Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

Отчет по лабораторной работе №4

по дисциплине «Базы данных»

Язык SQL-DML

Выполнил
студент гр. 43501/3 Е.А. Никитин

Преподаватель А.В. Мяснов

«__»_____2015г.

Санкт-Петербург
2015

Цели работы

Изучение SQL-DML, ознакомление с основными методами создания запросов

Программа работы

1. Изучите SQL-DML
2. Выполните все запросы из списка стандартных запросов. Продемонстрируйте результаты преподавателю.
3. Получите у преподавателя и реализуйте SQL-запросы в соответствии с **индивидуальным** заданием. Продемонстрируйте результаты преподавателю.
4. Выполненные запросы `SELECT` сохраните в БД в виде представлений, запросы `INSERT`, `UPDATE` или `DELETE` -- в виде ХП. Выложите скрипт в Subversion.

Список стандартных запросов

- Сделайте выборку всех данных из каждой таблицы
- Сделайте выборку данных из одной таблицы при нескольких условиях, с использованием логических операций, `LIKE`, `BETWEEN`, `IN` (не менее 3-х разных примеров)
- Создайте в запросе вычисляемое поле
- Сделайте выборку всех данных с сортировкой по нескольким полям
- Создайте запрос, вычисляющий несколько совокупных характеристик таблиц
- Сделайте выборку данных из связанных таблиц (не менее двух примеров)
- Создайте запрос, рассчитывающий совокупную характеристику с использованием группировки, наложите ограничение на результат группировки
- Придумайте и реализуйте пример использования вложенного запроса
- С помощью оператора `INSERT` добавьте в каждую таблицу по одной записи
- С помощью оператора `UPDATE` измените значения нескольких полей у всех записей, отвечающих заданному условию
- С помощью оператора `DELETE` удалите запись, имеющую максимальное (минимальное) значение некоторой совокупной характеристики
- С помощью оператора `DELETE` удалите записи в главной таблице, на которые не ссылается подчиненная таблица (используя вложенный запрос)

Ход работы:

1) **SQL** — формальный непроцедурный язык программирования, применяемый для создания, модификации и управления данными в произвольной реляционной базе данных, управляемой соответствующей системой управления базами данных (СУБД).

В SQL определены два подмножества языка:

- **SQL-DDL** (Data Definition Language) - язык определения структур и ограничений целостности баз данных. Сюда относятся команды создания и удаления баз данных; создания, изменения и удаления таблиц; управления пользователями и т.д.
- **SQL-DML** (Data Manipulation Language) - язык манипулирования данными: добавление, изменение, удаление и извлечение данных, управления транзакциями

2) Выполнить все запросы из списка стандартных запросов.

Сделайте выборку всех данных из каждой таблицы

```
create view car_typeV as select * from car_type;
```

```
create view ordersV as select * from orders;

create view passengerV as select * from passenger;

create view placev as select * from place;

create view place_typeV as select * from place_type;

create view route_stationV as select * from route_station;

create view sheduleV as select * from shedule;

create view stationV as select * from station;

create view ticketV as select * from ticket;

create view trainV as select * from train;

create view t_orderV as select * from t_order;

commit;
```

ROUTE_ID	TRAIN_ID	NAME	R_DATE
29	75	722A Moscow-SPB	12:15 24.12.2015
125	77	138 Moscow-N_Novgorod	23:01 25.12.2015
158	79	592 SPB-Kazan	16:53 26.12.2015
232	78	235 Vladimir-SPB	01:33 24.12.2015
38	76	199 Moscow-Vladimir	10:15 23.12.2015

- Сделайте выборку данных из одной таблицы при нескольких условиях, с использованием логических операций, LIKE, BETWEEN, IN (не менее 3-х разных примеров)

```
create view stationVUSL as select * from station where st_id like 1;

create view passengerVUSL as select * from passenger where

pas_id between 36 and 38;

create view trainVUSL as select * from train where train_id in (75, 76);

commit;
```

Выборка данных из таблицы station при условии, что st_id=1

ST_ID	NAME
1	MOSCOW

Выборка данных из таблицы passenger при соответствии значений pas_id от 36 до 38.

PAS_ID	NAME	SURNAME	DOCUMENT	DOC_NUM
36	Petr	Petrov	passport	9408244244
37	Alexander	Belov	militaryID	45286513
38	Vladimir	Krasnov	passport	9410235235

Выборка данных из таблицы train при соответствии значений train_id 75 или 76.

TRAIN_ID	TYPE	CARRIAGES
75	2EL5	15
76	2EL5	12

- Создайте в запросе вычисляемое поле

Для примера сложим значения полей route_id и st_id таблицы route_station:

```
create view vich_pole as

select route_id, st_id, route_id+st_id as summ from route_station;

commit;
```

Результат:

ROUTE_ID	ST_ID	SUMM
29	1	30
29	2	31
158	2	160
158	5	163
125	1	126
125	3	128
232	4	236
232	2	234
29	1	30
29	2	31
158	2	160
158	5	163
125	1	126
125	3	128
232	4	236
232	2	234
38	1	39
38	4	42

- Сделайте выборку всех данных с сортировкой по нескольким полям

```
create view v_sort as  select * from route_station

order by route_id asc, st_id asc;

commit;
```

Результат:

ROUTE_ID	ST_ID	ST_FUNC
29	1	start
29	1	start
29	2	finish
29	2	finish
38	1	start
38	4	finish
125	1	start
125	1	start
125	3	finish
125	3	finish
158	2	start
158	2	start
158	5	finish
158	5	finish
232	2	finish
232	2	finish
232	4	start
232	4	start

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

```
create view svyaz_tables as select route_station.st_func as func,

 station.name as name, route_station.num_in_route as num

 from route_station, station

 where route_station.st_id=station.st_id;
```

```
create view svyaz_tables1 as select place.carriage as carriage,

place.num_in_car as num, ticket.status, train."TYPE" from ticket, place, train

where place.place_id=ticket.place_id and train.train_id=place.train_id;

commit;
```

Результаты:

FUNC	NAME	NUM
start	MOSCOW	1
finish	SPB	4
start	SPB	1
finish	Kazan	29
start	MOSCOW	1
finish	N_Novgorod	11
start	Vladimir	1
finish	SPB	12
start	MOSCOW	1
finish	SPB	4
start	SPB	1
finish	Kazan	29
start	MOSCOW	1
finish	N_Novgorod	11
start	Vladimir	1
finish	SPB	12
start	MOSCOW	1
finish	Vladimir	17

CARRIAGE	NUM	STATUS	TYPE
1	25	buy	2EL5
3	3	wait	2EL5
3	20	buy	KZ4A
1	36	wait	VL40U
5	12	buy	E5K

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

Расчет количества билетов по начальной станции.

```
create view grouping as select  ticket.start_st as start_st,

count(ticket.start_st) as nums from ticket group by ticket.start_st;

commit;
```

Результат:

START_ST	NUMS
Moscow	3
SPB	1
Vladimir	1

- Придумайте и реализуйте пример использования вложенного запроса

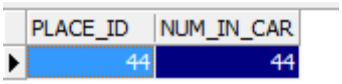
Вывод мест в поезде с id=75, на которые еще не оформлены билеты:

```
create view vlozh as select

place_id as place_id, num_in_car as num_in_car from place

where place.place_id not in (select place_id from ticket) and place.train_id=75;

commit;
```



PLACE_ID	NUM_IN_CAR
44	44

- С помощью оператора `INSERT` добавьте в каждую таблицу по одной записи

Создание процедур:

```
create procedure ins_station (i INTEGER, n char(255)) as begin

insert into station (st_id, name) values (:i, :n);

end;


create procedure ins_passenger (i INTEGER, n char(255), s char(255), d char(255),

num char(255)) as begin

insert into passenger (pas_id, name, surname, document, doc_num)

values (:i, :n, :s, :d, :num);

end;


create procedure ins_train (i INTEGER, t char(255), c INTEGER) as begin

insert into train (train_id, "TYPE", carriages) values (:i, :t, :c);

end;
```

```
create procedure ins_place (i INTEGER, ti integer, c integer,
num INTEGER, ct integer, pt integer) as begin
insert into place (place_id, train_id, carriage, num_in_car, car_type_id,
place_type_id) values (:i, :ti, :c, :num, :ct, :pt);
end;

create procedure ins_ticket (tn INTEGER, i integer, status char(255),
sta char(255), sto char(255)) as begin
insert into ticket (t_num, place_id, status, start_st, stop_st)
values (:tn, :i, :status, :sta, :sto);
end;

create procedure ins_shedule (ri INTEGER, ti integer, n char(255),rd char(255))
as begin
insert into shedule (route_id, train_id, name, r_date)
values (:ri, :ti, :n, :rd);
end;

create procedure ins_r_station (si INTEGER, ri integer, f char(255),
stime char(255), num integer)
as begin
insert into route_station (route_id, st_id, st_func, stop_time, num_in_route)
```

```
values (:si, :ri, :f, :stime, :num);

end;


create procedure ins_t_order (oi INTEGER, tnum integer, tsnum integer)

as begin

insert into t_order (order_id, t_num, tickets_num)

values (:oi, :tnum, :tsnum);

end;


create procedure ins_orders (i INTEGER, oi integer)

as begin

insert into orders (pas_id, order_id)

values (:i, :oi);

end;
```

Использование процедур для заполнения:

```
execute procedure ins_passenger

(34, 'Sergey', 'Smirnov', 'passport', '9809999999');

execute procedure ins_station (100002, 'Tver');

execute procedure ins_train (74, 'E5K', 12);

execute procedure ins_place (78, 74, 2, 24, 0, 0);

execute procedure ins_ticket (88, 78, 'buy', 'Moscow', 'Tver');

execute procedure ins_shedule (177, 74, '654 Moscow-Tver', '27.12.2015');
```

```
execute procedure ins_r_station (177, 1, 'start', '-', 1);

execute procedure ins_t_order (7546, 88, 1);

execute procedure ins_orders (34, 7546);

commit;
```

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

Создадим процедуру, которая изменяет имя и фамилию пассажиров, чей id от 55 до 63:

```
create procedure udpdt (n char(255), s char(255)) as begin

update passenger set name = :n, surname = :s

where passenger.pas_id between 55 and 63;

end;
```

Используем процедуру:

```
execute procedure udpdt ('Karl', 'Marks');

commit;
```

54	cyD$J<:#	6#Jh@J"tGCE{p}
55	Karl	Marks
56	Karl	Marks
57	Karl	Marks
58	Karl	Marks
59	Karl	Marks
60	Karl	Marks
61	Karl	Marks
62	Karl	Marks
63	Karl	Marks

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

Удалим запись с входным параметром-именем n и минимальным id из таблицы passenger:

```
create procedure del1 (n char(255)) as begin

delete from passenger where name= :n and pas_id=(select MIN(pas_id)

from passenger where name= :n);

end;
```

Использование процедуры:

```
execute procedure del1('Karl');

commit;
```

54	cyD$J<:#	6#]h@J"t
56	Karl	Marks
57	Karl	Marks
58	Karl	Marks
59	Karl	Marks
60	Karl	Marks
61	Karl	Marks
62	Karl	Marks
63	Karl	Marks

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

Удалим пользователей, у которых нет заказов. Сейчас в таблице passenger 100000 записей.

```
create procedure del2 as begin

delete from passenger where pas_id

not in (select pas_id from orders);

end;
```

Используем процедуру:

```
execute procedure del2;

commit;
```

PAS_ID	NAME	SURNAME	DOCUMENT	DOC_NUM
34	Sergey	Smirnov	passport	9809999999
35	Ivan	Ivanov	passport	9408255255
36	Petr	Petrov	passport	9408244244
37	Alexander	Belov	militaryID	45286513
38	Vladimir	Krasnov	passport	9410235235
39	Dmitriy	Chernov	passport	9411288288

Задания, выданные преподавателем:

1. Удалить неиспользуемые станции.

   Сейчас в таблице station 100001 запись. Удалим не использующиеся:

   ```
   create procedure del_station as begin
   delete from station where st_id
   not in (select st_id from route_station);
   end;
   ```

   Используем процедуру:

   ```
   execute procedure del_station;
   commit;
   ```

   Результат:

ST_ID	NAME
1	MOSCOW
2	SPB
3	N_Novgorod
4	Vladimir
5	Kazan
100 001	Tver

2. Отобразить 10 наиболее популярных маршрутов.

   ```
   create view top10routes as select first 10

   shedule.name as name, place.train_id as train_id, count(place.train_id) as nums

   from shedule, place

   where shedule.train_id=place.train_id group by shedule.name, place.train_id

   order by nums desc;
   ```

commit;

Результат:

NAME	TRAIN_ID	NUMS
234 SPB-Ekaterinburg	883	90
098 Krasnoyrask-Chelyabinsk	222	88
592 SPB-Kazan	79	85
743 Voronezh-Perm	223	83
475A Omsk-Moscow	99	81
475 Moscow-Omsk	99	81
722A Moscow-SPB	75	79
654 Moscow-Tver	74	77
985 SPB-Novosibirsk	555	76
199 Moscow-Vladimir	76	75

3. Отобразить 5 станций, на которых больше всего входят и выходят пассажиры.

create view top5stations as select first 5

station.name, (count(ticket.start_st) + count(ticket.stop_st))/2 as nums

from station, ticket

where station.name=ticket.start_st or station.name=ticket.stop_st

group by station.name order by nums desc;

commit;

Заказанные билеты:

PLACE_ID	STATUS	START_ST	STOP_ST
25	buy	Moscow	SPB
128	buy	Moscow	N_Novgorod
244	buy	SPB	Kazan
36	wait	Vladimir	SPB
111	wait	Moscow	Vladimir
78	buy	Moscow	Tver
380	buy	Kazan	Moscow

Результат:

NAME	NUMS
Moscow	5
SPB	3
Kazan	2
Vladimir	2
N_Novgorod	1

Вывод.

В ходе лабораторной работы был изучен язык SQL-DML и основные методы создания запросов, использование условий, логических операций.

Также был приобретен опыт работы с хранимыми процедурами, которые позволяют использовать скрипты написанные ранее командой вызова процедуры, что удобно, если процедуру нужно выполнить много раз (не нужно открывать скрипт и выполнять вручную).

Были изучены представления, которые можно сформировать в удобном для себя виде и которые будут хранить нужные на данный момент данные. Представление является отображением данных из основных таблиц, сформированным в нужном виде.