# Логирование

etc/mysql/my.cnf

# \* Logging and Replication

general\_log\_file = /var/log/mysql/query.log

general\_log = 1

# Консоль

По умолчанию создается пользователь root с админскими правами

## Подключение к админу (важно под sudo)

**sudo** mysql -uroot

## Пользователи

### Получить список всех пользователей

SELECT user,host FROM mysql.user;

### Текущий пользователь

SELECT user();

можно просмотреть через команду

status

Просмотреть всех активных пользователей

SELECT SUBSTRING\_INDEX(host, ':', 1) AS host\_short, GROUP\_CONCAT(DISTINCT user) AS users, COUNT(\*) AS threads FROM information\_schema.processlist GROUP BY host\_short ORDER BY COUNT(\*), host\_short;

### Создать пользователей

**CREATE USER** usr@localhost **IDENTIFIED BY** "123";

### Удалить пользователя

**DROP USER** user@localhost;

### Изменить пароль пользователя

вариант 1

SET PASSWORD FOR usr@localhost = 'new\_password';

вариант 2

ALTER USER usr@localhost IDENTIFIED BY '123456789';

alex — имя пользователя

По умолчанию прав не имеет.

### Изменение прав пользователя

GRANT ПРАВО, ПРАВО ON база\_даних.таблиця TO  ім’я\_користувача@хост;

ALL — дати всі права до бази даних, за винятком GRANT OPTION. Якщо ви не вкажете назву конкретної бази даних, користувач отримає повний доступ до всього сервера MySQL;

CREATE — право створювати нові бази даних та таблиці;

DELETE — право видаляти рядки з таблиці;

DROP — право видаляти бази даних або таблиці;

GRANT OPTION — право призначати чи відбирати права. Але вдасться дати або відібрати тільки ті права, якими володіє користувач, який використовує команду;

INSERT — право створювати рядки у таблиці;

SELECT — право переглядати рядки в таблицях;

UPDATE — право змінювати зміст рядків у таблицях.

Пример

GRANT CREATE ON music.\* to [usr@localhost](mailto:usr@localhost)

GRANT ALL ON bookstore.\* TO "bookstore\_administrator"@"localhost";

### Забрать права у пользователя

REVOKE ПРАВО ON база\_даних.таблиця FROM user@хост;

Пример

REVOKE DROP ON bookstore.\* FROM "bookstore\_administrator"@"localhost";

### Сбросить кешь привелегий

FLUSH PRIVILEGES;

### Просмотреть список прав пользователя

SHOW GRANTS FOR "user"@"localhost";

## Подключиться к mySql

**mysql** -h 127.0.0.1 -uroot -p

root – имя пользователя

пример 2 – с указанием пароля

**mysql** -h localhost -upma -p111

## База данных

### Подключиться к БД

**use** namebd

### Создать БД

**CREATE DATABASE** nameBD;

Просмотреть список всех БД

**SHOW** databases;

Выведет список ВСЕХ таблиц в Базе Данных base\_name.

Обязательно в конце надо поставить ;

## Импорт БД

Важно указать точный путь где лежит источник

**source** db\_name.sql

**source** opros\_list2.sql

## Выгрузить из БД

username – имя пользователя БД

password – пароль к БД

crosscuttinganalytics – имя БД

db.sql – файл в который импортировать

mysqldump -uusername -ppassword --routines crosscuttinganalytics > db.sql

## Посмотреть структуру таблицы

Обязательно в конце надо поставить ;

**describe name\_table;**

## Выйти с MySql

\q

# SHOW

***SHOW tables in base\_name;***

# SELECT

*SELECT \* FROM tbl\_name;*

## count

*Всего записей*

*SELECT count(\*) FROM tbl\_name;*

## limit

Выбирает (SELECT) из (FROM) таблицы tbl\_name лимит (LIMIT) 3 записи, начиная с 2.

*SELECT \* FROM tbl\_name LIMIT 2,3;*

## Order

Выбирает (SELECT) ВСЕ (\*) записи из (FROM) таблицы tbl\_name и сортирует их (ORDER BY) по полю id по порядку.  
*SELECT \* FROM tbl\_name ORDER BY id;*

### desc

Выбирает (SELECT) ВСЕ записи из (FROM) таблицы tbl\_name и сортирует их (ORDER BY) по полю id в ОБРАТНОМ порядке.

*SELECT \* FROM tbl\_name ORDER BY id DESC;*

Выбирает (SELECT) ВСЕ (\*) записи из (FROM) таблицы users и сортирует их (ORDER BY) по полю id в порядке возрастания, лимит (LIMIT) первые 5 записей.  
*SELECT \* FROM users ORDER BY id LIMIT 5;*

## where

Выбирает все записи из таблицы users, где поле fname соответствует значению Gena.  
*SELECT \* FROM users WHERE fname='Gena';*

### and

Select cad.id, cad.ref\_client, cad2.custom\_data , column\_json(custom\_data), cad2.last\_updated, cad2.added\_date

from screens s

join client\_assessment\_demographics cad on cad.ref\_assessment = s.ID

join client\_assessment\_data cad2 on cad2.ref\_client\_assessment\_demographics = cad.id

where s.`type` = 6

and cad2.custom\_data is NOT NULL

and cad2.custom\_data !=''

and ifnull(cad.deleted,0) =0

and s.status =1

and cad.ref\_client = 14667

### like

Выбирает все записи из таблицы users, где значение поля fname начинается с Ge.

*SELECT \* FROM users WHERE fname LIKE 'Ge%';*

Выбирает все записи из таблицы users, где fname заканчивается на na, и упорядочивает записи в порядке возрастания значения id.

*SELECT \* FROM users WHERE fname LIKE '%na' ORDER BY id;*

### *!=*

Выберет данные из таблицы users по полю name ГДЕ id НЕ РАВЕН 2.  
*SELECT name FROM users WHERE id!='2';*

### *DISTINCT*

Допустим у Вас в таблице пользовательских данных есть страна. Так вот если Вы хотите вывести ТОЛЬКО список встречающихся значений (чтобы, например, Россия не выводилось 20 раз, а только один), то используем DISTINCT. Выведет, из массы повторяющихся значений Россия, Украина, Беларусь. Таким образом, из таблицы users колонки country будут выведены ВСЕ УНИКАЛЬНЫЕ значения

*SELECT DISTINCT country FROM users;*

### in

12. Выбирает ВСЕ данные строк из таблицы users где age имеет значения 18,19 и 21.  
*SELECT \* FROM users WHERE age IN (18,19,21);*

### ifnull

Select cad.id, cad.ref\_client, cad2.custom\_data , column\_json(custom\_data)

from screens s

join client\_assessment\_demographics cad on cad.ref\_assessment = s.ID

where s.`type` = 6

and ifnull(cad.deleted,0) =0

and s.status =1

and cad.ref\_client = 14667

## Max/Min

Выбирает МАКСИМАЛЬНОЕ значение age в таблице users. То есть если у Вас в таблице самое большее значение age(с англ. возраст) равно 55, то результатом запроса будет 55.

*SELECT max(age) FROM users;*

Выберет данные из таблицы users по полям name и age ГДЕ age принимает самое маленькое значение.  
*SELECT name, min(age) FROM users;*

## Alias

*Select cad.id, cad.ref\_client, cad2.custom\_data , column\_json(custom\_data), cad2.last\_updated, cad2.added\_date*

*from screens s*

*join client\_assessment\_demographics cad on cad.ref\_assessment = s.ID*

*join client\_assessment\_data cad2 on cad2.ref\_client\_assessment\_demographics = cad.id*

*where s.`type` = 6*

*and cad2.custom\_data is NOT NULL*

## Join

### Join

*Select cad.id, cad.ref\_client, cad2.custom\_data , column\_json(custom\_data)*

*from screens s*

*join client\_assessment\_demographics cad on cad.ref\_assessment = s.ID*

*join client\_assessment\_data cad2 on cad2.ref\_client\_assessment\_demographics = cad.id*

*where s.`type` = 6*

*and cad2.custom\_data is NOT NULL*

## Types

### blob

*Преобразовать в json*

*custom\_data -имеет тип blob*

*Select cad.id, cad.ref\_client, cad2.custom\_data , column\_json(custom\_data)*

*from screens s*

*join client\_assessment\_demographics cad on cad.ref\_assessment = s.ID*

*join client\_assessment\_data cad2 on cad2.ref\_client\_assessment\_demographics = cad.id*

*where s.`type` = 6*

*and cad2.custom\_data is NOT NULL*

# DROP

**Удалить базу данных**

DROP DATABASE [IF EXISTS] database\_name;

**Удалить таблицу**

DROP TABLE users;

**Удалить колонку**

ALTER TABLE clients DROP name;

# AlTER TABLE

**Сделать поле уникальным**

ALTER TABLE `clients` ADD UNIQUE(`name`);

**Добавить колонку**

ALTER TABLE clients ADD Email varchar(255);

# TRIGERS

## Создание

DELIMITER **$$**

CREATE TRIGGER name AFTER INSERT ON users

FOR EACH ROW

BEGIN

INSERT INTO profiles(user\_id) VALUES (NEW.id)

END **$$**

DELIMITER;

DELIMITER **$$**

CREATE TRIGGER name1 AFTER UPDATE ON profiles

FOR EACH ROW

BEGIN

UPDATE users SET updated\_at = now() WHERE id = OLD.user\_id

END **$$**

DELIMITER;

## Удаление существующих

DROP TRIGGER IF EXIST name;

# Fixes

## Замена пользователя

Если выдает ошибку

*The user specified as a definer ('root'@'%') does not exist*

GRANT ALL ON \*.\* TO 'someuser'@'%' IDENTIFIED BY 'complex-password';

FLUSH PRIVILEGES;

## Transaction

### Show list

SHOW PROCESSLIST;

### Удалить транзакцию

1. показать список процессов с помощью команды SHOW PROCESSLIST;

2. удалить командой kill id