Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

Отчет по лабораторной работе №4

По дисциплине «Базы данных»

«Язык SQL-DML»

Работу выполнили студенты группы №43501/1 Кричевский В.А. \_\_\_\_\_\_\_\_

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

Санкт-Петербург

2016

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\_typetour as  select \* from typetour;  create view as\_additionalserv as  select \* from additionalserv;  create view as\_tourfirm as  select \* from tourfirm;  create view as\_tour as  select \* from tour;  create view as\_client as  select \* from client;  create view as\_country as  select \* from country;  create view as\_zakaz as  select \* from zakaz;  create view as\_room as  select \* from room;  create view as\_hotel as  select \* from hotel; |

* Сделайте выборку данных из одной таблицы при нескольких условиях, с использованием логических операций, LIKE, BETWEEN, IN (не менее 3-х разных примеров) (Рис. 1, 2 и 3 соответственно)

|  |
| --- |
| CREATE VIEW Z2\_1 AS  SELECT id,name FROM clients  where id BETWEEN 1 AND 600 AND name LIKE 'a%';  CREATE VIEW Z2\_2 AS  SELECT \* FROM tour  where arrival\_date BETWEEN '1.1.2014' AND '1.1.2017'  AND tourfirm\_id IN (1);  CREATE VIEW Z2\_3 AS  SELECT \* FROM zakaz  where ID BETWEEN 50 AND 100 AND Cost BETWEEN 1000 AND 3000; |

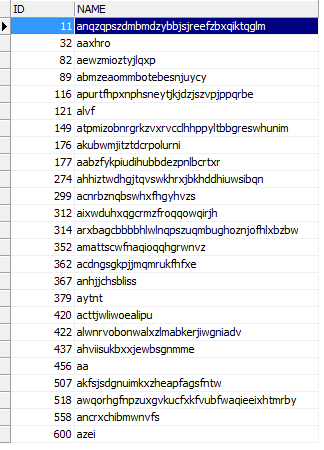


Рис.1



Рис.2

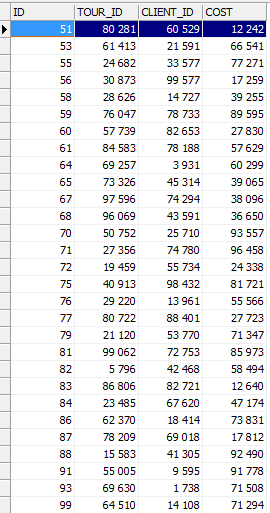
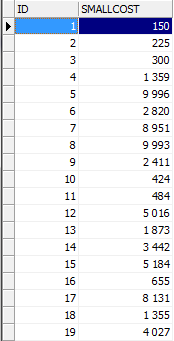


Рис.3

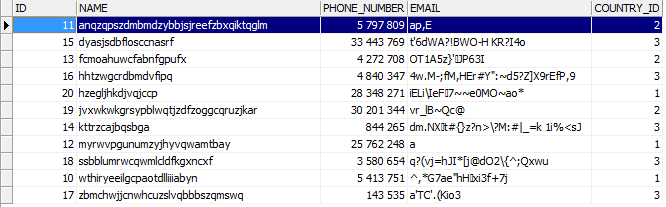
* Создайте в запросе вычисляемое поле

|  |
| --- |
| CREATE VIEW Z3 AS  SELECT id,(cost\*5)/100 as smallcost FROM zakaz |



* Сделайте выборку всех данных с сортировкой по нескольким полям

|  |
| --- |
| CREATE VIEW z4 AS  SELECT \*  FROM client  WHERE id between 10 and 20  ORDER BY client.name asc, client.country\_id desc |



* Создайте запрос, вычисляющий несколько совокупных характеристик таблиц

|  |
| --- |
| CREATE VIEW Z5 AS  SELECT min(cost) AS min\_ , max(cost) as max\_  FROM zakaz  WHERE zakaz.client\_id between 20 and 500 |



* Сделайте выборку данных из связанных таблиц

|  |
| --- |
| CREATE OR ALTER VIEW Z6\_1(  tourf,  tur  )  AS  SELECT tourfirm.name as tourf, tour.id as tur  FROM tourfirm, tour  WHERE tourfirm.id = tour.tourfirm\_id; |



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

|  |
| --- |
| CREATE VIEW Z7\_1(  TOUR\_ID,  NUM\_TUR)  AS  SELECT zakaz.client\_id, COUNT(tour.id) as num\_tur  FROM zakaz, tour  where tour.tourfirm\_id in (1)  GROUP by zakaz.client\_id |

* Придумайте и реализуйте пример использования вложенного запроса

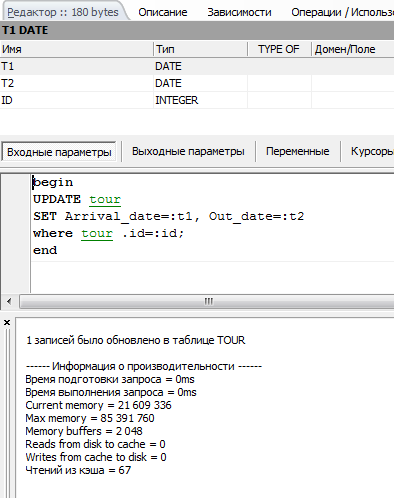
|  |
| --- |
| delete from typetour where id not in(select typetour\_id from tour); |

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

|  |
| --- |
| create procedure in\_rooms( i1 INTEGER,i2 INTEGER,i3 INTEGER, c1 CHAR(1), c2 CHAR(1))  as  begin  INSERT INTO rooms VALUES (:i1,:i2,:i3,:c1,:c2);  end;  create procedure in\_zakaz(i1 INTEGER,i2 INTEGER,i3 INTEGER, i4 INTEGER)  as  begin  INSERT INTO zakaz VALUES (:i1,:i2,:i3,:i4);  end;  create procedure in\_country(i INTEGER, c CHAR)  as  begin  INSERT INTO country VALUES (:i,:c);  end;  create procedure in\_tourfirm(i1 INTEGER, i2 INTEGER,c CHAR)  as  begin  INSERT INTO tourfirm VALUES (:i1,:i2,:c);  end;  create procedure in\_tour(i1 INTEGER,i2 INTEGER,i3 INTEGER, c CHAR(1),tin TIME, tout TIME)  as  begin  INSERT INTO tour VALUES (:i1,:i2,:i3,:c,:tin,:tout);  end;  create procedure in\_client(i1 INTEGER,c1 CHAR,i2 INTEGER, c2 CHAR, i3 INTEGER)  as  begin  INSERT INTO client VALUES (:i1,:c1,:i2,:c2,:i3);  end;  create procedure in\_typetour (i INTEGER, c CHAR)  as  begin  INSERT INTO country VALUES (:i,:c);  end;  create procedure in\_dostp (i INTEGER, c CHAR)  as  begin  INSERT INTO country VALUES (:i,:c);  end;  create procedure in\_addserv (i INTEGER, c CHAR)as  begin  INSERT INTO country VALUES (:i,:c);  end; |

* 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 tour  SET Arrival\_date=:t1, Out\_date=:t2  where tour .id=:id;  end; |
| UPDATE tour  Arrival\_date =’1.1.2016’, Out\_date =’1.2.2016’  where id=200 |



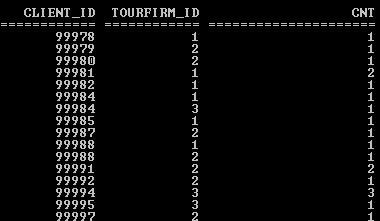
* С помощью оператора DELETE удалите запись, имеющую максимальное (минимальное) значение некоторой совокупной характеристики

|  |
| --- |
| create procedure in\_delete\_tiur\_min\_time (i INTEGER)  as  begin  delete from tour  where Arrival\_date= (select min(Arrival\_date) from your where id = :i)  AND id=:i;  end; |

|  |
| --- |
| delete from tour  where Arrival\_date = (select min(Arrival\_date) from tour where id = 1)  AND id=1 |

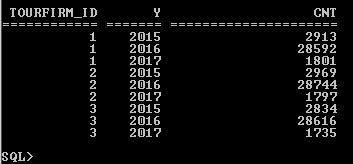
1. Выполнение индивидуальных заданий
2. Для каждого клиента вывести количество туров, в которых он был от каждой из турфирм.

|  |
| --- |
| create or alter view clienttour as  select client\_id, tourfirm\_id, count(tour\_id) as cnt  from zakaz join tour  on zakaz.tour\_id = [tour.id](https://vk.com/away.php?utf=1&to=http%3A%2F%2Ftour.id)  group by client\_id, tourfirm\_id;  select \* from clienttour; |



1. Для каждой турфирмы вывести количество туров в каждом году.

|  |
| --- |
| create or alter view yeartour as  select tourfirm\_id,  EXTRACT(Year FROM arrival\_date) AS Y,  count(tour\_id) as cnt  from tour join zakaz  on [tour.id](https://vk.com/away.php?utf=1&to=http%3A%2F%2Ftour.id) = zakaz.tour\_id  group by tourfirm\_id,Y;  select \* from yeartour; |



1. Удалить неиспользуемые типы туров.

|  |
| --- |
| delete from Trip where id not in(select typetour\_id from trip); |

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.