Санкт-Петербургский Политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

**Отчёт о лабораторной работе №3**

**Дисциплина**: Базы данных

**Тема**: Язык SQL-DML

Выполнила студентка гр. 43501/1 В.А.Чернышева

(подпись)

Руководитель А.В. Мяснов

(подпись)

“ ” 2015 г.

Санкт-Петербург

2015

1. **Цель работы**

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

1. **Программа работы**
2. Изучите SQL-DML
3. Выполните все запросы из списка стандартных запросов. Продемонстрируйте результаты преподавателю.
4. Получите у преподавателя и реализуйте SQL-запросы в соответствии с **индивидуальным** заданием. Продемонстрируйте результаты преподавателю.
5. Выполненные запросы SELECT сохраните в БД в виде представлений, запросы INSERT, UPDATE или DELETE -- в виде ХП. Выложите скрипт в Subversion.
6. **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

1. **Выполнение работы**
2. Выборка всех данных из каждой таблицы

SET NAMES WIN1251;

CONNECT 'C:\123\BIBLIOTEKA.FDB' USER 'SYSDBA' PASSWORD 'masterkey';

create view AuthorSel as select \* from Author;

create view BookSel as select \* from Book;

create view BookAuthorSel as select \* from book\_author;

create view ExempSel as select \* from exemp;

create view GenreSel as select \* from genre;

create view IntakeSel as select \* from intake;

create view OperationSel as select \* from operation;

create view ProvaiderSel as select \* from provaider;

create view ReserveSel as select \* from reserve;

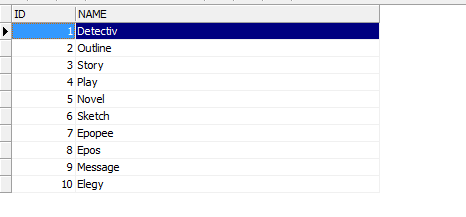
create view StatusSel as select \* from status;

create view TypeSel as select \* from type;

create view TypeOfIssueSel as select \* from type\_of\_issue;

create view WrittenOfSel as select \* from write\_off;

Выборка данных из таблицы жанров:



1. Выборка данных из одной таблицы при нескольких условиях, с использованием логических операций, LIKE, BETWEEN, IN

SET NAMES WIN1251;

CONNECT 'C:\123\BIBLIOTEKA.FDB' USER 'SYSDBA' PASSWORD 'masterkey';

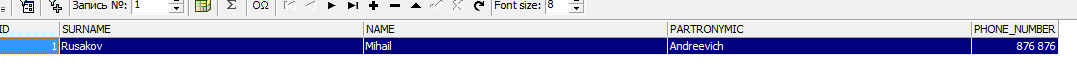
create view selectRusakov as select \* from Readership where SURNAME like 'Rusakov';

create view selectAuthor as select \* from Author where id between 0 and 20;

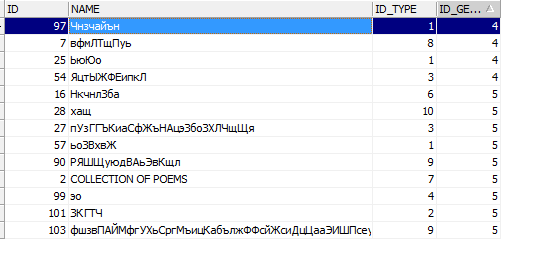
create view selectBook as select \* from Book where Id\_genre in (4,5);

Были созданы следующие выборки:

* Из таблицы Readership читатель с фамилией Русаков
* Из таблицы Author авторы с id от 0-20
* Из таблицы Book книги с номерами жанров 4 и 5



Пример выборки читателя с фамилией Русаков



Пример выборки книг с жанрами , id которых равен 4 и 5.

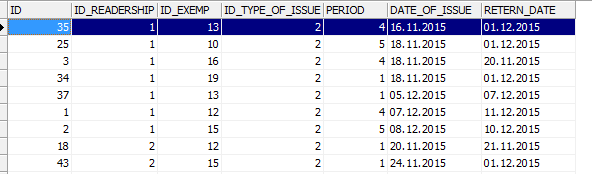
1. Выборка всех данных с сортировкой по нескольким полям

SET NAMES WIN1251;

CONNECT 'C:\123\BIBLIOTEKA.FDB' USER 'SYSDBA' PASSWORD 'masterkey';

create view SelectOperation as select \* from Operation order by Id\_readership asc, Date\_of\_issue asc;

Пример выборки из таблицы «Операции» с сортировкой по номеру читателя и даты выдчи ему книг.



1. Выборка данных из связанных таблиц

SET NAMES WIN1251;

CONNECT 'C:\123\BIBLIOTEKA.FDB' USER 'SYSDBA' PASSWORD 'masterkey';

create view selectGenreInBookName

as select Genre.Name as Genre,

Book.Name as Book

from Genre, Book

where Genre.id = Book.id\_genre;

create view selectNameBook

as select Book.Name as Book,

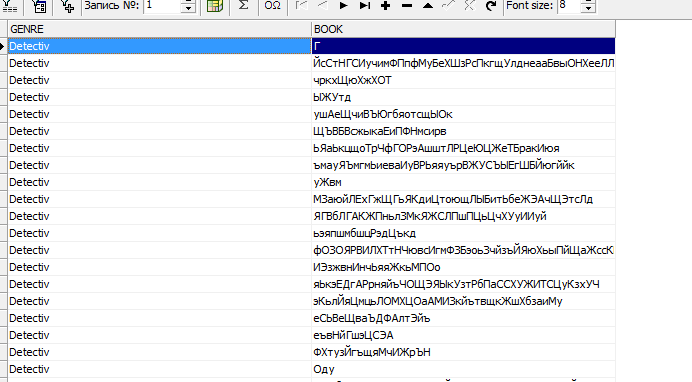
Exemp.Id as Exemp ,

Status.Name as Status

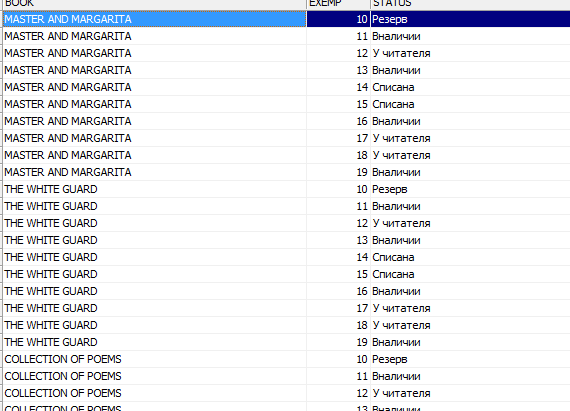
from Book, Exemp, Status

where Exemp.id\_status = Status.id;

Пример выборки , связывающий жанр и книги, имеющией такой жанр



Пример выборки связывающей название книги, ее экземпляр и статус



1. Запрос, рассчитывающий совокупную характеристику с использованием группировки, наложите ограничение на результат группировки

SET NAMES WIN1251;

CONNECT 'C:\123\BIBLIOTEKA.FDB' USER 'SYSDBA' PASSWORD 'masterkey';

create view selectExempCount

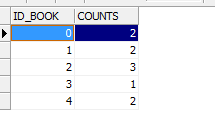
as select Exemp.id\_book as id\_book,

COUNT(Exemp.id\_book) as Counts

from Exemp

group by Exemp.id\_book;

Выборка id книги и количества ее экземпляров



1. Использование вложенного запроса

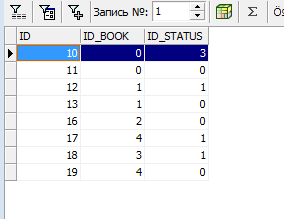
SET NAMES WIN1251;

CONNECT 'C:\123\BIBLIOTEKA.FDB' USER 'SYSDBA' PASSWORD 'masterkey';

create view selectEXempWrittenOff as select \* from Exemp where Id

not in (select Id\_Exemp from Write\_Off);

Выборка книг, которых нет в таблице списанных



1. Использование оператора INSERT для добавления по одной записи в каждую таблицу:

SET NAMES WIN1251;

CONNECT 'C:\123\BIBLIOTEKA.FDB' USER 'SYSDBA' PASSWORD 'masterkey';

-- Добавим запись в таблицу авторов

create procedure insert\_author (id integer, SURNAME varchar(50),

NAME varchar(50), PARTRONYMIC varchar(50))

as

begin

insert into AUTHOR (ID, SURNAME, NAME, PARTRONYMIC)

values (:ID, :SURNAME, :NAME, :PARTRONYMIC);

end;

-- Добавим запись в таблицу книг

create procedure insert\_book (id integer, NAME varchar(50),

ID\_TYPE integer, ID\_GENRE integer)

as

begin

insert into BOOK (id, NAME, ID\_TYPE, ID\_GENRE)

values (:id, :NAME, :ID\_TYPE, :ID\_GENRE);

end;

-- Добавим запись в таблицу связи книг и авторов

create procedure insert\_book\_author (id\_author integer, id\_book integer)

as

begin

insert into BOOK\_AUTHOR (id\_author, id\_book)

values (:id\_AUTHOR, :ID\_BOOK);

end;

-- Добавим запись в таблицу экземпляров

create procedure insert\_exemp (id integer, id\_book integer, id\_status integer)

as

begin

insert into EXEMP (id, id\_book, id\_status)

values (:id, :id\_book, :id\_status);

end;

-- Добавим запись в таблицу жанров

create procedure insert\_genre (ID integer, NAME varchar(50))

as

begin

insert into GENRE (ID, NAME)

values (:ID, :NAME);

end;

-- Добавим запись в таблицу поступлений книг

create procedure insert\_intake (ID integer, DATE\_INTAKE date, ID\_PROVAIDER integer, ID\_EXEMP integer)

as

begin

insert into intake (ID, DATE\_INTAKE, ID\_PROVAIDER, ID\_EXEMP)

values (:ID, :DATE\_INTAKE, :ID\_PROVAIDER, :ID\_EXEMP);

end;

-- Добавим запись в таблицу операций по книгам

create procedure insert\_OPERATION (ID integer, ID\_READERSHIP integer,

ID\_EXEMP integer, ID\_TYPE\_OF\_ISSUE integer, PERIOD integer, DATE\_OF\_ISSUE date, RETURN\_DATE date)

as

begin

insert into OPERATION (ID, ID\_READERSHIP, ID\_EXEMP, ID\_TYPE\_OF\_ISSUE, PERIOD, DATE\_OF\_ISSUE, RETERN\_DATE)

values (:ID, :ID\_READERSHIP, :ID\_EXEMP, :ID\_TYPE\_OF\_ISSUE, :PERIOD, :DATE\_OF\_ISSUE, :RETURN\_DATE) ;

end;

-- Добавим запись в таблицу читателей

create procedure insert\_READERSHIP (ID integer, SURNAME varchar(50), NAME varchar(50), PARTRONYMIC varchar(50), PHONE\_NUMBER integer)

as

begin

insert into READERSHIP (ID, SURNAME, NAME, PARTRONYMIC, PHONE\_NUMBER)

values (:ID, :SURNAME, :NAME, :PARTRONYMIC, :PHONE\_NUMBER) ;

end;

-- Добавим запись в таблицу резерва

create procedure insert\_RESERVE (ID integer, ID\_READERSHIP integer, ID\_EXEMP integer, DATE\_RESERVE date)

as

begin

insert into RESERVE (ID, ID\_READSHIP, ID\_EXEMP, DATE\_RESERVE)

values (:ID, :ID\_READERSHIP, :ID\_EXEMP, :DATE\_RESERVE) ;

end;

-- Добавим запись в таблицу поставщиков

create procedure insert\_PROVAIDER (ID integer, NAME varchar(50))

as

begin

insert into provaider (ID, NAME)

values (:ID, :NAME);

end;

-- Добавим запись в таблицу статусов

create procedure insert\_STATUS (ID integer, NAME varchar(100))

as

begin

insert into STATUS (ID, NAME)

values (:ID, :NAME);

end;

-- Добавим запись в таблицу типов книг

create procedure insert\_TYPE (ID integer, NAME varchar(50))

as

begin

insert into type (ID, NAME)

values (:ID, :NAME);

end;

-- Добавим запись в таблицу "тип выдачи"

create procedure insert\_TYPE\_OF\_ISSUE (ID integer, NAME varchar(50))

as

begin

insert into TYPE\_OF\_ISSUE (ID, NAME)

values (:ID, :NAME);

end;

-- Добавим запись в таблицу списания книг

create procedure insert\_write\_off (ID integer, DATE\_WRITE\_OFF date, ID\_EXEMP integer)

as

begin

insert into write\_off (ID, date\_write\_off, id\_exemp)

values (:ID, :date\_write\_off, :id\_exemp);

end;

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

SET NAMES WIN1251;

CONNECT 'C:\123\BIBLIOTEKA.FDB' USER 'SYSDBA' PASSWORD 'masterkey';

set term ^;

create procedure update\_PERIOD1 (period int) as

begin

update operation set PERIOD = :period where OPERATION.RETERN\_DATE = 0;

end^;

Увеличение периода сдачи книг, у не сдавших книги читателей

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

SET NAMES WIN1251;

CONNECT 'C:\123\BIBLIOTEKA.FDB' USER 'SYSDBA' PASSWORD 'masterkey';

set term ^;

create procedure del\_type as

begin

delete from TYPE where ID not in (select ID\_TYPE from BOOK);

end ^

Удаление не использованных типов.

Выполнение индивидуального задания:

Реализовать следующие запросы:

1. Вывести 10 наиболее популярных книг, которые не на руках.
2. Вывести 5 авторов, книги которых пользуются наибольшей популярностью.
3. Удалить неиспользуемые жанры.

1)Вывесть 10 наиболее популярных книг, которые не на руках

CREATE VIEW SELECTTOP10BOOK

AS

select first 10 EXEMP.ID, EXEMP.ID as EXEMP,

COUNT(OPERATION.ID\_EXEMP) as Sell

from operation, EXEMP

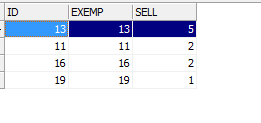
where (OPERATION.ID\_EXEMP=EXEMP.ID) and (exemp.id\_status=0)

group by EXEMP.ID

order by Sell desc

;

Результат работы скрипта:



**Дополнение по замечанию: ” В задании 1 необходимо популярность вычислять по книгам, а не по экземплярам.”**

1\_1)Вывести 10 наиболее популярных книг, которые не на руках

SET NAMES WIN1251;

CONNECT 'C:\123\BIBLIOTEKA.FDB' USER 'SYSDBA' PASSWORD 'masterkey';

CREATE VIEW SELECTTOP10BOOK2

AS

select first 10 book.name, book.name as book\_name,

COUNT(OPERATION.ID\_EXEMP) as Sell

from

book as book inner join exemp as exemp

on book.id=exemp.id\_book

inner join operation as operation

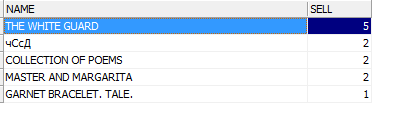
on exemp.id=operation.id\_exemp

where exemp.id\_status=0

group by book.name

order by Sell desc;

Результаты работы скрипта:



2)Вывести 5 авторов, книги которых пользуются наибольшей популярностью

SET NAMES WIN1251;

CONNECT 'C:\123\BIBLIOTEKA.FDB' USER 'SYSDBA' PASSWORD 'masterkey';

create view popular5author as

select first 5 author.id, author.id as author,

COUNT(OPERATION.ID\_EXEMP) as Sell

from operation, EXEMP,book\_author, AUTHOR

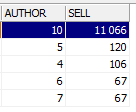
where (OPERATION.ID\_EXEMP=EXEMP.ID) and (EXEMP.ID\_BOOK=BOOK\_AUTHOR.ID\_BOOK) and (BOOK\_AUTHOR.ID\_AUTHOR=AUTHOR.ID)

group by author.id

order by Sell desc

;

Результат работы скрипта:



* 1. Удалить неиспользуемые жанры

SET NAMES WIN1251;

CONNECT 'C:\123\BIBLIOTEKA.FDB' USER 'SYSDBA' PASSWORD 'masterkey';

set term ^;

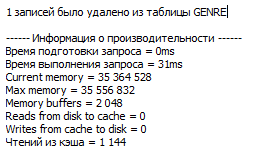
create procedure del\_genre as

begin

delete from genre where ID not in (select ID\_genre from BOOK);

end ^

Результат выполнения процедуры:



1. **Вывод**

В результате выполнения работы был изучен язык управления данными SQL-DML. Были выполенны стандартные запросы получения данных из БД и так же запросы в соответствии с индивидуальным заданием.

Приведенные ниже команды группы DML работают исключительно со строками и выполняются преимущественно клиентом:

INSERT — добавление строк(и);

SELECT — выборка строк(и);

UPDATE — изменение строк(и);

DELETE— удаление строк(и);

Так как язык стандартизирован, выполнять эти операции относительно удобно для решения задач различной сложности.

Так же были изучены хранимые процедуры и представления, с помощью которых с легкостью можно добавлять данные в БД. При выполнении работы затруднений не возникло.