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. 43501/3 Никитенко А.П.

(подпись)

Руководитель А.В. Мяснов

(подпись)

“ ” 2016 г.

Санкт-Петербург

2016

**Цель работы**

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

**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 преподавателю.
4. 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 (StructuredQueryLanguage) - язык структурированных запросов. Он позволяет формировать весьма сложные запросы к базам данных. В SQL определены два подмножества языка:

SQL-DDL (DataDefinitionLanguage) - 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 (DataManipulationLanguage) - язык манипулирования данными: добавление, изменение, удаление и извлечение данных, управления транзакциями.

**Выполнение работы**

Выполнение стандартных запросов:

1. Сделать выборку всех данных из каждой таблицы:

create view attachments as select \* from attachment;

create view buildings as select \* from building;

create view departments as select \* from department;

create view departmentsadmissions as select \* from departmentsadmission;

create view employees as select \* from employee;

create view factualvisits as select \* from factualvisit;

create view periodadmissions as select \* from periodadmission;

create view rooms as select \* from room;

create view temporaryadmissions as select \* from temporaryadmission;

create view visitoradmissions as select \* from visitoradmission;

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

CREATE VIEW stand\_2 AS

SELECT emp\_id,full\_name FROM employee

WHERE emp\_id BETWEEN 1 AND 30 AND full\_name LIKE '%Ivan%';

CREATE VIEW stand\_2\_2 AS

SELECT dep\_id,dep\_name FROM department

WHERE dep\_id in(9,10,15) AND dep\_id BETWEEN 5 AND 13;

CREATE VIEW stand\_2\_3 AS

SELECT room\_num,room\_name FROM room

WHERE room\_num in(1,17,18,19,107,115) AND room\_name LIKE '%bookkeep%';

3. Создайте в запросе вычисляемое поле

CREATE VIEW stand\_3 AS

SELECT emp\_id,full\_name,(work\_time2-work\_time1)/3600 as work\_mode FROM employee;

4. Сделайте выборку всех данных с сортировкой по нескольким полям

CREATE VIEW stand\_4 AS

SELECT \* FROM room

ORDER BY open\_time,room\_num;

5. Создайте запрос, вычисляющий несколько совокупных характеристик таблиц

CREATE VIEW stand\_5 AS

SELECT avg((work\_time2-work\_time1)/3600) as avg\_work\_mode,count(emp\_id) as emp\_amount FROM employee;

6. Сделайте выборку данных из связанных таблиц (не менее двух примеров)

CREATE OR ALTER VIEW STAND\_6(

DEP\_ID,

DEP\_NAME,

BUILDING\_NUM,

BUILDING\_NAME)

AS

SELECT department.dep\_id, department.dep\_name, building.building\_num,building.building\_name

FROM department,building

WHERE department.building\_num=building.building\_num;

CREATE OR ALTER VIEW STAND\_6\_2(

ROOM\_NUM,

ROOM\_NAME,

WEEK\_DAY,

EMP\_NAME)

AS

SELECT periodadmission.room\_num,room.room\_name,periodadmission.week\_day,employee.full\_name

FROM periodadmission,room,employee

WHERE periodadmission.emp\_id=employee.emp\_id AND

periodadmission.room\_num=room.room\_num;

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

CREATE VIEW stand\_7 AS

SELECT room.room\_name, count(departmentsadmission.dep\_id) as amount

FROM room,departmentsadmission

WHERE departmentsadmission.room\_num=room.room\_num

GROUP BY room.room\_name

HAVING count(\*)>1;

8. Придумайте и реализуйте пример использования вложенного запроса

CREATE VIEW stand\_8 AS

SELECT room.room\_name

FROM room

WHERE room.room\_num not in (SELECT departmentsadmission.room\_num

FROM departmentsadmission WHERE departmentsadmission.dep\_id =15)

AND room.room\_num<200;

9. С помощью оператора INSERT добавьте в каждую таблицу по одной записи

create procedure insert\_attachment ( eid int, did int)

as begin

insert into attachment (emp\_id, dep\_id)

values (:eid, :did);

end;

create procedure insert\_building ( bnum int, bname varchar(255))

as begin

insert into building (building\_num, building\_name)

values (:bnum, :bname);

end;

create procedure insert\_department ( did int, dname varchar(255), bnum int)

as begin

insert into department (dep\_id, dep\_name,building\_num)

values (:did, :dname, :bnum);

end;

create procedure insert\_departmentadm ( did int, rnum int)

as begin

insert into departmentsadmission (dep\_id, room\_num)

values (:did, :rnum);

end;

create procedure insert\_employee ( eid int, fname varchar(255), adr varchar(255), phone char(12), pst varchar(255), t1 time , t2 time)

as begin

insert into employee (emp\_id, full\_name, adress, phone\_num, post, work\_time1, work\_time2)

values (:eid, :fname, :adr, :phone, :pst, :t1, :t2);

end;

create procedure insert\_factvisit ( eid int, rnum int, vdate date, t1 time, t2 time)

as begin

insert into factualvisit (emp\_id, room\_num, visit\_date, start\_of\_vis, end\_of\_vis)

values (:eid, :rnum, :vdate, :t1, :t2);

end;

create procedure insert\_periodadm ( eid int, rnum int, wday varchar(255), t1 time, t2 time)

as begin

insert into periodadmission (emp\_id, room\_num, week\_day, start\_of\_adm, end\_of\_adm)

values (:eid, :rnum, :wday, :t1, :t2);

end;

create procedure insert\_room ( rnum int, rname varchar(255), t1 time, t2 time)

as begin

insert into room (room\_num, room\_name, open\_time, close\_time)

values (:rnum, :rname, :t1, :t2);

end;

create procedure insert\_temporaryadm ( eid int, rnum int, vdate date, t1 time, t2 time)

as begin

insert into temporaryadmission (emp\_id, room\_num, visit\_date, start\_of\_adm, end\_of\_adm)

values (:eid, :rnum, :vdate, :t1, :t2);

end;

create procedure insert\_visitoradm ( fname varchar(255), phone char(12), rnum int , vdate date , t1 time, t2 time)

as begin

insert into visitoradmission (full\_name, phone\_num, room\_num, visit\_date, start\_of\_vis, end\_of\_vis)

values (:fname, :phone, :rnum, :vdate, :t1, :t2);

end;

Теперь используем процедуры:

execute procedure insert\_employee(9,'Baranov Artem Viktorovich','Mira street 77.15','79054233848','cleaner','05:00:00','07:30:00');

execute procedure insert\_room(97,'Utility room','00:00:00','23:59:59');

execute procedure insert\_building(3,'Utility service office');

execute procedure insert\_department(17,'Utility service',3);

execute procedure insert\_departmentadm(17,97);

execute procedure insert\_attachment(9,17);

execute procedure insert\_periodadm(9,18,'monday','06:00:00','08:00:00');

execute procedure insert\_factvisit(9,18,'2015-12-23','06:15:23','07:43:11');

execute procedure insert\_temporaryadm(9,1,'2016-01-12','06:22:23','07:15:45');

execute procedure insert\_visitoradm('Arsenov Gleb Alekseevich','79650982742',18,'2016-01-13','12:30:00','13:00:00');

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

create procedure update\_department (dname varchar(255)) as

begin

update department set dep\_name = :dname where dep\_name like 'bookkeeping%';

end;

execute procedure update\_department('Bookkeeping service');

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

create procedure delete\_employee (pst varchar(255)) as

begin

delete from employee where post = :pst and

work\_time1 = (select max(work\_time1) from employee);

end;

execute procedure delete\_employee('accountant');

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

create procedure delete\_notadm as

begin

delete from employee where emp\_id not in (select emp\_id from periodadmission);

end;

execute procedure delete\_notadm;

Выполнение индивидуальных заданий:

1. Вывести сотрудников, имеющих доступ к комнате в заданное время.

Для постоянных пропусков по дням недели(на примере комнаты №19):

CREATE OR ALTER VIEW ROOM\_PERIODADM\_TIME(

EMP\_ID,

FULL\_NAME,

WEEK\_DAY,

ROOM\_NUM)

AS

select periodadmission.emp\_id,employee.full\_name,periodadmission.week\_day,periodadmission.room\_num from employee,periodadmission

where periodadmission.emp\_id=employee.emp\_id

and periodadmission.room\_num=19

and periodadmission.start\_of\_adm<'01:20:00'

and periodadmission.end\_of\_adm>'02:00:00';

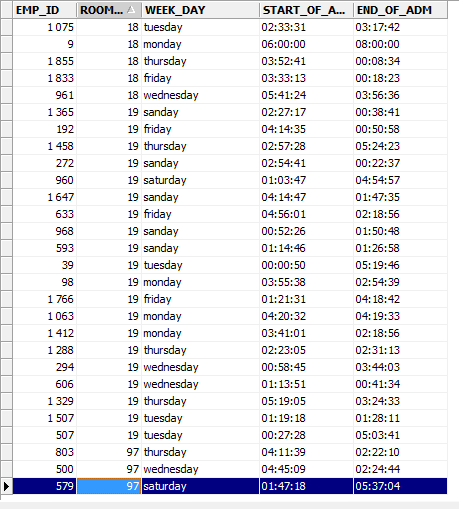


Рис.1 Существующие постоянные допуски

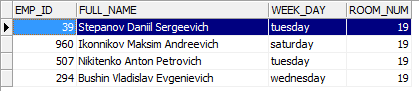


Рис.2 Результат

Для временных пропусков на определенную дату(на примере комнаты №1314):

CREATE OR ALTER VIEW ROOM\_TEMPORARYADM\_TIME(

EMP\_ID,

FULL\_NAME,

VISIT\_DATE,

ROOM\_NUM)

AS

select temporaryadmission.emp\_id,employee.full\_name,temporaryadmission.visit\_date,temporaryadmission.room\_num from employee,temporaryadmission

where temporaryadmission.emp\_id=employee.emp\_id

and temporaryadmission.room\_num=1314

and temporaryadmission.start\_of\_adm<'01:00:00'

and temporaryadmission.end\_of\_adm>'03:00:00';

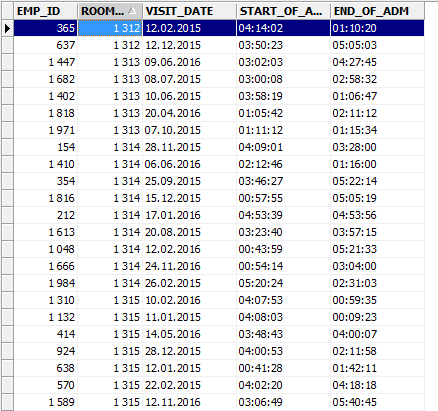


Рис.3 Существующие временные пропуски

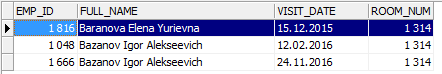


Рис.4 Результат

2. Вывести посещения сотрудника комнат заданного здания.

Для того, чтобы выполнить это задание пришлось добавить еще одну таблицу, которая устанавливает соответствие между номером комнаты и номером здания, в котором она находится, т.к. в существующей структуре БД такого соответствия нет, есть только соответствие между номерами отделов и номерами комнат, к которым у этого отдела есть доступ. Но не всегда различные отделы, у которых есть допуск к одной и той же комнате находятся в одном здании, например, служба уборки имеет доступ как к хозяйственным помещениям, так и к обычным рабочим помещениям(например, кабинеты бухгалтеров), которые могут находиться в любом здании.

create table room\_build (

room\_num int not null references room(room\_num),

building\_num int not null references building(building\_num));

Создана процедура для занесения данных в эту таблицу:

create procedure insert\_room\_build ( rnum int, bnum int)

as begin

insert into room\_build (room\_num, building\_num)

values (:rnum, :bnum);

end;

Далее были занесены записи принадлежности комнат к зданиям.

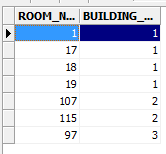


Рис. 5 Принадлежность комнат к зданиям

Запрос, выполняющий вывод посещений сотрудником комнат заданного здания(на примере сотрудника №9, здание №3):

CREATE OR ALTER VIEW EMP\_FACTVIS\_DEF\_BUILD(

EMP\_ID,

FULL\_NAME,

ROOM\_NUM,

VISIT\_DATE,

START\_OF\_VIS,

END\_OF\_VIS,

BUILDING\_NUM)

AS

select factualvisit.emp\_id,employee.full\_name,factualvisit.room\_num,factualvisit.visit\_date,factualvisit.start\_of\_vis,factualvisit.end\_of\_vis,room\_build.building\_num from factualvisit,employee,room\_build

where factualvisit.emp\_id=employee.emp\_id

and employee.emp\_id=9

and factualvisit.room\_num=room\_build.room\_num

and room\_build.building\_num=107;



Рис.6 Фактические посещения сотрудником комнат здания №3



Рис.7 Фактические посещения сотрудником комнат здания №107

Работа запросов была проверена на следующем наборе данных:

|  |  |
| --- | --- |
| Таблица | Количество записей |
| Attachment | 10000 |
| Building | 503 |
| Department | 10005 |
| Departmentsadmission | 30009 |
| Employee | 2004 |
| Factualvisit | 200005 |
| Periodadmission | 200004 |
| Room | 1007 |
| Room\_build | 10007 |
| Temporaryadmission | 55394 |
| Visitoradmission | 50001 |

3. Удалить неиспользуемые временные интервалы.

Задание было отменено преподавателем, т.к. относилось к заданию из предыдущей работы(Ввести учет промежутков времени (рабочее, нерабочее и пр.)), которое было выполнено не совсем так, как имел ввиду преподаватель, но его реализация была принята. В замен этому заданию преподаватель предложил придумать новое задание самостоятельно.

Вывести список всех сотрудников, работающих на территории заданного здания.

CREATE OR ALTER VIEW EMP\_WORK\_DEF\_BUILD(

EMP\_ID,

FULL\_NAME,

DEP\_ID)

AS

select employee.emp\_id,employee.full\_name,attachment.dep\_id from employee,attachment where

employee.emp\_id=attachment.emp\_id and

attachment.dep\_id in (select dep\_id from department where building\_num=357);



Рис. 8 Результат работы запроса(вывод работников здания №357)

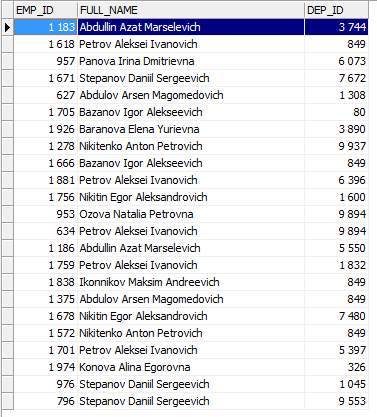


Рис. 9 Результат работы запроса(вывод работников здания №489)

**Вывод**

В результате работы было проведено знакомство и изучение языка SQL-DML. Были выполнены стандартные запросы извлечения данных, а также выполнены запросы в соответствии с индивидуальным заданием.

Научились создавать представления и процедуры.

Представления - это таблицы чье содержание выбирается или получается из других таблиц. Они работают в запросах и операторах DML точно также как и основные таблицы, но не содержат никаких собственных данных. Представления подобны окнам, через которые можно просматривать информацию, хранящуюся в базовых таблицах.

Хранимые процедуры позволяют объединить последовательность запросов и сохранить их на сервере. Хранимые процедуры удобно использовать, когда необходимо выполнить много похожих действий, не приходится каждый раз писать полный запрос.