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 16»***

РАЗРАБОТКА БАЗЫ ДАННЫХ В СУБД POSTGRESQL

Выполнил студент группы 0024-01

Невзоров Л.Е.

Проверил преподаватель

Вертешев А.С.

Псков

2022

**Задание**

Тема: сдача студентами дипломных проектов.

Объекты: студенты, группы, дипломные проекты, руководители, члены ГЭК.

Для моделирования задачи необходимо хранить следующую информацию:

\* ФИО студента

\* N группы

\* N зачетной книжки

\* Тема дипломного проекта

\* Итоговая оценка за дипломный проект

\* ФИО руководителя дипломного проекта

\* Оценки всех членов ГЭК за дипломный проект

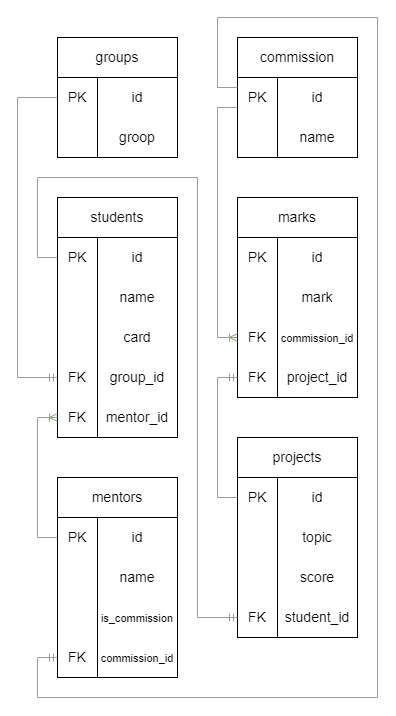
Дополнительные условия:

\* один студент защищает только одну дипломную работу;

\* каждую дипломную работу оценивает не менее 3 членов ГЭК;

\* член ГЭК может одновременно быть руководителем дипломного проекта.

**Схема**



**Создание таблиц**

CREATE TABLE "commission" (

  id SERIAL PRIMARY KEY,

  name VARCHAR(50) NOT NULL

);

INSERT INTO commission (name)

VALUES

  ('Jonah Osborn'),

  ('Piper Medina'),

  ('Hadley Colon'),

  ('Jeanette Roberson'),

  ('Darius Vazquez');

CREATE TABLE "groups" (

  id SERIAL PRIMARY KEY,

  groop VARCHAR(7) UNIQUE NOT NULL

);

INSERT INTO groups (groop)

VALUES

  ('0024-01'),

  ('0024-02'),

  ('0024-03'),

  ('0024-04');

CREATE TABLE "marks" (

  id SERIAL PRIMARY KEY,

  mark INTEGER NOT NULL,

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

  project\_id INTEGER NOT NULL REFERENCES projects(id) ON DELETE CASCADE

);

INSERT INTO marks (mark,commission\_id,project\_id)

VALUES

  (4,1,1),

  (5,2,1),

  (3,3,1),

  (5,4,2),

  (5,5,2);

CREATE TABLE "mentors" (

  id SERIAL PRIMARY KEY,

  name VARCHAR(50) NOT NULL,

  is\_commission VARCHAR(3) NOT NULL,

  commission\_id INTEGER DEFAULT NULL REFERENCES commission(id) ON DELETE CASCADE

);

INSERT INTO mentors (name,is\_commission,commission\_id)

VALUES

  ('Jonah Osborn','Yes',1),

  ('Miranda Bailey','No', NULL),

  ('Piper Medina','Yes',2),

  ('Emmanuel Guerrero','No', NULL),

  ('Hadley Colon','Yes',3);

CREATE TABLE "projects" (

  id SERIAL PRIMARY KEY,

  topic VARCHAR(120) NOT NULL,

  score INTEGER NOT NULL,

  student\_id INTEGER NOT NULL REFERENCES students(id) ON DELETE CASCADE

);

INSERT INTO projects (topic,score,student\_id)

VALUES

  ('Language representation',4,1),

  ('Evaluation vocabulary',5,2),

  ('Social language variation',4,3),

  ('Means of expressing modality',3,4),

  ('Metaphor as an object',2,5);

CREATE TABLE "students" (

  id SERIAL PRIMARY KEY,

  name VARCHAR(50) NOT NULL,

  card VARCHAR(5) UNIQUE NOT NULL,

  group\_id INTEGER NOT NULL REFERENCES groups(id) ON DELETE CASCADE,

  mentor\_id INTEGER NOT NULL REFERENCES mentors(id) ON DELETE CASCADE

);

INSERT INTO students (name,card,group\_id,mentor\_id)

VALUES

  ('Uma Goodman','98124',1,1),

  ('Signe Hopkins','55298',2,2),

  ('Zeph Castillo','89175',3,3),

  ('Kadeem Ferrell','47217',4,4),

  ('Chandler Galloway','21535',1,5);

**Запросы**

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

SELECT st.name,

       gr.groop,

       me.name

FROM mentors AS me

JOIN students AS st ON me.id=st.mentor\_id

JOIN groups AS gr ON gr.id=st.group\_id;



SELECT st.name,

       pr.topic,

       mk.mark AS ОценкаКоммиссии,

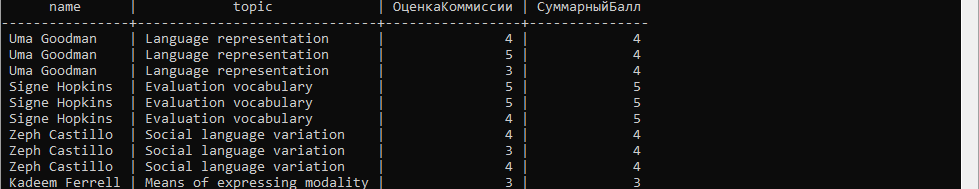
       pr.score AS СуммарныйБалл

FROM students AS st

JOIN projects AS pr ON st.id=pr.student\_id

JOIN marks AS mk ON pr.id=mk.project\_id

LIMIT 10;



SELECT pr.topic,

       mk.mark,

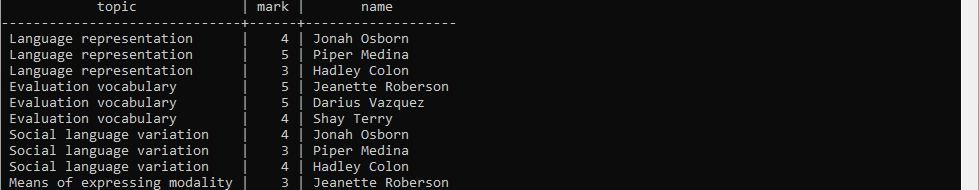
       co.name

FROM projects AS pr

JOIN marks AS mk ON pr.id=mk.project\_id

JOIN commission as co ON co.id=mk.commission\_id

LIMIT 10;



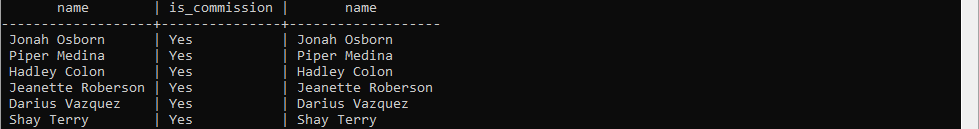
SELECT me.name,

       me.is\_commission,

       co.name

FROM mentors AS me

JOIN commission as co ON co.id=me.commission\_id;



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

SELECT name,

       card

FROM students

WHERE name LIKE '%S%'

UNION

SELECT name,

       card

FROM students

WHERE name LIKE '%M%';



SELECT topic,

       score

FROM projects

WHERE score <= 5

  AND score >= 3

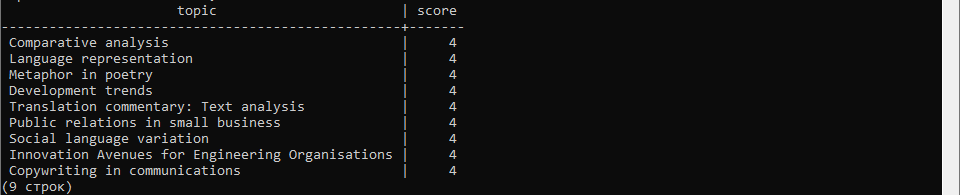
INTERSECT

SELECT topic,

       score

FROM projects

WHERE score = 4;



SELECT topic,

       score

FROM projects

EXCEPT

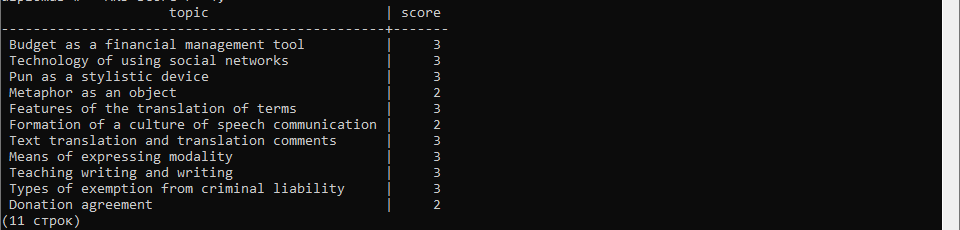
SELECT topic,

       score

FROM projects

WHERE score <= 5

  AND score >= 4;



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

SELECT st.name,

       gr.groop

FROM students AS st

LEFT OUTER JOIN groups AS gr ON gr.id=st.group\_id

LIMIT 10;



SELECT st.name,

       gr.groop

FROM students AS st

RIGHT OUTER JOIN groups AS gr ON gr.id=st.group\_id

LIMIT 10;



SELECT st.name,

       gr.groop

FROM students AS st

INNER JOIN groups AS gr ON gr.id=st.group\_id

LIMIT 10;



4) 2 Подзапроса: Простой, Сложный

SELECT st.name,

       st.card,

       gr.groop

FROM students AS st

RIGHT OUTER JOIN groups AS gr ON gr.id=st.group\_id

WHERE st.card LIKE '%3%' OR st.card LIKE '%6%'

LIMIT 10;



SELECT st.name,

       st.card,

       gr.groop,

       pr.score

FROM students AS st

RIGHT OUTER JOIN groups AS gr ON gr.id=st.group\_id

LEFT OUTER JOIN projects AS pr ON st.id=pr.student\_id

WHERE pr.score > (SELECT AVG(scores.score)

                  FROM (SELECT score

                        FROM projects

                        WHERE topic LIKE '%A%' OR topic LIKE '%a%') AS scores)

LIMIT 10;



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

SELECT st.name,

       st.card,

       pr.score

FROM students AS st

RIGHT OUTER JOIN groups AS gr ON gr.id=st.group\_id

LEFT OUTER JOIN projects AS pr ON st.id=pr.student\_id

WHERE pr.score > (SELECT AVG(scores.score)

                  FROM (SELECT score

                        FROM projects

                        WHERE topic LIKE '%B%') AS scores)

LIMIT 10;



SELECT tbl.name,

       tbl.card,

       tbl.score

FROM (SELECT \*

      FROM students AS st

      RIGHT OUTER JOIN groups AS gr ON gr.id=st.group\_id

      LEFT OUTER JOIN projects AS pr ON st.id=pr.student\_id) AS tbl

WHERE tbl.score > (SELECT AVG(scores.score)

                   FROM (SELECT score

                         FROM projects

                         WHERE topic LIKE '%B%') AS scores)

LIMIT 10;



SELECT st.name,

       AVG(mk.mark)

FROM students AS st

LEFT OUTER JOIN projects AS pr ON st.id=pr.student\_id

LEFT OUTER JOIN  marks AS mk ON pr.id=mk.project\_id

GROUP BY name

HAVING AVG(mk.mark) > (SELECT AVG(scores.score)

                       FROM (SELECT score

                             FROM projects

                             WHERE topic LIKE '%B%') AS scores);



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

DROP ROLE IF EXISTS test\_user;

-- создание пользователя

CREATE ROLE test\_user WITH LOGIN PASSWORD 'passworduser';

-- выдаем привелегии

GRANT ALL ON DATABASE diplomas TO test\_user;

-- просмотр зарегистрированных пользователей

\du

-- забираем привелегии

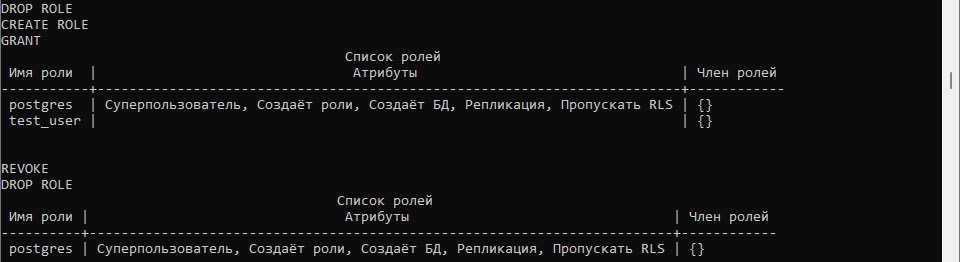
REVOKE ALL ON DATABASE diplomas FROM test\_user;

-- удаляем пользователя

DROP ROLE test\_user;

-- просмотр зарегистрированных пользователей

\du



DROP ROLE IF EXISTS test\_guest;

-- создание пользователя

CREATE ROLE test\_guest WITH LOGIN PASSWORD 'passwordguest';

-- выдаем привелегии

GRANT ALL ON students TO test\_guest;

-- просмотр зарегистрированных пользователей

\du

-- забираем привелегии

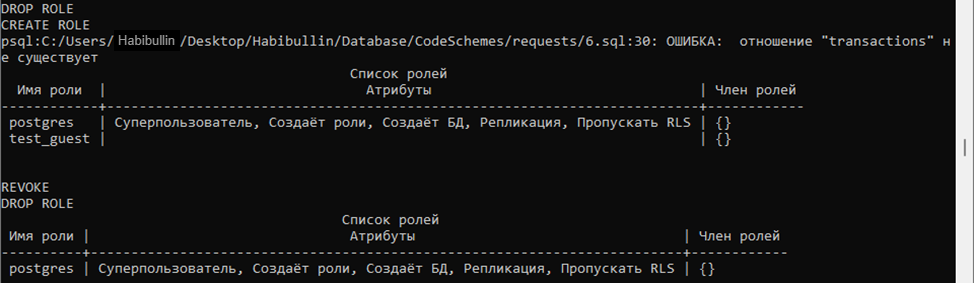
REVOKE ALL ON students FROM test\_guest;

-- удаляем пользователя

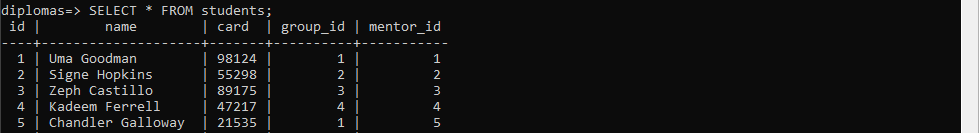
DROP ROLE test\_guest;

-- просмотр зарегистрированных пользователей

\du



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)

SELECT \*

FROM projects AS pr

ORDER BY pr.id ASC

LIMIT 3;

BEGIN;

-- транзакция 1

UPDATE projects SET score = score - 1

    WHERE topic = 'Evaluation vocabulary';

SAVEPOINT my\_savepoint;

UPDATE projects SET score = score + 1

    WHERE topic = 'Language representation';

SAVEPOINT my\_savepoint;

-- транзакция 2

UPDATE projects SET score = score - 1

    WHERE topic = 'Evaluation vocabulary';

SAVEPOINT my\_savepoint;

UPDATE projects SET score = score + 1

    WHERE topic = 'Social language variation';

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

-- корректировка транзакции 2

UPDATE projects SET score = score + 1

    WHERE topic = 'Language representation';

COMMIT;

SELECT \*

FROM projects AS pr

ORDER BY pr.id ASC

LIMIT 3;

До:



После:



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

-- создадим изменяемое представление

CREATE OR REPLACE VIEW groups\_view AS

    SELECT \*

    FROM groups

    WHERE groop = '0024-01'

       OR groop = '0024-02';

SELECT \*

FROM groups\_view;



-- изменим представление

CREATE OR REPLACE VIEW groups\_view AS

    SELECT \*

    FROM groups

    WHERE groop = '0024-03'

       OR groop = '0024-04';

SELECT \*

FROM groups\_view;



-- создадим неизменяемое представление

CREATE VIEW groups\_view AS

    SELECT \*

    FROM groups

    WHERE groop = '0024-01'

       OR groop = '0024-02'

    WITH CASCADED CHECK OPTION;

SELECT \*

FROM groups\_view;



-- попробуем его изменить

CREATE VIEW groups\_view AS

    SELECT \*

    FROM groups

    WHERE groop = '0024-03'

       OR groop = '0024-04'

    WITH CASCADED CHECK OPTION;

SELECT \*

FROM groups\_view;



[9) Создание функции](#_Toc91305843)

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

     SELECT x / y;

$$ LANGUAGE SQL;

SELECT mean((SELECT SUM(mark) FROM marks),

            (SELECT COUNT(mark) FROM marks)) AS answer;



[10) Создание триггера](#_Toc91305844)

-- создание функции

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

    BEGIN

        IF NEW.topic IS NULL THEN

            RAISE EXCEPTION 'topic cannot be null';

        END IF;

        IF NEW.score IS NULL THEN

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

        END IF;

        IF NEW.student\_id IS NULL THEN

            RAISE EXCEPTION '% student\_id cannot be null', NEW.student\_id;

        END IF;

        RETURN NEW;

    END;

$project\_stamp$ LANGUAGE plpgsql;

-- создание триггера

CREATE TRIGGER project\_stamp BEFORE INSERT OR UPDATE ON projects

    FOR EACH ROW EXECUTE FUNCTION project\_stamp();

-- проверка неправильными данными

INSERT INTO projects (topic, score, student\_id)

VALUES

  ('TEST VKR', 5, NULL);

DROP FUNCTION project\_stamp CASCADE;



[11). Создание процедуры](#_Toc91305845)

-- проектов ДО

SELECT COUNT(topic)

FROM projects;



-- проверка каскадного удаления

SELECT COUNT(mark)

FROM marks;



-- удалние через процедуру

CREATE PROCEDURE clean\_emp() AS '

    DELETE FROM projects

        WHERE score < 3;

' LANGUAGE SQL;

CALL clean\_emp();

DROP PROCEDURE clean\_emp;

-- проектов ПОСЛЕ

SELECT COUNT(topic)

FROM projects;



-- проверка каскадного удаления

SELECT COUNT(mark)

FROM marks;



[12) Шифрование столбцов](#_Toc91305846)

DROP EXTENSION pgcrypto;

DROP TABLE students\_passwords;

CREATE EXTENSION IF NOT EXISTS pgcrypto;

CREATE TABLE students\_passwords (

   id SERIAL PRIMARY KEY,

   name VARCHAR(50) NOT NULL,

   card VARCHAR(150) NOT NULL,

   group\_id INTEGER NOT NULL,

   mentor\_id VARCHAR(150) NOT NULL

);

INSERT INTO students\_passwords (name,card,group\_id,mentor\_id)

VALUES

  ('Uma Goodman',pgp\_sym\_encrypt('98124','mySmallSecret'), 1, encrypt('1','whoIsMentor', 'aes')),

  ('Signe Hopkins',pgp\_sym\_encrypt('55298','mySmallSecret'), 2, encrypt('2','whoIsMentor', 'aes')),

  ('Zeph Castillo',pgp\_sym\_encrypt('89175','mySmallSecret'), 3, encrypt('3','whoIsMentor', 'aes')),

  ('Kadeem Ferrell',pgp\_sym\_encrypt('47217','mySmallSecret'), 4, encrypt('4','whoIsMentor', 'aes')),

  ('Chandler Galloway',pgp\_sym\_encrypt('21535','mySmallSecret'), 1, encrypt('5','whoIsMentor', 'aes'));

-- прочитаем зашифрованные пароли

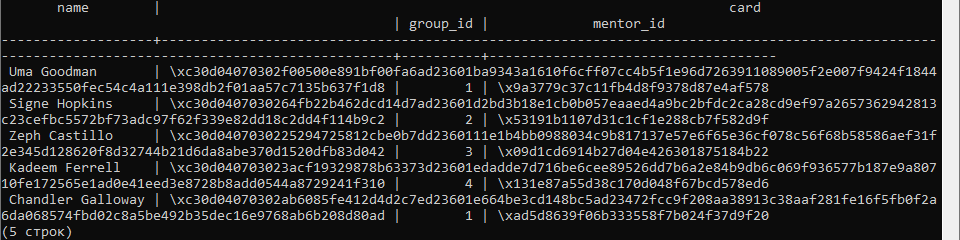
SELECT name,

       card,

       group\_id,

       mentor\_id

FROM students\_passwords;



13) Расшифровка столбцов

-- прочитаем расшифрованные пароли

SELECT  name,

        pgp\_sym\_decrypt(card::bytea, 'mySmallSecret'),

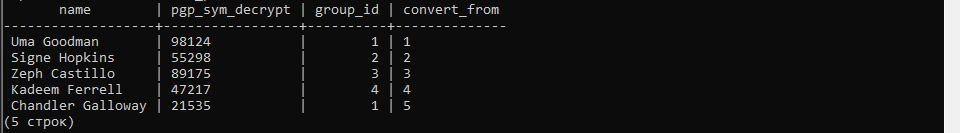
        group\_id,

        convert\_from(decrypt(mentor\_id::bytea, 'whoIsMentor', 'aes'), 'SQL\_ASCII')

FROM students\_passwords;

DROP EXTENSION pgcrypto;

DROP TABLE account\_passwords;



14) Создание объектных типов данных

CREATE TYPE graduate AS ENUM ('Bachelor', 'Master', 'Doctor');

CREATE TABLE graduate\_table (

    id serial,

    name text,

    grad graduate

);

INSERT INTO graduate\_table (id, name, grad)

VALUES

    (1, 'Truvor', 'Bachelor'),

    (2, 'Jon', 'Master'),

    (3, 'Emirald', 'Doctor');

    (4, 'Amigo', 'Another one guy with the red diploma');

SELECT \*

FROM graduate\_table;





16) Запрос на перевод в формат JSON

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

    SELECT st.name,

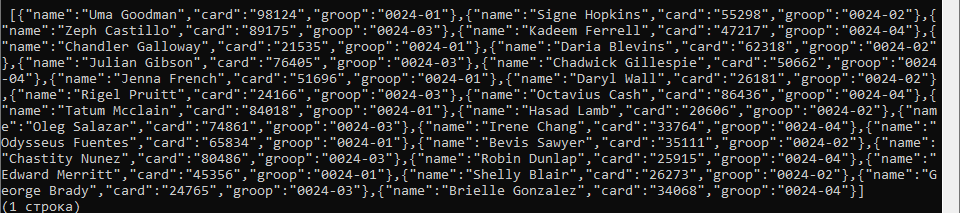
           st.card,

           gr.groop

    FROM students AS st

    RIGHT OUTER JOIN groups AS gr ON gr.id=st.group\_id

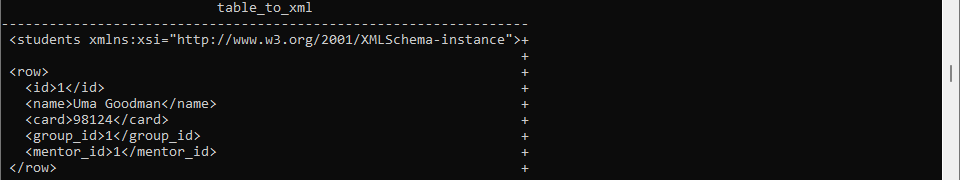
) st;



17) Запрос на перевод в формат XML

-- таблица в XML

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



-- query запрос в XML

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

