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

Отчёт о лабораторной работе №4

Дисциплина: Базы данных

Тема: Язык SQL-DML

Выполнил студент гр. 43501/1 _____ Нгуен Тиен Ву
(подпись)

Руководитель _____ А.В. Мяснов
(подпись)

“__” _____2015 г.

Санкт-Петербург

2015

Задание

1. **Цель работы**

Познакомить студентов с языком создания запросов управления данными SQL-DML.

2. **Программа работы**

Изучите SQL-DML

1.Выполните все запросы из списка стандартных запросов. Продемонстрируйте результаты преподавателю.

2.Получите у преподавателя и реализуйте SQL-запросы в соответствии с **индивидуальным** заданием. Продемонстрируйте результаты преподавателю.

3.запросы SELECT сохраните в БД в виде представлений, запросы INSERT, UPDATE или DELETE -- в виде ХП. Выложите скрипт в Subversion.

3. **Язык SQL**

Язык SQL (Structured Query Language) -- язык структурированных запросов. Он позволяет формировать весьма сложные запросы к базам данных. 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

4.Выполнение программы работы

3.1. Стандартные запросы

1) Выборка всех данных из каждой таблицы:

```
SELECT * FROM ADRESS;
```

```
SELECT * FROM CASE;
SELECT * FROM CLIENT;
SELECT * FROM INSURANCE_CASE_CON;
SELECT * FROM INSURANCE_CASE_POLIC;
SELECT * FROM POLIC;
SELECT * FROM PRICE;
SELECT * FROM SIGN_CONTRACT;
SELECT * FROM SUM;
SELECT * FROM TERM;
SELECT * FROM TYPE;
```

Результат для : ADRESS

ID_ADR...	CITY	STREET	BUILDING	ROOM
4	SAINT - PETERSBURG	GRAZHDANSKIY PR	30	123
2	SAINT - PETERSBURG	HEVSKIY PR	12	211
3	SAINT - PETERSBURG	KAZANCKAYA UL	32	642
1	SAINT - PETERSBURG	PROS-POPOVA UL	43	321

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

```
SELECT * FROM client
WHERE
name like '%NGUYEN%';
```

Результат:

ID_CLIENT	NAME	BIRTH_DAY	ADRESS
1 235	NGUYEN TIEN VU	08.04.1993	3

Из таблицы CLIENTS показать клиентов с фамилией NGUYEN

```
select * from "SUM"
```

ID_SUM	SUM_PAY
32	80.000 RUB
33	120.000 RUB
34	220.000 RUB
35	350.000 RUB

```
where
id_sum between 32 and 35;
```

Результат:

Из таблицы SUM показать id_sum от 32 до 35

```
select * from "CASE"
where
id_case in(11,13);
```

ID_CASE	CASE_IN
11	CAR
13	MEDICINE

Результат:

Из таблицы "CASE" показать id_case, которые расположены на площадках = 11 или 13

3) Выборка всех данных с сортировкой по нескольким полям:

```
select * from polic
order by insurance_case desc , type ;
```

Результат:

ID_POLIC	TYPE	TERM	INSURANCE_C...	ESTIMATE_DAMAGES
3	52	42	4	GOOD
4	52	42	4	INVALID
2	52	41	2	BED
1	51	42	1	NORMAL

Мы отсортировали уменшения по страховую состоянию и типе

4) Запрос, вычисляющий несколько совокупных характеристик таблиц:

```
select MAX(Price_ins.price_insurance) from Price_ins;
```

Результат:

MAX
19 000

5) Выборка данных из связанных таблиц:

```
select client.name, adress.city , sign_contract.id_contract
from client, adress , sign_contract
where client.adress = adress.id_adress and
sign_contract.client = client.id_client;;
```

Результат:

NAME	CITY	ID_CONTRACT
WAYNER ROONEY	SANKT - PETERSBURG	123 456
NGUYEN TIEN VU	SANKT - PETERSBURG	123 457
DAVID BECKHAM	SANKT - PETERSBURG	123 458
TIGER GOOD	SANKT - PETERSBURG	123 468
ADREI ARSHAVIN	SANKT - PETERSBURG	123 459
YURI ZHIRIKOV	MOSCOW	123 460
ALAN DZAGOV	MOSCOW	123 461
PAVEL MAMAYEV	MOSCOW	123 463
VIKTOR VASIN	MOSCOW	123 462

6) Запрос, рассчитывающий совокупную характеристику с использованием группировки, наложите ограничение на результат группировки:

```
select client.name, count(sign_contract."TERM") from sign_contract , client where
client.id_client = sign_contract.client
Group by client.name;
```

Результат:

NAME	COUNT
ADREI ARSHAVIN	1
ALAN DZAGOV	1
DAVID BECKHAM	1
NGUYEN TIEN VU	1
PAVEL MAMAYEV	1
TIGER GOOD	1
VIKTOR VASIN	1
WAYNER ROONEY	1
YURI ZHIRIKOV	1

7) Пример использования вложенного запроса:

```
select * from client where
client.adress in ( select polic.id_polic from polic);
```

Результат:

ID_CLIENT	NAME	BIRTH_DAY	ADRESS
1 234	WAYNER ROONEY	08.02.1985	4
1 235	NGUYEN TIEN VU	08.04.1993	3
1 236	DAVID BECKHAM	23.12.1987	2
1 237	TIGER GOOD	11.12.1977	2
1 238	ADREI ARSHAVIN	29.05.1981	1
1 239	YURI ZHIRIKOV	12.04.1987	5
1 240	ALAN DZAGOV	11.09.1993	6
1 242	VIKTOR VASIN	23.02.1989	8
1 243	PAVEL MAMAYEV	19.09.1992	7
1 244	DENIS GLUSHAKOV	22.11.1989	9

8) Добавление в каждую таблицу по одной записи с помощью оператора INSERT:

```
insert into adress values (11, 'moscow' , 'nizino', 4,212);
```

9) Изменение значения нескольких полей у всех записей, отвечающих заданному условию, с помощью оператора UPDATE:

```
update sum
set sum_pay = sum_pay + 5000
where sum_pay < 50000;
```

10) Удаление записи, имеющей максимальное (минимальное) значение некоторой совокупной характеристики, с помощью оператора DELETE:

```
DELETE FROM ADRESS
WHERE ADRESS.ID_ADRESS = ( select POLIC.ID_POLIC select from polic);
```

11) Удаление записей в главной таблице, на которые не ссылается подчиненная таблица (используя вложенный запрос), с помощью оператора DELETE:

```
DELETE FROM client
WHERE clien.id_client not in
(SELECT sign_contract.Id_contract FROM sign_contract);
```

3.2. **Индивидуальные задания**

Реализовать следующие запросы:

1. Вывести суммарные выплаты по страховкам за выбранный период.
2. Вывести 5 типов страховых случаев, по которым были максимальные выплаты за выбранный период.
3. Удалит неиспользуемые типы полисов

Вывести суммарные выплаты по страховкам за выбранный период.

```
CREATE OR ALTER VIEW SUM_PRICE(
    SUMPRICE)
AS
select sum(s1) as SUMPRICE from (select "SUM".SUM_PAY *
COUNT(INSURANCE_CASE_CON.CASE)
 AS s1 from INSURANCE_CASE_CON, "SUM" where INSURANCE_CASE_CON.SUM =
"SUM".ID_SUM
 group BY INSURANCE_CASE_CON.SUM)
```

;Результат:

SUMPRICE
199 000

```
Plan
PLAN JOIN (TERM NATURAL, SIGN_CONTRACT INDEX (FK_SIGN_CONTRACT_4), PRICE_INS INDEX (PK_PRICE_INS))

------ Performance info ------
Prepare time = 31ms
Execute time = 0ms
Avg fetch time = 0,00 ms
Current memory = 9 869 512
Max memory = 9 951 608
Memory buffers = 2 048
Reads from disk to cache = 0
Writes from cache to disk = 0
Fetches from cache = 74
```

Вывести 5 типов страховых случаев, по которым были максимальные выплаты за выбранный период.

```
CREATE OR ALTER VIEW MAXPRICE(
    NAME_POLIC,
    COUNTER)
AS
select first 5 INSURANCE_CASE_CON.CASE, "SUM".SUM_PAY *
COUNT(INSURANCE_CASE_CON.CASE)
 AS COUNTER from INSURANCE_CASE_CON, "SUM", "CASE"where
INSURANCE_CASE_CON.SUM = "SUM".ID_SUM and
INSURANCE_CASE_CON.SUM = "CASE".ID_CASE

 group BY CASE.CASE_IN,SUM.SUM_PAY order by COUNTER desc
```

;Результат:

CASE_IN	COUNTER
ALL	60 000
TRA+ACC	60 000
PRO+MED	26 000
CAR+PRO	24 000
CAR	15 000

1) Удалить неиспользуемые типы абонементов

```
begin

   delete POLIC.TYPE from POLIC where TYPE.ID_TYPE not in (select
TYPE.ID_TYPE from TYPE);

end;
```

1. **Выводы:**

В результате выполнения работы был изучен язык управления данными SQL-DML. Были выполнены стандартные запросы извлечения данных. Также были выполнены запросы в соответствии с индивидуальным заданием. Были изучены представления и хранимые процедуры, с помощью которых можно спокойно добавлять данные в БД. При выполнении работы проблем не было.

Использовались такие команды языка DML: insert (добавить), update (обновить), delete (удалить), select (выборка данных). Данный язык удобен для написания запросов разной сложности. При обращении к нескольким таблицам в запросе следует выбирать подходящую связь между таблицами для более быстрого выполнения SQL-запроса.