**Федеральное агентство по образованию**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**Институт информационных технологий и управления**

**Кафедра компьютерных систем и программных технологий**

# ОТЧЕТ

**По лабораторной работе №3**

**Язык SQL-DML**

**Студентка гр.43501/4 Перетятько Е.В.**

**Преподаватель Мяснов А. В.**

**Санкт-Петербург**

**2015**

## 1. Цель работы

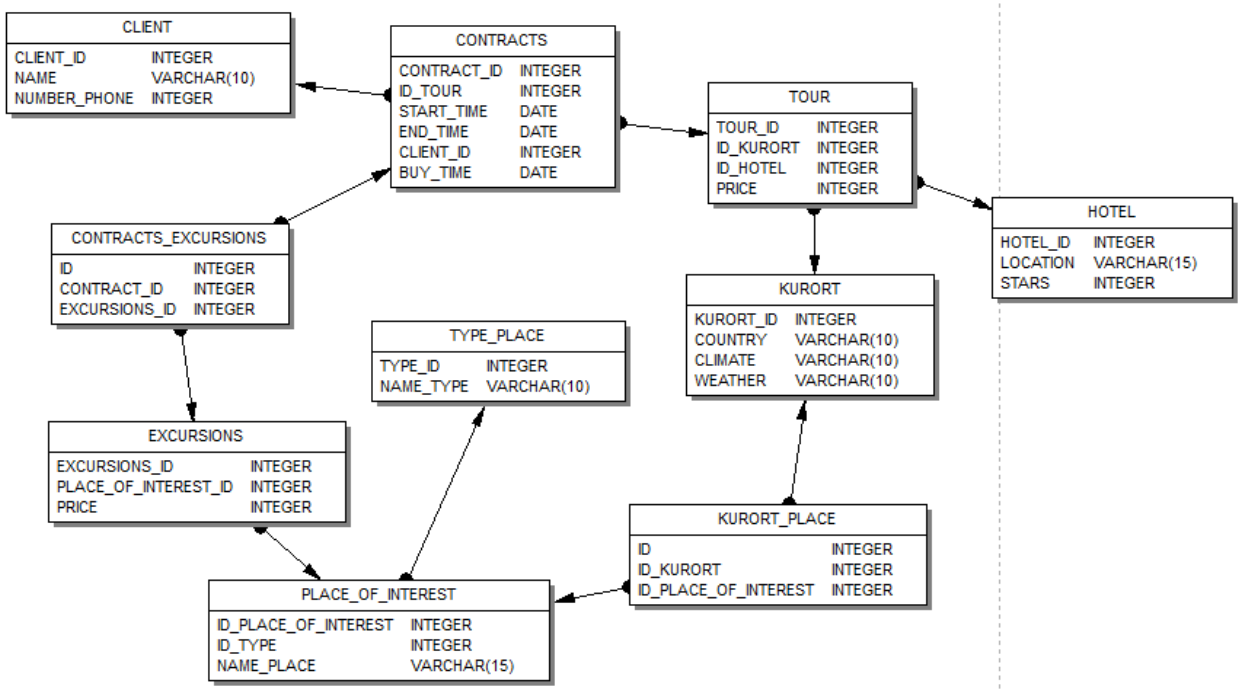
Познакомиться с языком создания запросов управления данными SQL-DML.

## 2. Программа работы

1) Изучить SQL-DML
2) Выполнить все запросы из списка стандартных запросов. Продемонстрировать результаты преподавателю.
3) Получить у преподавателя и реализовать SQL-запросы в соответствии с индивидуальным заданием. Продемонстрировать результаты преподавателю.
4) Выполненные запросы SELECT сохранить в БД в виде представлений, запросы INSERT, UPDATE или DELETE -- в виде ХП. Выложить скрипт в Subversion.

## 3. Выполнение программы работы

Была модернизирована схема. Ниже представлена ER-диаграмма, по которой выполнялись запросы:



Выполнение стандартных запросов:

1) Выборка всех данных из каждой таблицы:

**Листинг 1. Представления для выборки всех данных из каждой таблицы**

```
create view Client_Select as select * from client;

create view Contracts_Select as select * from contracts;

create view contracts_excursions_Select as select * from contracts_excursions;

create view excursions_Select as select * from excursions;

create view hotel_Select as select * from hotel;

create view kurort_Select as select * from kurort;

create view kurort_place_Select as select * from kurort_place;

create view place_of_interest_Select as select * from place_of_interest;

create view tour_Select as select * from tour;

create view ype_place_Select as select * from type_place;
```

Полученная выборка данных из таблицы place_of_interest:

| ID_PLACE_OF_INTEREST | ID_TYPE | NAME_PLACE |
|---|---|---|
| 1 | 8 | TV tower |
| 2 | 1 | Himeji castle |
| 3 | 1 | Potala Palace |
| 4 | 6 | Museum ships |
| 5 | 9 | statue of lbert |
| 6 | 5 | Marienplatz |
| 7 | 4 | temple of sun |
| 8 | 3 | red Fort |
| 9 | 7 | Royal palace |
| 10 | 10 | English garden |
| 11 | 8 | big Ben |
| 12 | 10 | Hyde park |
| 13 | 1 | Tibidabo |
| 14 | 6 | Picasso Museum |
| 15 | 8 | tower of Pisa |
| 16 | 3 | Coliseum |
| 17 | 4 | pyramid |
| 18 | 6 | national museum |
| 19 | 10 | national park |
| 20 | 8 | afalava tower |

2) Выборка данных из одной таблицы при нескольких условиях, с использованием логических операций, LIKE, BETWEEN, IN:

**Листинг 2. Выборка с использованием логических операций.**

```
create view Select_IN as select * from kurort where country IN ('London');

create view Select_BETWEEN as select * from client where client_id BETWEEN 4 AND 7;

create view Select_WHERE as select * from client WHERE Name LIKE 'Masha';
```

Полученные результаты для оператора BETWEEN (выборка клиентов со значением ID от 4 до 7)

| CLIENT_ID | NAME | NUMBER_PHONE |
|---|---|---|
| 4 | Nikita | 7 412 585 |
| 5 | Vlad | 7 458 632 |
| 6 | Ruslan | 7 258 632 |
| 7 | Anastasiya | 7 455 896 |

3) Создайте в запросе вычисляемое поле

**Листинг 3. Создание вычисляемого поля.**

```
create view CalcField as select excursions_id, price, (Price/70) as Euro from excursions;
```

Полученные результаты:

| EXCURSIONS_ID | PRICE | EURO |
|---|---|---|
| 1 | 2 600 | 37 |
| 2 | 3 800 | 54 |
| 3 | 1 200 | 17 |
| 4 | 500 | 7 |
| 5 | 1 000 | 14 |
| 6 | 15 000 | 214 |
| 7 | 9 000 | 128 |
| 8 | 4 500 | 64 |
| 9 | 2 500 | 35 |
| 10 | 3 000 | 42 |
| 11 | 5 000 | 71 |
| 12 | 8 000 | 114 |
| 13 | 10 000 | 142 |
| 14 | 1 500 | 21 |
| 15 | 12 000 | 171 |

4) Сделайте выборку всех данных с сортировкой по нескольким полям

**Листинг 4. Выборка с сортировкой.**

```
create view SortContracts as select * from contracts order by client_id desc, start_time asc;
```

Полученные результаты сортировки(контракты отсортированные по номеру клиента ID и времени начала путешествия):

| CONTRACT_ID | ID_TOUR | START_TIME | END_TIME | CLIENT_ID | BUY_TIME |
|---|---|---|---|---|---|
| 3 | 9 | 15.12.2014 | 27.12.2014 | 18 | 12.08.2014 |
| 2 | 2 | 06.03.2014 | 20.03.2014 | 17 | 02.02.2014 |
| 1 | 3 | 05.04.2014 | 19.04.2014 | 16 | 01.03.2014 |
| 4 | 8 | 07.03.2014 | 21.03.2014 | 15 | 01.02.2014 |
| 12 | 7 | 14.08.2014 | 29.08.2014 | 14 | 12.03.2014 |
| 13 | 1 | 13.10.2014 | 26.10.2014 | 13 | 01.10.2014 |
| 18 | 6 | 30.09.2015 | 20.10.2015 | 12 | 10.03.2015 |
| 14 | 5 | 29.06.2015 | 15.07.2015 | 11 | 15.02.2015 |
| 15 | 4 | 17.04.2015 | 30.04.2015 | 10 | 14.02.2015 |
| 16 | 2 | 14.12.2015 | 30.12.2015 | 9 | 10.08.2015 |
| 17 | 1 | 01.02.2014 | 15.02.2014 | 8 | 15.01.2014 |
| 5 | 7 | 22.09.2015 | 01.10.2015 | 7 | 25.07.2015 |
| 6 | 6 | 12.05.2014 | 25.05.2014 | 6 | 12.04.2014 |
| 7 | 5 | 14.11.2014 | 29.11.2014 | 5 | 01.11.2014 |
| 8 | 4 | 05.11.2014 | 14.11.2014 | 4 | 06.08.2014 |
| 9 | 3 | 18.09.2015 | 30.09.2015 | 3 | 16.06.2015 |
| 10 | 2 | 14.05.2015 | 20.05.2015 | 2 | 12.04.2015 |
| 11 | 1 | 25.07.2015 | 08.08.2015 | 1 | 02.05.2015 |

5) Создайте запрос, вычисляющий несколько совокупных характеристик таблиц

**Листинг 5.**

```
create view Max_And_Min as select max(price) as "max_price", min(price) as "min_prise" from tour;
```

Полученные результаты(максимальная и минимальная стоимость за тур):

| max_price | min_prise |
|---|---|
| 84 000 | 8 000 |

6) Сделайте выборку данных из связанных таблиц (не менее двух примеров)

**Листинг 6.**

```
create view ClientCountry as select client.client_id, client.name, kurort.country from client, contracts, tour, kurort  where client.client_id=contracts.client_id and contracts.id_tour=tour.tour_id and tour.id_kurort=kurort.kurort_id;
```

Была осуществлена выборка клиента и страны, в которую он едет.

| CLIENT_ID | NAME | COUNTRY |
|---|---|---|
| 1 | Dariya | Paris |
| 13 | Alex | Paris |
| 8 | Olga | Paris |
| 17 | Svatlana | London |
| 2 | Artem | London |
| 9 | Katya | London |
| 16 | Marina | Germany |
| 3 | Andrey | Germany |
| 4 | Nikita | Thailand |
| 10 | ElenUlyaa | Thailand |
| 5 | Vlad | China |
| 11 | Igor | China |
| 6 | Ruslan | USA |
| 12 | Pavel | USA |
| 7 | Anastasiya | Spain |
| 14 | Sasha | Spain |
| 15 | Masha | Finland |
| 18 | Nataly | Italy |

```
create view ClientPrice as select client.client_id, client.name, tour.price from client, contracts, tour where client.client_id=contracts.client_id and contracts.id_tour=tour.tour_id;
```

Выборка клиента и стоимости путевки

| CLIENT_ID | NAME | PRICE |
|---|---|---|
| 1 | Dariya | 60 000 |
| 13 | Alex | 60 000 |
| 8 | Olga | 60 000 |
| 17 | Svatlana | 32 000 |
| 2 | Artem | 32 000 |
| 9 | Katya | 32 000 |
| 16 | Marina | 54 000 |
| 3 | Andrey | 54 000 |
| 4 | Nikita | 84 000 |
| 10 | ElenUlyaa | 84 000 |
| 5 | Vlad | 24 000 |
| 11 | Igor | 24 000 |
| 6 | Ruslan | 65 000 |
| 12 | Pavel | 65 000 |
| 7 | Anastasiya | 8 000 |
| 14 | Sasha | 8 000 |
| 15 | Masha | 35 000 |
| 18 | Nataly | 60 000 |

7) Создайте запрос, рассчитывающий совокупную характеристику с использованием группировки, наложите ограничение на результат группировки

**Листинг 7.**

```
create view ClientPhone as select name as "Name", sum(number_phone) as "Phone"

from client group by name having sum(number_phone)>7856948;
```

Определение клиентов, имеющего номер телефона больше задаваемого значения

| Name | Phone |
|---|---|
| Alex | 7 951 364 |
| Nataly | 7 896 235 |

8) Придумайте и реализуйте пример использования вложенного запроса

**Листинг 8.**

```
create view View_excursions as select * from excursions where excursions_id

in (select excursions_id from contracts_excursions);
```

Определим экскурсии, на которые оформлены заказы

| EXCURSIONS_ID | PLACE_OF_INTEREST_ID | PRICE |
|---|---|---|
| 5 | 5 | 1 000 |
| 6 | 6 | 15 000 |
| 7 | 7 | 9 000 |
| 10 | 10 | 3 000 |
| 11 | 11 | 5 000 |
| 14 | 14 | 1 500 |
| 15 | 15 | 12 000 |

9) С помощью оператора INSERT добавьте в каждую таблицу по одной записи

**Листинг 9. Процедуры для добавления записей.**

```
create procedure insert_client (client_ID integer, name varchar(10), number_phone integer)

as

begin

    insert into client(client_ID, name, number_phone) values (:client_ID, :name,
:number_phone);

end;

create procedure insert_contracts (contract_ID integer, id_tour integer,  start_time date,
end_time date, client_id integer,  buy_time date)

as

begin

    insert into contracts(contract_ID, id_tour, start_time, end_time, client_id, buy_time) values
(:contract_ID, :id_tour, :start_time, :end_time, :client_id, :buy_time);
```

```
end;

create procedure insert_tour(tour_id integer, id_kurort integer, id_hotel integer, price integer)

as

begin

   insert into tour(tour_id, id_kurort, id_hotel, price) values (:tour_id, :id_kurort, :id_hotel,
:price);

end;

create procedure insert_hotel(hotel_ID integer,  location varchar(15), stars integer)

as

begin

   insert into hotel(hotel_ID, location,  stars) values (:hotel_ID, :location,  :stars);

end;

create procedure insert_kurort(kurort_ID integer,  country varchar(10),  climate varchar(10),
weather varchar(10))

as

begin

   insert into kurort(kurort_ID, country, climate, weather) values(:kurort_ID, :country,
:climate, :weather);

end;

create procedure insert_kurort_place(ID integer,  ID_kurort integer, ID_place_of_interest
integer)

as

begin

   insert into kurort_place(ID, id_kurort, id_place_of_interest) values(:ID, :id_kurort,
:id_place_of_interest);

end;

create procedure insert_place_of_interest(id_place_of_interest integer, id_type integer,
name_place varchar(15))

as

begin

   insert into place_of_interest(id_place_of_interest, id_type, name_place)
values(:id_place_of_interest, :id_type, :name_place);

end;
```

```
create procedure insert_type_place(type_id integer, name_type varchar(15))

as

begin

    insert into type_place(type_id, name_type) values(:type_id, :name_type);

end;

create procedure insert_excursions(excursions_id integer,  place_of_interest_id integer,  price integer)

as

begin

    insert into excursions(excursions_id, place_of_interest_id, price) values(:excursions_id, :place_of_interest_id, :price);

end;

create procedure insert_contract_excursions(id integer,  contract_id integer,  excursions_id integer)

as

begin

    insert into contracts_excursions(id, contract_id, excursions_id) values(:id, :contract_id, :excursions_id);

end;
```

- С помощью оператора `UPDATE` измените значения нескольких полей у всех записей, отвечающих заданному условию

```
SQL> update client set name='Sasha' where name='Masha';

SQL> select * from client;



NAME                    SURNAME                 NUMBER_PHONE   PAS

PORT_ID INSURANCE_PRICE

============================== ============================== ========== =====

======= ===============

Sasha                   Petrova                 4608345
```

```
 789345          1

Dasha                   Vetrova                 4567345

 784567          2

Ivan                    Ivanov                  5672341

 744384          3

Pavel                   Sidorov                 3848432

 734829          1

Igor                    Kuznezov                7423847

 785499          2

Anna                    Alexeeva                7348957

 783243          3

Ksenia                  Bistrova                7388957

 785667          5
```

- С помощью оператора `DELETE` удалите запись, имеющую максимальное (минимальное) значение некоторой совокупной характеристики

```
SQL> select * from hotel;


        ID LOCATION_HOTEL                  STARS LANGUAGE


========== ============================== ============
======================

=======

         1 center                         5 English


         2 center                         3 English


         3 near the sea                    5 English
```

```
        4 suburb                        4 English

        5 near the sea                  2 English


SQL> delete from hotel where stars in (select min(stars) from hotel);

SQL> select * from hotel;

        ID LOCATION_HOTEL                   STARS LANGUAGE


=========== ============================== ===========
======================

=======

        1 center                        5 English

        2 center                        3 English

        3 near the sea                  5 English

        4 suburb                        4 English
```

- С помощью оператора `DELETE` удалите записи в главной таблице, на которые не ссылается подчиненная таблица (используя вложенный запрос)

```
SQL> INSERT INTO TYPE (id, typeName) VALUES (7, 'Monument');

SQL> select * from type;
```

```
        ID TYPENAME
========== ==============================
         1 Tower
         2 Pastle
         3 Museum
         4 Palace
         5 Park
         6 Platz
         7 Monument


SQL> select id, typename from type where not exists(select * from place_of_inter
est where type.id=place_of_interest.type_id);


        ID TYPENAME
========== ==============================
         7 Monument
SQL> delete from type where not exists(select * from place_of_interest where typ
e.id=place_of_interest.type_id);
SQL> select * from type;
        ID TYPENAME
========== ==============================
         1 Tower
         2 Pastle
         3 Museum
         4 Palace
         5 Park
```

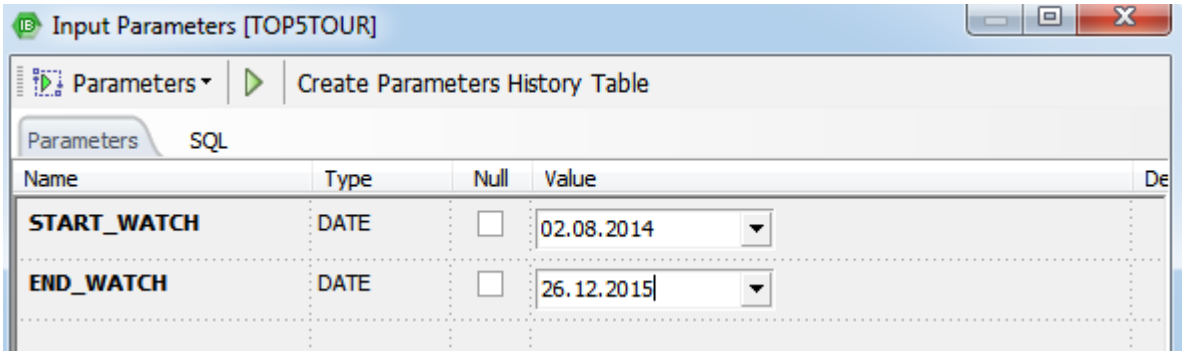
| 6 Platz |
|---|

## Выполнение индивидуального задания

1. Вывести 5 туров, по которым были максимальные продажи за выбранный период.

```
create procedure top5tour(start_watch date, end_watch date)
returns(
   Tour_ID integer, Count_Sale integer
)as
begin
   for select first 5 id_tour, count(contract_id) as count_s from contracts
   where (buy_time>=:start_watch)and(buy_time<=:end_watch)
   group by id_tour order by count_s desc into :Tour_ID, :Count_Sale do
   begin
      suspend;
   end
end
```

Был задан период с 02.08.2014 по 26.12.2015




Результат работы скрипта:

| TOUR_ID | COUNT_SALE |
|---|---|
| 1 | 2 |
| 2 | 2 |
| 4 | 2 |
| 5 | 2 |
| 3 | 1 |

2. Вывести 10 достопримечательностей, которые наиболее популярны (по экскурсиям).

```
create view top10place as select first 10 id_place_of_interest, name_type, name_place, count(contracts_excursions.excursions_id) as count_p

from contracts_excursions, excursions,  place_of_interest, type_place

where(contracts_excursions.excursions_id=excursions.excursions_id)and(excursions.place_of_interest_id=place_of_interest.id_place_of_interest)

and(place_of_interest.id_type=type_place.type_id)

group by id_place_of_interest, name_type, name_place  order by count_p desc;
```

Результат работы скрипта

| ID_PLACE_OF_INTEREST | NAME_TYPE | NAME_PLACE | COUNT_P |
|---|---|---|---|
| 5 | Monument | statue of lbert | 1 |
| 6 | Place | Marienplatz | 1 |
| 7 | Holy_place | temple of sun | 1 |
| 10 | Park | English garden | 1 |
| 11 | Tower | big Ben | 1 |
| 14 | Museum | Picasso Museum | 1 |
| 15 | Tower | tower of Pisa | 1 |

3. Удалить ненужные типы достопримечательностей.

```
create procedure DeleteUnusedType as

declare variable deltype integer;

begin

   for select type_id from type_place into :deltype do

   begin

      delete from type_place where (type_id not in(select id_type from place_of_interest where (id_type=:deltype)))and(type_id=:deltype);

   end

end
```

## Вывод

В ходе работы были изучены необходимые для работы с реляционными базами данных запросы языка SQL-DML: insert, update, delete. Изучены внутреннее и внешнее объединение таблиц, группировки и агрегатные функции, вложенные запросы, а также представления. Полученные навыки работы с SQL-запросами будут использованы в дальнейшей работе с базой данных.DML запросы имеют довольно большие возможности для манипуляции базами данных.