|  |
| --- |
| [$QLcommand](#_top): |

SHOW DATABASES; - выводит все базы данных.

SHOW TABLES; - выводит все таблицы в use\_db;

CREATE DATABASE \* ; - создание базы данных (num\_lul, numDfg, root).

DROP DATABASE \*; - удаление базы данных.

USE \*; - выбираем базу данных, где будем работать.

!СОЗДАНИЕ СТРУКТУРЫ ТАБЛИЦЫ:

|  |
| --- |
| *id - название поля(столбца), INT – тип данных .*  AUTO\_INCREMENT – пишется для автозаполнения.  PRIMARY KEY – определяет первичный ключ.  NOT NULL – значение не может быть нулевым.  *Если забыли добавить колонку, смотри* [*ALTER*](#_.ALTER)*!* |

CREATE TABLE \* - \*-название таблицы.

🡪 (

🡪 id INT AUTO\_INCREMENT, *-*

🡪 fname VARCHAR(30) NOT NULL,

🡪 enail VARCHAR(30),

🡪 PRIMARY KEY(id)

🡪); - *создание таблицы с первичным ключом*.

DROP TABLE \*; - *удаление таблицы.*

DESC \*; - *просмотр структуры таблицы;*

|  |
| --- |
| возможен вариант: INSERT INTO название таблицы (name, email) VALUES - добавить только нужные данные(строки, записи) |

INSERT INTO \* VALUES - добавление инф. в таблицу

🡪 ( 1, Petr, dfool@mail.ru),

🡪 (2, Slava, floot@mail.ru);

Удаление записей:

DELETE FROM таблица; - удалит все записи в таблице;

DELETE FROM таблица

🡪 WHERE (условие); - удаление определенной записи;

|  |
| --- |
| В первом случае, значение передастся всему столбцу(полю). Удобно если надо изменить значения всех данных в столбце.  Во втором случае мы добавили уточнение построчно, чтобы попасть 1:1 в таблице; |

Обновление записей:

UPDATE таблица SET поле = значение;

UPDATE таблица SET поле = значение

WHERE поле, столбец = запись, строка

## .SELECT

|  |
| --- |
| \* - Если оставить пустым, выведет все столбцы(поля)  \* - Название таблицы |

**SELECT** \* **FROM** \*; - выводит нужные нам данные с помощью определенных условий.

SELECT \* FROM таблица

🡪**WHERE** поле < <= >= >, = значение (ГДЕ)

SELECT \* FROM таблица

|  |
| --- |
| % - используется как точка отсчета, можно сделать выборку по окончанию слова и т.д.  **\_** - выборка по любой букве, на этом месте  ‘\_eke%’**,** ‘%Col\_’ |

🡪WHERE поле < <= >= >, = значение

🡪**AND/O**R поле < <= >= >, = значение (И/ИЛИ)

SELECT \* FROM таблица

🡪WHERE поле **LIKE** (‘%ok%’), (‘%ok’), (‘ok%’), (‘oka\_’)

|  |
| --- |
| IN - сокращает запись, используем меньше AND/OR |

SELECT \* FROM таблица

🡪WHERE поле **IN** выборка поля;

|  |
| --- |
| Значения записываются от малого к большему:  🡪 BETWEEN 50 AND 190; |

SELECT \* FROM таблица

🡪WHERE поле **BETWEEN** значение AND значение;

|  |
| --- |
| NOT желательно пишется с после WHERE, обозначает «НЕ» |

SELECT \* FROM таблица

🡪WHERE **NOT** поле BETWEEN 50 AND 190;

Также можно создавать условия в условиях: SELECT \* FROM (таблица)

🡪WHERE (условие > 70 OR условие < 100)

🡪AND условие < 25; -короче, все зависит от воображения;

|  |
| --- |
| SELECT fnames\_list, SUM(price) FROM sales  -> **GROUP BY** fnames\_list;  SELECT fnames\_list, MAX(price) FROM sales  -> **GROUP BY** fnames\_list; |

SELECT поля + агрегатная функция FROM таблица

🡪 **GROUP BY** поле; - группирует по признаку;

|  |
| --- |
| SELECT fnames\_list, s\_product, price FROM sales  🡪 ORDER BY price DESC  🡪 LIMIT 1; - лимит выведет один самый большой результат.  SELECT fnames\_list, s\_product, price FROM sales  -> ORDER BY price DESC  -> LIMIT 0,3; - когда два числа, правило массива [0][1][2]; |

SELECT **DISTINCT** FROM таблица; - убирает дубли; **SELECT DISTINCT fnames\_list FROM sales;**

SELECT .. **LIMIT** число/**LIMIT** число, число; - Выводит числа по указанному лимиту: | fnames\_list | s\_product | price |

| Аня | Торт | 120 |

| Аня | Арбуз | 70 |

| Николай | Кола 2л | 65 |

|  |
| --- |
| Пример с id:  ALTER TABLE (таблица)  🡪ADD COLUMNid INT AUTO\_INCREMENT FIRST,  🡪ADD PRIMARY KEY (id);  (FIRST означает что будет первым столбцом (полем) в таблице, также можно указать “AFTER название колонки” – что означает после) |

## .**ALTER**

ALTER TABLE таблица

🡪**ADD COLUMN название тип;**

ALTER TABLE таблица **DROP COLUMN** колонка; - удаляем колонку;

ALTER TABLE таблица

🡪**RENAME TO** новое название таблицы; - меняем название таблицы в базе данных;

ALTER TABLE таблица

🡪 **CHANGE COLUMN** старое имя поля новое имя поля ТИП АТРИБУТЫ(NOT NULL); -если меняем название, тип атрибуты колонки без запятых через пробел. Пример: name rename\_r VARCHAR(10) NOT NULL;

ALTER TABLE таблица

🡪**MODIFY COLUMN** имя ТИП АТРИБУТЫ(NOT NULL); – если не надо менять имя;

|  |
| --- |
| Пример: **UPDATE** user  **SET** user **=** **CASE**  **WHEN** people > 0 **THEN** ‘many people’  **WHEN** people < 0 **THEN** ‘few people’  **ELSE** ‘Not people’  **END;** |

# CASE и ORDER BY:

… CASE

🡪WHEN выражение THEN возвращаемое значение

…. бесконечное множество КОГДА/ТОГДА …..

🡪WHEN выражение THEN возвращаемое значение

🡪ELSE возвращаемое значение

🡪END; -конец;

|  |
| --- |
| **SELECT** fnames\_list,lnames\_list,rank **FROM** user 🡪**ORDER BY** fnames\_list**;**  **SELECT** fnames\_list,lnames\_list,rank **FROM** user 🡪**ORDER BY** fnames\_list **DESC;**  Где  **DESC** от > к <, а **ASK** от < к >; |

… ORDER BY…; - сортировка по алфавиту или <>

… ORDER BY…DESC; - сортировка в обратном порядке;

(приоритеты: NULL, цифры, спецсимволы, символы)

|  |
| --- |
| UPDATE таблица  SET  Новое Поле = SUBSTRING INDEX  (Делимое поле, « »(делитель пробел), 1); |

# **ФУНКЦИИ ДЛЯ РАБОТЫ СО СТРОКАМИ:**

**SUBSTRING\_INDEX**(столбец, маркер, 1 это лево (-1 это право)); Пример:

|  |
| --- |
| SUBSTRING\_INDEX разделяет строки в столбце и переносит в новый столбец, к примеру есть ИМЯ ФАМИЛИЯ в одном столбце, после использования будет ИМЯ | ФАМИЛИЯ в разных столбцах. ВОТЬ |

… RIGHT/LEFT (поле, количество символов)

Пример 1: SELECT LEFT(название поля, количество) FROM таблица;

Пример 2: UPDATE таблица 🡪SET поле = RIGHT(поле, количество);

UPPER/LOWER - меняет регистр;

Пример: UPDATE таблица 🡪SET поле = UPPER(поле) 🡪 WHERE id = 1; (изменит регистр в строке один)

LTRIM/RTRIM -удаляет пробелы слева/справа

# **ФУНКЦИИ ДЛЯ РАБОТЫ С ЧИСЛАМИ:**

SUM(поле) 🡪 выводит сумму всех чисел; SELECT SUM(price) FROM sales;

MIN/MAX(поле) 🡪 выводит мин/макс число; SELECT MAX(price) FROM sales;

SELECT fnames\_list, lnames\_list, s\_product FROM sales

🡪WHERE price = (SELECT MAX(price) FROM sales); (Чтобы работали функции, вызываем SELECT повторно)

COUNT(поле) 🡪 выводит количество записей; SELECT COUNT(price) FROM sales;

AVG(поле) 🡪 выводит среднее арифметическое; SELECT AVG(price) FROM sales;

-- Change a database

ALTER DATABASE database\_name

*CHARACTER* SET = utf8mb4 COLLATE = utf8mb4\_unicode\_ci;

-- Change a table

ALTER TABLE table\_name

CONVERT TO *CHARACTER* SET utf8mb4 COLLATE utf8mb4\_unicode\_ci;

-- Change a column

ALTER TABLE table\_name

CHANGE column\_name column\_name *VARCHAR*(255)

*CHARACTER* SET utf8mb4 COLLATE utf8mb4\_unicode\_ci;