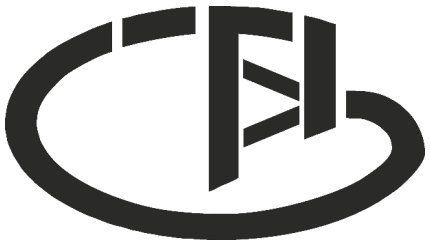

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНЖЕНЕРНЫХ ТЕХНОЛОГИЙ

БЕЗОПАСНОСТЬ СИСТЕМ БАЗ ДАННЫХ

Отчет

по практической работе

«ВВЕДЕНИЕ В ЯЗЫК SQL»

Отчет подготовил

студент 3 курса группы УБ-01

Хомутов Константин

Воронеж 2023

1. Средства определения данных языка SQL

1.1 Оператор CREATE TABLE

Основная функция этого оператора — создание новой таблицы и описание ее столбцов и типов данных. Также, оператор позволяет определять первичные ключи, внешние ключи и альтернативные ключи.

```
itcourses=# create table course (
itcourses(# course_id int NOT NULL,
itcourses(# course_name varchar(50) NOT NULL,
itcourses(# constraint coursePk
itcourses(# primary key (course_id));
CREATE TABLE
itcourses=#
```

course_id принадлежит к типу данных integer и имеет свойство primary key. Столбец course_name имеет тип данных varchar(50) и является строкой ограниченной переменной длины.

1.2 Оператор ALTER TABLE

Этим оператором можно изменять содержимое таблицы после ее создания.

```
courses2=# ALTER TABLE course ADD principal_name varchar(50) NOT NULL;
ALTER TABLE
courses2=# \d course
                          ╥рсышЎр "public.course"
    ╥Єюысхў     |          ╥шя          | ╧Ёртшыю ёюЁЄшЁютъш | ─юяєёЄшьюёЄ№ NULL | ╧ю єьюыўрэш■
----------------+------------------------+--------------------+-------------------+-------------
 course_id      | integer                |                    | not null          |
 course_name    | character varying(50) |                    | not null          |
 principal_name | character varying(50) |                    | not null          |
╚эфхъёў:
    "coursepk" PRIMARY KEY, btree (course_id)
```

1.3 Операторы DROP TABLE и DROP

Оператор DROP TABLE позволяет удалить таблицу, но при этом нужно следить, чтобы не нарушалась ссылочная целостность.

```
postgres=# drop table course2
postgres-# ;
DROP TABLE
postgres=#
```

Оператор DROP используется для удаления. С его помощью можно удалить базу данных (DROP TABLE;) или использовать ее в сочетании с командой ALTER TABLE для удаления определенного столбца (ALTER TABLE *название таблицы* DROP COLUMN *название столбца*;)

3.2 Средства запроса данных языка SQL

3.2.1 Чтение заданных столбцов из одиночной таблицы

Следующий запрос производит выборку (чтение) трех из пяти столбцов таблицы course:

SELECT course_name, principal_name FROM course;

```
courses2=# SELECT course_name, principal_name FROM course;
      course_name       |        principal_name
------------------------+-----------------------------
 программирование       | Карпов Марк Русланович
 устройство компьютера  | Морозов Эрик Владимирович
 операционные системы   | Марков Александр Миронович
(3 строки)
```

Порядок столбцов в результирующей таблице определяется порядком следования их имен после ключевого слова SELECT.

SELECT principal_name, course_name FROM course;

```
courses2=# SELECT principal_name, course_name FROM course;
       principal_name       |      course_name
----------------------------+------------------------
 Карпов Марк Русланович     | программирование
 Морозов Эрик Владимирович  | устройство компьютера
 Марков Александр Миронович | операционные системы
(3 строки)
```

Если нужно при выводе, чтобы СУБД удаляла повторяющиеся строки, при записи необходимо использовать ключевое слово DISTINCT. Применение этого слова выглядело бы следующим образом:

SELECT DISTINCT course_name FROM course;

3.2.2 Чтение заданных строк из одиночной таблицы

Выбор столбцов определенных строк

SELECT course_id FROM course WHERE course_name = ‘программирование’;

```
courses2=# SELECT course_id FROM course WHERE course_name = 'программирование';
 course_id
-----------
       101
(1 строка)
```

3.2.3 Чтение заданных строк и столбцов из одиночной таблицы

Из таблицы можно выбирать определенные столбцы и строки. Следующая команда выводит идентификационный номер курса, его название и имя декана, но только те, идентификационный номер которых больше 101.

SELECT course_id, course_name, principal_name FROM course WHERE course_id > 101;

```
courses2=# SELECT course_id, course_name, principal_name FROM course WHERE course_id > 101
courses2-# ;
 course_id |      course_name      |       principal_name
-----------+-----------------------+----------------------------
       102 | устройство компьюетра | Морозов Эрик Владимирович
       103 | операционные системы  | Марков Александр Миронович
(2 строки)
```

3.2.4 Диапазоны, специальные символы и пустые значения в предложениях WHERE

Для создания диапазонов используется слово BETWEEN. Например, запрос

SELECT course_id, course_name FROM course WHERE course_id BETWEEN 101 AND 103;

```
courses2=# SELECT course_id, course_name FROM course WHERE course_id BETWEEN 101 AND 103;
 course_id |      course_name
-----------+-----------------------
       101 | программирование
       102 | устройство компьюетра
       103 | операционные системы
(3 строки)
```

Оператор BETWEEN использует нестрогие границы <=, >=. Если границы должны быть строгими, то лучше записать следующим образом:

SELECT course_id, course_name FROM course WHERE course_id > 101 AND course_id < 103;

```
courses2=# SELECT course_id, course_name FROM course WHERE course_id > 101 AND course_id < 103;
 course_id |      course_name
-----------+----------------------
       102 | устройство компьютера
(1 строка)
```

3.2.5 Сортировка результатов

Порядок строк в таблице, возвращаемой оператором SELECT, является произвольным. Если нужно сортировать строки результата, это можно сделать с помощью конструкции ORDER BY. Например, следующий запрос возвращает названия курсов, отсортированных в алфавитном порядке:

SELECT course_id, course_name FROM course ORDER BY course_name;

```
courses2=# SELECT course_name FROM course ORDER BY course_name;
      course_name
-----------------------
 операционные системы
 программирование
 устройство компьютера
(3 строки)
```

По умолчанию, сортировка идет по возрастанию, чтобы это поменять, можно использовать ключевые слова ASC (по возрастанию) и DESC (по убыванию).

SELECT course_name FROM course ORDER BY course_name DESC;

Результат противоположен предыдущему:

```
courses2=# SELECT course_name FROM course ORDER BY course_name DESC;
      course_name
-----------------------
 устройство компьютера
 программирование
 операционные системы
(3 строки)
```

Сортировка может происходить по двум столбцам, причем столбцы с одинаковыми значениями могут сортироваться в обратном

SELECT course_id, course_name, principal_name FROM course ORDER BY principal_name DESC, course_name ASC;

```
 course_id |       course_name        |       principal_name
-----------+--------------------------+----------------------------
       102 | устройство компьютера    | Морозов Эрик Владимирович
       104 | администрирование сетей  | Марков Александр Миронович
       103 | операционные системы     | Марков Александр Миронович
       101 | программирование         | Карпов Марк Русланович
(4 строки)
```

Здесь, сортировка сначала идет по убыванию по именам управляющих, но если декан управляет двумя курсами, то сортировка происходит по возрастанию по названию курса.

3.2.6 Агрегатные состояния SQL

В SQL имеется пять агрегатных функций: COUNT, SUM, AVG, MAX, MIN. Они выполняют различные действия над результатами оператора SELECT. Функция COUNT работает вне зависимости от типа данных столбца, а функции SUM, AVG, MAX, MIN оперируют только с числовыми столбцами (integer, numeric и т.д.).

Функция COUNT подсчитывает количество строк в результате, а функция SUM вычисляет сумму значений числового столбца. Например, запрос

SELECT COUNT (*) FROM course;

Даст следующий результат:

```
courses2=# SELECT COUNT (*) FROM course;
 count
-------
     4
(1 строка)
```

3.2.7 Агрегатные функции и группировка

Агрегатные функции можно применять к группам строк данных. Например, запрос

SELECT principal_name, COUNT(*) FROM course GROUP BY principal_name;

Выводит на экран количество курсов с разными заведующими:

```
      principal_name       | count
---------------------------+-------
 Морозов Эрик Владимирович |     1
 Марков Александр Миронович |     2
 Карпов Марк Русланович    |     1
(3 строки)
```

К группам можно применять различные условия. Например, выборке подлежат только те экземпляры, идентификатор которых больше 102

SELECT principal_name, COUNT(*) FROM course WHERE course_id > 102 GROUP BY principal_name;

```
      principal_name       | count
---------------------------+-------
 Марков Александр Миронович |     2
(1 строка)
```

3.2.8 Оконные функции

При использовании группировки результирующая таблица содержит по одной строке для каждого группируемого значения, а оконная функция не свертывает результаты, принадлежащие одной группе.

SELECT name, course, COUNT(*) FROM student GROUP BY name, course;

```
courses2=# SELECT name, course, COUNT(*) FROM student GROUP BY name, course;
           name             | course | count
----------------------------+--------+-------
 Кузьмина Ксения Григорьевна |    101 |     1
 Лебедев Фёдор Дмитриевич    |    103 |     1
 Смирнова Анна Серафимовна   |    101 |     1
 Петров Платон Богданович    |    104 |     1
 Прохоров Тимур Эминович     |    102 |     1
(5 строк)
```

Следующий запрос содержит оператор PARTITION BY для реализации оконной функции и имеет вид:

SELECT name, passport, COUNT(*) OVER (PARTITION BY course) FROM student ORDER BY passport;

Данный запрос выводит имена студентов и общее число студентов, которые учатся с ними на одном курсе.

```
courses2=# SELECT name, passport, COUNT(*) OVER (PARTITION BY course) FROM student ORDER BY passport;
           name            |  passport  | count
---------------------------+------------+-------
 Смирнова Анна Серафимовна | 2022110483 |     2
 Петров Платон Богданович  | 2022183922 |     1
 Кузьмина Ксения Григорьевна | 2022420416 |     2
 Прохоров Тимур Эминович   | 2022592815 |     1
 Лебедев Фёдор Дмитриевич  | 2022997893 |     1
(5 строк)
```

3.2.9 Чтение данных из нескольких таблиц с применением вложенных запросов

Допустим, мы хотим вывести имена студентов, которые учатся на курсе «программирование». Для этого нам нужно ввести тако1 запрос:

SELECT name FROM student WHERE course == ‘программирование’.

Но тут возникает проблема - поле course имеет целочисленный тип, а не символьный, и ссылается на идентификатор курса. Используя другую команду мы можем вызвать идентификатор курса, который имеет название «программирование».

SELECT course_id FROM course WHERE course_name = ‘программирование’;

Соединив эти две команды, мы получаем запрос, который выводит имена студентов, которые учатся на курсе «программирование»:

SELECT name FROM student WHERE ((SELECT course_id FROM course WHERE course_name = ‘программирование’) = course);

```
courses2=# SELECT name FROM student WHERE ((S
e);
           name
---------------------------
 Смирнова Анна Серафимовна
 Кузьмина Ксения Григорьевна
(2 строки)
```

3.2.10 Чтение данных из нескольких таблиц с помощью операции соединения

Операция соединения JOIN позволяет извлечь данные из двух и более таблиц, когда это не удается сделать при помощи вложенных запросов. Основной принцип — создать новое отношение, связав между собой содержимое двух иди более исходных отношений. Например:

SELECT name, course_name FROM student, course WHERE course = course_id;

```
courses2=# SELECT name, course_name FROM student, course WHERE c
            name              |       course_name
------------------------------+-------------------------
 Смирнова Анна Серафимовна    | программирование
 Прохоров Тимур Эминович      | устройство компьютера
 Лебедев Фёдор Дмитриевич     | операционные системы
 Петров Платон Богданович     | администрирование сетей
 Кузьмина Ксения Григорьевна  | программирование
(5 строк)
```

Такой запрос говорит СУБД найти в двух таблицах одинаковое поле, которое ссылается на экземпляр курса, и, установив между ними связь, вывести столбцы, которые мы указали при помощи команды SELECT.

При этом имеется другой способ соединения, при котором используются ключевые слова JOIN и ON

SELECT name, course FROM student JOIN course ON course = course_id;

```
courses2=# SELECT name, course_name FROM student JOIN cours
id;
            name              |       course_name
------------------------------+-------------------------
 Смирнова Анна Серафимовна    | программирование
 Прохоров Тимур Эминович      | устройство компьютера
 Лебедев Фёдор Дмитриевич     | операционные системы
 Петров Платон Богданович     | администрирование сетей
 Кузьмина Ксения Григорьевна  | программирование
(5 rows)
```

Команда выдает такой же результат, однако запрос намного проще читается.

Соединение трех таблиц выглядело бы следующим образом:

SELECT course_name, teacher_name, name FROM course JOIN student ON course_id = course JOIN teacher ON course_id = course_id;

course_name	teacher_name	name
администрирование сетей	Nazarov Gennadiy Gordeevich	Петров Платон Богданович
устройство компьютера	Petrov Aristarch Christophovich	Прохоров Тимур Эминович
администрирование сетей	Ponomareva Nina Mironovna	Петров Платон Богданович
устройство компьютера	Nazarov Gennadiy Gordeevich	Прохоров Тимур Эминович
программирование	Ponomareva Nina Mironovna	Смирнова Анна Серафимовна
программирование	Voroncova Elliana Denisovna	Кузьмина Ксения Григорьевна
программирование	Petrov Aristarch Christophovich	Кузьмина Ксения Григорьевна
программирование	Nazarov Gennadiy Gordeevich	Кузьмина Ксения Григорьевна

3.2.11 Внешние соединения

Смысл оператора в том, чтобы соединять строки таблиц студент и курс, даже если среди студентов есть те, которые не учатся ни на одном курсе. Внешне это выглядело бы как имя студента, напротив которого не стоит название курса.

SELECT name, course_name FROM student LEFT JOIN course ON course = course_id;

```
courses2=# SELECT name, course_name FROM student LEFT JO
            name             |       course_name
-----------------------------+-------------------------
 Смирнова Анна Серафимовна   | программирование
 Прохоров Тимур Эминович     | устройство компьютера
 Лебедев Фёдор Дмитриевич    | операционные системы
 Петров Платон Богданович    | администрирование сетей
 Кузьмина Ксения Григорьевна | программирование
(5 rows)
```

Также имеется RIGHT JOIN (включаются все строки из таблицы, находящейся справа от ключевого слова JOIN) и FULL JOIN для полного внешнего соединения.

3.2.12 Объединения

Если возникает ситуация, когда необходимо объединить результаты двух и более запросов в единую таблицу, состоящую из схожих строк, то можно воспользоваться оператором UNION.

В следующем запросе происходит объединение таблиц имен студентов преподавателей, которые связаны между между собой кусом.

SELECT name FROM student WHERE course = 101

UNION

SELECT teacher_name FROM teacher WHERE course = 101;

```
courses2=# SELECT name FROM student WHERE course = 101
courses2-# UNION
courses2-# SELECT teacher_name FROM teacher WHERE course =  101;
              name
--------------------------------
 Petrov Aristarch Christophovich
 Кузьмина Ксения Григорьевна
 Смирнова Анна Серафимовна
(3 rows)
```

3. Средства модификации данных языка SQL

Под модификацией данных подразумевается вставка, удаление, изменение.

3.3.1 Вставка данных, оператор INSERT

Данный оператор имеет две формы записи в зависимости от того, заполняются все поля или нет

INSERT INTO course VALUES

(‘программирование’);

Строковые значения обязательно должны вводиться в кавычках, в то время как целочисленные и порядковые значения вводятся без кавычек.

Следующий запрос представляет собой пример ввода команды INSERT, где не все поля заполняются. Так, например, выглядел бы ввод значений в таблицу студент:

INSERT INTO student (name, course, passport) VALUES

(‘Кузьмина Ксения Григорьевна’, „программирование“, 2022543654);

Для столбцов, для которых значение не было установлено, будут установлено значение по умолчанию.

Комбинируя операторы INSERT и SELECT можно скопировать дынные из одной таблицы в другую.

INSERT INTO student (name, course, passport) VALUES

SELECT name, course, passport FROM student;

3.3.2 Изменение данных, оператор UPDATE

Значение существующих данных могут быть изменены с помощью SQL-оператора UPDATE. Следующий пример устанавливает курс студента Кузьмина Ксения Григорьевна с «программирование» (101) на «администрирование сетей» (104)

UPDATE student SET course = 104 WHERE student_id = 205;

```
courses2=# UPDATE student SET course = 104 WHERE student_id = 205;
UPDATE 1
courses2=# SELECT name, course FROM student;
            name             | course
-----------------------------+--------
 Смирнова Анна Серафимовна   |    101
 Прохоров Тимур Эминович     |    102
 Лебедев Фёдор Дмитриевич    |    103
 Петров Платон Богданович    |    104
 Кузьмина Ксения Григорьевна |    104
(5 строк)
```

При этом нельзя забывать оператор WHERE, иначе все созданные экземпляры сущности студент поменяют свой курс на «администрирование сетей» (104).

3.3.3 Удаление данных, оператор DELETE

Для удаления строк служит SQL-оператор DELETE. Следующий оператор удаляет из таблицы student строку, в которой столбец student_id равен 101:

DELETE FROM student WHERE student_id = 101;

Ограничения такие же, как и на оператор UPDATE. Если забыть WHERE, то изменения затронут все экземпляры.

3.4 Транзакции

Транзакция — механизм СУБД, обеспечивающий непротиворечивость данных. Явное использование механизма транзакций при модификации данных

гарантирует, что все операции будут произведены успешно, а в случае возникновения ошибки будет вызван процесс отката транзакции (rollback) и состояние базы данных вернется к первоначальному (состояние до начала выполнения транзакции)

```
BEGIN TRANSACTION;
DELETE FROM student;
DELETE FROM course;
ROLLBACK TRANSACTION;
SELECT * FROM student;
SELECT * FROM course;
```

```
courses2=# BEGIN TRANSACTION;
BEGIN
courses2=*# DELETE FROM student;
DELETE 5
courses2=*# DELETE FROM course;
DELETE 4
courses2=*# ROLLBACK TRANSACTION;
ROLLBACK
courses2=# SELECT * FROM student;
 student_id |            name             |        adress         |  passport  | course
------------+-----------------------------+-----------------------+------------+--------
        201 | Смирнова Анна Серафимовна   | пер. Ладыгина, 30     | 2022110483 |    101
        202 | Прохоров Тимур Эминович     | проезд 1905 года, 75  | 2022592815 |    102
        203 | Лебедев Фёдор Дмитриевич    | спуск Гоголя, 60      | 2022997893 |    103
        204 | Петров Платон Богданович    | ул. Будапештсткая, 86 | 2022183922 |    104
        205 | Кузьмина Ксения Григорьевна | наб. Ленина, 27       | 2022420416 |    104
(5 строк)


courses2=# SELECT * FROM course;
 course_id |       course_name       |       principal_name
-----------+-------------------------+----------------------------
       101 | программирование        | Карпов Марк Русланович
       102 | устройство компьюетра   | Морозов Эрик Владимирович
       103 | операционные системы    | Марков Александр Миронович
       104 | администрирование сетей | Марков Александр Миронович
(4 строки)
```

Таблицы не были удалены, так как над оператор DELETE был произведен откат оператором ROLLBACK TRANSACTION.