifier primary key not null,  userId uniqueidentifier foreign key references USERS(userId) not null,  tourId uniqueidentifier foreign key references TOURS(tourId) not null,  mark float not null  ); |

### Заказы

|  |
| --- |
| create table ORDERS(  orderId uniqueidentifier primary key,  tourId uniqueidentifier foreign key references TOURS(tourId) not null,  userId uniqueidentifier foreign key references USERS(userId) not null,  adultsCount int not null check(adultsCount>0),  childrenCount int not null check(childrenCount>=0),  DateStart date not null,  cost float not null check(cost>0)  ); |

### Скидки

|  |
| --- |
| create table DISCOUNTS(  discountId uniqueidentifier primary key not null,  tourId uniqueidentifier foreign key references TOURS(tourId) not null,  discountValue float not null check(discountValue>0 and discountValue<1)  ); |

## Индексы

### Индекс на таблицу со странами

|  |
| --- |
| create index COUNTRIES\_INDX on COUNTRIES(countryId); |

### Индекс на таблицу с городами для столбцов с идентификаторами страны и города

|  |
| --- |
| create index CITIES\_IDX on CITIES(cityId, countryId); |

### Индекс на таблицу с городами

|  |
| --- |
| create index TOURS\_IDX on TOURS(tourId) include (tourName); |

## Процедуры

### Процедура получения самого дорогого тура

|  |
| --- |
| create or alter procedure GetTheMostExpensiveTour  as  begin  set nocount on;  begin try  declare @EXPTOUR money=(select MAX(costForOnePerson) from TOURS);  select \* from DETAILED\_VIEW\_TOUR where costForOnePerson=@EXPTOUR;  return 1;  end try  begin catch  declare @ERROR\_MESSAGE nvarchar(max), @ERROR\_SEVERITY int, @ERROR\_STATE int;  select @ERROR\_MESSAGE = ERROR\_MESSAGE(),  @ERROR\_SEVERITY = ERROR\_SEVERITY(),  @ERROR\_STATE = ERROR\_STATE();  raiserror (@ERROR\_MESSAGE, @ERROR\_SEVERITY, @ERROR\_STATE);  return -1;  end catch  end; |

### Процедура добавления нового тура

|  |
| --- |
| create or alter procedure AddNewTour  @TOUR\_NAME nvarchar(100),  @CITY\_NAME nvarchar(30),  @COUNTRY\_NAME nvarchar(30),  @CATEGORY\_NAME nvarchar(100),  @DESCR nvarchar(1000),  @DUR int,  @COSTFORONE money,  @TRNAME nvarchar(20),  @HNAME nvarchar(40)  as  begin  set nocount on;  begin try  DECLARE @TOUR\_ID uniqueidentifier=NEWID(),  @CITY\_ID char(3)=(select cityId from CITIES where cityName=@CITY\_NAME),  @COUNTRY\_ID char(2)=(select countryId from COUNTRIES where countryName=@COUNTRY\_NAME),  @CATEGORY\_ID uniqueidentifier=(select categoryId from CATEGORIES where categoryName=@CATEGORY\_NAME),  @TRID uniqueidentifier=(select transportId from TRANSPORTS where transportName=@TRNAME),  @HID uniqueidentifier=(select hotelId from HOTELS where hotelName=@HNAME);  print 'CITY\_ID: '+cast(@CITY\_ID as nvarchar)  begin tran  insert into TOURS(tourId,tourName,countryId,cityId,categoryId,description,duration,costForOnePerson,average\_mark,transportId,hotelId) values (@TOUR\_ID,@TOUR\_NAME,@COUNTRY\_ID,@CITY\_ID,@CATEGORY\_ID,@DESCR,@DUR,@COSTFORONE,0,@TRID,@HID);  commit;  select \* from TOURS where tourId=@TOUR\_ID;  print 'Added new tour!';  return 1;  end try  begin catch  declare @ERROR\_MESSAGE nvarchar(max), @ERROR\_SEVERITY int, @ERROR\_STATE int;  select @ERROR\_MESSAGE = ERROR\_MESSAGE(),  @ERROR\_SEVERITY = ERROR\_SEVERITY(),  @ERROR\_STATE = ERROR\_STATE();  rollback;  raiserror (@ERROR\_MESSAGE, @ERROR\_SEVERITY, @ERROR\_STATE);  return -1;  end catch  end; |

### Процедура добавления комментария

|  |
| --- |
| create or alter procedure SetTheFeedBack  @USERNAME nvarchar(50),  @USERSURNAME nvarchar(50),  @TOURNAME nvarchar(100),  @MARK float  as  begin  set nocount on;  begin try  declare @FEEDBACK\_ID uniqueidentifier=NEWID(),  @USER\_ID uniqueidentifier=(select userId from USERS where name=@USERNAME and surname=@USERSURNAME),  @TOUR\_ID uniqueidentifier=(select tourId from TOURS where tourName=@TOURNAME)  begin tran  insert into FEEDBACKS values (@FEEDBACK\_ID,@USER\_ID,@TOUR\_ID,@MARK);  commit;  select \* from TOURS where tourId=@TOUR\_ID;  print (cast(@USERNAME as nvarchar))+' '+(cast(@USERSURNAME as nvarchar))+' set the mark'+cast(@MARK as nvarchar)+'to the tour '+cast(@TOURNAME as nvarchar);  end try  begin catch  end catch  end; |

## Функции

### Получение числа стран

|  |
| --- |
| create or alter function GetCountOfCountries()  returns int  as  begin  declare @COUNTR int=(select COUNT(\*) from COUNTRIES);  return @COUNTR;  end; |

### Получение городов по странам

|  |
| --- |
| create or alter function GetCitiesByCountry(  @COUNTRY\_ID char(2)  ) returns table  as  return (select cityId, cityName from CITIES where countryId=@COUNTRY\_ID); |

### Получение туров по определённому виду транспорта

|  |
| --- |
| create or alter function GetToursByTransport(  @TRNAME nvarchar(20)  )  returns table  as  return (select \* from TOURS where transportId=(select transportId from TRANSPORTS where transportName=@TRNAME)); |

## Триггеры

### Триггер обновления средней оценки при добавлении нового отзыва по туру

|  |
| --- |
| create or alter trigger costForTour on ORDERS  after insert, update  as  update ORDERS  set cost=(select costForOnePerson from TOURS where TOURS.tourId=ORDERS.tourId)\*(adultsCount+0.5\*childrenCount)  where orderId=(select orderId from inserted); |

### Триггер обновления стоимости тура в зависимости от числа детей и взрослых

|  |
| --- |
| create or alter trigger avgMark on FEEDBACKS  after insert, update, delete  as  UPDATE TOURS  set average\_mark=(select SUM(mark) from FEEDBACKS where FEEDBACKS.tourId=TOURS.tourId)/(select COUNT(\*) from FEEDBACKS where FEEDBACKS.tourId=TOURS.tourId)  where tourId=(select tourId from inserted); |

### Триггер обновления цены тура на одного взрослого человека при добавлении скидки на тур

|  |
| --- |
| create or alter trigger setDiscount on DISCOUNTS  after insert, update  as  UPDATE TOURS  set costForOnePerson=costForOnePerson-costForOnePerson\*(select discountValue from DISCOUNTS where DISCOUNTS.tourId=TOURS.tourId)  where tourId=(select tourId from inserted); |

## Представления

### Представление информации о туре без идентификаторов в атрибутах

|  |
| --- |
| alter view DETAILED\_VIEW\_TOUR  as  select t.tourId,  t.tourName,  co.countryId,  ci.cityId,  cat.categoryName,  t.description,  t.duration,  t.costForOnePerson,  t.average\_mark,  tr.transportName,  h.hotelName  from TOURS t  join COUNTRIES co on t.countryId=co.countryId  join CITIES ci on t.cityId=ci.cityId  join CATEGORIES cat on t.categoryId=cat.categoryId  join TRANSPORTS tr on t.transportId=tr.transportId  join HOTELS h on t.hotelId=h.hotelId; |

### Представление информации о заказе без идентификаторов в атрибутах

|  |
| --- |
| create or alter view DETAILED\_VIEW\_ORDER  as  select o.orderId, u.name, u.surname, u.email,  t.tourname, o.adultsCount, o.childrenCount, o.DateStart, o.cost  from ORDERS o join USERS u on o.userId=u.userId join TOURS t on o.tourId =t.tourId; |

### Представление из имени страны, имени города и имени отеля

|  |
| --- |
| create or alter view CITY\_COUNTRY\_HOTEL  as  select ci.cityName, co.countryName, h.hotelName from CITIES ci  join COUNTRIES co on ci.countryId=co.countryId join HOTELS h on ci.cityId=h.cityId; |

# Oracle

## Таблицы

### Роли

|  |
| --- |
| create table ROLES(  roleId NUMBER GENERATED ALWAYS AS IDENTITY PRIMARY KEY,  roleName VARCHAR2(20) NOT NULL  ); |

### Пользователи

|  |
| --- |
| create table USERS(  userId number generated always as identity primary key not null,  name varchar(50) not null,  surname varchar(50) not null,  email varchar(50) not null unique,  roleId number,  password varchar(32) not null,  constraint fk\_role foreign key(roleId) references ROLES(roleId)  ); |

### Категории

|  |
| --- |
| create table CATEGORIES(  categoryId number generated always as identity primary key not null,  categoryName varchar2(100) not null  ); |

### Страны

|  |
| --- |
| create table COUNTRIES(  countryId char(2) primary key not null,  countryName varchar2(30) not null  ); |

### Города

|  |
| --- |
| create table CITIES(  cityId char(3) primary key not null,  cityName varchar2(30) not null,  countryId char(2),  constraint fk\_city foreign key(countryId) references COUNTRIES(countryId)  ); |

### Отели

|  |
| --- |
| create table HOTELS(  hotelId number generated always as identity primary key not null,  hotelName varchar(40) not null,  cityId char(3),  constraint fk\_hotel foreign key(cityId) references CITIES(cityId)  ); |

### Транспорт

|  |
| --- |
| create table TRANSPORTS(  transportId number generated always as identity primary key not null,  transportName varchar(20) not null  ); |

### Туры

|  |
| --- |
| create table TOURS(  tourId number generated always as identity primary key not null,  tourName varchar(100) not null,  countryId char(2) not null,  cityId char(3) not null,  categoryId number not null,  description varchar(1000) not null,  duration number not null,  costForOnePerson number(10,2) not null check(costForOnePerson>0),  average\_mark number(10,2) not null,  transportId number not null,  hotelId number not null,  constraint fk\_country\_tours foreign key(countryId) references COUNTRIES(countryId),  constraint fk\_city\_tours foreign key(cityId) references CITIES(cityId),  constraint fk\_category\_tours foreign key(categoryId) references CATEGORIES(categoryId),  constraint fk\_transport\_tours foreign key(transportId) references TRANSPORTS(transportId),  constraint fk\_hotel\_tours foreign key(hotelId) references HOTELS(hotelId)  ); |

### Отзывы

|  |
| --- |
| create table FEEDBACKS(  feedbackId number generated always as identity primary key not null,  userId number,  tourId number,  mark number(10,2) not null,  constraint fk\_feed\_user foreign key(userId) references USERS(userId),  constraint fk\_feed\_tour foreign key(tourId) references TOURS(tourId)  ); |

### Заказы

|  |
| --- |
| create table ORDERS(  orderId number generated always as identity primary key,  tourId number,  userId number,  adultsCount number not null check(adultsCount>0),  childrenCount number not null check(childrenCount>=0),  DateStart date not null,  cost number(10,2) not null check(cost>0),  constraint fk\_order\_tours foreign key(tourId) references TOURS(tourID),  constraint fk\_order\_users foreign key(userId) references USERS(userId)  ); |

### Скидки

|  |
| --- |
| create table DISCOUNTS(  discountId number generated always as identity primary key not null,  tourId number,  discountValue number(10,2) not null check(discountValue>0 and discountValue<1),  constraint fk\_discount\_tours foreign key(tourId) references TOURS(tourId)  ); |

## Индексы

### Индекс на таблицу со странами

|  |
| --- |
| create index COUNTRIES\_INDX on COUNTRIES(countryName); |

### Индекс на таблицу с городами для столбцов с идентификаторами страны и города

|  |
| --- |
| create index CITIES\_IDX on CITIES(cityName, countryId); |

### Индекс на таблицу с городами

|  |
| --- |
| create index TOURS\_IDX on TOURS(tourId) include (tourName); |

## Процедуры

### Процедура получения самого дорогого тура

|  |
| --- |
| create or replace procedure GetTheMostExpensiveTour  is  exptour number;  begin  select max(costForOnePerson) into exptour from TOURS;  select \* from DETAILED\_VIEW\_TOUR where costForOnePerson=cast(exptour as number);  end; |

### Процедура добавления нового тура

|  |
| --- |
| create or replace procedure AddNewTour(  TOUR\_NAME varchar,  CITY\_ID char,  COUNTRY\_ID char,  CATEGORY\_NAME varchar2,  DESCR varchar,  DUR int,  COSTFORONE number,  TRNAME varchar,  HNAME varchar)  is  begin  insert into Tours(tourName,cityId,countryId,categoryId,description,duration,costForOnePerson, average\_mark, transportId, hotelId) values(  tour\_name,  city\_id,  country\_id,  cast((select categoryId from CATEGORIES where categoryName=category\_name) as number),  descr,  dur,  costforone,  0,  cast((select transportId from TRANSPORTS where transportName=trname) as number),  cast((select hotelId from HOTELS where hotelName=hname) as number)  );  end; |

### Процедура добавления комментария

|  |
| --- |
| create or replace procedure SetTheFeedBack(  user\_id number,  tour\_id number,  set\_mark number)  is  begin  insert into FEEDBACKS(userId,tourId,mark) values(user\_id, tour\_id, set\_mark);  update tours t set t.average\_mark=cast((select avg(mark) from FEEDBACKS f where f.tourId=tour\_id) as number) where t.tourId=tour\_id;  end; |

## Функции

### Получение числа стран

|  |
| --- |
| create or replace function GetCountOfCountries  return number  is  countr number(6);  begin  select COUNT(\*) into countr from COUNTRIES;  return COUNTR;  end; |

### Получение городов по странам

|  |
| --- |
| create or replace function GetCitiesByCountry(  COUNTRY\_ID char  ) return number  is  emp\_c sys\_refcursor;  v\_c\_id cities.cityId%type;  v\_c\_name cities.cityName%type;  begin  open emp\_c for  select cityId, cityName from Cities where countryId=COUNTRY\_ID;  LOOP  FETCH emp\_c  INTO v\_c\_id, v\_c\_name;  EXIT WHEN emp\_c%NOTFOUND;  DBMS\_OUTPUT.PUT\_LINE(v\_c\_id || ' - ' || v\_c\_name);  END LOOP;  CLOSE emp\_c;  return 1;  end; |

### Получение туров по определённому виду транспорта

|  |
| --- |
| create or replace function GetToursByTransport(  TRNAME varchar  ) return number  is  emp\_c sys\_refcursor;  v\_t\_name tours.tourName%type;  v\_tr\_id transports.transportId%type;  begin  select transportId into v\_tr\_id from transports where transportName=TRNAME;  open emp\_c for  select tourName from Tours where transportId=v\_tr\_id;  LOOP  FETCH emp\_c  INTO v\_t\_name;  EXIT WHEN emp\_c%NOTFOUND;  DBMS\_OUTPUT.PUT\_LINE(v\_t\_name);  END LOOP;  CLOSE emp\_c;  return 1;  end; |

## Триггеры

### Триггер обновления средней оценки при добавлении нового отзыва по туру

|  |
| --- |
| create or replace procedure SetTheFeedBack(  user\_id number,  tour\_id number,  set\_mark number)  is  begin  insert into FEEDBACKS(userId,tourId,mark) values(user\_id, tour\_id, set\_mark);  update tours t set t.average\_mark=cast((select avg(mark) from FEEDBACKS f where f.tourId=tour\_id) as number) where t.tourId=tour\_id;  end; |

### Триггер обновления стоимости тура в зависимости от числа детей и взрослых

|  |
| --- |
| create or replace procedure SetTheOrder(  tour\_id number,  user\_id number,  adults\_Count number,  children\_Count number)  is  c number;  begin  select costForOnePerson into c from TOURS t where t.tourId=tour\_id;  insert into ORDERS(tourId, userId, adultsCount, childrenCount, DateStart, cost) values(tour\_id, user\_id, adults\_Count, children\_Count, SYSDATE, (cast(c as number))\*(adults\_Count+0.5\*children\_Count));  end; |

### Триггер обновления цены тура на одного взрослого человека при добавлении скидки на тур

|  |
| --- |
| create or replace trigger setDiscount  after insert or update on DISCOUNTS  for each row  begin  UPDATE TOURS  set costForOnePerson=costForOnePerson-costForOnePerson\*:new.discountValue  where tourId=:new.tourId;  end; |

## Представления

### Представление информации о туре без идентификаторов в атрибутах

|  |
| --- |
| create or replace view DETAILED\_VIEW\_TOUR  as  select t.tourId,  t.tourName,  co.countryId,  ci.cityId,  cat.categoryName,  t.description,  t.duration,  t.costForOnePerson,  t.average\_mark,  tr.transportName,  h.hotelName  from TOURS t  join COUNTRIES co on t.countryId=co.countryId  join CITIES ci on t.cityId=ci.cityId  join CATEGORIES cat on t.categoryId=cat.categoryId  join TRANSPORTS tr on t.transportId=tr.transportId  join HOTELS h on t.hotelId=h.hotelId; |

### Представление информации о заказе без идентификаторов в атрибутах

|  |
| --- |
| create or replace view DETAILED\_VIEW\_ORDER  as  select o.orderId, u.name, u.surname, u.email,  t.tourname, o.adultsCount, o.childrenCount, o.DateStart, o.cost  from ORDERS o join USERS u on o.userId=u.userId join TOURS t on o.tourId =t.tourId; |

### Представление из имени страны, имени города и имени отеля

|  |
| --- |
| create or replace view CITY\_COUNTRY\_HOTEL  as  select ci.cityName, co.countryName, h.hotelName from CITIES ci  join COUNTRIES co on ci.countryId=co.countryId join HOTELS h on ci.cityId=h.cityId; |