Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

ПСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет информатики и вычислительной техники

Кафедра вычислительной техники

ПРЕДМЕТ «Базы Данных»

# *КУРСОВОЙ ПРОЕКТ*

***«Вариант 7»***

РАЗРАБОТКА БАЗЫ ДАННЫХ В СУБД POSTGRESQL

Выполнил студент группы 0024-01

Невзоров Л.Е.

Проверил преподаватель

Вертешев А.С.

Псков

2022

**Задание**

Тема: учет работы операторов с программным обеспечением в организации.

Объекты: сервера, рабочие станции, программное обеспечение.

Для моделирования задачи необходимо хранить следующую информацию:

\* Наименование рабочей станции (РС)

\* Наименование серверов, с которыми работают РС

\* Программное обеспечение, установленное на РС

\* ФИО оператора

\* Должность оператора

\* На каких РС может работать данный оператор

Дополнительные условия:

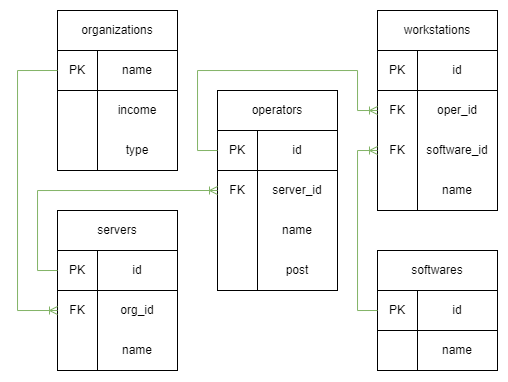
\* на одной рабочей станции может быть установлено несколько программ;

\* на разных рабочих станциях может быть установлено одинаковое программное

обеспечение;

\* один оператор может работать на разных рабочих станциях.

**Схема**



**Создание таблиц**

CREATE TABLE "organizations" (

  name VARCHAR PRIMARY KEY,

  income INTEGER NOT NULL,

  type VARCHAR(50) NOT NULL

);

INSERT INTO organizations (name,income,type)

VALUES

  ('Euismod Corporation',84218,'Product Company'),

  ('Euismod Urna Associates',56997,'Outsourcing Company'),

  ('Mattis Ornare LLP',149151,'Consulting Company'),

  ('Quis Urna Nunc Institute',53537,'Software Development');

CREATE TABLE "servers" (

  id SERIAL PRIMARY KEY,

  org\_id VARCHAR(50) NOT NULL REFERENCES organizations(name) ON DELETE CASCADE,

  name VARCHAR(50) NOT NULL

);

INSERT INTO servers (org\_id,name)

VALUES

  ('Euismod Corporation','Server 1'),

  ('Euismod Urna Associates','Server 1'),

  ('Mattis Ornare LLP','Server 1'),

  ('Quis Urna Nunc Institute','Server 1');

CREATE TABLE "operators" (

  id SERIAL PRIMARY KEY,

  server\_id INTEGER NOT NULL REFERENCES servers(id) ON DELETE CASCADE,

  name VARCHAR(50) NOT NULL,

  post VARCHAR(50) NOT NULL

);

INSERT INTO operators (server\_id,name,post)

VALUES

  (1,'Jackson Ochoa','Data Engeneer'),

  (1,'Hanae Crawford','ML-Engeneer'),

  (2,'Jameson Sandoval','Data Analyst'),

  (2,'Holmes Decker','Biznes Analyst'),

  (3,'Illana Branch','Data Scientist');

CREATE TABLE "workstations" (

  id SERIAL PRIMARY KEY,

  oper\_id INTEGER NOT NULL REFERENCES operators(id) ON DELETE CASCADE,

  software\_id INTEGER NOT NULL REFERENCES softwares(id) ON DELETE CASCADE,

  name VARCHAR(50) NOT NULL

);

INSERT INTO workstations (oper\_id,software\_id,name)

VALUES

  (1,9,'Station 1'),

  (1,5,'Station 1'),

  (1,10,'Station 1'),

  (1,8,'Station 1'),

  (1,2,'Station 1');

CREATE TABLE "softwares" (

  id SERIAL PRIMARY KEY,

  name VARCHAR(50) NOT NULL

);

INSERT INTO softwares (name)

VALUES

  ('PyCharm'),

  ('Visual Studio'),

  ('Anaconda'),

  ('Iputhon'),

  ('PostgreSQL');

**Запросы**

1)Три запроса на выборку с использованием join, использовать не мене 3 таблиц

SELECT og.name,

       se.name,

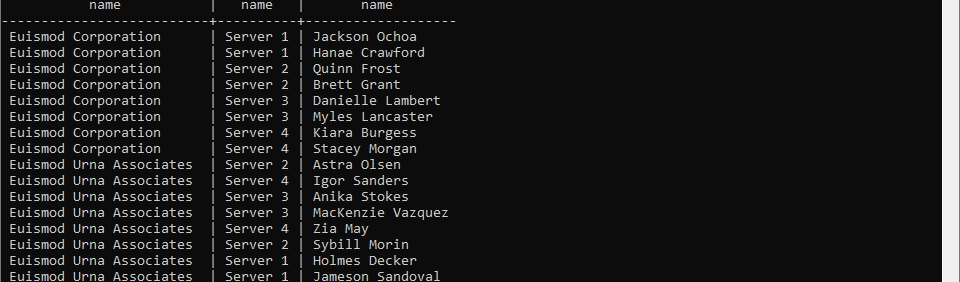
       op.name

FROM organizations AS og

JOIN servers AS se ON og.name=se.org\_id

JOIN operators AS op ON se.id=op.server\_id

ORDER BY og.name;



SELECT wo.name,

       op.name,

       wo.software\_id

FROM servers AS se

JOIN operators AS op ON se.id=op.server\_id

JOIN workstations AS wo ON op.id=wo.oper\_id

LIMIT 10;



SELECT wo.name,

       op.name,

       so.name

FROM operators AS op

JOIN workstations AS wo ON op.id=wo.oper\_id

JOIN softwares AS so ON so.id=wo.software\_id

LIMIT 10;



[2) Запросы на объединение, пересечение, разность, произведение, проекцию, деление](#_Toc91305837)

SELECT name,

       post

FROM operators

WHERE post = 'Data Engeneer'

UNION

SELECT name,

       post

FROM operators

WHERE post = 'Data Scientist';



SELECT name,

       post

FROM operators

WHERE post LIKE '%D%'

   OR post LIKE '%A%'

INTERSECT

SELECT name,

       post

FROM operators

WHERE post LIKE '%E%';



SELECT name,

       income

FROM organizations

EXCEPT

SELECT name,

       income

FROM organizations

WHERE income < 75000;



[3) Запросы на соединение: left join, right join, full outer join](#_Toc91305838)

SELECT og.name,

       se.name

FROM organizations AS og

LEFT OUTER JOIN servers AS se ON og.name=se.org\_id;



SELECT og.name,

       se.name

FROM organizations AS og

RIGHT OUTER JOIN servers AS se ON og.name=se.org\_id;



SELECT og.name,

       se.name

FROM organizations AS og

FULL OUTER JOIN servers AS se ON og.name=se.org\_id;



4) 2 Подзапроса: Простой, Сложный

SELECT og.name,

       se.name,

       op.name

FROM organizations AS og

JOIN servers AS se ON og.name=se.org\_id

JOIN operators AS op ON se.id=op.server\_id

WHERE og.income > (SELECT AVG(income)

                   FROM organizations)

ORDER BY og.name

LIMIT 10;



SELECT og.name,

       se.name,

       op.name

FROM organizations AS og

JOIN servers AS se ON og.name=se.org\_id

JOIN operators AS op ON se.id=op.server\_id

WHERE og.income < (SELECT AVG(tbl.income)

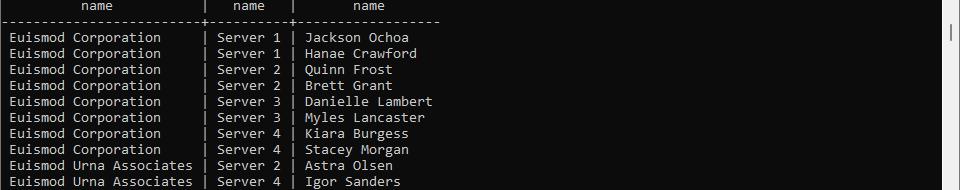
                   FROM (SELECT \*

                         FROM organizations

                         WHERE type LIKE '%Company') AS tbl)

ORDER BY og.name

LIMIT 10;



5) Запросы с различными уровнями вложенности подзапросов в частях SELECT, FROM, WHERE, HAVING

SELECT og.name,

       og.income,

       se.name,

       op.name,

       op.post

FROM organizations AS og

JOIN servers AS se ON og.name=se.org\_id

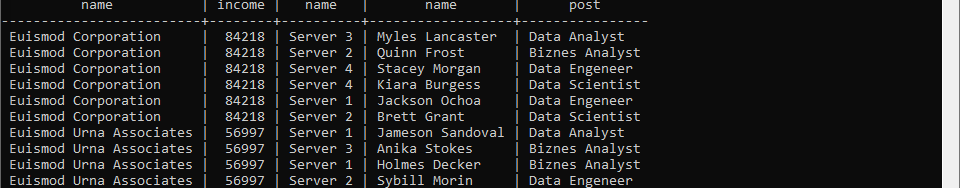
JOIN operators AS op ON se.id=op.server\_id

WHERE op.post LIKE '%Data%'

   OR op.post LIKE '%Analyst%'

ORDER BY og.name

LIMIT 10;



SELECT \*

FROM (SELECT og.name,

             og.income,

             se.name,

             op.name,

             op.post

      FROM organizations AS og

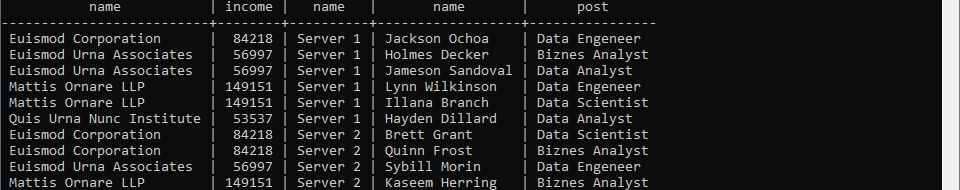
      JOIN servers AS se ON og.name=se.org\_id

      JOIN operators AS op ON se.id=op.server\_id) AS tbl

WHERE tbl.post LIKE '%Data%'

   OR tbl.post LIKE '%Analyst%'

LIMIT 10;



SELECT tbl.org\_name,

       AVG(tbl.income)

FROM (SELECT og.name AS org\_name,

             og.income,

             se.name AS server\_name,

             op.name AS oper\_name,

             op.post

      FROM organizations AS og

      JOIN servers AS se ON og.name=se.org\_id

      JOIN operators AS op ON se.id=op.server\_id) AS tbl

WHERE tbl.post LIKE '%Data%'

   OR tbl.post LIKE '%Analyst%'

GROUP BY tbl.org\_name

HAVING AVG(tbl.income) > 75000

LIMIT 10;



[6) Создание ролей пользователей user, guest](#_Toc91305840)

-- создание пользователя

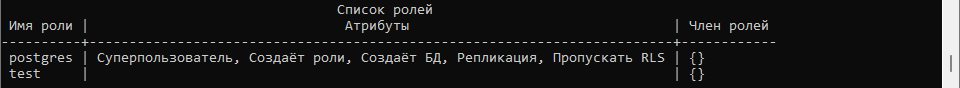
CREATE ROLE test WITH LOGIN PASSWORD 'passworduser';

-- выдаем привелегии

GRANT ALL ON DATABASE organisations TO test;

-- забираем привелегии

REVOKE ALL ON DATABASE organisations FROM test

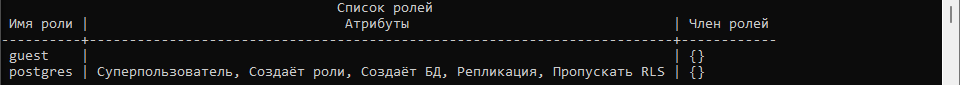


-- создание пользователя

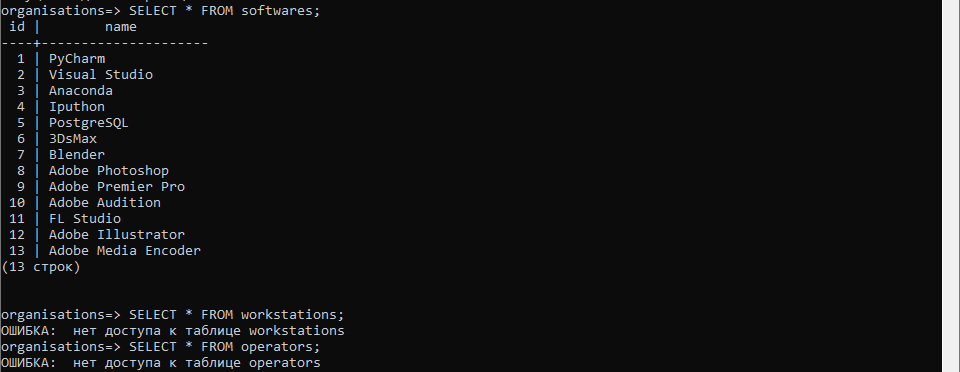
CREATE ROLE guest WITH LOGIN PASSWORD 'password';

-- выдаем привелегии

GRANT ALL ON softwares TO guest;



Проверка, может ли GUEST (гость) зайти в недоступные ему таблицы в базе данных:



[7) Создание двух транзакций с тремя точками восстановления на удаление, создание транзакции на вставку](#_Toc91305841)

BEGIN;

-- транзакция 1

UPDATE organizations SET income = income - 10000

    WHERE name = 'Euismod Corporation';

SAVEPOINT my\_savepoint;

UPDATE organizations SET income = income + 10000

    WHERE name = 'Euismod Urna Associates';

SAVEPOINT my\_savepoint;

-- транзакция 2

UPDATE organizations SET income = income - 10000

    WHERE name = 'Euismod Corporation';

SAVEPOINT my\_savepoint;

UPDATE organizations SET income = income + 10000

    WHERE name = 'Mattis Ornare LLP';

ROLLBACK TO my\_savepoint; -- откат на операцию 2.1 и отмена операции 2.2

-- корректировка транзакции 2

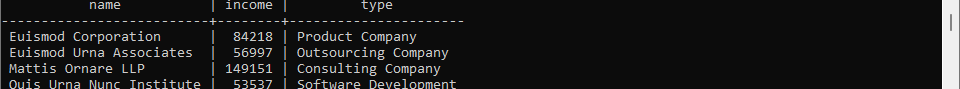
UPDATE organizations SET income = income + 10000

    WHERE name = 'Euismod Urna Associates';

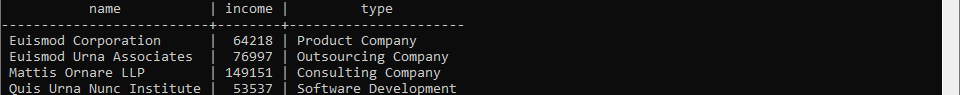
COMMIT;



ДО:



ПОСЛЕ:



[8) Создание двух представлений: изменяемое, неизменяемое](#_Toc91305842)

-- создадим изменяемое представление

CREATE OR REPLACE VIEW org\_view AS

    SELECT wo.name AS station\_name,

           so.name AS software\_name

    FROM workstations AS wo

    LEFT OUTER JOIN softwares AS so ON so.id=wo.software\_id

    WHERE so.name = 'PyCharm'

       OR so.name = 'Anaconda';



-- изменим представление

CREATE OR REPLACE VIEW org\_view AS

    SELECT wo.name AS station\_name,

           so.name AS software\_name

    FROM workstations AS wo

    LEFT OUTER JOIN softwares AS so ON so.id=wo.software\_id

    WHERE so.name = 'PostgreSQL'

       OR so.name = 'Visual Studio';



-- создадим неизменяемое представление

CREATE VIEW org\_view AS

    SELECT wo.name AS station\_name,

           so.name AS software\_name

    FROM workstations AS wo

    LEFT OUTER JOIN softwares AS so ON so.id=wo.software\_id

    WHERE so.name = 'PyCharm'

       OR so.name = 'Anaconda';



-- попробуем его изменить

CREATE VIEW org\_view AS

    SELECT wo.name AS station\_name,

           so.name AS software\_name

    FROM workstations AS wo

    LEFT OUTER JOIN softwares AS so ON so.id=wo.software\_id

    WHERE so.name = 'PostgreSQL'

       OR so.name = 'Visual Studio';



[9) Создание функции](#_Toc91305843)

CREATE FUNCTION mean(x float, y float) RETURNS INTEGER AS $$

     SELECT x / y;

$$ LANGUAGE SQL;

SELECT mean((SELECT SUM(income) FROM organizations),

            (SELECT COUNT(income) FROM organizations)) AS mean\_income;



[10) Создание триггера](#_Toc91305844)

-- создание функции

CREATE FUNCTION oper\_stamp() RETURNS trigger AS $oper\_stamp$

    BEGIN

        IF NEW.server\_id IS NULL THEN

            RAISE EXCEPTION 'server\_id cannot be null';

        END IF;

        IF NEW.name IS NULL THEN

            RAISE EXCEPTION '% name cannot be null', NEW.name;

        END IF;

        IF NEW.post IS NULL THEN

            RAISE EXCEPTION '% post cannot be null', NEW.post;

        END IF;

        RETURN NEW;

    END;

$oper\_stamp$ LANGUAGE plpgsql;

-- создание триггера

CREATE TRIGGER oper\_stamp BEFORE INSERT OR UPDATE ON operators

    FOR EACH ROW EXECUTE FUNCTION oper\_stamp();

-- проверка неправильными данными

INSERT INTO exams (server\_id, name, post)

VALUES

  (2, NULL, NULL);



[11). Создание процедуры](#_Toc91305845)

-- удалние через процедуру

CREATE PROCEDURE clean\_emp() AS '

    DELETE FROM organizations

        WHERE income < 75000;

' LANGUAGE SQL;

CALL clean\_emp();



[12) Шифрование столбцов](#_Toc91305846)

CREATE EXTENSION IF NOT EXISTS pgcrypto;

CREATE TABLE organisation\_table (

   id SERIAL PRIMARY KEY,

   name VARCHAR(150) NOT NULL,

   income VARCHAR(150) NOT NULL,

   type VARCHAR(150) NOT NULL

);

INSERT INTO organisation\_table (name,income,type)

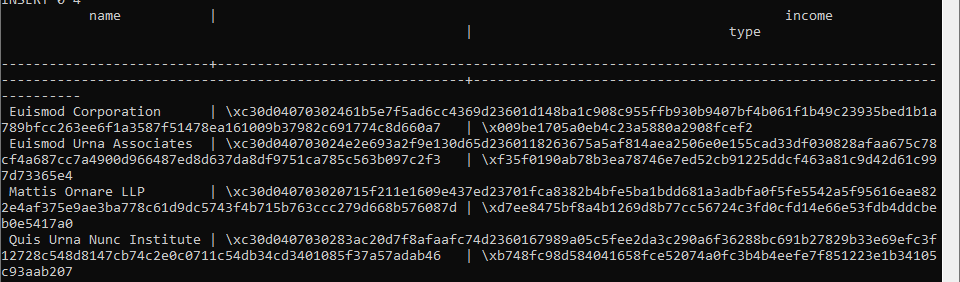
VALUES

  ('Euismod Corporation', pgp\_sym\_encrypt('84218','passwordpassword'), encrypt('Product Company','passwordpassword', 'aes')),

  ('Euismod Urna Associates', pgp\_sym\_encrypt('56997','passwordpassword'), encrypt('Outsourcing Company','passwordpassword', 'aes')),

  ('Mattis Ornare LLP', pgp\_sym\_encrypt('149151','passwordpassword'), encrypt('Consulting Company','passwordpassword', 'aes')),

  ('Quis Urna Nunc Institute', pgp\_sym\_encrypt('53537','passwordpassword'), encrypt('Software Development','passwordpassword', 'aes'));



13) Расшифровка столбцов

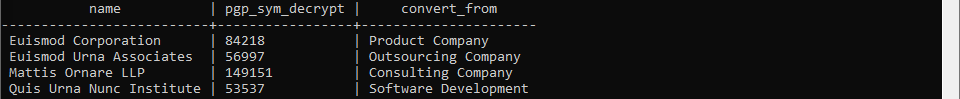
-- прочитаем расшифрованные пароли

SELECT  name,

        pgp\_sym\_decrypt(income::bytea, 'passwordpassword'),

        convert\_from(decrypt(type::bytea, 'passwordpassword', 'aes'), 'SQL\_ASCII')

FROM organisation\_table;



14) Создание объектных типов данных

CREATE TYPE classification AS ENUM ('license', 'trial version', 'pirated version');

CREATE TABLE classification\_table (

    id SERIAL,

    name TEXT,

    est classification

);

INSERT INTO classification\_table (id, name, est)

VALUES

    (1, 'PostgreSQL', 'license'),

    (2, 'Anaconda', 'trial version'),

    (3, '3DsMax', 'pirated version');

    (4, 'PyCharm', 'community version');



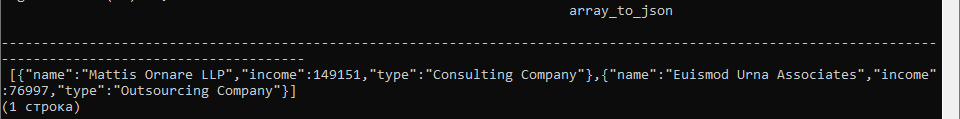
16) Запрос на перевод в формат JSON

SELECT array\_to\_json(array\_agg(row\_to\_json (fc))) FROM (

    SELECT \*

    FROM organizations

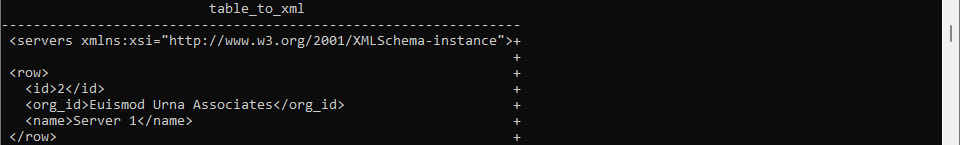
) fc;



17) Запрос на перевод в формат XML

-- таблица в XML

SELECT table\_to\_xml('servers', true, false, '');



-- query запрос в XML

SELECT query\_to\_xml('SELECT \* FROM servers', true, false, '');

