Основные понятия языка SQL. Синтаксис операторов

SQL (*structured query language)* — «язык структурированных запросов»

План

1. Определение языка SQL
2. Составные части
3. DML
4. DDL
5. DCL
6. Операции сравнения

Определение

- Язык SQL -язык который предназначен для манипулирования данными в реляционных базах данных, определения структуры баз данных и для управления правами доступа к данным в многопользовательской среде.

Составные части SQL

Язык манипулирования данными (Data Manipulation Language, DML)

Язык определения данных (Data Definition Language, DDL)

Язык Управления данными (Data Control Language, DCL)

Язык манипулирования данными (Data Manipulation Language, DML)

для манипулирования данными в таблицах баз данных

SELECT (выбрать)

INSERT (вставить)

UPDATE (обновить)

DELETE (удалить)

MERGE (слияние данных)

Язык определения данных (Data Definition Language, DDL)

для создания и изменения структуры базы данных и ее составных частей - таблиц, индексов, представлений (виртуальных таблиц), а также триггеров и сохраненных процедур.

CREATE DATABASE	(создать базу данных)
CREATE TABLE	(создать таблицу)
CREATE VIEW	(создать виртуальную таблицу)
CREATE INDEX	(создать индекс)
CREATE TRIGGER	(создать триггер)
CREATE PROCEDURE	(создать сохраненную процедуру)
ALTER DATABASE	(модифицировать базу данных)
ALTER TABLE	(модифицировать таблицу)
ALTER VIEW	(модифицировать виртуальную таблицу)
ALTER INDEX	(модифицировать индекс)
ALTER TRIGGER	(модифицировать триггер)
ALTER PROCEDURE	(модифицировать сохраненную процедуру)
DROP DATABASE	(удалить базу данных)
DROP TABLE	(удалить таблицу)
DROP VIEW	(удалить виртуальную таблицу)
DROP INDEX	(удалить индекс)
DROP TRIGGER	(удалить триггер)
DROP PROCEDURE	(удалить сохраненную процедуру)

Язык Управления данными (Data Control Language, DCL)

для управления правами доступа к данным и выполнением процедур в многопользовательской среде

GRANT (дать права)

REVOKE (забрать права)

SET ROLE (разрешает или запрещает роли)

Синтаксис SELECT

- SELECT [ALL | DISTINCT]
  <список_выбора>
  FROM <имя_таблицы>, ...
  [WHERE <условие>]
  [GROUP BY<имя_столбца>,...]
  [HAVING <условие>]
  [ORDER BY <имя_столбца> [ASC | DESC],...]

Пример Выборка данных SELECT (выбрать)

получить список всех авторов

SELECT author FROM authors;

получить список всех полей таблицы authors:

SELECT * FROM authors;

найдем все книги, опубликованные после 1996 года

SELECT title FROM titles WHERE yearpub > 1996;

найти все публикации за интервал 1995 - 1997 гг.:

SELECT title FROM titles WHERE yearpub>=1995 AND yearpub<=1997;

Другой вариант с использованием логической операции проверки на вхождение в интервал:

SELECT title FROM titles WHERE yearpub BETWEEN 1995 AND 1997;

При использовании конструкции NOT BETWEEN находятся все строки, не входящие в указанный диапазон.

ИЛИ

SELECT title FROM titles WHERE yearpub IN (1995,1996,1997);

Синтаксис INSERT

INSERT INTO <имя_таблицы> [
(<имя_столбца>,<имя_столбца>,...)]
VALUES (<значение>,<значение>,..)

Пример Вставка данных INSERT (вставить)

Вставка с указанием списка столбцов:

```
INSERT INTO publishers (publisher,pub_id)
    VALUES ("Super Computer Publishing",17);
```

Синтаксис UPDATE

UPDATE <имя_таблицы> SET <имя_столбца>=<значение>,... [WHERE <условие>]

Пример Обновления данных
UPDATE (обновить)

*в таблице **publishers** все неопределенные значения столбца **url** и заменяет их строкой "**url not defined**“*

UPDATE publishers SET url="url not defined"
WHERE url IS NULL;

idStudent	surname	name	stipend	kurs	city
1	Иванов	петя	150	1	Орел
3	Петров	Петр	200	3	Курск
6	Сидоров	Вадим	150	4	Москва

```
update Student
set name = "ПЁТР"
where idStudent =1
```

idStudent	surname	name	stiper
1	Иванов	ПЁТР	150
3	Петров	Петр	200
6	Сидоров	Вадим	150

Синтаксис DELETE

DELETE FROM <имя_таблицы> [WHERE <условие>]

Пример Удаления данных

удаляет запись об издательстве Super Computer Publishing

```
DELETE FROM publishers WHERE publisher =
"Super Computer Publishing";
```

Базовые операции реляционных баз данных

- **выборка**(Restriction)
- **проекция**(Projection)
- **соединение**(Join)
- **объединение**(Union)

Комментарии в языке SQL

-- однострочный комментарий

/ многострочный*
*комментарий */*

Создание виртуальной таблицы (представления)

CREATE VIEW <имя_представления>
[<имя_столбца>,...] AS <запрос>

Язык Управления данными (Data Control Language, DCL)

позволяют управлять доступом к информации, находящейся внутри *базы данных*.

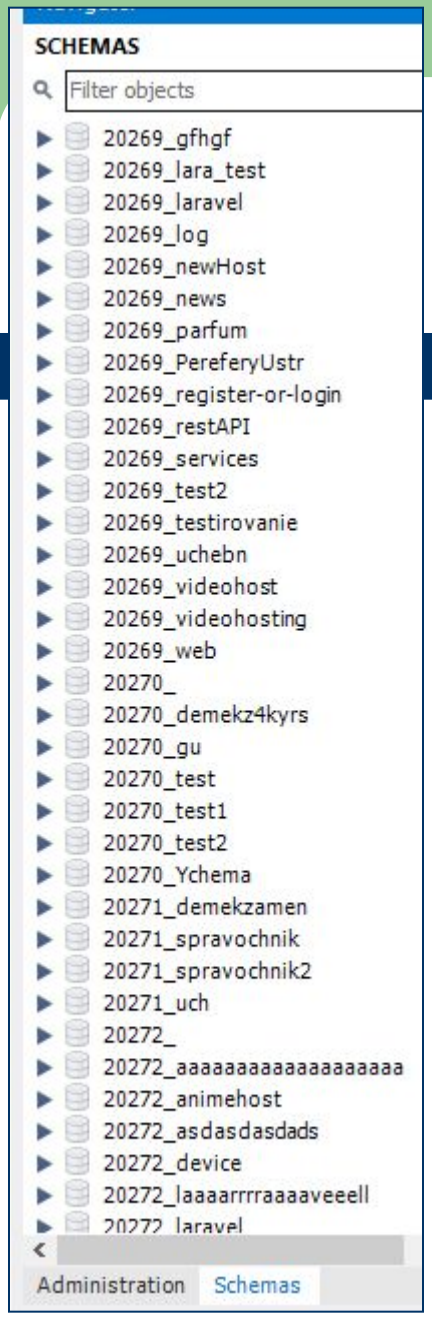
GRANT применяется для присвоения привилегии;

REVOKE применяется для отмены привилегии;

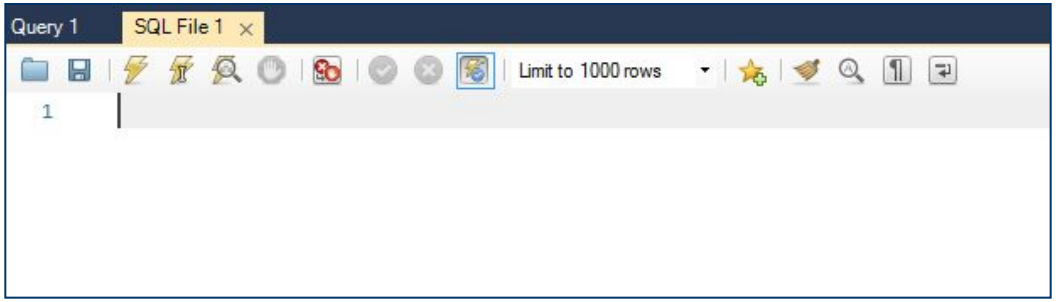
SET ROLE разрешает или запрещает роли для текущего сеанса

Операции сравнения

Обозначение	Операции
>, (больше) <,(меньше) >=(больше чем или равно) <=,(меньше чем или равно) =, <>,(сравнение на неравенство) !<, (не меньше чем) !>(не больше чем)	Операции сравнения
Null: IS NULL, IS NOT NULL	Проверка поля на значение
BETWEEN, NOT BETWEEN	Проверка на вхождение в диапазон
IN, NOT IN	Проверка вхождения в список
LIKE, NOT LIKE	Проверка вхождения подстроки
AND,OR,NOT	Операции соединения связями


SCHEMAS
Filter objects
20269_gfhgf
20269_lara_test
20269_laravel
20269_log
20269_newHost
20269_news
20269_parfum
20269_PereferyUstr
20269_register-or-login
20269_restAPI
20269_services
20269_test2
20269_testirovanie
20269_uchebn
20269_videohost
20269_videohosting
20269_web
20270_
20270_demekz4kyrs
20270_gu
20270_test
20270_test1
20270_test2
20270_Ychema
20271_demekzamen
20271_spravochnik
20271_spravochnik2
20271_uch
20272_
20272_aaaaaaaaaaaaaaaaaa
20272_animehost
20272_asdasdasdads
20272_device
20272_laaaarrrraaaaveeell
20272_laravel
Administration
Schemas


SQL


Query 1
SQL File 1
Limit to 1000 rows
1

Использование кавычек в запросах

- `текст` - для создания названий таблиц и атрибутов

- “текст”
- ‘текст’

– для создание значений атрибутов

Создание БД

Create database `название`

*название (на сервере wed.edu)
базы данных начинается с логина, далее пишется название,
например, 232300_sklad

Создание таблицы

```
1  create table
2  `user` (
3  `id_user` int not null auto_increment,
4  `fio` char (30) not null,
5  `adress` char(30) not null,
6  primary key (`id_user`)
7  )
8
```

Создание таблицы

Название таблицы

Создание поля id

Создание поля fio

Создание поля adress

Создание первичного Ключа поля id

Вставка данных

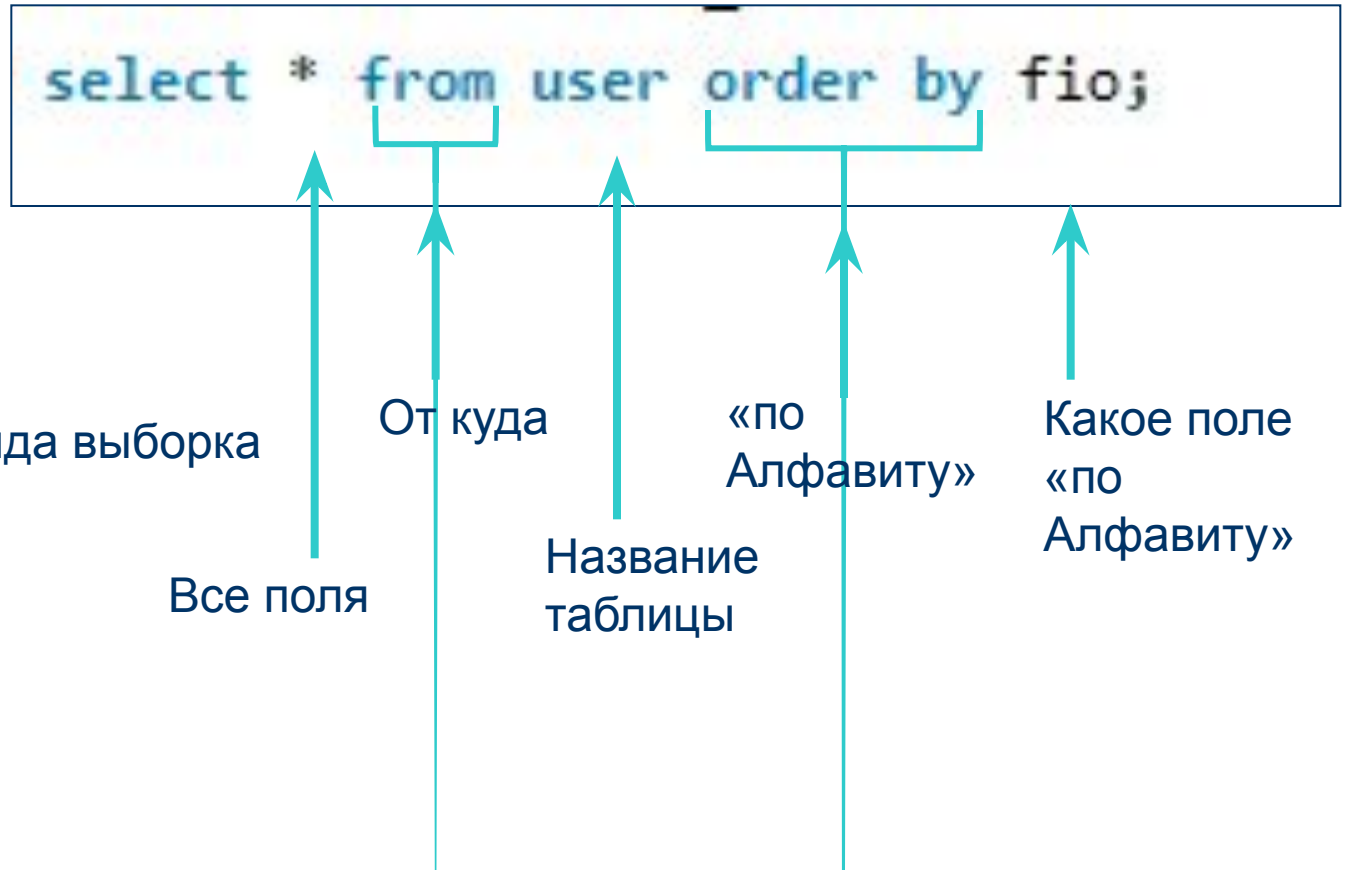
```
insert into
    user(fio,adress)
    values("Иванов Иван Инванович", "Иркутск ул Ленина 5 а");
```

Вставка данных

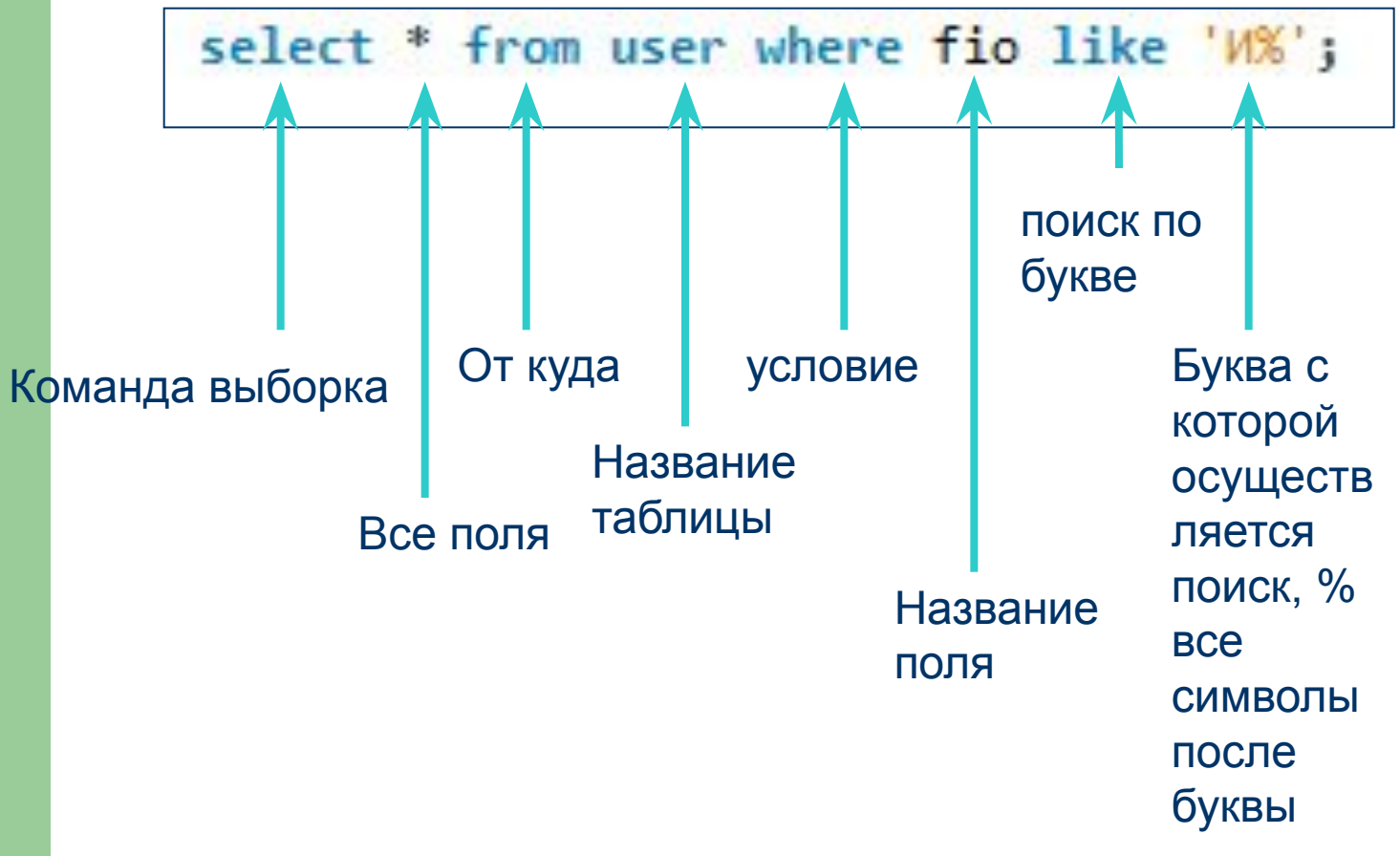
Название таблицы и ее полей

Перечисление значений для каждого поля

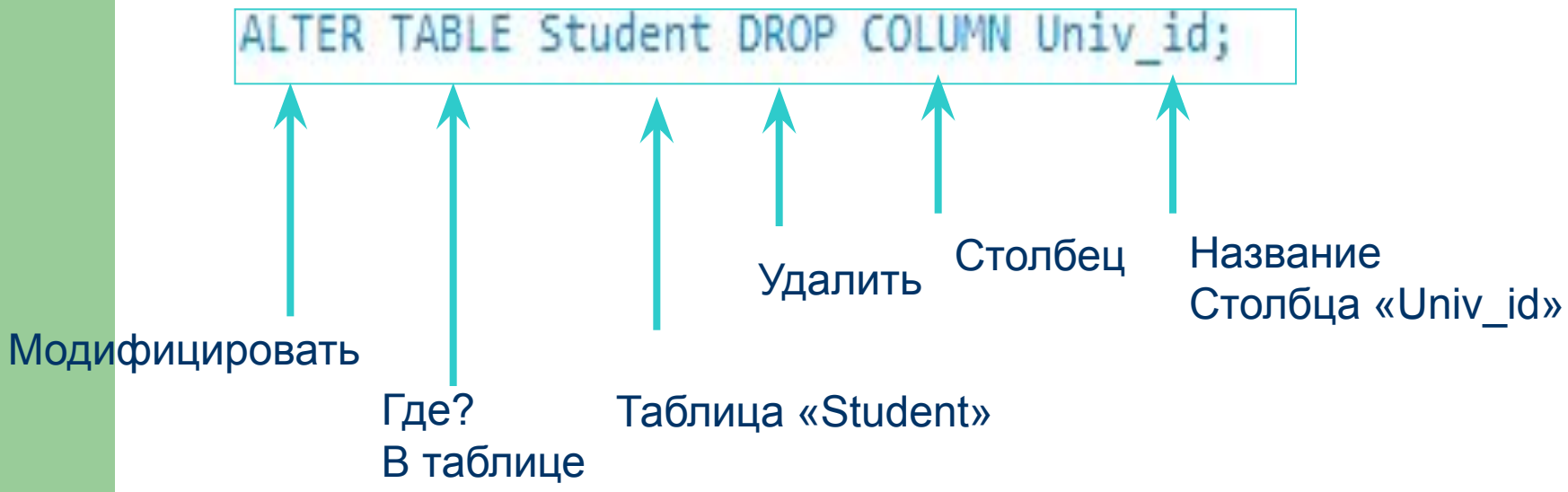
Сортировка по алфавиту



Выборка по условию

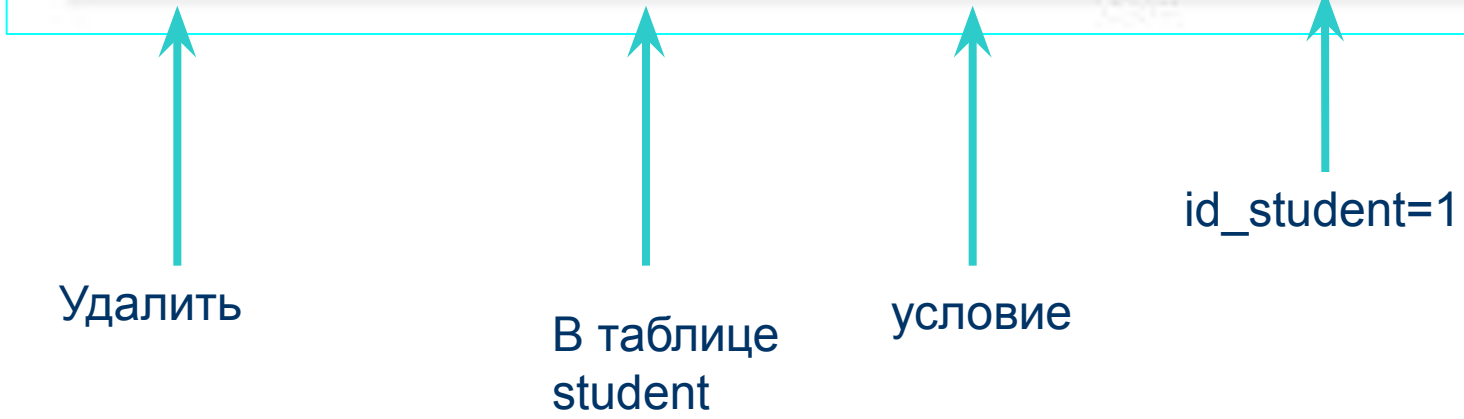


Удаление столбцов в таблице



Удаление строк в таблице по id

```
delete from student where id_student=1
```



Создание внешнего ключа

Общий синтаксис установки внешнего ключа на уровне таблицы:

```
[CONSTRAINT имя_ограничения]
FOREIGN KEY (столбец1, столбец2, ... столбецN)
REFERENCES главная_таблица (столбец_главной_таблицы1, столбец_главной_таблицы2, ... столбец_главной_таблицыN)
[ON DELETE действие]
[ON UPDATE действие]
```

Пример:

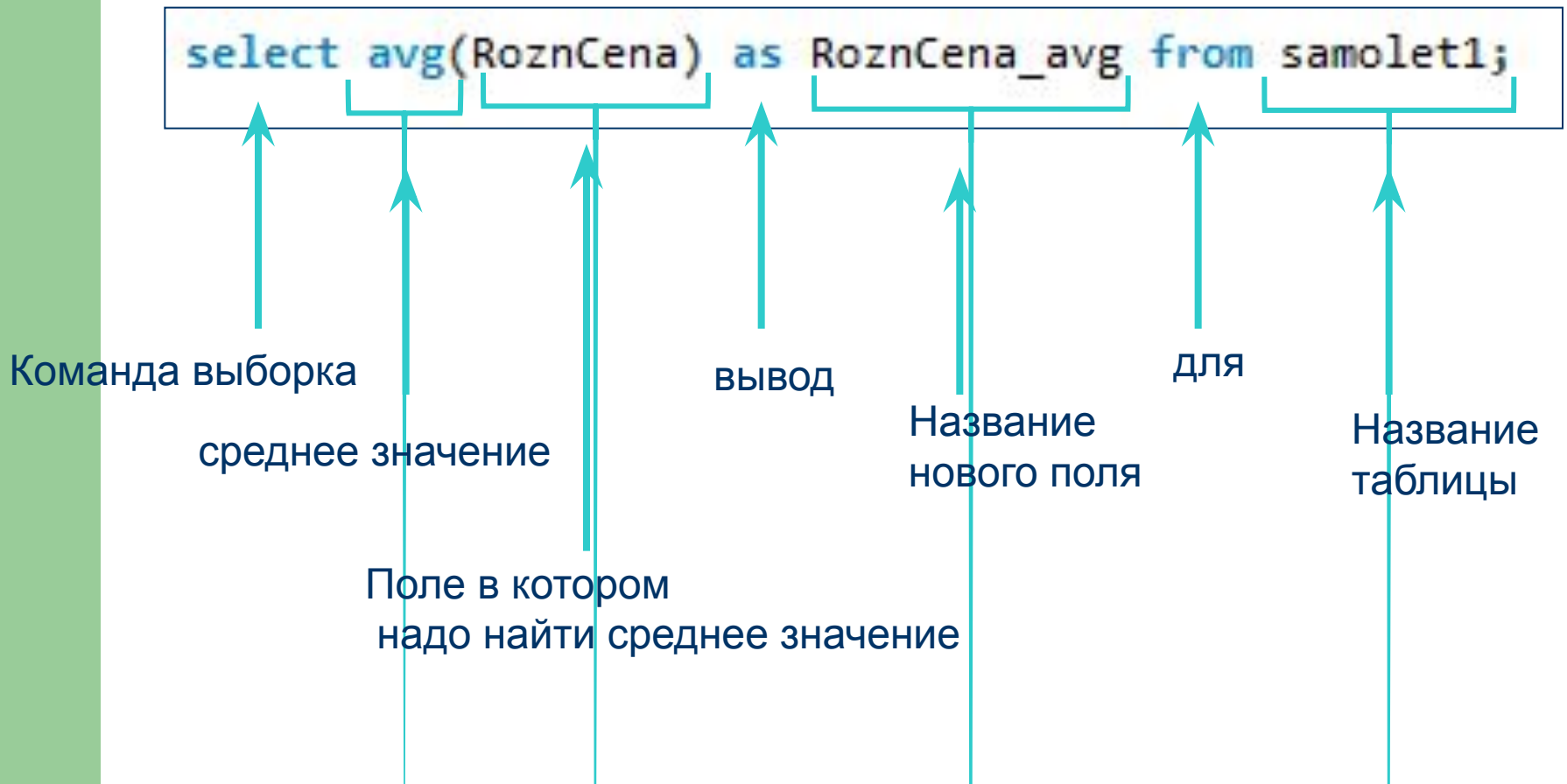
```
CREATE TABLE Orders (
    OrderID int NOT NULL,
    OrderNumber int NOT NULL,
    PersonID int,
    PRIMARY KEY (OrderID),
    FOREIGN KEY (PersonID) REFERENCES Persons(PersonID)
);
```

Подсчет минимального, максимального и среднего значения

AVG(<имя поля>) - среднее по всем значениям данного поля
COUNT(<имя поля>) или **COUNT (*)** - число записей
MAX(<имя поля>) - максимальное из всех значений данного поля
MIN(<имя поля>) - минимальное из всех значений данного поля
SUM(<имя поля>) - сумма всех значений данного поля

Подсчет среднего значения

```
select avg(RoznCena) as RoznCena_avg from samolet1;
```



Сортировка результата ORDER BY

Вывод месяца на русском языке

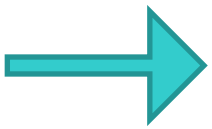
Перед выполнением запроса прописать:

SET @@lc_time_names=`ru_Ru`;

Внутри прописать
Date_format (столбец день рождения,'%d %M %Y')

Пример:

```
SET @@lc_time_names=`ru_Ru`;
    select
    date_format(birthday,'%d.%M.%Y')
    as "День рождения"
    from Student
```



День рождения
03.Декабря.1982
01.Декабря.1980
07.Июня.1979
08.Декабря.1981
01.Мая.1981
NULL
07.Января.1980
05.Ноября.1979
05.Августа.1981
01.Декабря.1981

Преобразование вывода и встроенные функции

Арифметические операции для преобразования числовых данных

- Унарный (одиночный) оператор - (знак минус) изменяет знак числового значения, перед которым он стоит, на противоположный.
- • Бинарные операторы +, -, * и / предоставляют возможность выполнения арифметических операций сложения, вычитания, умножения и деления.

Например, результат запроса

```
SELECT SURNAME, NAME, STIPEND, - ( STIPEND*KURS)/2
FROM STUDENT
WHERE KURS = 4 AND STIPEND >0;
```

SURNAME	NAME	STIPEND	KURS	
Сидоров	Вадим	150	4	-300
Петров	Антон	200	4	-400

Символьная операция конкатенации строк

SELECT **CONCAT (столбец, “;”, столбец) as название столбца нового from таблица**

Конкатенация позволяет соединять ("склеивать") значения двух или более столбцов символьного типа или символьных констант в одну строку

Символьные функции преобразования букв различных слов в строке

LOWER — перевод в строчные символы (нижний регистр)

LOWER (<строка>)

UPPER — перевод в прописные символы (верхний Регистр)

UPPER (<строка >)

INITCAP — перевод первой буквы каждого слова строки в заглавную (прописную)

INITCAP (<строка>)

Символьные функции преобразования букв различных слов в строке

Например :

```
SELECT LOWER(SURNAME), UPPER(NAME)
FROM STUDENT
WHERE KURS = 4 AND STIPEND > 0;
```

Результат запроса будет выглядеть следующим образом

SURNAME	NAME
Сидоров Петров	Вадим Антон

Символьные, строковые функции

LPAD — дополнение строки слева

LPAD (<строка>, <длина> [,<подстрока.>]

<строка> дополняется слева указанной в <подстроке> последовательностью символов до указанной <длины>.(возможно, с повторением последовательности);

если <подстрока> не указана, то по умолчанию <строка> дополняется пробелами;

если <длина> меньше длины <строки>, то исходная <строка> усекается слева до заданной <длины>.

Символьные, строковые функции

RPAD — дополнение строки справа

RPAD(<строка>,<длина>[, <подстрока>])

<строка> дополняется справа указанной в <подстроке> последовательностью символов до указанной <длины> (возможно, с повторением последовательности);

— если <подстрока> не указана, то по умолчанию <строка> дополняется пробелами;

— если <длина> меньше длины <строки>, то исходная <строка> усекается справа до заданной <длины>.

Символьные, строковые функции

LTRIM — удаление левых граничных символов

```
LTRIM (<строка> [, <подстрока>])
```

из <строки> удаляются слева символы, указанные в <подстроке>;

— если <подстрока> не указана; то по умолчанию удаляются пробелы;

— в <строку> справа добавляется столько пробелов, сколько символов слева было удалено, т.е. длина <строки> остается неизменной.

Символьные, строковые функции

RTRIM — удаление правых граничных символов

RTRIN(<строка>[,<подстрока>])

- *из <строки> удаляются справа символы, указанные в <подстроке>;*
- *- если <подстрока> не указана, то по умолчанию удаляются пробелы;*
- *- в <строку> слева добавляется, столько пробелов, сколько символов справа было удалено, т.е. длина <строки> остается неизменной.*

Выбор первой буквы слова

- **Левый**

LEFT(ColumnX, 1)

Функция IFNULL

IFNULL (NULL, «IFNULL function») возвращает строку **IFNULL function**, потому что первый аргумент - **NULL**.

Пример фрагмента SQL-запроса:

…..(ifnull(city, "не указан"))…….

Возвращает строку «не указан», если аргумент NULL

Пример использования ifnull

Вставка данных в таблицу контактов с указанием домашнего и рабочего телефона.

```
1  INSERT INTO contacts(contactname,bizphone,homephone)
2  VALUES('John Doe','(541) 754-3009',NULL),
3       ('Cindy Smith',NULL,'(541) 754-3110'),
4       ('Sue Greenspan','(541) 754-3010','(541) 754-3011'),
5       ('Lily Bush',NULL,'(541) 754-3111');
```

	contactName	bizphone	homephone
▶	John Doe	(541) 754-3009	NULL
	Cindy Smith	NULL	(541) 754-3110
	Sue Greenspan	(541) 754-3010	(541) 754-3011
	Lily Bush	NULL	(541) 754-3111

Получить домашний телефон контакта, если рабочий телефон недоступен. If null вернет домашний телефон, если рабочий телефон - **NULL**.

```
1  SELECT
2      contactname, IFNULL(bizphone, homephone) phone
3  FROM
4      contacts;
```

	contactname	phone
▶	John Doe	(541) 754-3009
	Cindy Smith	(541) 754-3110
	Sue Greenspan	(541) 754-3010
	Lily Bush	(541) 754-3111

Функции работы с числами

ABS (<значимое числовое выражение>)

FLOOR — наибольшее целое, не превосходящее заданное число с плавающей точкой

- FLOOR(<значимое числовое выражение>)

CEIL — наименьшее целое, которое равно или больше заданного числа

- CEIL(<значимое числовое выражение>)

Функция округления — ROUND

ROUND (<значимое числовое выражение>, <точность>)

- аргумент <точность> задает точность округления

Функция усечения — TRUNC

- TRUNC (<значимое числовое выражение>, <точность>)

Тригонометрические функции — COS, SIN, TAN

- COS(<значимое числовое выражение>)
- SIN(<значимое числовое выражение>)
- TAN(значимое числовое выражение>)

Функции работы с числами

- Гиперболические функции — COSH, SINH, TANH
- COSH(<значимое числовое выражение>)
- SINH(<значимое числовое выражение>)
- TANH(<значимое числовое выражение>)
- • Экспоненциальная функция — EXP
- ***EXP****(<значимое числовое выражение>)*
- • Логарифмические функции — ***L N****,* LOG
- ***LN****(<значимое числовое выражение>)*
- ***LOG****(<значимое числовое выражение>)*
- • Функция возведения в степень — POWER
- POWER *(<значимое числовое выражение>,<показатель степени>)*
- • Определение знака числа — SIGN
- SIGN*(<значимое числовое выражение>)*
- • Вычисление квадратного корня — SQRT
- SQRT *(<значимое числовое выражение>)*

Функции преобразования значений

- Преобразование в символьную строку — **TO_CHAR**
- **TO_CHAR** *(<значимое выражение> [, <символьный формат>])*
- — *<значимое выражение>* должно представлять числовое значение или значение типа дата-время;
- — для числовых значений <символьный формат> должен иметь синтаксис [S]9[9…][.9[9…]], где S – представление знака числа (при отсутствии предполагается без отображения знака), 9 — представление цифр-знаков числового значения (для каждого знакоместа). Символьный формат определяет вид отображения чисел. По умолчанию для числовых значений используется формат '999999.99';
- для значений типа ДАТА-ВРЕМЯ <символьный формат> имеет вид (т. е. вид отображения значений даты и времени)

Функции преобразования значений

- — в части даты:
- 'DD-Mon-YY'
- 'DD-Mon-YYYY'
- 'MM/DD/YY'
- 'MM/DD/YYYY'
- 'MM.DD.YY'
- 'MM.DD.YYYY'
- В части времени:
- 'HH24'
- 'HH24:Ml'
- 'HH24:MI.SS’
- 'HH24:Ml:SS.FF’
- где: - HH24 — часы в диапазоне от 0 до 24;
- Ml — минуты;
- SS — секунды;
- FF — тики (сотые доли секунды)

Задание формата ДАТЫ

- DATE_FORMAT()

```
SELECT DATE_FORMAT("2008-11-19",'%d.%m.%Y');
результат

19.11.2008
```

Определитель	Описание
%M	Название месяца (январь...декабрь)
%W	Название дня недели (воскресенье...суббота)
%D	День месяца с английским суффиксом (0st, 1st, 2nd, 3rd и т.д.)
%Y	Год, число, 4 разряда
%y	Год, число, 2 разряда
%X	Год для недели, где воскресенье считается первым днем недели, число, 4 разряда, используется с '%V'
%x	Год для недели, где воскресенье считается первым днем недели, число, 4 разряда, используется с '%v'
%a	Сокращенное наименование дня недели (Вс...Сб)
%d	День месяца, число (00..31)
%e	День месяца, число (0..31)
%m	Месяц, число (00..12)
%c	Месяц, число (0..12)
%b	Сокращенное наименование месяца (Янв...Дек)
%j	День года (001..366)
%H	Час (00..23)
%k	Час (0..23)
%h	Час (01..12)
%I	Час (01..12)
%l	Час (1..12)
%i	Минуты, число (00..59)
%r	Время, 12-часовой формат (hh:mm:ss [AP]M)
%T	Время, 24-часовой формат (hh:mm:ss)
%S	Секунды (00..59)
%s	Секунды (00..59)
%p	AM или PM
%w	День недели (0=воскресенье..6=суббота)
%U	Неделя (00..53), где воскресенье считается первым днем недели
%u	Неделя (00..53), где понедельник считается первым днем недели
%V	Неделя (01..53), где воскресенье считается первым днем недели. Используется с `%X'
%v	Неделя (01..53), где понедельник считается первым днем недели. Используется с `%x'
%%	Литерал `%'.