

Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

Отчет по лабораторной работе № 4

«SQL-DML»

по дисциплине «Базы данных»

Работу выполнил:

студент гр. 43501/3

Хуторной Я. В.

Руководитель

Мяснов А. В.

«__» _____ 2015 г

Санкт-Петербург

2015

Цели работы

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

Программа работы

- Выполните все запросы из списка стандартных запросов. Продемонстрируйте результаты преподавателю.
- Получите у преподавателя и реализуйте SQL-запросы в соответствии с индивидуальным заданием. Продемонстрируйте результаты преподавателю.
- 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)

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

В SQL-DML определены команды select (выбрать), insert (вставить), update (обновить), delete (удалить).

Выполнение работы

1. Был самостоятельно изучен язык SQL-DML.
 2. Были выполнены все запросы из списка стандартных запросов.
- Выборка всех данных из каждой таблицы (в виде представлений)

```
connect 'C:\DB\MUSICLIBRARY.FDB' user 'SYSDBA' password 'masterkey';

CREATE OR ALTER VIEW ARTISTSSELECT(
  ARTISTS_ID,
  ARTISTS_NAME,
  ARTISTS_ROLE,
  ARTISTS_BIOGRAPHY)
AS
select * from ARTISTS
;
CREATE OR ALTER VIEW CDSELECT(
  CD_ID,
  CD_NAME,
  CD_COUNTRY,
  CD_FORMAT,
  CD_RATING,
  CD_TIME,
  CD_YEAR)
AS
select * from CD
```

```

;
CREATE OR ALTER VIEW CD_GENRESELECT(
    CD_ID,
    GENRE_NAME)
AS
select * from CD_GENRE
;
CREATE OR ALTER VIEW CD_PLAYERSSELECT(
    CD_ID,
    PLAYER_ID)
AS
select * from CD_PLAYERS
;
CREATE OR ALTER VIEW CD_RECORDING_STUDIOSSELECT(
    CD_ID,
    RECORDING_STUDIO_ID)
AS
select * from CD_RECORDING_STUDIO
;
CREATE OR ALTER VIEW CD_TRACKSELECT(
    CD_ID,
    TRACK_ID)
AS
select * from CD_TRACK
;
CREATE OR ALTER VIEW CLIENTSELECT(
    CLIENT_ID,
    CLIENT_NAME,
    CLIENT_LOGIN)
AS
select * from CLIENT
;
CREATE OR ALTER VIEW GENRESELECT(
    GENRE_NAME)
AS
select * from GENRE
;
CREATE OR ALTER VIEW GENRE_ARTISTSSELECT(
    ARTISTS_ID,
    GENRE_NAME)
AS
select * from GENRE_ARTISTS
;
CREATE OR ALTER VIEW GENRE_PLAYERSSELECT(

```

```

    PLAYER_ID,
    GENRE_NAME)
AS
select * from GENRE_PLAYERS
;
CREATE OR ALTER VIEW MYPLAYLISTSELECT(
    MYPLAYLIST_NAME,
    MYPLAYLIST_DESCRIPTION)
AS
select * from MYPLAYLIST
;
CREATE OR ALTER VIEW MYPLAYLIST_TRACKSELECT(
    MYPLAYLIST_NAME,
    TRACK_ID)
AS
select * from MYPLAYLIST_TRACK
;
CREATE OR ALTER VIEW PLAYERSSELECT(
    PLAYER_ID,
    PLAYER_NAME,
    PLAYER_SITE,
    PLAYER_DESCRIPTION)
AS
select * from PLAYERS
;
CREATE OR ALTER VIEW PLAYERS_ARTISTSSELECT(
    PLAYER_ID,
    ARTISTS_ID)
AS
select * from PLAYERS_ARTISTS
;
CREATE OR ALTER VIEW PLAYERS_TRACKSELECT(
    PLAYER_ID,
    TRACK_ID)
AS
select * from PLAYERS_TRACK
;
CREATE OR ALTER VIEW RECORDING_STUDIOSELECT(
    RECORDING_STUDIO_ID,
    RECORDING_STUDIO_NAME,
    RECORDING_STUDIO_COUNTRY,
    RECORDING_STUDIO_DESCRIPTION,
    RECORDING_STUDIO_SITE)
AS

```

```

select * from RECORDING_STUDIO
;
CREATE OR ALTER VIEW SALE_CDSELECT(
    SALE_CD_ID,
    CD_ID,
    CLIENT_ID,
    SALE_DATA,
    SALE_PRICE)
AS
select * from SALE_CD
;
CREATE OR ALTER VIEW SALE_CD_CLIENTSELECT(
    SALE_CD_ID,
    CLIENT_ID)
AS
select * from SALE_CD_CLIENT
;
CREATE OR ALTER VIEW SALE_TRACKSELECT(
    SALE_TRACK_ID,
    TRACK_ID,
    CLIENT_ID,
    SALE_DATA,
    SALE_PRICE)
AS
select * from SALE_TRACK
;
CREATE OR ALTER VIEW SALE_TRACK_CLIENTSELECT(
    SALE_TRACK_ID,
    CLIENT_ID)
AS
select * from SALE_TRACK_CLIENT
;
CREATE OR ALTER VIEW TRACKSELECT(
    TRACK_ID,
    TRACK_NAME,
    TRACK_TIME,
    TRACK_GENRE,
    TRACK_YEAR,
    TRACK_RATING)
AS
select * from TRACK
;
CREATE OR ALTER VIEW TRACK_ARTISTSSELECT(
    TRACK_ID,

```

```

ARTISTS_ID)
AS
select * from TRACK_ARTISTS
;
commit;

```

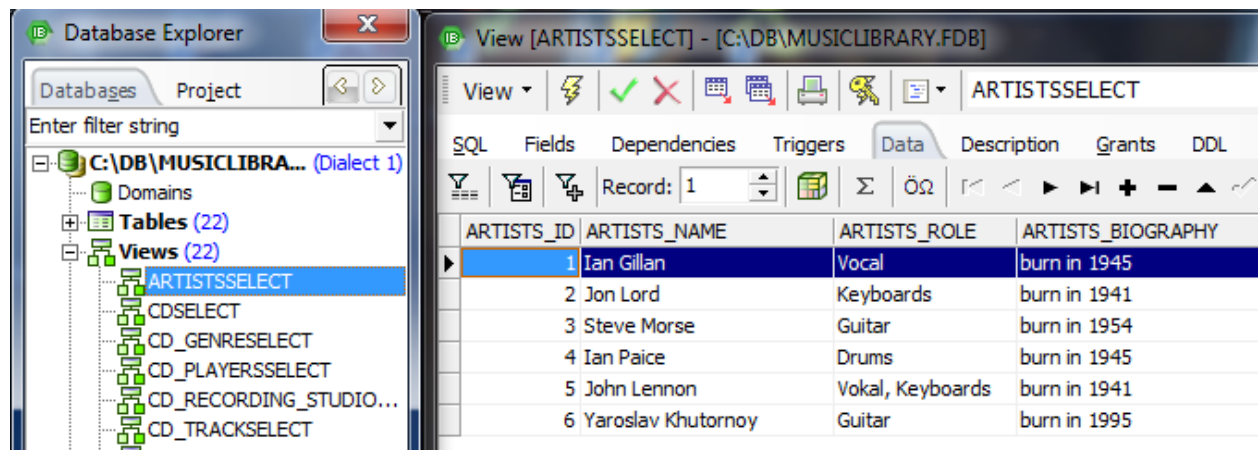


Рис. 1. Выборка данных из таблицы артистов

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

```

connect 'C:\DB\MUSICLIBRARY.FDB' user 'SYSDBA' password 'masterkey';

/*Выборка артистов с именем Ian Gillan из таблицы ARTISTS*/
create view LIKE_IANGILLAN as select * from ARTISTS where
ARTISTS_NAME like 'Ian Gillan%';

/*Выборка альбомов выпущенных в период 1969-1971 годов из таблицы CD*/
create view BETWEEN_CD69_71 as select * from CD where CD_YEAR
between 1969 and 1971;

/*Выборка групп, у которых сайт deeppurple.com из таблицы PLAYERS*/
create view IN_PLAYERS6 as select * from PLAYERS where PLAYER_SITE in
('deeppurple.com');

commit;

```

CD_ID	CD_NAME	CD_COUNTRY	CD_FORMAT	CD_RATING	CD_TIME	CD_YEAR
25	DPRUGQ	pd_ucEVsH	MP3	5	00:06:42	1969
26	EKSMTDXYJDCUZW	ZjJQrDTjxvtwnS	MP3	9	00:06:42	1970
37	JUBOMTQRBFXBW	fuapHfl	MP3	6	00:06:42	1970
80	APECOSEFZHJ	APgLmAm_Z\^f	MP3	9	00:06:42	1969
85	RTREIVR	oAxxQO	MP3	8	00:06:42	1970
87	VFLVWWEKZP]eRJEffAm	MP3	7	00:06:42	1969
100	NDTCTUWCEXHROJH	AKbXAhcWTF	MP3	8	00:06:42	1971
2	Deep Purple - In Rock	UK	flac	10	00:45:15	1970

Рис. 2. Результат для представления BETWEEN_CD69_71

- Выборка всех данных с сортировкой по нескольким полям

```
connect 'C:\DB\MUSICLIBRARY.FDB' user 'SYSDBA' password 'masterkey';
/*Выборка дисков по рейтингу в порядке возрастания из таблицы CD */
create view ORDER_CD_RAITING1 as select * from CD order by CD_RATING
asc;

/*Выборка дисков по рейтингу и по ID в порядке убывания из таблицы CD */
create view ORDER_CD_RAITING2 as select * from CD order by CD_RATING
desc, CD_ID desc;

commit;
```

CD_ID	CD_NAME	CD_COUNTRY	CD_FORMAT	CD_RATING	CD_TIME	CD_YEAR
98	PCAFPWSVNMXL	PoAupoK_nwvYr	MP3	10	00:06:42	1966
95	TYBKM	d^VuOE	MP3	10	00:06:42	1991
89	NRJYNFPHIMFTU	IvOzZLx	MP3	10	00:06:42	1965
84	XVNSKMTB	Er\dyZ	MP3	10	00:06:42	1950
83	XKSHNPH	qWMPMNIS\SU	MP3	10	00:06:42	1981
79	AQXEQNTBF	Mpuzhd	MP3	10	00:06:42	1956
78	LIMXZPGMCE	UoCck	MP3	10	00:06:42	1992
77	WWLDVKKTZVHQBL	dnAMhedTcFL	MP3	10	00:06:42	2007
72	WZVWWQYFW	oaAg^RQXAkzhiNc	MP3	10	00:06:42	1975
70	VZKUOFCSTXJGE	MduLhOX	MP3	10	00:06:42	2009
69	DGUZAEHA	YetXWQrNRpTNeF	MP3	10	00:06:42	1980
65	DXSHQL	SeGPA	MP3	10	00:06:42	1956
61	VKWDEPG	MuCZkDKOxZAH	MP3	10	00:06:42	2008
47	VXYYP	NLhnRNr	MP3	10	00:06:42	2002
40	XDTSUIVQ	S^EfpUo	MP3	10	00:06:42	1997
33	XNKATZKTZJQZCF	FN\VgubSpSno	MP3	10	00:06:42	1955
19	KYHFVOLXOVDSVJV	Gropi	MP3	10	00:06:42	1953
18	SJNSBLBQFBQEHG	Kf\PFSc	MP3	10	00:06:42	2016
17	LLFRTC	XWMOoCVwzGX	MP3	10	00:06:42	2011
8	UQMIJG	sJckyJ^t	MP3	10	00:06:42	2013
2	Deep Purple - In Rock	UK	flac	10	00:45:15	1970
1	Deep Purple - Machine Head	UK	MP3	10	00:50:25	1972

Рис. 3. Результат для представления ORDER_CD_RAITING2

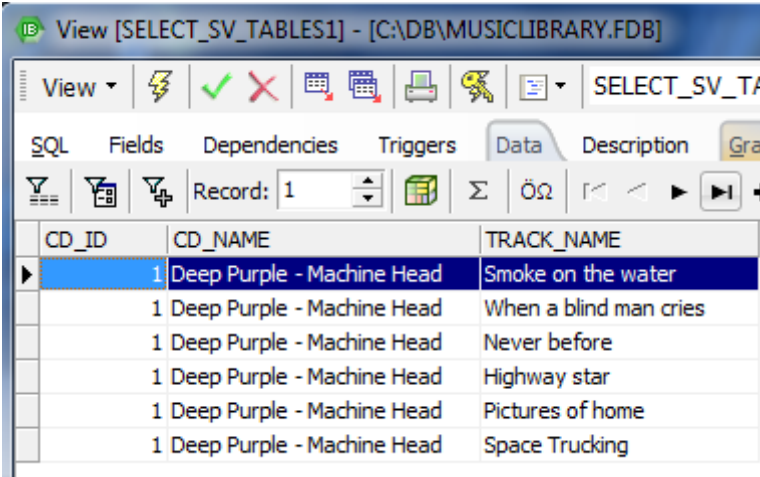
- Выборка данных из связанных таблиц (не менее двух примеров)

```
connect 'C:\DB\MUSICLIBRARY.FDB' user 'SYSDBA' password 'masterkey';
/*Выборка из связанных таблиц*/

/*Выборка названия альбома из таблицы CD и соответствующих ему треков из
таблицы TRECK при условии, что
мы смотрим CD.ID = 1 */
create view SELECT_SV_TABLES1 as select CD.CD_ID, CD.CD_NAME ,
TRACK.TRACK_NAME
from CD, TRACK, CD_TRACK where CD.CD_ID = CD_TRACK.CD_ID and
TRACK.TRACK_ID = CD_TRACK.TRACK_ID and CD.CD_ID = 1;

/*Выборка треков из таблицы TRACK соответствующих автору Ian Gillan из
таблицы ARTISTS */
create view SELECT_SV_TABLES2 as select ARTISTS.artists_name,
TRACK.track_name from ARTISTS, TRACK, track_artists
where ARTISTS.artists_id = TRACK_ARTISTS.artists_id and TRACK.track_id =
TRACK_ARTISTS.track_id and ARTISTS.artists_name like 'Ian Gillan%';

commit;
```



CD_ID	CD_NAME	TRACK_NAME
1	Deep Purple - Machine Head	Smoke on the water
1	Deep Purple - Machine Head	When a blind man cries
1	Deep Purple - Machine Head	Never before
1	Deep Purple - Machine Head	Highway star
1	Deep Purple - Machine Head	Pictures of home
1	Deep Purple - Machine Head	Space Trucking

Рис. 4. Результат для представления SELECT_SV_TABLES1

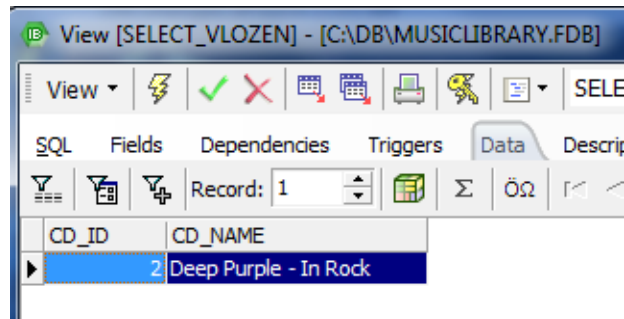
- Пример использования вложенного запроса

```
connect 'C:\DB\MUSICLIBRARY.FDB' user 'SYSDBA' password 'masterkey';
/*Пример вложенных запросов*/

/* Отображение названий тех дисков, в которых содержится песня Fools */
create view SELECT_VLOZEN as select CD_TRACK.CD_ID, CD.CD_NAME
from CD_TRACK, CD, TRACK where
CD_TRACK.CD_ID = CD.CD_ID and CD_TRACK.TRACK_ID =
```

TRACK.TRACK_ID and
CD_TRACK.TRACK_ID in (select TRACK.TRACK_ID from TRACK where
TRACK_NAME like 'Fools%');

commit;



CD_ID	CD_NAME
2	Deep Purple - In Rock

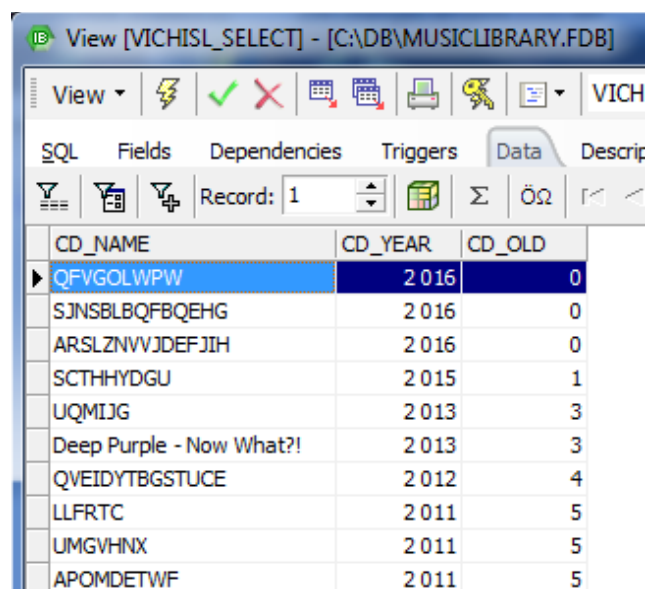
Рис. 5. Результат для представления SELECT_VLOZEN

- Создайте в запросе вычисляемое поле

```
connect 'C:\DB\MUSICLIBRARY.FDB' user 'SYSDBA' password 'masterkey';
/*Создать в запросе вычисляемое поле*/
```

```
/*Отобразить названия дисков с их годом выпуска и посчитать возраст дисков*/
create view VICHISL_SELECT as select CD.CD_NAME as CD_NAME,
CD.CD_YEAR as CD_YEAR,
2016 - CD.CD_YEAR as CD_OLD from CD order by CD_OLD asc;
;
```

commit;



CD_NAME	CD_YEAR	CD_OLD
QFVGOLWPW	2016	0
SJNSBLBQFBQEHG	2016	0
ARSLZNVVJDEFJIH	2016	0
SCTHHYDGU	2015	1
UQMIJG	2013	3
Deep Purple - Now What?!	2013	3
QVEIDYTBGSTUCE	2012	4
LLFRTC	2011	5
UMGVHIX	2011	5
APOMDETWF	2011	5

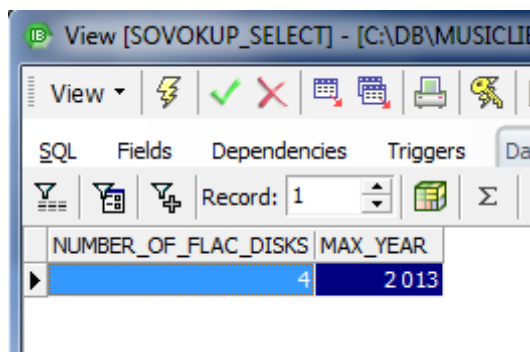
Рис. 6. Результат для представления VICHISL_SELECT

- Создайте запрос, вычисляющий несколько совокупных характеристик таблиц

```
connect 'C:\DB\MUSICLIBRARY.FDB' user 'SYSDBA' password 'masterkey';
/*Пример запроса, вычисляющего несколько совокупных характеристик
таблицы*/

/*Отображение количества дисков с форматом flac и отображение даты выпуска
диска с самым поздним годом выпуска*/
create view SOVOKUP_SELECT as select COUNT(CD.CD_FORMAT) as
NUMBER_OF_FLAC_DISKS, MAX(CD.CD_YEAR) as MAX_YEAR from CD
where CD.CD_FORMAT like 'flac%'
;

commit;
```



NUMBER_OF_FLAC_DISKS	MAX_YEAR
4	2013

Рис. 7. Результат для представления SOVOKUP_SELECT

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

```
connect 'C:\DB\MUSICLIBRARY.FDB' user 'SYSDBA' password 'masterkey';
/*Пример запроса, рассчитывающий совокупную характеристику с
использованием группировки с
наложением ограничения на результат группировки*/

/*Сгруппировать альбомы по форматам, подсчитать кол-во альбомов каждого
формата и показать только
те группы, в которых кол-во альбомов больше 1*/
create view GROUP_SELECT as select CD.CD_FORMAT as FORMAT,
count(CD.CD_FORMAT) as CD_COUNT
from CD group by CD.CD_FORMAT having count(CD.CD_FORMAT)>1;
;

commit;
```

FORMAT	CD_COUNT
MP3	101
flac	4

Рис. 8. Результат для представления GROUP_SELECT

- С помощью оператора INSERT добавьте в таблицы по одной записи

```
connect 'C:\DB\MUSICLIBRARY.FDB' user 'SYSDBA' password 'masterkey';
/*С помощью INSERT добавить запись в каждую таблицу*/

/*ХП для заполнения таблицы CD*/
create procedure INS_CD (CD_IDp integer, CD_NAMEp varchar(255),
CD_COUNTRYp varchar(255),
CD_FORMATp varchar(255), CD_RATINGp smallint, CD_TIMEp varchar(255),
CD_YEARp smallint)
as
begin
insert into CD (CD_ID, CD_NAME, CD_COUNTRY, CD_FORMAT,
CD_RATING, CD_TIME, CD_YEAR)
values (:CD_IDp, :CD_NAMEp, :CD_COUNTRYp, :CD_FORMATp,
:CD_RATINGp, :CD_TIMEp, :CD_YEARp);
end;

/*ХП для заполнения таблицы ARTISTS*/
create procedure INS_ARTISTS (ARTISTS_IDp integer, ARTISTS_NAMEp
varchar(255), ARTISTS_ROLEp varchar(255),
ARTISTS_BIOGRAPHYp varchar(255))
as
begin
insert into ARTISTS (ARTISTS_ID, ARTISTS_NAME, ARTISTS_ROLE,
ARTISTS_BIOGRAPHY)
values (:ARTISTS_IDp, :ARTISTS_NAMEp, :ARTISTS_ROLEp,
:ARTISTS_BIOGRAPHYp);
end;

/*ХП для заполнения таблицы CD_GENRE*/
create procedure INS_CD_GENRE (CD_IDp integer, GENRE_NAMEp
varchar(255))
```

```

as
begin
    insert into CD_GENRE (CD_ID, GENRE_NAME)
    values (:CD_IDp, :GENRE_NAMEp);
end;

/*ХП для заполнения таблицы CD_PLAYERS*/
create procedure INS_CD_PLAYERS (CD_IDp integer, PLAYER_IDp integer)
as
begin
    insert into CD_PLAYERS (CD_ID, PLAYER_ID)
    values (:CD_IDp, :PLAYER_IDp);
end;

/*ХП для заполнения таблицы CD_RECORDING_STUDIO*/
create procedure INS_CD_RECORDING_STUDIO (CD_IDp integer,
RECORDING_STUDIO_IDp integer)
as
begin
    insert into CD_RECORDING_STUDIO (CD_ID, RECORDING_STUDIO_ID)
    values (:CD_IDp, :RECORDING_STUDIO_IDp);
end;

/*ХП для заполнения таблицы CD_TRACK*/
create procedure INS_CD_TRACK (CD_IDp integer, TRACK_IDp integer)
as
begin
    insert into CD_TRACK (CD_ID, TRACK_ID)
    values (:CD_IDp, :TRACK_IDp);
end;

/*ХП для заполнения таблицы CLIENT*/
create procedure INS_CLIENT (CLIENT_IDp integer, CLIENT_NAMEp
varchar(255), CLIENT_LOGINp varchar(255))
as
begin
    insert into CLIENT (CLIENT_ID, CLIENT_NAME, CLIENT_LOGIN)
    values (:CLIENT_IDp, :CLIENT_NAMEp, :CLIENT_LOGINp);
end;

/*ХП для заполнения таблицы GENRE*/
create procedure INS_GENRE (GENRE_NAMEp varchar(255))
as
begin

```

```

insert into GENRE (GENRE_NAME)
values (:GENRE_NAMEp);
end;

/*ХП для заполнения таблицы GENRE_ARTISTS*/
create procedure INS_GENRE_ARTISTS (ARTISTS_IDp integer,
GENRE_NAMEp varchar(255))
as
begin
insert into GENRE_ARTISTS (ARTISTS_ID, GENRE_NAME)
values (:ARTISTS_IDp, :GENRE_NAMEp);
end;

/*ХП для заполнения таблицы GENRE_PLAYERS*/
create procedure INS_GENRE_PLAYERS (PLAYER_IDp integer,
GENRE_NAMEp varchar(255))
as
begin
insert into GENRE_PLAYERS (PLAYER_ID, GENRE_NAME)
values (:PLAYER_IDp, :GENRE_NAMEp);
end;

/*ХП для заполнения таблицы MYPLAYLIST*/
create procedure INS_MYPLAYLIST (MYPLAYLIST_NAMEp varchar(255),
MYPLAYLIST_DESCRIPTIONp varchar(255))
as
begin
insert into MYPLAYLIST (MYPLAYLIST_NAME,
MYPLAYLIST_DESCRIPTION)
values (:MYPLAYLIST_NAMEp, :MYPLAYLIST_DESCRIPTIONp);
end;

/*ХП для заполнения таблицы MYPLAYLIST_TRACK*/
create procedure INS_MYPLAYLIST_TRACK (MYPLAYLIST_NAMEp
varchar(255), TRACK_IDp integer)
as
begin
insert into MYPLAYLIST_TRACK (MYPLAYLIST_NAME, TRACK_ID)
values (:MYPLAYLIST_NAMEp, :TRACK_IDp);
end;

/*ХП для заполнения таблицы PLAYERS*/

```

```

create procedure INS_PLAEYRS (PLAYER_IDp integer, PLAYER_NAMEp
varchar(255),    PLAYER_SITEp    varchar(255),    PLAYER_DESCRIPTIONp
varchar(255))
as
begin
    insert into PLAYERS (PLAYER_ID, PLAYER_NAME, PLAYER_SITE,
PLAYER_DESCRIPTION)
    values (:PLAYER_IDp,      :PLAYER_NAMEp,      :PLAYER_SITEp,
:PLAYER_DESCRIPTIONp);
end;

/*ХП для заполнения таблицы PLAYERS_ARTISTS*/
create procedure INS_PLAYERS_ARTISTS (PLAYER_IDp integer, ARTISTS_IDp
integer)
as
begin
    insert into PLAYERS_ARTISTS (PLAYER_ID, ARTISTS_ID)
    values (:PLAYER_IDp, :ARTISTS_IDp);
end;

/*ХП для заполнения таблицы PLAYERS_TRACK*/
create procedure INS_PLAYERS_TRACK (PLAYER_IDp integer, TRACK_IDp
integer)
as
begin
    insert into PLAYERS_TRACK (PLAYER_ID, TRACK_ID)
    values (:PLAYER_IDp, :TRACK_IDp);
end;

/*ХП для заполнения таблицы RECORDING_STUDIO*/
create procedure INS_RECORDING_STUDIO (RECORDING_STUDIO_IDp
integer, RECORDING_STUDIO_NAMEp varchar(255),
RECORDING_STUDIO_COUNTRYp          varchar(255),
RECORDING_STUDIO_DESCRIPTIONp      varchar(255),
RECORDING_STUDIO_SITEp varchar(255))
as
begin
    insert into RECORDING_STUDIO (RECORDING_STUDIO_ID,
RECORDING_STUDIO_NAME, RECORDING_STUDIO_COUNTRY,
RECORDING_STUDIO_DESCRIPTION, RECORDING_STUDIO_SITE)
    values (:RECORDING_STUDIO_IDp, :RECORDING_STUDIO_NAMEp,
:RECORDING_STUDIO_COUNTRYp,
:RECORDING_STUDIO_DESCRIPTIONp, :RECORDING_STUDIO_SITEp);
end;

```

```

/*ХП для заполнения таблицы SALE_CD*/
create procedure INS_SALE_CD (SALE_CD_IDp integer, CD_IDp integer,
CLIENT_IDp integer, SALE_DATAp date,
SALE_PRICEp integer)
as
begin
insert into SALE_CD (SALE_CD_ID, CD_ID, CLIENT_ID, SALE_DATA,
SALE_PRICE)
values (:SALE_CD_IDp, :CD_IDp, :CLIENT_IDp, :SALE_DATAp,
:SALE_PRICEp);
end;

/*ХП для заполнения таблицы SALE_CD_CLIENT*/
create procedure INS_SALE_CD_CLIENT (SALE_CD_IDp integer, CLIENT_IDp
integer)
as
begin
insert into SALE_CD_CLIENT (SALE_CD_ID, CLIENT_ID)
values (:SALE_CD_IDp, :CLIENT_IDp);
end;

/*ХП для заполнения таблицы SALE_TRACK*/
create procedure INS_SALE_TRACK (SALE_TRACK_IDp integer, TRACK_IDp
integer, CLIENT_IDp integer, SALE_DATAp date,
SALE_PRICEp integer)
as
begin
insert into SALE_TRACK (SALE_TRACK_ID, TRACK_ID, CLIENT_ID,
SALE_DATA, SALE_PRICE)
values (:SALE_TRACK_IDp, :TRACK_IDp, :CLIENT_IDp, :SALE_DATAp,
:SALE_PRICEp);
end;

/*ХП для заполнения таблицы SALE_TRACK_CLIENT*/
create procedure INS_SALE_TRACK_CLIENT (SALE_TRACK_IDp integer,
CLIENT_IDp integer)
as
begin
insert into SALE_TRACK_CLIENT (SALE_TRACK_ID, CLIENT_ID)
values (:SALE_TRACK_IDp, :CLIENT_IDp);
end;

/*ХП для заполнения таблицы TRACK*/

```



```

create procedure INS_TRACK (TRACK_IDp integer, TRACK_NAMEp
varchar(255), TRACK_TIMEp varchar(255),
TRACK_YEARp smallint, TRACK_RATINGp smallint)
as
begin
insert into TRACK (TRACK_ID, TRACK_NAME, TRACK_TIME,
TRACK_YEAR, TRACK_RATING)
values (:TRACK_IDp, :TRACK_NAMEp, :TRACK_TIMEp, :TRACK_YEARp,
:TRACK_RATINGp);
end;

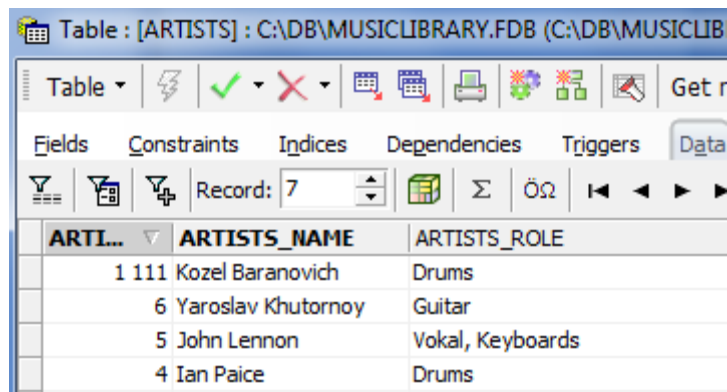
/*ХП для заполнения таблицы TRACK_ARTISTS*/
create procedure INS_TRACK_ARTISTS (TRACK_IDp integer, ARTISTS_IDp
integer)
as
begin
insert into TRACK_ARTISTS (TRACK_ID, ARTISTS_ID)
values (:TRACK_IDp, :ARTISTS_IDp);
end;

commit;

```

Name	Type	Null	Value	Descri
ARTISTS_IDP	INTEGER	<input type="checkbox"/>	1111	
ARTISTS_NAMEP	VARCHAR(...)	<input type="checkbox"/>	Kozel Baranovich	
ARTISTS_ROLEP	VARCHAR(...)	<input type="checkbox"/>	Drums	
ARTISTS_BIOGRAPHYP	VARCHAR(...)	<input type="checkbox"/>	Simple Kozel.	

Рис. 9. Процедура добавления новой записи в таблицу ARTISTS



ARTISTS_ID	ARTISTS_NAME	ARTISTS_ROLE
1 111	Kozel Baranovich	Drums
6	Yaroslav Khutornoy	Guitar
5	John Lennon	Vokal, Keyboards
4	Ian Paice	Drums

Рис. 10. Результат добавления

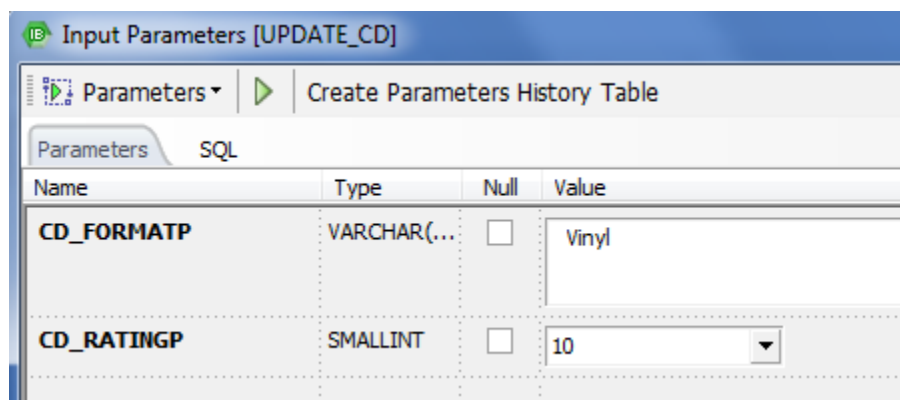
Аналогичные действия были проведены для всех таблиц.

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

```
connect 'C:\DB\MUSICLIBRARY.FDB' user 'SYSDBA' password 'masterkey';
/*С помощью UPDATE изменить значения нескольких полей у всех записей,
отвечающих заданному условию*/

/*ХП для модификации таблицы CD*/
create procedure UPDATE_CD (CD_FORMATp varchar(255), CD_RATINGp
smallint)
as
begin
    update    CD set CD_FORMAT = :CD_FORMATp, CD_RATING =
:CD_RATINGp where CD_YEAR between 1950 and 1985;
end;

commit;
```



Name	Type	Null	Value
CD_FORMATP	VARCHAR(255)	<input type="checkbox"/>	Vinyl
CD_RATINGP	SMALLINT	<input type="checkbox"/>	10

Рис. 9. Процедура изменения в таблице CD

Table : [CD] : C:\DB\MUSICLIBRARY.FDB (C:\DB\MUSICLIBRARY.FDB)

Table ▾ ⚡ ✓ ✗ 📅 📄 🖨️ 🌐 🔄 📡 Get record count CD

Fields Constraints Indices Dependencies Triggers Data Master/Detail View Description DDL Grants Logging

Record: 25 Σ Ω ⏪ ⏩ + - 🔍 ↺ Font size: 8

CD_ID	CD_NAME	CD_COUNTRY	CD_FORMAT	CD_RATING	CD_TIME	CD_YEAR
3	Deep Purple - Burn	UK	Vinyl	10	00:49:18	1975
102	AYQUFNHMPIESJFM	jc]px`uvEB	Vinyl	10	00:06:42	1972
49	QZXVWSBUHONDWAX	Q\HAWrmVQBk	Vinyl	10	00:06:42	1973
100	NDTCTUWCEXHRJH	AKbXAhcWTF	Vinyl	10	00:06:42	1971
12	YKKIUVVS	\McGo	Vinyl	10	00:06:42	1959
98	PCAFPWSVNMXL	PoAupoK_nwvYr	Vinyl	10	00:06:42	1966
14	RYDHCUXGFOJAL	VAS]UMMZxPpbQJ	Vinyl	10	00:06:42	1960
29	DBILYTKRX	LzWqAvoBkxf	Vinyl	10	00:06:42	1981

Рис. 10. Результат изменения таблицы CD

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

```
connect 'C:\DB\MUSICLIBRARY.FDB' user 'SYSDBA' password 'masterkey';
/*С помощью DELETE удалить запись, имеющую
максимальное(минимальное) значение некоторой совокупной
характеристики*/

/*Удаляем самый новый диск с таблицы CD*/
create procedure DELETE_CD_1
as
begin
delete from CD where CD.CD_YEAR = (select max(CD_YEAR) from CD);
end;

commit;
```

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

```
connect 'C:\DB\MUSICLIBRARY.FDB' user 'SYSDBA' password 'masterkey';
/*С помощью DELETE удалить записи в главной таблице, на которые не
ссылается подчиненная таблица
(используя вложенный запрос)*/

/*Удаляем те диски с таблицы CD, в которых не содержится треков (не
ссылается таблица со списком треков)*/
create procedure DELETE_CD_2
as
begin
delete from CD where CD_ID not in (select CD_ID from CD_TRACK);
```

```
end;
```

```
commit;
```

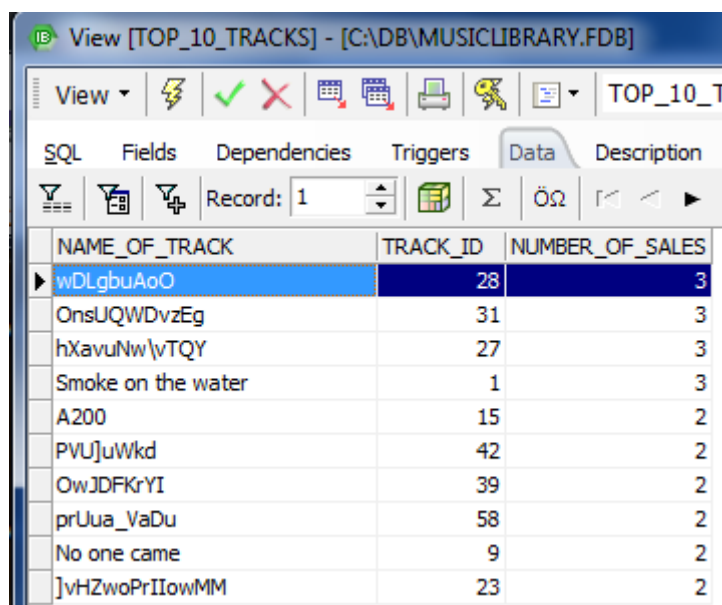
3. У преподавателя были получены индивидуальные задания:

- Вывести топ-10 композиций, проданных за заданный месяц

```
connect 'C:\DB\MUSICLIBRARY.FDB' user 'SYSDBA' password 'masterkey';
```

```
create view TOP_10_TRACKS as select first 10 TRACK.TRACK_NAME as  
NAME_OF_TRACK,  
SALE_TRACK.TRACK_ID as TRACK_ID,  
count(SALE_TRACK.SALE_TRACK_ID) as NUMBER_OF_SALES  
from TRACK, SALE_TRACK where TRACK.TRACK_ID = TRACK_ID and  
SALE_TRACK.SALE_DATA between "2015-01-01" and "2015-02-01"  
group by SALE_TRACK.TRACK_ID, TRACK.TRACK_NAME order by  
NUMBER_OF_SALES desc;
```

```
commit;
```



NAME_OF_TRACK	TRACK_ID	NUMBER_OF_SALES
wDLgbuAoO	28	3
OnsUQWDvzEg	31	3
hXavuNw\vtQY	27	3
Smoke on the water	1	3
A200	15	2
PVU]uWkd	42	2
OwJDFKrYI	39	2
prUua_VaDu	58	2
No one came	9	2
]vHZwoPrIIowMM	23	2

Рис. 11. ТОП-10 композиций, проданных за январь 2015 года

Время выполнения с количеством записей в таблицах равным 100 000 составило 105 мс.

- Вывести топ-5 наиболее популярных исполнителей за заданный промежуток времени.

```
connect 'C:\DB\MUSICLIBRARY.FDB' user 'SYSDBA' password 'masterkey';
```

```

create view TOP_5_PLAYERS as select first 5 PLAYERS.PLAYER_NAME as
PLAYERS,
count(SALE_CD.SALE_CD_ID) as NUMBERS_OF_SALES_CD
from SALE_CD, PLAYERS, CD_PLAYERS, CD where
SALE_CD.CD_ID = CD.cd_id and
CD.CD_ID = CD_PLAYERS.cd_id and
CD_PLAYERS.player_id = PLAYERS.PLAYER_ID
and SALE_CD.SALE_DATA between "2015-01-01" and "2015-07-01"
group by PLAYERS.PLAYER_NAME, PLAYERS.PLAYER_NAME order by
NUMBERS_OF_SALES_CD desc;

commit;

```

PLAYERS	NUMBERS_OF_SALES_CD
Led Zeppelin	1 000
The Doors	886
DDT	707
Joe Satriani	697
Steve Morse	672

Рис. 12. ТОП-5 самых популярных исполнителей за январь – июнь 2015 года

Время выполнения с количеством записей в таблицах равным 100 000 составило 125 мс.

- Вывести количество композиций по каждому жанру

```

connect 'C:\DB\MUSICLIBRARY.FDB' user 'SYSDBA' password 'masterkey';

create view NUMBER_OF_TRACK_ON_GENRE as select
GENRE.GENRE_NAME as GENRE,
count(CD_TRACK.TRACK_ID) as NUMBER_OF_TRACKS from
CD_TRACK, GENRE, CD_GENRE, CD where
CD_GENRE.GENRE_NAME = GENRE.GENRE_NAME
and CD_GENRE.CD_ID = CD_TRACK.CD_ID
and CD_GENRE.CD_ID = CD.CD_ID
group by GENRE.GENRE_NAME, GENRE.GENRE_NAME;

commit;

```

GENRE	NUMBER_OF_TRACKS
Blues	13
Classic	23
Hard Rock	28
Jazz	18
Rock	31

Рис. 13. Количество треков по каждому жанру

Время выполнения с количеством записей в таблицах равным 100 000 составило 87 мс.

Вывод

В результате выполнения работы был изучен язык манипулирования данными SQL-DML. Были написаны запросы извлечения данных из БД в соответствии с индивидуальным заданием.

Были изучены представления и хранимые процедуры SQL. Практический опыт создания приложений обработки данных показывает, что ряд операций над данными, реализующих общую для всех пользователей логику и не связанных с пользовательским интерфейсом, целесообразно вынести на сервер. Представления и хранимые процедуры позволяют хранить запросы и скрипты в самой БД. SQL-DML удобен для написания запросов разной сложности.

