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

Выполнила студентка гр. 43501/1 Я.Н. Олефир

(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

connect 'D:\bd\avia.fdb' user 'SYSDBA' password 'masterkey';

create view TicketsSel as select \* from tickets;

create view SheduleSel as select \* from shedule;

create view FlightsSel as select \* from flights;

create view AirlinesSel as select \* from airlines;

create view AirplanesSel as select \* from airplanes;

create view Ticket\_typeSel as select \* from ticket\_type;

create view RoutesSel as select \* from routes;

create view ManufacturersSel as select \* from manufacturers;

create view ReturnSel as select \* from return;

create view PrioritySel as select \* from priority;

create view StatusSel as select \* from status;

Выборка данных из таблицы Tickets:



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

create view selectIvanov as select \* from Priority where Pax like 'Ivanov';

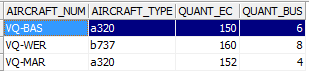
create view selectQE as select \* from airplanes where Quant\_bus between 0 and 10;

create view selectShed as select \* from Shedule where Shed\_id in (2,3);

Были созданы следующие выборки:

* Из таблицы Priority показать пассажиров с фамилией Ivanov.
* Из таблицы Airplanes показать самолеты с количеством мест в бизнес-классе от 0 до 10
* Из таблицы Shedule показать строки расписания, имеющие номера 2 и 3.

Пример выборки самолетов с количеством мест в бизнес-классе от 0 до 10.



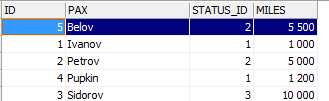
Пример выборки строк расписания, имеющих номера 2 и 3.



1. Выборка всех данных с сортировкой по нескольким полям

create view selectSortPaxMiles as select \* from Priority order by Pax asc, Miles asc;

Пассажиры, отсортированные по имени и количеству статусных миль.



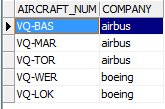
1. Выборка данных из связанных таблиц

create view selectAircraftNumCompany as select Airplanes.Aircraft\_Num as Aircraft\_Num,

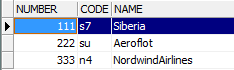
Manufacturers.company as Company from Airplanes, Manufacturers where Airplanes.Aircraft\_type = Manufacturers.Aircraft\_type;

create view selectFlightsCodeName as select Flights.Number as Number, Flights.Code as Code, Airlines.Name as Name from Flights,Airlines where Flights.Code = Airlines.Code;

Номер борта и производитель



* Номер рейса, код авиакомпании и название авиакомпании.

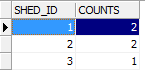


1. Запрос, рассчитывающий совокупную характеристику с использованием группировки, наложите ограничение на результат группировки

create view selectShedCount as select Tickets.shed\_id as Shed\_id,

COUNT(Tickets.Shed\_id) as Counts from Tickets group by Tickets.Shed\_id;

Вывод количества купленных билетов на каждый рейс в расписании.



1. Использование вложенного запроса

create view selectUnusedAirlines as select \* from Airlines where Code

not in (select Code from Flights);

Отображение авиакомпаний, которых нет в списке рейсов.



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

set term ^;

create procedure insert\_airlines ( i varchar(2), d varchar(20))

as begin

insert into Airlines (Code, Name)

values (:i, :d);

end ^

create procedure insert\_manufacturers ( i varchar(7), d varchar(10))

as begin

insert into Manufacturers (Aircraft\_type, Company)

values (:i, :d);

end ^

create procedure insert\_ttype ( i integer, d varchar(15))

as begin

insert into Ticket\_type (Ticket\_type\_id, Ticket\_type)

values (:i, :d);

end ^

create procedure insert\_routes ( i varchar(10), d varchar(7))

as begin

insert into Routes (Route\_id, Route)

values (:i, :d);

end ^

create procedure insert\_flights ( i integer, d varchar(3), s varchar(3),t varchar(5),v varchar(5),q varchar(2),c varchar(7), w integer, b integer)

as begin

insert into Flights (Number, Departure, Arrival, Time\_dep,Time\_arr, Code, Day\_week, Price\_ec, Price\_bus)

values (:i, :d, :s, :t, :v, :q, :c, :w, :b);

end ^

create procedure insert\_airlplanes ( i varchar(7), d varchar(7), w integer, b integer)

as begin

insert into Airplanes (Aircraft\_num, Aircraft\_type, Quant\_ec, Quant\_bus)

values (:i, :d, :w, :b);

end ^

create procedure insert\_shedule ( i integer, d date, s integer, t varchar(3),

v varchar(3),q varchar(5),c varchar(5), w varchar(7))

as begin

insert into Shedule (Shed\_id, Date\_fl, Number, Departure, Arrival,

Time\_dep, Time\_arr, Aircraft\_num)

values (:i, :d, :s, :t, :v, :q, :c, :w);

end ^

create procedure insert\_tickets ( i integer, d integer, s varchar(20), t varchar(20),

v integer,q integer,c integer, w varchar(10))

as begin

insert into Tickets (Id, Ticket\_id, Byer, Pax, Shed\_id, Price, Ticket\_type\_id, Route\_id)

values (:i, :d, :s, :t, :v, :q, :c, :w);

end ^

create procedure insert\_status ( i integer, d varchar(15))

as begin

insert into Status (Status\_id, Status)

values (:i, :d);

end ^

create procedure insert\_return ( i integer, d integer, v integer, s date)

as begin

insert into Return (Id, Ticket\_id, Price, Return\_date)

values (:i, :d, :v, :s);

end ^

create procedure insert\_priority ( i integer, d varchar(20), v integer, s integer)

as begin

insert into Priority (Id, Pax, Status\_id, Miles)

values (:i, :d, :v, :s);

end ^

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

set term ^;

create procedure update\_priority (miles int) as

begin

update Priority set Miles = :miles where Priority.Status\_id > 2;

end^;

Добавление количества миль для пассажиров, имеющих статус Platinum.

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

set term ^;

create procedure del\_price(i int) as

begin

delete from Tickets where Ticket\_type\_id = :i and

Price = (select MIN(Price) from Tickets where Ticket\_type\_id = :i);

end ^

Данная процедура удаляет билет с наименьшей стоимостью, имеющий заданный способ возврата.

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

set term ^;

create procedure del\_airplanes as

begin

delete from Airplanes where Aircraft\_num not in (select Aircraft\_num from Shedule);

end ^

Удаление самолетов, которые не задействованы в расписании.

Выполнение индивидуального задания:

Выполнить следующие запросы:

1. Отобразить 5 наиболее популярных маршрутов.
2. Вывести рейтинг авиакомпаний по выручке.
3. Удалить неиспользуемые маршруты.
4. Отобразить 5 наиболее популярных маршрутов

connect 'D:\bd\avia.fdb' user 'SYSDBA' password 'masterkey';

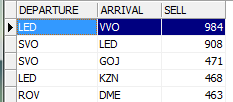
create view top5r as select first 5 Shedule.departure, Shedule.arrival,

COUNT (Tickets.shed\_id) as Sell from Tickets, Shedule where Tickets.shed\_id=Shedule.shed\_id

group by Shedule.departure, Shedule.arrival order by Sell desc ;

Для тестирования данного скрипта было сгенирировано 4000 записей в таблицу Tickets.

Результат работы скрипта. 5 самых популярных маршрутов:



1. Вывести рейтинг авиакомпаний по выручке

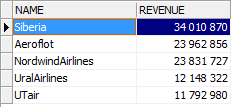
connect 'D:\bd\avia.fdb' user 'SYSDBA' password 'masterkey';

create view topAirlines as select Airlines.Name, SUM(Tickets.Price) AS Revenue from Flights,Shedule,Tickets, Airlines

where (Tickets.Shed\_id=Shedule.Shed\_id) and (Shedule.Number=Flights.Number) and (Airlines.Code=Flights.Code)

group by Airlines.Name order by Revenue desc ;

Результат работы скрипта:



1. Удалить неиспользуемые маршруты

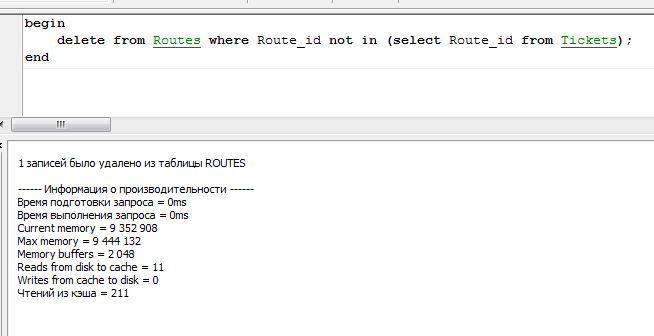
set term ^;

create procedure del\_routes as begin

delete from Routes where Route\_id not in (select Route\_id from Tickets);

end ^;

Результат выполнения процедуры:



1. **Вывод**

В результате выполнения работы был изучен язык управления данными SQL-DML. Это семейство компьютерных языков позволяет получать, вставлять, удалять и изменять данные при работе с базами данных.

Основные операторы языка DML:

* [SELECT](http://ru.wikipedia.org/wiki/Select_(SQL)) считывает данные, удовлетворяющие заданным условиям,
* [INSERT](http://ru.wikipedia.org/wiki/Insert_(SQL)) добавляет новые данные,
* [UPDATE](http://ru.wikipedia.org/wiki/Update_(SQL)) изменяет существующие данные,
* [DELETE](http://ru.wikipedia.org/wiki/Delete_(SQL)) удаляет данные;

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.