Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

Отчет по лабораторной работе №3

По дисциплине «Базы данных»

«Язык SQL-DML»

Работу выполнили студенты группы №43501/4 Кушнаренко П.В. \_\_\_\_\_\_\_\_

Работу принял преподаватель Мяснов А.В.\_\_\_\_\_\_\_\_\_\_\_\_\_

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.

Список стандартных запросов

1. Сделайте выборку всех данных из каждой таблицы
2. Сделайте выборку данных из одной таблицы при нескольких условиях, с использованием логических операций, LIKE, BETWEEN, IN (не менее 3-х разных примеров)
3. Создайте в запросе вычисляемое поле
4. Сделайте выборку всех данных с сортировкой по нескольким полям
5. Создайте запрос, вычисляющий несколько совокупных характеристик таблиц
6. Сделайте выборку данных из связанных таблиц (не менее двух примеров)
7. Создайте запрос, рассчитывающий совокупную характеристику с использованием группировки, наложите ограничение на результат группировки
8. Придумайте и реализуйте пример использования вложенного запроса
9. С помощью оператора INSERT добавьте в каждую таблицу по одной записи
10. С помощью оператора UPDATE измените значения нескольких полей у всех записей, отвечающих заданному условию
11. С помощью оператора DELETE удалите запись, имеющую максимальное (минимальное) значение некоторой совокупной характеристики
12. С помощью оператора DELETE удалите записи в главной таблице, на которые не ссылается подчиненная таблица (используя вложенный запрос)
13. Ход работы
14. SQL-DML изучен в теории по методическим указаниям приведенным на ресурсе “trac”.
15. Создание стандартных запросов.

* Сделайте выборку всех данных из каждой таблицы

|  |
| --- |
| create view as\_addservcond as  select \* from addservicecondition;  create view as\_addservices as  select \* from addservices;  create view as\_categories as  select \* from categories;  create view as\_cleanrooms as  select \* from cleanrooms;  create view as\_clients as  select \* from clients;  create view as\_emp as  select \* from employees;  create view as\_posts as  select \* from posts;  create view as\_roomcond as  select \* from roomconditions;  create view as\_rooms as  select \* from rooms; |

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

|  |
| --- |
| CREATE VIEW Z2\_1 AS  SELECT id,name FROM clients  where id BETWEEN 1 AND 600 AND name LIKE 'CA%';  CREATE VIEW Z2\_2 AS  SELECT \* FROM roomconditions  where INROOM BETWEEN '1.1.2013' AND '1.1.2016'  AND ROOMID IN (1);  CREATE VIEW Z2\_3 AS  SELECT \* FROM employees  where ID BETWEEN 50 AND 100 AND NAME LIKE 'DA%'; |

* Создайте в запросе вычисляемое поле

|  |
| --- |
| CREATE VIEW Z3 AS  SELECT id,inroom,(cost\*2)/100 as smallcost FROM roomconditions |

* Сделайте выборку всех данных с сортировкой по нескольким полям

|  |
| --- |
| CREATE VIEW z4 AS  SELECT \*  FROM posts,rooms  ORDER BY rooms.id,posts.name |

* Создайте запрос, вычисляющий несколько совокупных характеристик таблиц

|  |
| --- |
| CREATE VIEW Z5 AS  SELECT min(cleanrooms.cleandate) AS min\_ , max(cleanrooms.number) as max\_,  count(room\_id) as Number  FROM cleanrooms  WHERE cleanrooms.id=1 |

* Сделайте выборку данных из связанных таблиц

|  |
| --- |
| CREATE OR ALTER VIEW Z6\_1(  man,  dolj)  AS  SELECT employees.name as man, posts.name as dolj  FROM posts, employees  WHERE posts.id = employees.postid; |

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

|  |
| --- |
| CREATE VIEW Z7 AS  SELECT rooms.id, COUNT(roomconditions.roomid)  FROM roomconditions, rooms  WHERE roomconditions.roomid=rooms.id  GROUP by rooms.id  HAVING COUNT(roomconditions.roomid)>100 |

* Придумайте и реализуйте пример использования вложенного запроса

|  |
| --- |
| select  rooms.numberroom as TOP10ROOMS  from  (  select  first 10 count(roomconditions.roomid) as schit,roomconditions.roomid  from  roomconditions  where  roomconditions.id IN (    select  roomconditions.id from roomconditions  where  roomconditions.InRoom> '25.6.2014' and OutRoom< '13.2.2016')  group by roomconditions.roomid  order by schit desc) as coof, rooms  where rooms.id = coof.roomid; |

* С помощью оператора INSERT добавьте в каждую таблицу по одной записи

|  |
| --- |
| create procedure in\_rooms(i INTEGER, c CHAR)  as  begin  INSERT INTO rooms VALUES (:i,:c);  end;  create procedure in\_sensors(i1 INTEGER,i2 INTEGER,i3 INTEGER, c CHAR)  as  begin  INSERT INTO sensors VALUES (:i1,:i2,:i3,:c);  end;  create procedure in\_worker(i INTEGER, c CHAR)  as  begin  INSERT INTO worker VALUES (:i,:c);  end;  create procedure in\_cards(i1 INTEGER,c CHAR, i2 INTEGER)  as  begin  INSERT INTO cards VALUES (:i1,:c,:i2);  end;  create procedure in\_department(i1 INTEGER,c CHAR(255), i2 INTEGER)  as  begin  INSERT INTO department VALUES (:i1,:c,:i2);  end;  create procedure in\_access\_by\_card(i1 INTEGER,i2 INTEGER,tin TIME, tout TIME)  as  begin  INSERT INTO access\_by\_card VALUES (:i1,:i2,:tin,:tout);  end;  create procedure in\_access\_by\_dep (i1 INTEGER,i2 INTEGER,tin TIME, tout TIME)  as  begin  INSERT INTO access\_by\_dep VALUES (:i1,:i2,:tin,:tout);  end;  create procedure in\_worker\_to\_dep (i1 INTEGER,i2 INTEGER)  as  begin  INSERT INTO worker\_to\_dep VALUES (:i1,:i2);  end;  create procedure in\_entry (i1 INTEGER,i2 INTEGER, t timestamp)  as  begin  INSERT INTO entry VALUES (1,100001,'2015-10-29 8:55:00');  end;  create procedure in\_exit (i1 INTEGER,i2 INTEGER, t timestamp)  as  begin  INSERT INTO exit VALUES (1,100001,'2015-10-29 8:55:00');  end; |

|  |
| --- |
| connect 'C:\Users\Bocman\Documents\Database\Hotel.fdb' user 'SYSDBA' password 'masterkey';  insert into Posts (id, Name) values (1, 'manager');  insert into Employees (id,Name,Birthday,Education,PostID,Adres,Phone,Salary )  values (1, 'Obama B.V.','25.6.1985','High',1,'USA','5555555',20000 );  insert into AddServices (id, Name, Cost) values (1, 'Digit TV in Room', 1000);  insert into Categories (id, Name,NumBeds,DopInfo,Cost)  values (11, 'Lux',6,'Very good',28000);  insert into Rooms (id, NumberRoom,CategoryID)  values (1, 101,1);  insert into Clients (id, Name,Birthday,Sex,Passport)  values (1, 'John','19.04.1985','Male',48569);  insert into RoomConditions (id, Number,RoomID,InRoom,OutRoom,Cost,ClientID,StateArmor)  values (1,1,5,'26.11.2015','28.11.2015',25000,3,'F');  insert into AddServiceCondition (id,Number,AddServiceId,EmployeeID,StartAS,EndAS,ClientID)  values (1, 200,1,1,'26.11.2015','28.11.2015',4);  insert into CleanRooms(id,number, roomid, employeeId,cleandate) values (1,135001,0,9,'11.11.2015');  commit; |

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

|  |
| --- |
| create procedure in\_update\_access\_time (t1 date,t2 date,id int)  as  begin  UPDATE roomconditions  SET Inroom=:t1, Outroom=:t2  where roomconditions .id=:id;  end; |

|  |
| --- |
| UPDATE roomconditions  Inroom =’1.1.2014’, Outroom =’1.1.2016’  where id=200 |

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

|  |
| --- |
| create procedure in\_delete\_ roomconditions \_min\_time (i INTEGER)  as  begin  delete from roomconditions  where inroom= (select min(inroom) from roomconditions where id = :i)  AND id=:i;  end; |

|  |
| --- |
| delete from roomconditions  where inroom= (select min(inroom) from roomconditions where id = 1)  AND id=1 |

1. Выполнение индивидуальных заданий
2. Вывести 10 наиболее популярных номеров за заданный промежуток времени.

|  |
| --- |
| select  rooms.numberroom as TOP10ROOMS  from  (  select  first 10 count(roomconditions.roomid) as schit,roomconditions.roomid  from  roomconditions  where  roomconditions.id IN (    select  roomconditions.id from roomconditions  where  roomconditions.InRoom> '25.6.2014' and OutRoom< '13.2.2016')  group by roomconditions.roomid  order by schit desc) as coof, rooms  where rooms.id = coof.roomid; |

1. Вывести 10 клиентов, которые заказали дополнительных услуг на наибольшую сумму..

|  |
| --- |
| select first 10 clients.name, schit  from ( select (addservicecondition.clientid),addservicecondition.clientid as client, sum(addservicecondition.cost) as schit  from addservicecondition  group by addservicecondition.clientid  order by schit desc), clients  where clients.id = client ; |

Затраты при 100 000 записях в таблице

------ Performance info ------

Prepare time = 46ms

Execute time = 312ms

Avg fetch time = 44,57 ms

Current memory = 9 749 104

Max memory = 9 765 200

Memory buffers = 2 048

Reads from disk to cache = 0

Writes from cache to disk = 0

Fetches from cache = 302 459

1. Удалить неиспользуемые категории номеров.

|  |
| --- |
| delete from categories where id not in(select categoryid from rooms); |

1. Создание представлений и хранимых процедур

В ходе работы все запросы уже были сохранены в виде представлений, а выполняемые действия в виде процедур.

1. Выводы

В данной работе изучен язык SQL-DML. Получены навыки работы с запросами. В данной работе мы научились извлекать необходимые данные из таблиц и представлять их в нужной форме. Также мы научились создавать представления и процедуры.

Представления дают нам возможность создавать некий интерфейс для разных типов пользователей. Они позволяют пользователю выполнять все необходимые запросы как запросы к обычным таблицам, но при этом не позволяют ему изменять данные. Таким образом получается некоторая система безопасности.

Хранимые процедуры дают возможность выполнять заранее прописанные действия, возможно, с некоторым изменением параметров. Это удобно, когда при работе необходимо часто выполнять схожие действия, требующие некоторого количества строк кода.