

Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

**Отчёт о лабораторной работе**

**Дисциплина:** Базы данных

**Тема:** Язык SQL-DML

Выполнил студент гр. 43501/1

Дроздовский А.А.

Руководитель

Мяснов А.В.

Санкт –Петербург

2016

## 1. Цель работы

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

## 2. 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

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

## 3. Ход работы

### 3.1. Выполнение стандартных запросов

#### Выборка всех данных из каждой таблицы

```
connect 'D:\BD\Lab4\Kinopoisk.fdb' user 'SYSDBA' password 'masterkey';
commit;

select * from Films;
select * from Worker;
select * from Film_Worker;
select * from Film_Studia;
select * from Studia;
select * from Film_Studia;
select * from Country_film;
select * from Country;
select * from Film_Genre;
select * from Genre;
select * from UserK;
select * from Rate_userK;
select * from critic;
select * from Rate_critic;
select * from Collections;
select * from period;
```

Результат работы представлен ниже:

SQL> select \* from Films;

ID	NAME	YEARS	TAGLINE	AGE_LIMIT	TIMES	BUDGET
1	The Shawshank Redemption	1994	Fear - it shackles. Hope - is freedom	16	142	2500000
2	The Green Mile	1999	Paul Edgecombe did not believe in miracles. While not faced with one of them.	16	189	60000000
3	Forrest Gump	1994	The world will never be the same after you see it through the eyes of Forrest Gump	12	142	55000000
4	Schindlers List	1993	This list of - life	16	195	22000000
5	Intouchables	2011	Sometimes you have to reach into someone else is world to find out what is missing in your own	16	112	9500000

SQL> select \* from Worker;

ID	NAME	DATE_OF_BIRTHDAY	SEX	ID_COUNT
1	Olivier Nakache	1973-04-15	M	4
2	Eric Toledano	1971-07-03	M	4
3	Frank Darabont	1959-01-28	M	4
4	François Cluzet	1955-09-21	M	4
5	Omar Sy	1978-01-20	M	4
6	Anne Le Ny	1962-12-16	W	4
7	Tim Robbins	1958-10-16	M	3
8	Morgan Freeman	1937-06-01	M	3

SQL> select \* from Film\_Worker;

ID_FP	ID_FILM	ID_WORKER	SALARY	ID_PROF
1	5	1	1500	1
2	5	2	2000	1
3	1	3	4300	1
4	5	1	5790	2
5	5	2	1333	2
6	5	3	890	2
7	1	4	1290	2
8	1	5	1780	2

SQL> select \* from Film\_Studia;

ID	ID_FILM	ID_NAME
1	1	1
2	2	1

SQL> select \* from Studia;

ID	NAME
1	Warner brothers
2	Marvel studios
3	21st century fox
4	Universal Studios
5	Pixar

SQL> select \* from Film\_Studia;

ID	ID_FILM	ID_NAME
1	1	1
2	2	1

SQL> select \* from Country\_film;

ID_COU	ID_FILM	ID_COUNTRY
1	1	3
2	2	3
3	3	3
4	4	3
5	5	4

```
SQL> select * from Country;
```

ID_COUNTRY	COUNTRY
1	Russia
2	Italy
3	USA
4	France
5	Latvia
6	Ukraine
7	Germany
8	China
9	Canada
10	Brazil

```
SQL> select * from Film_Genre;
```

ID	ID_FILM	ID_GENRE
1	1	4
2	1	5
3	2	4
4	2	5
5	2	11
6	3	4
7	3	3
8	4	4
9	5	4
10	5	3
11	5	12
12	4	12

```
SQL> select * from Genre;
```

ID	NAME
1	action
2	adventure
3	comedy
4	drama
5	crime
6	horror
7	fantasy
8	thriller
9	family
10	documentary
11	detective
12	biography
13	animation

```
SQL> select * from UserK;
```

ID_USERK	NAME	SEX	DATE_OF_BIRTHDAY	ID_COUNTRY
1	White	M	1996-05-26	1
2	Alexey_D	M	1976-12-31	1
3	Veronica	W	1969-08-19	1
4	Diego	M	1993-08-29	10
5	Masha	W	1995-07-04	1
6	Janna	W	1964-12-08	5

```
SQL> select * from critic;
```

ID_CRITIC	NAME	SEX	DATE_OF_BIRTHDAY	ID_COUNTRY
1	Alex	M	1996-05-26	1
2	ZLO	M	1976-12-31	1
3	Jobs	W	1969-08-19	1
4	BigRussianBoss	M	1993-08-29	10
5	Nastya	W	1995-07-04	1
6	John Cena	W	1964-12-08	5

## Выборка данных из таблицы при нескольких условиях

Выведем имя, пол (M) и дату рождения (старше 1995 года) из таблицы критиков;

Выведем название, год, возрастное ограничение (16) и время (от 140 до 190 минут) из таблицы фильмов;

Выведем имя (в котором есть «А» или «а») и id страны(1 – Россия) из таблицы пользователей:

```
select NAME, SEX, DATE_OF_BIRTHDAY from critic where DATE_OF_BIRTHDAY < '31.12.1995' and SEX in('M');
select NAME, YEARS, AGE_LIMIT, TIMES from Films where AGE_LIMIT in(16) and TIMES between 140 and 190;
select NAME, ID_COUNTRY from USERK where (NAME like('%a%') or Name like('%A%')) and ID_COUNTRY in(1);
```

Результат работы:

NAME	SEX	DATE_OF_BIRTHDAY
ZLO	M	1976-12-31
BigRussianBoss	M	1993-08-29

  

NAME	YEARS	AGE_LIMIT	TIMES
The Shawshank Redemption	1994	16	142
The Green Mile	1999	16	189

  

NAME	ID_COUNTRY
Alexey_D	1
Veronica	1
Masha	1

## Вычисляемое поле в запросе

Выведем сумму, потраченную на зарплату работников в одном из фильмов:

```
select sum(SALARY) from Film_Worker where id_film in(1);
```

Результат работы:

SUM
7370

## Выборка всех данных с сортировкой по нескольким полям

Отсортируем таблицу пользователей сначала по id страны, затем по именам:

```
select NAME, SEX, DATE_OF_BIRTHDAY, ID_COUNTRY from USERK order by ID_COUNTRY, Name;
```

Результат работы:

NAME	SEX	DATE_OF_BIRTHDAY	ID_COUNTRY
Alexey_D	M	1976-12-31	1
Masha	W	1995-07-04	1
Veronica	W	1969-08-19	1
White	M	1996-05-26	1
Janna	W	1964-12-08	5
Diego	M	1993-08-29	10

## Запрос, вычисляющий несколько совокупных характеристик таблиц

Выведем таблицу с минимальной, средней и максимальной зарплатой работников:

```
select min(SALARY) as Minimum, avg(SALARY) as Average, max(SALARY) as Maximum from Film_Worker;
```

Результат работы:

MINIMUM	AVERAGE	MAXIMUM
890	2360	5790

## Выборка данных из связанных таблиц

Выведем название фильма, имя работника и зарплату (больше 1300\$);

Выведем имя пользователя из России:

```
select Films.Name, Worker.Name, Film_Worker.salary
from Films, Film_Worker, Worker
where Film_Worker.salary > 1300 and
Films.ID=Film_Worker.ID_film
and Film_Worker.ID_worker=Worker.ID;

select USERK.Name, COUNTRY.COUNTRY
from USERK, COUNTRY
where COUNTRY.id_country in(1) and
USERK.ID_COUNTRY=COUNTRY.ID_COUNTRY;
```

Результат работы:

NAME	NAME	SALARY
The Shawshank Redemption	Frank Darabont	4300
The Shawshank Redemption	Omar Sy	1780
Intouchables	Olivier Nakache	1500
Intouchables	Eric Toledano	2000
Intouchables	Olivier Nakache	5790
Intouchables	Eric Toledano	1333

  

NAME	COUNTRY
White	Russia
Alexey_D	Russia
Veronica	Russia
Masha	Russia

## Запрос, рассчитывающий совокупную характеристику с использованием группировки

Сгруппируем имя критика и дату рождения, если он старше 1990 года:

```
select Name, min(AGE)
from critic group by Name having min(AGE) > 1990;
```

Результат работы:

ID_CRITIC	NAME	SEX	DATE_OF_BIRTHDAY	ID_COUNTRY
2	ZLO	M	1976-12-31	1
3	Jobs	W	1969-08-19	1
4	BigRussianBoss	M	1993-08-29	10
5	Nastya	W	1995-07-04	1
6	John Cena	W	1964-12-08	5

  

```
SQL> select Name, min(AGE)
CON> from critic group by Name having min(AGE) > 1990;
```

NAME	MIN
Jobs	1969-08-19
John Cena	1964-12-08
ZLO	1976-12-31

## Использование вложенного запроса

Выберем работников, которые принимали участие в создании фильма, в название которого есть «Shawshank»:

```
select worker.name from worker, Film_Worker
where Film_Worker.id_film in
(select ID from films where Name like('%Shawshank%'))
and Film_Worker.id_worker=worker.id;
```

Результат работы:

```
NAME
=====
Frank Darabont
Frank@ois Cluzet
Omar Sy
```

## Добавление записей в таблицы

С помощью оператора INSERT добавим записи:

```
--Страна:
insert into Country values(11, 'Belarus');
--Студия:
insert into Studio values(6, 'Columbia Pictures');
--Работник:
insert into Worker values (9, 'Jennifer Lawrence', '15.08.1990', 'W', 3);
```

**Измените значений нескольких полей у всех записей, отвечающих заданному условию**

Заменяем имя пользователя в таблице пользователей на Имя+«PRO», если он мужского пола:

```
select * from USERK;
update USERK set NAME= NAME || 'PRO' where SEX in('M');
select * from USERK;
```

Результат работы:

ID_USERK	NAME	SEX	DATE_OF_BIRTHDAY	ID_COUNTRY
1	White	M	1996-05-26	1
2	Alexey_D	M	1976-12-31	1
3	Veronica	W	1969-08-19	1
4	Diego	M	1993-08-29	10
5	Masha	W	1995-07-04	1
6	Janna	W	1964-12-08	5

```
SQL> update USERK set NAME=NAME || 'PRO' where SEX in('M');
SQL> select * from USERK;
```

ID_USERK	NAME	SEX	DATE_OF_BIRTHDAY	ID_COUNTRY
1	WhitePRO	M	1996-05-26	1
2	Alexey_DPRO	M	1976-12-31	1
3	Veronica	W	1969-08-19	1
4	DiegoPRO	M	1993-08-29	10
5	Masha	W	1995-07-04	1
6	Janna	W	1964-12-08	5

## Удаление записи, имеющей максимальное (минимальное) значение некоторой совокупной характеристики

Удалим самого молодого критика:

```
select * from critic;
delete from critic where DATE_OF_BIRTHDAY = (select max(DATE_OF_BIRTHDAY) from critic);
select * from critic;
```

Результат работы:

ID_CRITIC	NAME	SEX	DATE_OF_BIRTHDAY	ID_COUNTRY
1	Alex	M	1996-05-26	1
2	ZLO	M	1976-12-31	1
3	Jobs	W	1969-08-19	1
4	BigRussianBoss	M	1993-08-29	10
5	Nastya	W	1995-07-04	1
6	John Cena	W	1964-12-08	5

```
SQL> delete from critic where DATE_OF_BIRTHDAY = (select max(DATE_OF_BIRTHDAY) from critic);
SQL> select * from critic;
```

ID_CRITIC	NAME	SEX	DATE_OF_BIRTHDAY	ID_COUNTRY
2	ZLO	M	1976-12-31	1
3	Jobs	W	1969-08-19	1
4	BigRussianBoss	M	1993-08-29	10
5	Nastya	W	1995-07-04	1
6	John Cena	W	1964-12-08	5

## Удаление записи в главной таблице, на которые не ссылается подчиненная таблица

Удалим работника из списка, который не принимал участия в создании фильмов:

```
select * from WORKER;
delete from WORKER where ID not in
(select ID_worker from Film_Worker);
select * from WORKER;
```

Результат работы:

ID	NAME	DATE_OF_BIRTHDAY	SEX	ID_COUNT
1	Olivier Nakache	1973-04-15	M	4
2	Eric Toledano	1971-07-03	M	4
3	Frank Darabont	1959-01-28	M	4
4	François Cluzet	1955-09-21	M	4
5	Omar Sy	1978-01-20	M	4
6	Anne Le Ny	1962-12-16	W	4
7	Tim Robbins	1958-10-16	M	3
8	Morgan Freeman	1937-06-01	M	3
9	Jennifer Lawrence	1990-08-15	W	3

```
SQL> delete from WORKER where ID not in
CON> (select ID_worker from Film_Worker);
SQL> select * from WORKER;
```

ID	NAME	DATE_OF_BIRTHDAY	SEX	ID_COUNT
1	Olivier Nakache	1973-04-15	M	4
2	Eric Toledano	1971-07-03	M	4
3	Frank Darabont	1959-01-28	M	4
4	François Cluzet	1955-09-21	M	4
5	Omar Sy	1978-01-20	M	4



### 3.2. Индивидуальное задание

**Вывести 10 человек, которые поучаствовали в наибольшем количестве фильмов в разных ролях**

```
select first 10 worker.name, sum(film_worker.kol) as quantity
from worker
join film_worker on (film_worker.id_worker=worker.id)
group by worker.name
order by quantity desc;
```

Результат работы:

NAME	QUANTITY
Olivier Nakache	4
Eric Toledano	2
Frank Darabont	2
Omar Sy	1
Anne Le Ny	1
Tim Robbins	1
Chris Pratt	1
Morgan Freeman	1
Fran�ois Cluzet	1
Jennifer Lawrence	1

**Вывести 5 фильмов с максимальной разницей между средними оценками зрителей и критиков**

```
select first 5 films.name, avg(abs(Rate_critic.Rate-rate_userk.rate)) as Average
from films
join rate_critic on (rate_critic.id_film=films.id)
join Rate_USERK on (Rate_USERK.id_film=films.id)
group by films.name
order by Average desc;
```

Результат работы:

NAME	AVERAGE
DDD	8.000000000000000
The Green Mile	8.000000000000000
EEE	5.000000000000000
Forrest Gump	3.857142857142857
FFF	3.500000000000000

**Удалить неиспользуемые жанры**

```
select * from genre;
delete from genre where id not in
(select id_genre from Film_Genre);
select * from genre;
```

Результат работы:

ID	NAME
1	action
2	adventure
3	comedy
4	drama
5	crime
6	horror
7	fantasy
8	thriller
9	family
10	documentary
11	detective
12	biography
13	animation

```

SQL> delete from genre where id not in
CON> (select id_genre from Film_Genre);
SQL> select * from genre;

```

ID	NAME
3	comedy
4	drama
5	crime
11	detective
12	biography

### 3.3. Сохранение выполненных запросов в виде хранимых процедур

#### Insert

```

set term ^ ;
create procedure insert_tables
as begin
--Страна:
insert into Country values(11,' Belarus');
--Студия:
insert into Studia values(6,'Columbia Pictures');
--Работник:
insert into Worker values (9,'Jennifer Lawrence','15.08.1990','W',3);
--Фильм:
insert into Films values (6,'Passengers',2016,'Nothing is by chance',16,116,110000000);
--Фильм-Студия:
insert into Film_Studia values(3,6,6);
--Фильм-Страна:
insert into Country_film values(6,6,3);
--Профессия:
insert into profession values (3,'Operator');
--Фильм-Работник:
insert into Film_Worker values (9,6,9,7800,1,1);
--Жанры:
insert into Genre values(13,'animation');
--Фильм-Жанр:
insert into Film_Genre values(13,6,7);
--Пользователи:
insert into UserK values(7,'Alesha','M','26.05.1996',1);
--Рецензии пользователя:
insert into Rate_userK values(3,'9',7,6,'Excellent romantic melodrama about two people...');
--Критик:
insert into critic values(7,'NoName201','M','12.12.1987',4);
--Рецензия критика:
insert into Rate_critic values(3,'7',7,6,'Well, so-so');
--Период сборов:
insert into period values(5,'All time');
--Регионы:
insert into region values(6,3,'New York');
--Сборы:
insert into Collections values(6,6,5,4000000,1);
end ^

```

#### Update

```

create procedure updates_tables
as begin
update UserK set Name=Name || ' PRO' where sex in('M');
end ^

```

Результат работы скрипта:

ID_USERK	NAME	SEX	DATE_OF_BIRTHDAY	ID_COUNTRY
1	WhitePRO PRO	M	1996-05-26	1
2	Alexey_DPRO PRO	M	1976-12-31	1
3	Veronica	W	1969-08-19	1
4	DiegoPRO PRO	M	1993-08-29	10
5	Masha	W	1995-07-04	1
6	Janna	W	1964-12-08	5

## Delete

```

create procedure del_tables
as begin
delete from WORKER where ID not in
(select ID_worker from Film_Worker);
end ^

```

Результат работы скрипта:

ID	NAME	DATE_OF_BIRTHDAY	SEX	ID_COUNT
1	Olivier Nakache	1973-04-15	M	4
2	Eric Toledano	1971-07-03	M	4
3	Frank Darabont	1959-01-28	M	4
4	François Cluzet	1955-09-21	M	4
5	Omar Sy	1978-01-20	M	4
6	Anne Le Ny	1962-12-16	W	4
7	Tim Robbins	1958-10-16	M	3
8	Morgan Freeman	1937-06-01	M	3

  

ID	NAME	DATE_OF_BIRTHDAY	SEX	ID_COUNT
1	Olivier Nakache	1973-04-15	M	4
2	Eric Toledano	1971-07-03	M	4
3	Frank Darabont	1959-01-28	M	4
4	François Cluzet	1955-09-21	M	4
5	Omar Sy	1978-01-20	M	4

## 4. Выводы

- Язык DML позволяет достаточно просто выполнять простейшие запросы на модификацию записей, выборку данных, группировку и вычисление совокупных характеристик, операции объединения и сортировку.
- Поскольку представленная база данных имеет сложную нормализованную структуру, проводить операции добавления, удаления и редактирования данных вручную слишком трудоемко. Поэтому были реализованы хранимые процедуры, упрощающие эти действия.