**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**Институт информационных технологий и управления**

**Кафедра компьютерных систем и программных технологий**

**ОТЧЕТ**

**по лабораторной работе №4**

**«Язык SQL-DML»**

**Информационное обеспечение систем управления**

Студент гр. 43501/1 Васильев А.М.

Преподаватель Мяснов А.В.

**Санкт-Петербург**

**2015**

# 1.Цель работы

Ознакомиться с языком создания запросов управления данными SQL-DML.

## Программа работы:

1. Изучить SQL-DML

2. Выполнить все запросы из списка стандартных запросов.

3. Реализовать SQL-запросы в соответствии с индивидуальным заданием.

4. Выполненные запросы SELECT сохранить в БД в виде представлений, запросы INSERT, UPDATE или DELETE -- в виде ХП.

## Индивидуальное задание:

БД хранит информацию об абонентах мобильного оператора, услугах (с фиксированной и посекундной тарификацией), использованию абонентами услуг, счетах, списаниях и зачислениях на счета. ИС дает возможность управлять различными акциями и начислением бонусов.

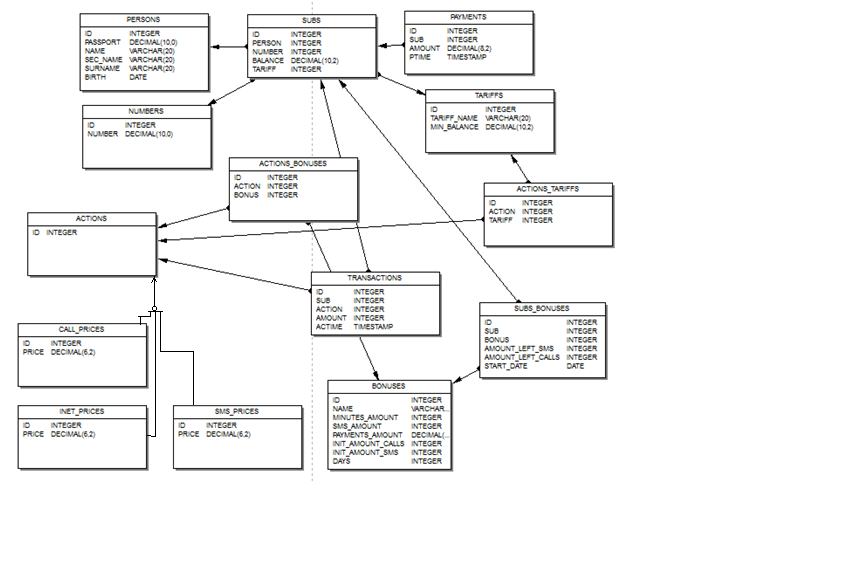


Рис 1. ER-диаграмма, построенная в IBExpert.

# 2. Стандартные запросы

## 1. Сделайте выборку всех данных из каждой таблицы

### **SQL> select \* from persons;**

|  |  |
| --- | --- |
| ID PASSPORT NAME SEC\_NAME SURNAME==== ===================== ==================== ==================== ==========1 4001111222 ИВАН ИВАНОВИЧ АНДРЕЕВ2 4001113222 ИВАН ОЛЕГОВИЧ ВИЙ3 4002342422 ИГОРЬ ПЕТРОВИЧ ИВАНОВ4 4023456422 ИВАН ИВАНОВИЧ ИВАНОВ5 4111116422 АЛЕКСАНДР АРТЕМОВИЧ СЕЧИН6 4100556422 АЛЕКСАНДР ИВАНОВИЧ БУЙНОВ7 4113759052 ИВАН ГЕННАДЬЕВИЧ БАРАНОВ8 4113243443 ИГОРЬ ИГОРЕВИЧ СУХОВ9 4123876543 АЛЕКСАНДР АЛЕКСАНДРОВИЧ СИДОРОВ10 4109566332 АНДРЕЙ ПЕТРОВИЧ АНДРЕЕВ11 4122222243 ИННА ИВАНОВНА СУХОВА12 4191986043 АЛЕНА ИВАНОВНА БАРАНОВА13 4198764643 ИРИНА ОЛЕГОВНА СЕЧИНА14 4145334223 ОЛЬГА АЛЕКСАНДРОВНА ОЛЬГОВА | BIRTH===========1990-10-041990-10-041956-11-141978-12-241989-01-241949-11-241977-03-031987-01-031989-01-241999-01-241989-01-281988-01-241948-07-211978-12-21 |

### **SQL> select \* from numbers;**

### ID NUMBER

### ============ =====================

### 1 9214994816

### 2 9217764333

### 3 9211643790

### 4 9217786968

### 5 9212945569

### 6 9215530482

### 7 9210524327

### 8 9217207873

### 9 9210983112

### 10 9216038440

### 11 9210962657

### 12 9213219902

### 13 9211158428

### 14 9210290094

### 15 9218908475

Данная таблица была сгенерирована скриптом типа:

### create sequence gen\_numid;

### insert into numbers values (gen\_id(gen\_numid,1),rand()\*9999999+9210000000);

### ….

### insert into numbers values (gen\_id(gen\_numid,1),rand()\*9999999+9210000000);

### **SQL> select \*from subs;**

### ID PERSON NUMBER BALANCE TARIFF

### ============ ============ ============ ===================== ============

### 1 1 1 100.00 1

### 2 2 2 150.00 2

### 3 3 3 -50.00 3

### 4 4 4 152.34 2

### 5 5 5 12.22 1

### 6 6 6 100.00 1

### 7 7 7 100.00 1

### 8 8 8 100.00 1

### 9 9 9 100.00 1

### 10 10 10 100.00 4

### 11 11 11 100.00 1

### 12 12 12 100.00 6

### 13 13 13 100.00 1

### 14 14 14 100.00 5

### **SQL> select \*from tariffs;**

### ID TARIFF\_NAME MIN\_BALANCE

### ============ ==================== =====================

### 1 Простецкий 0.00

### 2 SMS-зависимый 0.00

### 3 Звонить маме -10.00

### 4 Все по 3 0.00

### 5 Модем 0.00

### 6 Стабильный 0.00

### **SQL> select \*from actions;**

### ID

### ============

### 1

### 2

### 3

### 4

### 5

### 6

### 7

### 8

### 9

### 10

### 11

### **SQL> select \*from actions\_tariffs;**

### ID ACTION TARIFF

### ============ ============ ============

### 1 3 1

### 2 6 1

### 3 2 2

### 4 7 2

### 5 3 3

### 6 5 3

### 7 8 1

### 8 8 2

### 9 9 3

### 10 10 4

### 11 7 4

### 12 11 4

### 13 3 5

### 14 7 5

### 15 11 5

### 16 3 6

### 17 7 6

### 18 9 6

### **SQL> select \*from bonuses;**

### ID NAME MINUTES\_AMOUNT SMS\_AMOUNT PAYMENTS\_AMOUNT INIT\_AMOUNT\_CALLS INIT\_AMOUNT\_SMS

### DAYS

### ============ ==================== ============== ============ ===================== ================= =============== ==

### ==========

### 1 Болтун 100 <null> <null> <null> <null>

### 30

### 2 Телеграфист <null> 300 <null> <null> 10

### 30

### 3 Меценат <null> <null> 1000.00 100 100

### <null>

### **SQL> select \*from actions\_bonuses;**

### ID ACTION BONUS

### ============ ============ ============

### 1 5 1

### 2 1 2

### 3 1 3

### 4 4 3

### **SQL> select \*from sms\_prices;**

### ID PRICE

### ============ ============

### 1 0.00

### 2 2.00

### 3 5.00

### 10 3.00

### **SQL> select \*from call\_prices;**

### ID PRICE

### ============ ============

### 4 0.00

### 5 1.00

### 6 2.00

### 7 3.00

### **SQL> select \*from inet\_prices;**

### ID PRICE

### ============ ============

### 8 9.00

### 9 15.00

### 11 3.00

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

### SQL> select persons.name,persons.surname from persons where birth between '1920-01-01' and '1960-12-31';

### NAME SURNAME

### ==================== ====================

### ИГОРЬ ИВАНОВ

### АЛЕКСАНДР БУЙНОВ

### ИРИНА СЕЧИНА

### SQL> select persons.name,persons.surname,persons.birth from persons where extract(month from birth) in (10,11,12);

### NAME SURNAME BIRTH

### ==================== ==================== ===========

### ИВАН АНДРЕЕВ 1990-10-04

### ИВАН ВИЙ 1990-10-04

### ИГОРЬ ИВАНОВ 1956-11-14

### ИВАН ИВАНОВ 1978-12-24

### АЛЕКСАНДР БУЙНОВ 1949-11-24

### ОЛЬГА ОЛЬГОВА 1978-12-21

### SQL> select persons.name,persons.surname,persons.birth from persons where extract(month from birth) in (10,11,12) and bi

### rth > '1960-01-01';

### NAME SURNAME BIRTH

### ==================== ==================== ===========

### ИВАН АНДРЕЕВ 1990-10-04

### ИВАН ВИЙ 1990-10-04

### ИВАН ИВАНОВ 1978-12-24

### ОЛЬГА ОЛЬГОВА 1978-12-21

### create view a\_names as select name,surname from persons

### where name like 'А%';

### SQL> select \* from a\_names;

### NAME SURNAME

### ==================== ====================

### АЛЕКСАНДР СЕЧИН

### АЛЕКСАНДР БУЙНОВ

### АЛЕКСАНДР СИДОРОВ

### АНДРЕЙ АНДРЕЕВ

### АЛЕНА БАРАНОВА

## 3. Создайте в запросе вычисляемое поле

Вывести цены на sms в долларах

### SQL> select sms\_prices.price,sms\_prices.price/60.0 as dollars from sms\_prices;

### PRICE DOLLARS

### ============ =====================

### 0.00 0.000

### 2.00 0.033

### 5.00 0.083

### 3.00 0.050

## 4. Сделайте выборку всех данных с сортировкой по нескольким полям

## SQL> select \* from persons order by name,surname;

## ID PASSPORT NAME SEC\_NAME SURNAME BIRTH

## ============ ===================== ==================== ==================== ==================== ===========

## 6 4100556422 АЛЕКСАНДР ИВАНОВИЧ БУЙНОВ 1949-11-24

## 5 4111116422 АЛЕКСАНДР АРТЕМОВИЧ СЕЧИН 1989-01-24

## 9 4123876543 АЛЕКСАНДР АЛЕКСАНДРОВИЧ СИДОРОВ 1989-01-24

## 12 4191986043 АЛЕНА ИВАНОВНА БАРАНОВА 1988-01-24

## 10 4109566332 АНДРЕЙ ПЕТРОВИЧ АНДРЕЕВ 1999-01-24

## 1 4001111222 ИВАН ИВАНОВИЧ АНДРЕЕВ 1990-10-04

## 7 4113759052 ИВАН ГЕННАДЬЕВИЧ БАРАНОВ 1977-03-03

## 2 4001113222 ИВАН ОЛЕГОВИЧ ВИЙ 1990-10-04

## 4 4023456422 ИВАН ИВАНОВИЧ ИВАНОВ 1978-12-24

## 3 4002342422 ИГОРЬ ПЕТРОВИЧ ИВАНОВ 1956-11-14

## 8 4113243443 ИГОРЬ ИГОРЕВИЧ СУХОВ 1987-01-03

## 11 4122222243 ИННА ИВАНОВНА СУХОВА 1989-01-28

## 13 4198764643 ИРИНА ОЛЕГОВНА СЕЧИНА 1948-07-21

## 14 4145334223 ОЛЬГА АЛЕКСАНДРОВНА ОЛЬГОВА 1978-12-21

## 5. Создайте запрос, вычисляющий несколько совокупных характеристик таблиц

Найти среднюю и максимальную стоимость sms для всех тарифов:

### SQL> select avg(price),max(price) from sms\_prices;

### AVG MAX

### ===================== ============

### 2.50 5.00

Найти суммарное пополнение счета абонентом за все время:

### SQL> select sum(amount) from payments where sub=1;

### SUM

### =====================

### 100.00

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

Вывести список денег на счету для каждого активного номера:

### SQL> select numbers.number,subs.balance from numbers,subs where subs.number = numbers.id;

### NUMBER BALANCE

### ===================== =====================

### 9218689746 100.00

### 9213631714 150.00

### 9213326438 -50.00

### 9217041711 152.34

### 9212566612 12.22

### 9218121878 100.00

### 9216827227 100.00

### 9211474503 100.00

### 9210706846 100.00

### 9212451464 100.00

### 9214644158 100.00

### 9216557497 100.00

### 9213074226 100.00

### 9217535088 100.00

Вывести список всех id услуг для всех тарифов:

### SQL> select tariffs.tariff\_name,actions.id from tariffs,actions,actions\_tariffs where actions\_tariffs.tariff=tariffs.id

### and actions\_tariffs.action = actions.id;

### TARIFF\_NAME ID

### ==================== ============

### Простецкий 3

### Простецкий 6

### Простецкий 8

### SMS-зависимый 2

### SMS-зависимый 7

### SMS-зависимый 8

### Звонить маме 3

### Звонить маме 5

### Звонить маме 9

### Все по 3 10

### Все по 3 7

### Все по 3 11

### Модем 3

### Модем 7

### Модем 11

### Стабильный 3

### Стабильный 7

### Стабильный 9

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

### SQL> select name,count(id) from persons group by name order by count(id) desc;

### NAME COUNT

### ==================== ============

### ИВАН 4

### АЛЕКСАНДР 3

### ИГОРЬ 2

### ИННА 1

### ОЛЬГА 1

### АЛЕНА 1

### ИРИНА 1

### АНДРЕЙ 1

### SQL> select name,count(id) from persons group by name having count(id)>=2;

### NAME COUNT

### ==================== ============

### АЛЕКСАНДР 3

### ИВАН 4

### ИГОРЬ 2

## 8. Придумайте и реализуйте пример использования вложенного запроса

Вывести баланс для заданного номера:

### SQL> select subs.balance from subs where subs.number = (select numbers.id from numbers where numbers.number=9218826198);

### BALANCE

### =====================

### 100.00

## 9. Оператор insert

Добавить в таблицы по одной записи

### insert into tariffs values

### (6,'Стабильный');

### insert into bonuses values

### (3,'Меценат',NULL,NULL,1000,100,100,NULL);

### insert into sms\_prices values (10,3.0);

### insert into call\_prices values (7,3.0);

### insert into inet\_prices values (11,3.0);

### insert into actions values (11);

### insert into actions\_tariffs values (9,9,3);

### insert into actions\_bonuses values (4,4,3);

### insert into persons values

### (14,4145334223,'ОЛЬГА','АЛЕКСАНДРОВНА','ОЛЬГОВА','1978-12-21');

### insert into numbers values (gen\_id(gen\_numid,1),rand()\*9999999+9210000000);

### insert into subs values (

### 14,14,14,100.0,5);

### insert into payments values

### (5,5,25.50,current\_timestamp (0));

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

Изменить фамилию и паспорт абонента по номеру паспорта:

### SQL> update persons set passport = 4114847886, surname = 'БУЙНОВА' where passport = 4145334223;

### SQL> select \* from persons;

### ID PASSPORT NAME SEC\_NAME SURNAME BIRTH

### ============ ===================== ==================== ==================== ==================== ===========

### 1 4001111222 ИВАН ИВАНОВИЧ АНДРЕЕВ 1990-10-04

### 2 4001113222 ИВАН ОЛЕГОВИЧ ВИЙ 1990-10-04

### 3 4002342422 ИГОРЬ ПЕТРОВИЧ ИВАНОВ 1956-11-14

### 4 4023456422 ИВАН ИВАНОВИЧ ИВАНОВ 1978-12-24

### 5 4111116422 АЛЕКСАНДР АРТЕМОВИЧ СЕЧИН 1989-01-24

### 6 4100556422 АЛЕКСАНДР ИВАНОВИЧ БУЙНОВ 1949-11-24

### 7 4113759052 ИВАН ГЕННАДЬЕВИЧ БАРАНОВ 1977-03-03

### 8 4113243443 ИГОРЬ ИГОРЕВИЧ СУХОВ 1987-01-03

### 9 4123876543 АЛЕКСАНДР АЛЕКСАНДРОВИЧ СИДОРОВ 1989-01-24

### 10 4109566332 АНДРЕЙ ПЕТРОВИЧ АНДРЕЕВ 1999-01-24

### 11 4122222243 ИННА ИВАНОВНА СУХОВА 1989-01-28

### 12 4191986043 АЛЕНА ИВАНОВНА БАРАНОВА 1988-01-24

### 13 4198764643 ИРИНА ОЛЕГОВНА СЕЧИНА 1948-07-21

### 14 4114847886 ОЛЬГА АЛЕКСАНДРОВНА БУЙНОВА 1978-12-21

Поздравить абонентов с днем рождения 100 рублями

### SQL> update subs set balance = balance+100 where subs.person = (select id from persons where extract(month from birth)=extract(month from current\_date

### ) and extract(day from birth)=extract(day from current\_date));

### SQL> select persons.surname,subs.balance from persons,subs where subs.person = persons.id;

### SURNAME BALANCE

### ==================== =====================

### АНДРЕЕВ 100.00

### ВИЙ 150.00

### ИВАНОВ -50.00

### ИВАНОВ 152.34

### СЕЧИН 12.22

### БУЙНОВ 100.00

### БАРАНОВ 100.00

### СУХОВ 100.00

### СИДОРОВ 100.00

### АНДРЕЕВ 100.00

### СУХОВА 100.00

### БАРАНОВА 100.00

### СЕЧИНА 100.00

### БУЙНОВА 200.00

Выполнено успешно.

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

Требуется удалить абонента с максимальной задолженностью. Для этого сначала требуется удалить записи о зачислениях на его счет, а затем саму запись в таблице абонентов.

### SQL> delete from payments where sub = (select first 1 id from subs order by balance);

### SQL> delete from subs where balance = (select min(balance) from subs);

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

Найдем номера, на которые нет оформленных договоров

### SQL> select id,number from numbers nn where not exists(select \* from subs where nn.id=subs.number);

### ID NUMBER

### ============ =====================

### 3 9216660853

### 15 9213947042

Удалим неиспользуемые номера

### SQL> delete from numbers nn where not exists(select \* from subs where nn.id=subs.number);

До:

### SQL> select \* from numbers;

### ID NUMBER

### ============ =====================

### 1 9214994816

### 2 9217764333

### 3 9211643790

### 4 9217786968

### 5 9212945569

### 6 9215530482

### 7 9210524327

### 8 9217207873

### 9 9210983112

### 10 9216038440

### 11 9210962657

### 12 9213219902

### 13 9211158428

### 14 9210290094

### 15 9218908475

После:

### SQL> select \* from numbers;

### ID NUMBER

### ============ =====================

### 1 9218049473

### 2 9216499136

### 3 9211068219

### 4 9211086349

### 5 9214878637

### 6 9213411520

### 7 9218971538

### 8 9219712036

### 9 9215527844

### 10 9216068250

### 11 9210028728

### 12 9212263143

### 13 9216613510

### 14 9219288779

На 15 номер не было договора, и он был удален.

# 3. Индивидуальное задание

Перед выполнением задания требуется заполнить таблицы пополнений счета и транзакций данными. Для этого создадим хранимые процедуры:

## 1. ХП для пополнения счета (по id номера телефона)

### create sequence gen\_payid;

### set term ^ ;

### create procedure do\_payment (num\_id int,amount decimal(8,2),t timestamp ) as

### declare variable sub\_id int;

### begin

### sub\_id = (select id from subs where number = :num\_id);

### insert into payments values (gen\_id(gen\_payid,1),:sub\_id,:amount,:t);

### update subs set balance = balance+:amount where subs.id = :sub\_id;

### end ^

### set term ;^

Для задания id платежа используется генератор. Также требуется заменить терминальный символ ; на время задания процедуры в скрипте.

### SQL> execute procedure do\_payment(1,50.0,dateadd(1 day to current\_timestamp));

### SQL> execute procedure do\_payment(2,250.0,dateadd(1 day to current\_timestamp));

### SQL> select \* from payments;

### ID SUB AMOUNT PTIME

### ============ ============ ============ =========================

### 1 1 50.00 2014-12-23 18:44:43.5060

### 2 2 250.00 2014-12-23 18:44:47.3330

### SQL> select \* from subs;

### ID PERSON NUMBER BALANCE TARIFF

### ============ ============ ============ ===================== ============

### 1 1 1 150.00 1

### 2 2 2 400.00 2

### 3 3 3 -50.00 3

### 4 4 4 152.34 2

### 5 5 5 12.22 1

### 6 6 6 100.00 1

### 7 7 7 100.00 1

### 8 8 8 100.00 1

### 9 9 9 100.00 1

### 10 10 10 100.00 4

### 11 11 11 100.00 1

### 12 12 12 100.00 6

### 13 13 13 100.00 1

### 14 14 14 100.00 5

Создадим скрипт для пополнения баланса всем пользователям:

### execute procedure do\_payment(1,50.0,dateadd(1 day to current\_timestamp));

### execute procedure do\_payment(2,50.0,dateadd(2 day to current\_timestamp));

### execute procedure do\_payment(3,150.0,dateadd(15 day to current\_timestamp));

### execute procedure do\_payment(5,350.0,dateadd(2 day to current\_timestamp));

### execute procedure do\_payment(4,20.0,dateadd(8 day to current\_timestamp));

### execute procedure do\_payment(8,90.0,dateadd(3 day to current\_timestamp));

### execute procedure do\_payment(4,20.0,dateadd(9 day to current\_timestamp));

### execute procedure do\_payment(2,40.0,dateadd(1 day to current\_timestamp));

### execute procedure do\_payment(1,130.0,dateadd(1 day to current\_timestamp));

### execute procedure do\_payment(10,200.0,dateadd(12 day to current\_timestamp));

### execute procedure do\_payment(12,110.0,dateadd(17 day to current\_timestamp));

### SQL> select \* from subs;

### ID PERSON NUMBER BALANCE TARIFF

### ============ ============ ============ ===================== ============

### 1 1 1 280.00 1

### 2 2 2 240.00 2

### 3 3 3 100.00 3

### 4 4 4 192.34 2

### 5 5 5 362.22 1

### 6 6 6 100.00 1

### 7 7 7 100.00 1

### 8 8 8 190.00 1

### 9 9 9 100.00 1

### 10 10 10 300.00 4

### 11 11 11 100.00 1

### 12 12 12 210.00 6

### 13 13 13 100.00 1

### 14 14 14 100.00 5

## 2. ХП для выполнения действия (sms)

Процедура должна заносить в таблицу транзакций сведения об операции и снимать сумму со счета абонента. Аргументы – id номера телефона и количество расчетных единиц (sms или минут или мегабайт).

### create sequence gen\_transid;

Сама процедура:

create procedure sms (num\_id int,amount int) as

declare variable sub\_id int;

declare variable cost decimal(6,2);

declare variable act int;

declare variable tariff int;

begin

sub\_id = (select id from subs where number = :num\_id);

tariff = (select tariff from subs where id=:sub\_id);

cost = (select price from sms\_prices,actions\_tariffs where sms\_prices.id = actions\_tariffs.action and actions\_tariffs.tariff=:tariff);

act = (select sms\_prices.id from sms\_prices,actions\_tariffs where sms\_prices.id = actions\_tariffs.action and actions\_tariffs.tariff=:tariff);

insert into transactions (id,sub,action,amount,actime) values

(gen\_id(gen\_transid,1),:sub\_id,:act,:amount,dateadd(-rand()\*31 day to current\_timestamp));

update subs set balance = balance - :cost\*:amount where subs.id = :sub\_id;

end ^

Результат:

### SQL> execute procedure sms(1,1);

### SQL> select \* from transactions;

### ID SUB ACTION AMOUNT ACTIME

### ============ ============ ============ ============ =========================

### 1 1 3 1 2014-12-22 20:10:33.3650

### SQL> select \* from subs;

### ID PERSON NUMBER BALANCE TARIFF

### ============ ============ ============ ===================== ============

### 1 1 1 275.00 1

### 2 2 2 240.00 2

### 3 3 3 100.00 3

### 4 4 4 192.34 2

### 5 5 5 362.22 1

### 6 6 6 100.00 1

### 7 7 7 100.00 1

### 8 8 8 190.00 1

### 9 9 9 100.00 1

### 10 10 10 300.00 4

### 11 11 11 100.00 1

### 12 12 12 210.00 6

### 13 13 13 100.00 1

### 14 14 14 100.00 5

SMS тарифицировалась правильно и со счета снято 5 рублей.

## Процедура для совершения звонков

### create procedure call (num\_id int,amount int) as

### declare variable sub\_id int;

### declare variable cost decimal(6,2);

### declare variable act int;

### declare variable tariff int;

### begin

### sub\_id = (select id from subs where number = :num\_id);

### tariff = (select tariff from subs where id=:sub\_id);

### 

### cost = (select price from call\_prices,actions\_tariffs where call\_prices.id = actions\_tariffs.action and actions\_tariffs.tariff=:tariff);

### act = (select call\_prices.id from call\_prices,actions\_tariffs where call\_prices.id = actions\_tariffs.action and actions\_tariffs.tariff=:tariff);

### insert into transactions (id,sub,action,amount,actime) values

### (gen\_id(gen\_transid,1),:sub\_id,:act,:amount,dateadd(-rand()\*31 day to current\_timestamp));

### update subs set balance = balance - :cost\*:amount where subs.id = :sub\_id;

### end ^

Скрипт для заполнения таблицы транзакций:

### execute procedure call(1,6);

### execute procedure call(3,4);

### execute procedure call(1,1);

### execute procedure call(8,2);

### execute procedure call(3,9);

### execute procedure call(8,1);

### execute procedure call(4,2);

### execute procedure call(5,9);

### execute procedure call(9,3);

### execute procedure call(6,1);

### execute procedure call(7,4);

### execute procedure call(7,1);

### execute procedure call(8,12);

### execute procedure call(9,10);

### execute procedure call(10,9);

### execute procedure call(11,3);

### execute procedure call(12,3);

### execute procedure call(13,2);

### execute procedure call(14,1);

### execute procedure sms(1,1);

### execute procedure sms(2,2);

### execute procedure sms(3,1);

### execute procedure sms(4,1);

### execute procedure sms(5,1);

### execute procedure sms(6,1);

### execute procedure sms(7,1);

### execute procedure sms(8,1);

### execute procedure sms(9,1);

### execute procedure sms(10,1);

### execute procedure sms(11,1);

### execute procedure sms(12,1);

### execute procedure sms(13,1);

### execute procedure sms(14,1);

### execute procedure sms(14,1);

### execute procedure sms(14,1);

### execute procedure sms(3,1);

### execute procedure sms(6,1);

### execute procedure sms(8,1);

### execute procedure sms(1,1);

### execute procedure sms(3,1);

### execute procedure sms(4,1);

### execute procedure sms(1,1);

### execute procedure sms(2,1);

### execute procedure sms(5,1);

### execute procedure sms(1,1);

Изменим скрипт так, чтобы дата транзакции получалась случайно в пределах прошедших 30 дней – вставим фрагмент вместо current\_timestamp:

### dateadd(-rand()\*31 day to current\_timestamp)

Результат:

### SQL> select \* from transactions;

### ID SUB ACTION AMOUNT ACTIME

### ============ ============ ============ ============ =========================

### 1 1 6 6 2014-12-15 20:52:15.5130

### 2 3 5 4 2014-11-26 20:52:15.5130

### 3 1 6 1 2014-12-02 20:52:15.5130

### 4 8 6 2 2014-11-21 20:52:15.5130

### 5 3 5 9 2014-12-03 20:52:15.5130

### 6 8 6 1 2014-12-18 20:52:15.5130

### 7 4 7 2 2014-11-29 20:52:15.5130

### 8 5 6 9 2014-12-06 20:52:15.5130

### 9 9 6 3 2014-11-27 20:52:15.5130

### 10 6 6 1 2014-12-07 20:52:15.5130

### 11 7 6 4 2014-11-25 20:52:15.5130

### 12 7 6 1 2014-12-16 20:52:15.5130

### 13 8 6 12 2014-11-30 20:52:15.5130

### 14 9 6 10 2014-11-26 20:52:15.5130

### 15 10 7 9 2014-12-07 20:52:15.5130

### 16 11 6 3 2014-11-30 20:52:15.5130

### 17 12 7 3 2014-12-13 20:52:15.5130

### 18 13 6 2 2014-12-20 20:52:15.5130

### 19 14 7 1 2014-12-03 20:52:15.5130

### 20 1 3 1 2014-11-24 20:52:15.5130

### ID SUB ACTION AMOUNT ACTIME

### ============ ============ ============ ============ =========================

### 21 2 2 2 2014-12-22 20:52:15.5130

### 22 3 3 1 2014-12-18 20:52:15.5130

### 23 4 2 1 2014-12-05 20:52:15.5130

### 24 5 3 1 2014-11-26 20:52:15.5130

### 25 6 3 1 2014-12-17 20:52:15.5130

### 26 7 3 1 2014-12-16 20:52:15.5130

### 27 8 3 1 2014-12-13 20:52:15.5130

### 28 9 3 1 2014-12-02 20:52:15.5130

### 29 10 10 1 2014-12-14 20:52:15.5130

### 30 11 3 1 2014-11-29 20:52:15.5130

### 31 12 3 1 2014-12-02 20:52:15.5130

### 32 13 3 1 2014-12-07 20:52:15.5130

### 33 14 3 1 2014-12-08 20:52:15.5130

### 34 14 3 1 2014-12-20 20:52:15.5130

### 35 14 3 1 2014-12-03 20:52:15.5130

### 36 3 3 1 2014-12-05 20:52:15.5130

### 37 6 3 1 2014-12-21 20:52:15.5130

### 38 8 3 1 2014-12-18 20:52:15.5130

### 39 1 3 1 2014-12-13 20:52:15.5130

### 40 3 3 1 2014-12-21 20:52:15.5130

### ID SUB ACTION AMOUNT ACTIME

### ============ ============ ============ ============ =========================

### 41 4 2 1 2014-12-14 20:52:15.5130

### 42 1 3 1 2014-11-24 20:52:15.5130

### 43 2 2 1 2014-11-25 20:52:15.5130

### 44 5 3 1 2014-11-29 20:52:15.5130

### 45 1 3 1 2014-12-06 20:52:15.5130

Количество денег на счетах абонентов уменьшилось:

### SQL> select \* from subs;

### ID PERSON NUMBER BALANCE TARIFF

### ============ ============ ============ ===================== ============

### 1 1 1 246.00 1

### 2 2 2 234.00 2

### 3 3 3 72.00 3

### 4 4 4 182.34 2

### 5 5 5 334.22 1

### 6 6 6 88.00 1

### 7 7 7 85.00 1

### 8 8 8 150.00 1

### 9 9 9 69.00 1

### 10 10 10 270.00 4

### 11 11 11 89.00 1

### 12 12 12 196.00 6

### 13 13 13 91.00 1

### 14 14 14 82.00 5

## Индивидуальные задания:

## 1. Вывести пять тарифов, по которым было максимальное поступление денег за выбранный период с суммами поступлений.

### SQL> select first 5 tariffs.tariff\_name,sum(payments.amount) as total from payments,tariffs,subs where subs.tariff=tariffs.id and payments.sub = subs.

### id group by tariffs.tariff\_name order by total desc;

### TARIFF\_NAME TOTAL

### ==================== =====================

### Простецкий 620.00

### Все по 3 200.00

### Звонить маме 150.00

### SMS-зависимый 130.00

### Стабильный 110.00

Сгруппировали записи по тарифу и вывели название и сумму зачислений, отсортировали и вывели первые 5.

## 2. Вывести десять пользователей, отправивших максимальное количество сообщений за выбранный период

### SQL> select first 10 sub,count(sub) from transactions t where exists (select \* from sms\_prices where id=t.action) and actime between '2014-11-25' and

### current\_timestamp group by sub order by count(sub) desc;

### SUB COUNT

### ============ ============

### 3 3

### 14 3

### 1 2

### 4 2

### 5 2

### 6 1

### 7 1

### 8 1

### 9 1

### 10 1

## 3. Удалить транзакции, которые старше выбранного временного интервала.

### SQL> delete from transactions where actime <='2014-11-30';

### SQL> select \* from transactions;

### ID SUB ACTION AMOUNT ACTIME

### ============ ============ ============ ============ =========================

### 1 1 6 6 2014-12-01 21:08:32.5640

### 2 3 5 4 2014-12-02 21:08:32.5640

### 3 1 6 1 2014-12-21 21:08:32.5640

### 4 8 6 2 2014-12-10 21:08:32.5640

### 5 3 5 9 2014-12-02 21:08:32.5640

### 6 8 6 1 2014-12-18 21:08:32.5740

### 7 4 7 2 2014-12-12 21:08:32.5740

### 8 5 6 9 2014-12-16 21:08:32.5740

### 9 9 6 3 2014-12-05 21:08:32.5740

### 10 6 6 1 2014-12-13 21:08:32.5740

### 11 7 6 4 2014-12-05 21:08:32.5740

### 12 7 6 1 2014-12-13 21:08:32.5740

### 14 9 6 10 2014-12-19 21:08:32.5740

### 16 11 6 3 2014-12-01 21:08:32.5740

### 18 13 6 2 2014-12-03 21:08:32.5740

### 22 3 3 1 2014-12-17 21:08:32.5740

### 24 5 3 1 2014-12-21 21:08:32.5740

### 26 7 3 1 2014-12-21 21:08:32.5740

### 28 9 3 1 2014-12-20 21:08:32.5740

### 33 14 3 1 2014-12-04 21:08:32.5740

### ID SUB ACTION AMOUNT ACTIME

### ============ ============ ============ ============ =========================

### 34 14 3 1 2014-12-01 21:08:32.5740

### 40 3 3 1 2014-12-17 21:08:32.5740

### 41 4 2 1 2014-12-11 21:08:32.5740

### 42 1 3 1 2014-12-22 21:08:32.5740

### 44 5 3 1 2014-12-14 21:08:32.5740

Старые транзакции (ноябрьские) удалены.

## Создание представления (по п.1.):

### SQL> create view top\_tariffs as select first 5 tariffs.tariff\_name,sum(payments.amount) as total from payments,tariffs,subs where subs.tariff=tariffs

### id and payments.sub = subs.

### CON> id group by tariffs.tariff\_name order by total desc;

### SQL> select \* from top\_tariffs;

### TARIFF\_NAME TOTAL

### ==================== =====================

### Простецкий 620.00

### Все по 3 200.00

### Звонить маме 150.00

### SMS-зависимый 130.00

### Стабильный 110.00

## Создание хранимой процедуры: (по п3)

### set term ^ ;

### create procedure delete\_older (d date) as

### begin

### delete from transactions where actime <=:d;

### end ^

### set term ; ^

### commit;

Вызов:

### SQL> execute procedure delete\_older('2014-11-30');

Результат аналогичен результату запроса из п.3.

# 4.Запросы к БД увеличенного объема

Требуется заполнить некоторые таблицы БД как минимум 100000 записей. Для этого создадим следующие хранимые процедуры и последовательно вызовем их в скрипте:

Создадим таблицы значений имен, отчеств и фамилий для случайного заполнения.

create table t\_names (id int primary key,name varchar(20));

create table t\_sec\_names (id int primary key,sec\_name varchar(20));

### create table t\_surnames (id int primary key,surname varchar(20));

И заполним эти таблицы данными.

Процедура заполнения таблицы клиентов:

### create sequence gen\_personid;

### set term ^ ;

### create procedure fill\_persons as

### declare variable i int;

### declare variable pass decimal(10);

### declare variable name varchar(20);

### declare variable sec\_name varchar(20);

### declare variable surname varchar(20);

### declare variable days int;

### declare variable random int;

### begin

### i=0;

### while (i < 100000) do

### begin

### pass = 4010000000+rand()\*999999;

### if (not exists(select \* from persons where passport=:pass)) then

### begin

### random = trunc(rand()\*14)+1;

### name = (select name from t\_names where id = :random);

### random = trunc(rand()\*14)+1;

### sec\_name = (select sec\_name from t\_sec\_names where id = :random);

### random = trunc(rand()\*8)+1;

### surname = (select surname from t\_surnames where id = :random);

### days = -rand()\*30\*12\*50;

### insert into persons values (gen\_id(gen\_personid,1),:pass,:name,:sec\_name,:surname,

### dateadd(:days day to date '1995-01-01'));

### i=i+1;

### end

### end

### end ^

### set term ; ^

### commit;

Вызов и отображение результата:

### execute procedure fill\_persons;

### select count(\*) as persons from persons;

Заполнение номеров:

### create sequence gen\_numid;

### set term ^ ;

### create procedure fill\_numbers as

### declare variable i int;

### declare variable num decimal(10);

### begin

### i=0;

### while (i < 110000) do

### begin

### num = 9210000000+rand()\*9999999;

### if (not exists(select \* from numbers where number=:num)) then

### begin

### insert into numbers values (gen\_id(gen\_numid,1),:num);

### i=i+1;

### end

### end

### end ^

### set term ; ^

### commit;

### execute procedure fill\_numbers;

### select count(\*) as numbers from numbers;

Заполнение абонентов(договоров)

### set term ^ ;

### create procedure fill\_subs as

### declare variable i int;

### begin

### i=1;

### while (i <= 100000) do

### begin

### insert into subs values (:i,:i,:i,rand()\*20,rand()\*5+1);

### i=i+1;

### end

### end ^

### set term ; ^

### commit;

### execute procedure fill\_subs;

### select count(\*) as subs from subs;

Заполнение таблицы платежей:

### set term ^ ;

### create procedure fill\_payments as

### declare variable i int;

### begin

### i=1;

### while (i <= 100000) do

### begin

### execute procedure do\_payment (rand()\*99999+1,rand()\*90+10,dateadd(-rand()\*31 day to current\_timestamp));

### i=i+1;

### end

### end ^

### set term ; ^

### commit;

### execute procedure fill\_payments;

### select count(\*) as payments from payments;

Заполнение звонков и sms:

### set term ^ ;

### create procedure fill\_actions as

### declare variable i int;

### begin

### i=1;

### while (i <= 50000) do

### begin

### execute procedure call(rand()\*99999+1,rand()\*29+1);

### execute procedure sms(rand()\*99999+1,1);

### i=i+1;

### end

### end ^

### set term ; ^

### commit;

### execute procedure fill\_actions;

### select count(\*) as transactions from transactions;

Результат выполнения всего скрипта:

### PERSONS

### ============

### 100000

### NUMBERS

### ============

### 110000

### SUBS

### ============

### 100000

### PAYMENTS

### ============

### 100000

### TRANSACTIONS

### ============

### 100000

Приведено количество записей в таблицах.

Выполнение запросов для большой БД:

## 1. Вывести пять тарифов, по которым было максимальное поступление денег за выбранный период с суммами поступлений.

### SQL> select first 5 tariffs.tariff\_name,sum(payments.amount) as total from payments,tariffs,subs where subs.tariff=tariffs.id and payments.sub = subs.

### id group by tariffs.tariff\_name order by total desc;

### TARIFF\_NAME TOTAL

### ==================== =====================

### SMS-зависимый 1111148.99

### Звонить маме 1104233.58

### Модем 1096771.99

### Все по 3 1092135.77

### Простецкий 553617.59

## 2. Вывести десять пользователей, отправивших максимальное количество сообщений за выбранный период

### SQL> select first 10 sub,count(sub) from transactions t where exists (select \* from sms\_prices where id=t.action) and actime between '2014-11-25' and

### current\_timestamp group by sub order by count(sub) desc;

### SUB COUNT

### ============ ============

### 63493 6

### 67131 6

### 72896 6

### 54509 5

### 57521 5

### 59557 5

### 48170 5

### 38035 5

### 30979 5

### 32344 5

## 3. Удалить транзакции, которые старше выбранного временного интервала.

### SQL> select count(\*) from transactions;

### COUNT

### ============

### 100000

### SQL> commit;

### **SQL> delete from transactions where actime < '2015-01-01';**

### SQL> select count(\*) from transactions;

### COUNT

### ============

### 56438

Транзакции были удалены.

# 5. Вывод

В ходе работы были изучены необходимые для работы с реляционными базами данных запросы языка SQL-DML: insert, update, delete. Изучены внутреннее и внешнее объединение таблиц, группировки и агрегатные функции, вложенные запросы, а также представления.

Поскольку представленная база данных имеет сложную нормализованную структуру, проводить операции добавления платежа, звонка или SMS в базу вручную слишком трудоемко. Поэтому были реализованы хранимые процедуры, упрощающие эти действия. Полученные навыки работы с SQL-запросами будут использованы в дальнейшей работе с базой данных.