MySQL

- MySQL-ը շատ տարածված տվյալների բազա է.
- Հապավումը բացվում է որպես “my Ess Que Ell” (ոչ որպես my sequel).
- Աշխատում է շատ բեծ բազաների հետ և բավականին արագ.

Ինչու՞ ենք մենք օգտագործում MySQL

- Անվճար է
- Ցանկացած սկսնակ կարող է հեշտությամբ տեղադրել.
- Հեշտությամբ տերմինալի (Shell) միջոցով կարլի է ստեղծել աղյուսակներ, օգտվել աղյուսակներից և այլն. . .
- Աշխատում է շատ ծրագրավորման լեզուների հետ (PHP, JAVA, NOD. . .)
- Ունի մեծ թվով գրաֆիկական ինտերֆեյսով գործիքներ

MySQL-ին միացումը

MySQL- ը տրամադրում է ինտերակտիվ Shell, աղյուսակներին ստեղծման, տվյալների տեղադրման և այլ հրահանգնլ կատարելու համար:

```
mysql –u <user_login> -p <user_pass>
```

Ավջատելու հրահանգը՝

```
mysql> QUIT
```

```
mysql> exit
```

Հիմնական հարցումներ

```
mysql> SELECT VERSION(), CURRENT_DATE;
+-----------+--------------+
| VERSION() | CURRENT_DATE |
+-----------+--------------+
| 3.23.49   | 2002-05-26   |
+-----------+--------------+
1 row in set (0.00 sec)
```

Հրահանգների մեծամասնությունը ավարտվում են <;>-ով,

MySQL- ը վերադարձնում է գտնված տողերի ընդհանուր թիվը եւ հարցման կատարման ընդհանուր ժամանակը:

Հիմնական հարցումներ

Մուտքագրված բառերում մեծատառ կամ փոքրատառը Նշանակություն չունի

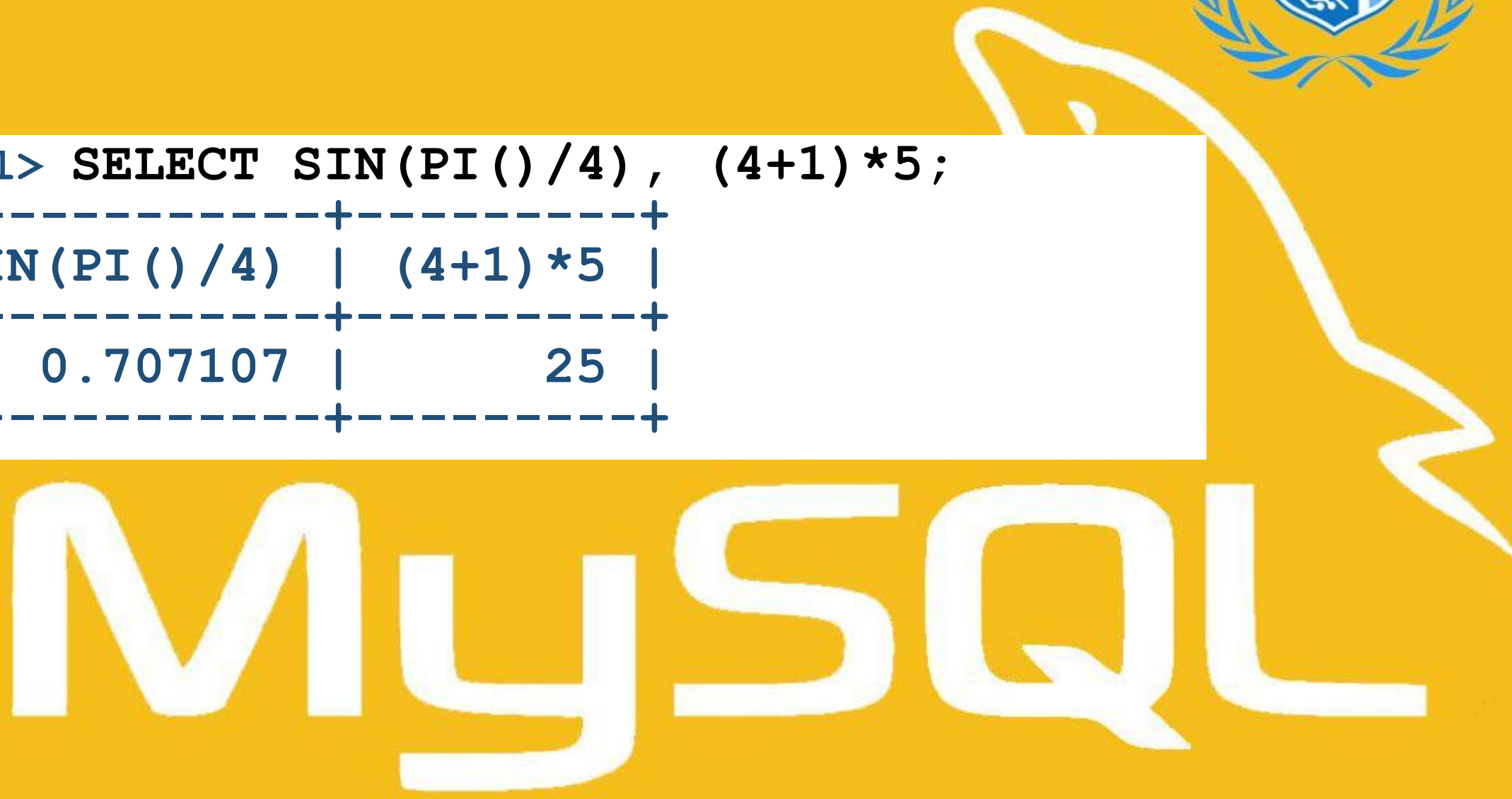
```
mysql> SELECT VERSION(), CURRENT_DATE;
```

```
mysql> select version(), current_date;
```

```
mysql> SeLeCt vErSiOn(), current_DATE;
```

Հիմնական հարցումներ

```
mysql> SELECT SIN(PI()/4), (4+1)*5;
+---------------+---------+
| SIN(PI()/4)   | (4+1)*5 |
+---------------+---------+
|      0.707107 |      25 |
+---------------+---------+
```



Հիմնական հարցումներ

Դուք նաեւ կարող եք մուտքագրել բազմաթիվ հրահանգներ մեկ տողում, արզապես վերջացրեք յուրաքանչյուր հրահանգները <;>-ի օգնությամբ

```
mysql> SELECT VERSION(); SELECT NOW();
+---------------+
| VERSION()     |
+---------------+
| 3.22.20a-log |
+---------------+
+---------------------+
| NOW()               |
+---------------------+
| 2004 00:15:33       |
+---------------------+
```

Հիմնական հարցումներ

```
mysql> SELECT
    -> USER()
    -> ,
    -> CURRENT_DATE;
+--------------------+--------------+
| USER()             | CURRENT_DATE |
+--------------------+--------------+
| joesmith@localhost | 1999-03-18   |
+--------------------+--------------+
```

Հիմնական հարցումներ

Եթե դուք գտնվում եք մուտքագրման գործընթացում և չեք ուզում որ հրահանգը կատարվի, ապա այն չեղարկելու համար մուտքագրելով <\ c>

```
mysql> SELECT
    -> USER()
    -> \c
mysql>
```

Տվյալների բազայի օգտագործումը

Մինշ որէ բազաի օգտագործելը պետ է տեսնել թե ինչ բազաներ գոյություն ունեն:

Դրա համար օգտագործվում է <**SHOW**> հրահանգը

```
mysql> show databases;
+----------+
| Database |
+----------+
| mysql    |
| test     |
| gTea     |
+----------+
2 rows in set (0.01 sec)
```

Տվյալների բազայի օգտագործումը

Տվյալներ բազայի ստեղծումը՝

```
mysql> create database webdb;
```

Տվյալներ բազայի ընտրելը կատարվում է հետևյալ հրահանգի միջոցով

```
mysql> use webdb;
```

Տվյալների բազայի օգտագործումը

Եթե արդեն ընտրել ենք բազան, ապա կարող ենք տեսնել նանում առկա աղյուսակները

```
mysql> show tables;
Empty set (0.02 sec)
```

Սա նշանակում է որ մեր ՏԲ-ում դեռևս աղյուսակներ չկան

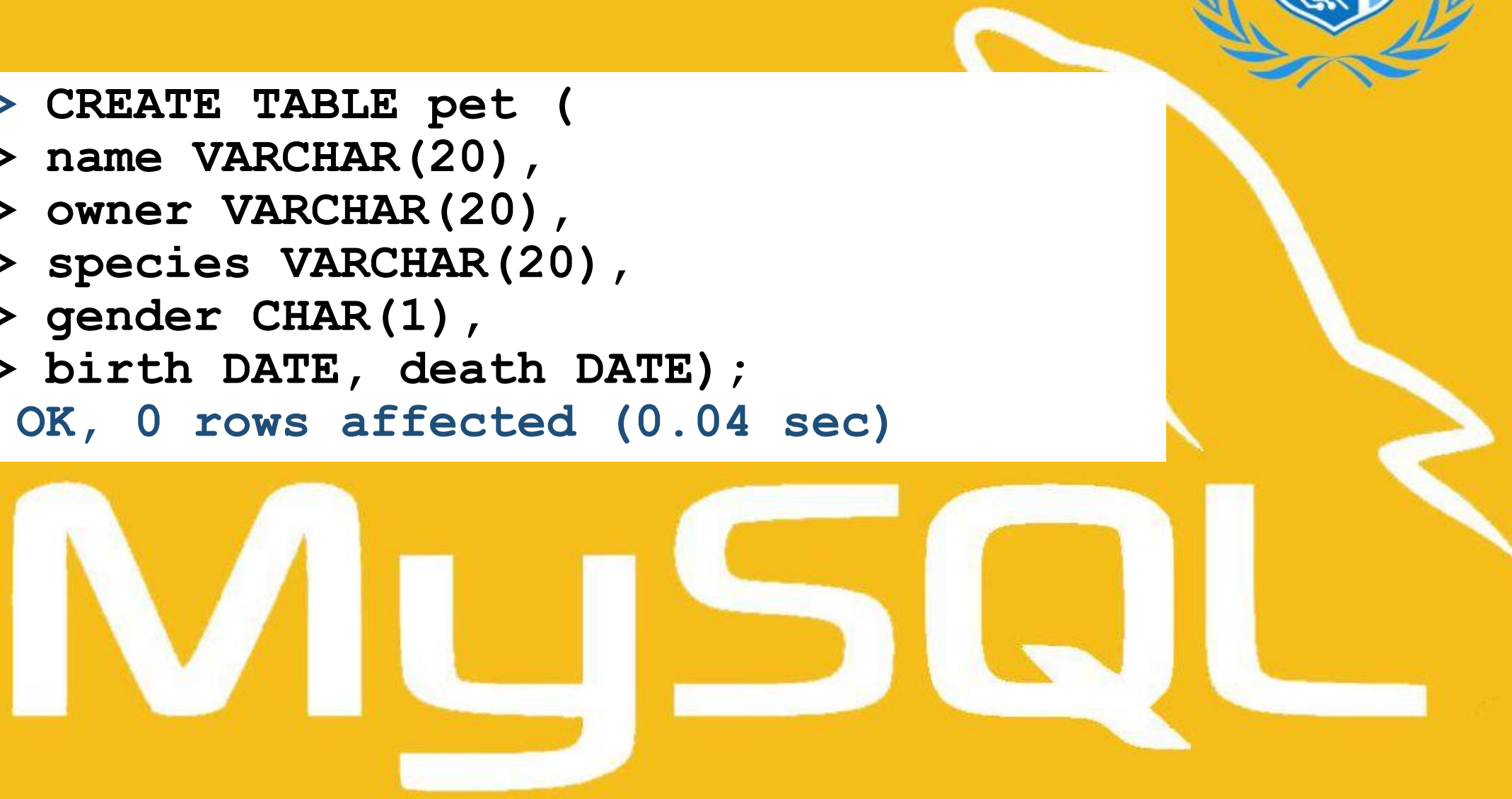
Տվյալների բազայի օգտագործումը

Ստեղծենք մեր առաջին աղուսակը, որը կպարունակի տնային կենդանիների մասին տվյալներ

- **Table: pets**
  - **name: VARCHAR(20)**
  - **owner: VARCHAR(20)**
  - **species: VARCHAR(20)**
  - **gender: CHAR(1)**
  - **birth: DATE**
  - **date: DATE**

Տվյալների բազայի օգտագործումը

```
mysql> CREATE TABLE pet (
    -> name VARCHAR(20),
    -> owner VARCHAR(20),
    -> species VARCHAR(20),
    -> gender CHAR(1),
    -> birth DATE, death DATE);
Query OK, 0 rows affected (0.04 sec)
```



Տվյալների բազայի օգտագործումը

Համոզվենք որ աղյուսակը ստեղծվել է

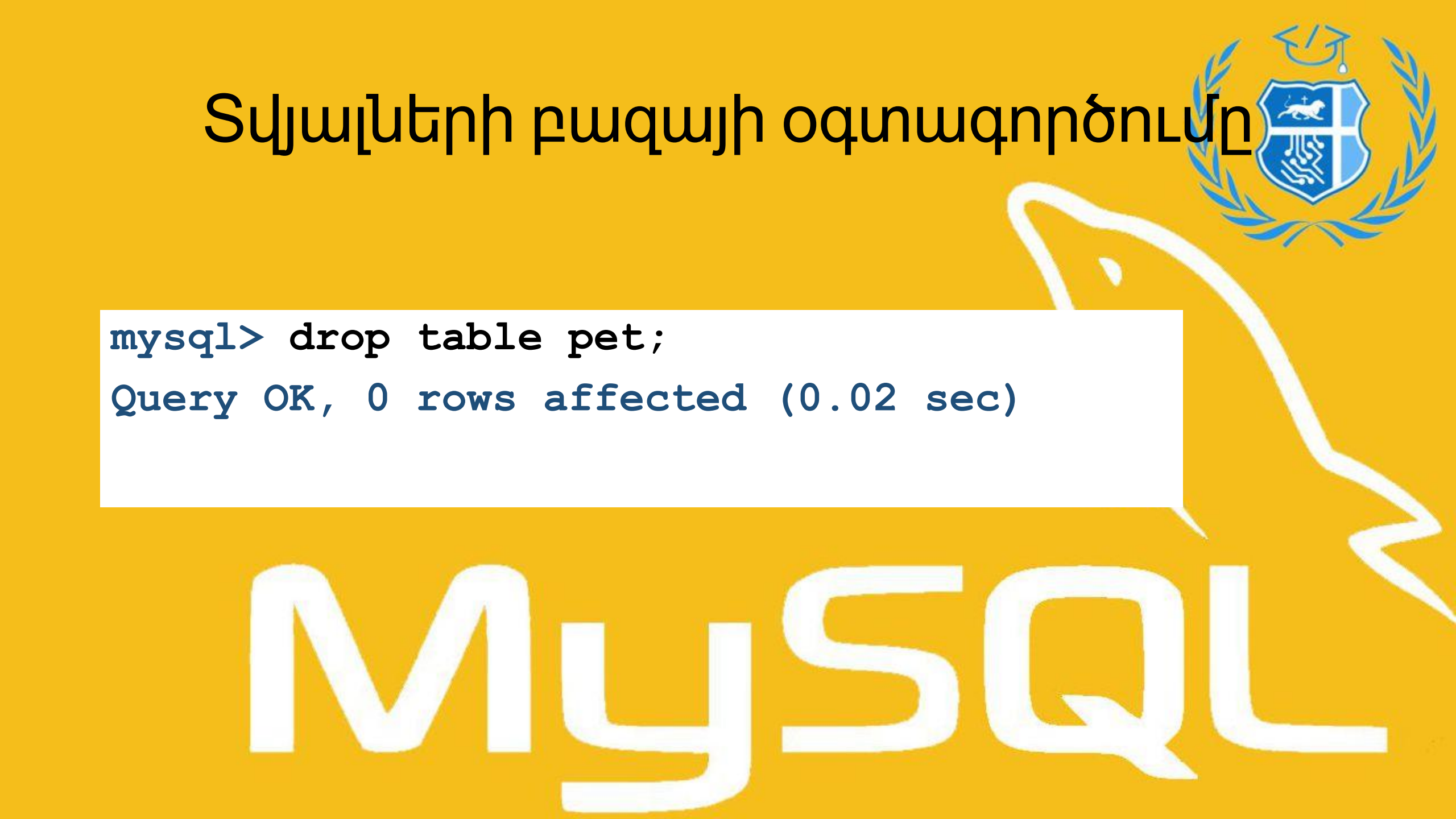
```
mysql> show tables;
+------------------+
| Tables_in_test   |
+------------------+
| pet              |
+------------------+
1 row in set (0.01 sec)
```

Տվյալների բազայի օգտագործումը

Աղյուսակների կառուցվածքը դիտելու համար օգտագործեք <**DESCRIBE**> հրամանը:

```
mysql> describe pet;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| name    | varchar(20) | YES  |     | NULL    |       |
| owner   | varchar(20) | YES  |     | NULL    |       |
| species | varchar(20) | YES  |     | NULL    |       |
| gender  | char(1)     | YES  |     | NULL    |       |
| birth   | date        | YES  |     | NULL    |       |
| death   | date        | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
6 rows in set (0.02 sec)
```

Տվյալների բազայի օգտագործումը



```
mysql> drop table pet;
Query OK, 0 rows affected (0.02 sec)
```

Տվյալների բազայի օգտագործումը

Տվյալների մուտքագրման համար օգտագոր <**INSERT**> հրամանը :

```
mysql> INSERT INTO pet VALUES
      ('Bob', 'Jon', 'cat', 'm','1999-02-04',NULL);
```

name	owner	species	gender	birth	death
Fluffy	Harold	cat	f	1993-02-04	
Claws	Gwen	cat	m	1994-03-17	
Buffy	Harold	dog	f	1989-05-13	
Fang	Benny	dog	m	1990-08-27	
Bowser	Diane	dog	m	1998-08-31	1995-07-29
Chirpy	Gwen	bird	f	1998-09-11	
Whistler	Gwen	bird		1997-12-09	
Slim	Benny	snake	m	1996-04-29	

SQL Select

Աղյուսակից տվյալներ ստանալու համար օգտագործում են <**SELECT**> հրամանը

```
mysql> SELECT <որ սյուները>
       FROM <որ աղյուսակից>
       WHERE <ին պայմանների պետք է բավարարի>
```

Ամբեղջ աղյուսկի տվյալների ստացումը

```
mysql> select * from pet;
+----------+--------+---------+------+------------+------------+
| name     | owner  | species |gender| birth      | death      |
+----------+--------+---------+------+------------+------------+
| Fluffy   | Harold | cat     | f    | 1999-02-04 | NULL       |
| Claws    | Gwen   | cat     | f    | 1994-03-17 | NULL       |
| Buffy    | Harold | dog     | f    | 1989-05-13 | NULL       |
| Fang     | Benny  | dog     | m    | 1999-08-27 | NULL       |
| Bowser   | Diane  | dog     | m    | 1998-08-31 | 1995-07-29 |
| Chirpy   | Gwen   | bird    | f    | 1998-09-11 | NULL       |
| Whistler | Gwen   | bird    |      | 1997-12-09 | NULL       |
| Slim     | Benny  | snake   | m    | 1996-04-29 | NULL       |
+----------+--------+---------+------+------------+------------+
8 rows in set (0.00 sec)
```

Աղյուսակից մի տվյալների մասի ստացումը

```
mysql> SELECT * FROM pet WHERE name = "Bowser";
+--------+-------+---------+------+------------+------------+
| name   | owner | species |gender| birth      | death      |
+--------+-------+---------+------+------------+------------+
| Bowser | Diane | dog     | m    | 1998-08-31 | 1995-07-29 |
+--------+-------+---------+------+------------+------------+
1 row in set (0.00 sec)
```

Աղյուսակից մի տվյալների մասի ստացումը

Կենդանիներ որոնք ծնվել են 1998-ից ուշ

```
SELECT * FROM pet WHERE birth >= "1998-1-1";
```

Գտնենք բոլոր շներին որոնց սեռը իգական է (**AND**)

```
SELECT * FROM pet WHERE species = "dog" AND gender = "f";
```

Գտնենք բոլոր թռչուններին և օձերին (**OR**)

```
SELECT * FROM pet WHERE species = "snake" OR species = "bird";
```

Աղյուսակից մի տվյալների մասի ստացումը

Եթե մեզ բետք է ստանալ միայն մի քանի սյուներ, ապա <*>-ի փոխարեն կարող ենք թվակել մեզ անհրաժեշտ սյուները

```
mysql> select name, birth from pet;
+----------+------------+
| name     | birth      |
+----------+------------+
| Fluffy   | 1999-02-04 |
| Claws    | 1994-03-17 |
| Buffy    | 1989-05-13 |
| Fang     | 1999-08-27 |
| Bowser   | 1998-08-31 |
| Chirpy   | 1998-09-11 |
| Whistler | 1997-12-09 |
| Slim     | 1996-04-29 |
+----------+------------+
8 rows in set (0.01 sec)
```

Ինֆորմացիայի սորթավորում

Ստացված արդյունքները կարող ենք դասավորել < **ORDER BY** > հրահանգի օգնությամբ

```
mysql> SELECT name, birth FROM pet ORDER BY birth;
+----------+------------+
| name     | birth      |
+----------+------------+
| Buffy    | 1989-05-13 |
| Claws    | 1994-03-17 |
| Slim     | 1996-04-29 |
| Fluffy   | 1999-02-04 |
| Fang     | 1999-08-27 |
+----------+------------+
8 rows in set (0.02 sec)
```

Ինֆորմացիայի սորթավորում

Սորթավորը կարող ենք շրջել <**DESC**> բանալի բառի օգնությամբ

Եթե <**DESC**> (լռելյան <**ASC**>)

```
mysql> SELECT name, birth FROM pet ORDER BY birth DESC;
+----------+------------+
| name     | birth      |
+----------+------------+
| Fang     | 1999-08-27 |
| Fluffy   | 1999-02-04 |
| Chirpy   | 1998-09-11 |
| Buffy    | 1989-05-13 |
+----------+------------+
8 rows in set (0.02 sec)
```

Աշխատանք <NULL>- ի հետ

<**NULL**>-ը համարվում է անհայտ կամ բացակայող արժեք

<**NULL**>- ի համար դուք չեք կարող օգտագործել թվաբանական համեմատության օպերատորները, ինչպիսիք են =, <կամ <>:

Փոխարենը, դուք պետք է օգտագործեք IS NULL եւ NOT NULL օպերատորները:

```
mysql> select name from pet where death IS NOT NULL;
```

LIKE

Ստանանք <b>-տոռաով սկսվող տողերը

```
mysql> SELECT * FROM pet WHERE name LIKE "b%";
+--------+--------+---------+------+------------+------------+
| name   | owner  | species | sex  | birth      | death      |
+--------+--------+---------+------+------------+------------+
| Buffy  | Harold | dog     | f    | 1989-05-13 | NULL       |
| Bowser | Diane  | dog     | m    | 1989-08-31 | 1995-07-29 |
+--------+--------+---------+------+------------+------------+
```

LIKE

Ստանանք <fy>-ով վերջացող տողերը

```
mysql> SELECT * FROM pet WHERE name LIKE "%fy";
+--------+--------+---------+------+------------+-------+
| name   | owner  | species | sex  | birth      | death |
+--------+--------+---------+------+------------+-------+
| Fluffy | Harold | cat     | f    | 1993-02-04 | NULL  |
| Buffy  | Harold | dog     | f    | 1989-05-13 | NULL  |
+--------+--------+---------+------+------------+-------+
```

LIKE

<_> օգտագործում ենք երբ <like>-ով պայմանի մեջ պետք է ասել որ տվյա հատվածում կարող է լինել ցանկացած տեկստ

```
mysql> SELECT * FROM pet WHERE name LIKE "_____";
+-------+--------+---------+------+------------+-------+
| name  | owner  | species | sex  | birth      | death |
+-------+--------+---------+------+------------+-------+
| Claws | Gwen   | cat     | m    | 1994-03-17 | NULL  |
| Buffy | Harold | dog     | f    | 1989-05-13 | NULL  |
+-------+--------+---------+------+------------+-------+
```

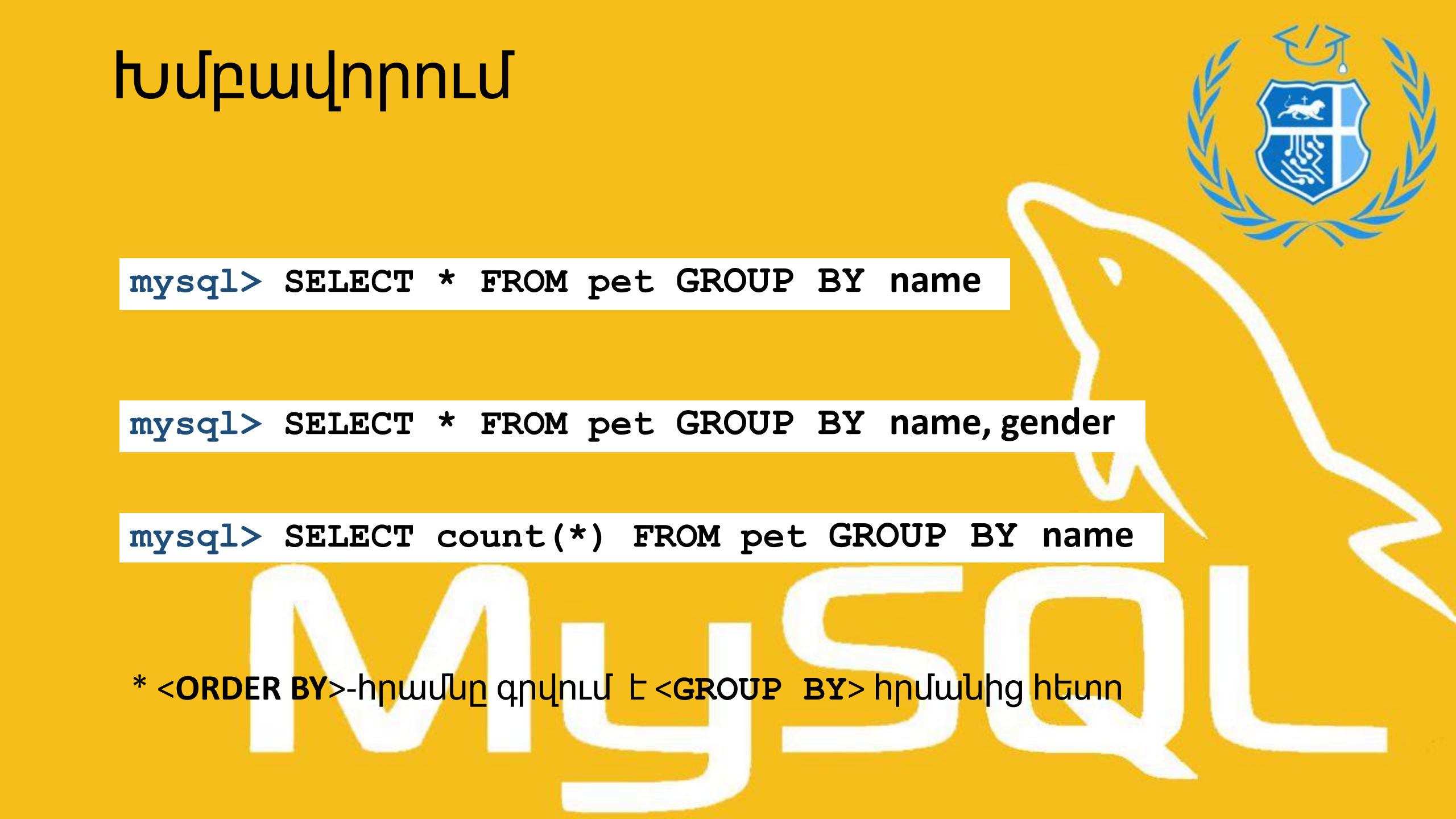
REGEXP

<MySql>-ը ունի նաև regular expressions-ի հնարավորություն:
Այն կիրառվում է <**REGEXP**> բանալի բառի օգնությամբ

```
mysql> SELECT * FROM pet WHERE name REGEXP "^b";
```

```
mysql> SELECT * FROM pet WHERE name NOT REGEXP "fy$";
```

Խմբավորում



```
mysql> SELECT * FROM pet GROUP BY name
```

```
mysql> SELECT * FROM pet GROUP BY name, gender
```

```
mysql> SELECT count(*) FROM pet GROUP BY name
```

* <**ORDER BY**>-հրամանը գրվում է <**GROUP BY**> հրամանից հետո

Սահմանափակումներ

Վորադարձվող տողարի քանակը կարող ենք սահմանափակել <LIMIT> հրամանի օգնությամբ

```
mysql> SELECT * FROM pet LIMIT 1

+--------+---------+----------+----------+---------------+--------+
| name | owner | species | gender | birth         | death |
+--------+---------+----------+----------+---------------+--------+
| Rex    | Valod  | dog      | f        | 1998-07-01  | NULL  |
+--------+---------+----------+----------+---------------+--------+
```

Սահմանափակումներ

<**OFSET**> հրամանի օգնությամբ կարող ենք նշել՝ թե քանի տող է անհրաժեշտ բաց թողնել

```
mysql> SELECT * FROM pet LIMIT 1 OFFSET 2

+--------+----------+----------+------------+-------------+--------+
| name | owner  | species | gender   | birth       | death |
+--------+----------+----------+------------+-------------+--------+
| jeck   | Vardan | snake   | m        |1988-02-04| NULL  |
+--------+----------+----------+------------+-------------+--------+
```

Սահմանափակումներ

<**BETWEEN**> Օպերատորի միջեցով կարող ենք սահմանապակել ստավող արժեքնորի միջակայքը

```
mysql> SELECT * FROM pet WHERE birth BETWEEN '1997-07-01' AND '2000-01-01'
```

IN օպերատոր

<**IN**> օպերատորի միջոցով կարող ենք տվարկել մի քնաի պայման

```
mysql> SELECT * FROM pet WHERE NAME IN('Rex', 'Jeck')
```

```
mysql> SELECT * FROM pet WHERE NAME NOT IN('Rex', 'Jeck')
```

Տվյալենրի ջնջում

Եթե ցանկանում եք ջնջել որևէ տվյալ պետք է օգտագործենք <**DELETE**> հրահանգը:

```
mysql>  DELETE FROM `pet` WHERE name = 'Rex';
```

Տվյալների Թարմացում

Եթե ցանկանում եք փոփոխություններ անել տվյալների մեջ, ապա օգնության է գալիս <**UPDATE**> հրամանը:

```
mysql>  UPDATE `pet` SET `birth` = '1998-07-01' WHERE `name` = 'Rex';
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Աղյուսակի վերափոխում

Աղյուսակը փոփոխելու համար օգտագործում ենք
< **ALTER** > հրահանգը:

Օրինակ եթե անհրաժեշտ է փոխել աղկուսակի անունը

```
mysql>  ALTER TABLE `pet` RENAME `pet_table`;
```

Աղյուսակի վերափոխում

Եթե պետք է աղյուսակում նոր սյուն ավելացնել

```
mysql> ALTER TABLE `pet` ADD `address` VARCHAR(255) AFTER `gender`;
```

Աղյուսակում սյունը տեղափոխելու համար

```
mysql>  ALTER TABLE `pet` CHANGE  `gender`  `gender` CHAR(1) AFTER `name`;
```

Աղյուսակի վերափոխում

Ջնջել սյունը

```
mysql> ALTER TABLE  pet DROP  name;
```

Փոխել սյունի տիպը

```
mysql> ALTER TABLE  pet MODIFY   name VARCHAR(50);
```

Աղյուսակի վերափոխում

Ջնջել սյունը

```
mysql> ALTER TABLE  pet DROP  name;
```

Փոխել սյունի տիպը

```
mysql> ALTER TABLE  pet MODIFY  name VARCHAR(50);
```

Ստատիստիկայի որոշ ֆունկցիաներ

Name	Description
AVG()	Ստացված արժեքների միջինը
COUNT()	Ստացված տողերի քանակը
COUNT(DISTINCT)	Ստացված չկրկնվող տողերի քանակը
MAX()	Ստացված տողերի մաքսիմում արժեքը
MIN()	Ստացված տողերի մինիմում արժեքը
SUM()	Ստացված տողերի գումարը

```
mysql> CREATE TABLE pet (
    -> id int NOT NULL AUTO_INCREMENT,
    -> name VARCHAR(20) NOT NULL,
    -> owner VARCHAR(20),
    -> species VARCHAR(20),
    -> gender CHAR(1) DEFAULT 'm',
    -> birth DATE, death DATE,
    -> PRIMARY KEY (ID) );
Query OK, 0 rows affected (0.09 sec)
```

PRIMARY KEY

Աղյուսակում < **PRIMARY** >-ով առանձնացված դաշտը ցույց է տալիս որ տվյալ դաշտը աղյուսակի համար առաջնային է և չկրկնվող: Այն չի կարող լինել դատարկ կամ NULL:

Աղյուսակում կարող է լինել ընդամենը մեկ < PRIMARY >-ով նշված դաշտ

Նայև հնարավոր է որ երկու կամ ավելի սյուներ միյասին նշվեն որպես < PRIMARY >

Դաշտի սահմանափակում

Աղյուսակ ստեղծելիս մենք կարող ենք <CHECK> ֆունկցիայի միջոցով սահմանափակել որևէ սյան արժեքների տիրույթը

```
mysql> CREATE TABLE Persons (
            ID int NOT NULL,
            LastName varchar(255) NOT NULL,
            FirstName varchar(255),
            Age int CHECK (Age>=18)
        );
```

Մի քանի սահմանափակումներ կատարելու օրինակ

```
mysql> CREATE TABLE Persons (
        ID int NOT NULL,
        LastName varchar(255) NOT NULL,
        FirstName varchar(255),
        Age int,
        City varchar(255),
        CONSTRAINT CHK_Person CHECK (Age>=18 AND City='Sandnes')
    );
```

Ալիասներ

- Ալիասը օգտագործվում է աղյուսակին կամ սյանը վերանվանելու համար:
- Ալիսին սովորաբար տալիս են ավելի կարճ անվանում

```
mysql>  SELECT full_name AS name from users;
```

```
mysql>  SELECT full_name  from  users as  u;
```

```
mysql>  SELECT full_name  from  users  u;
```

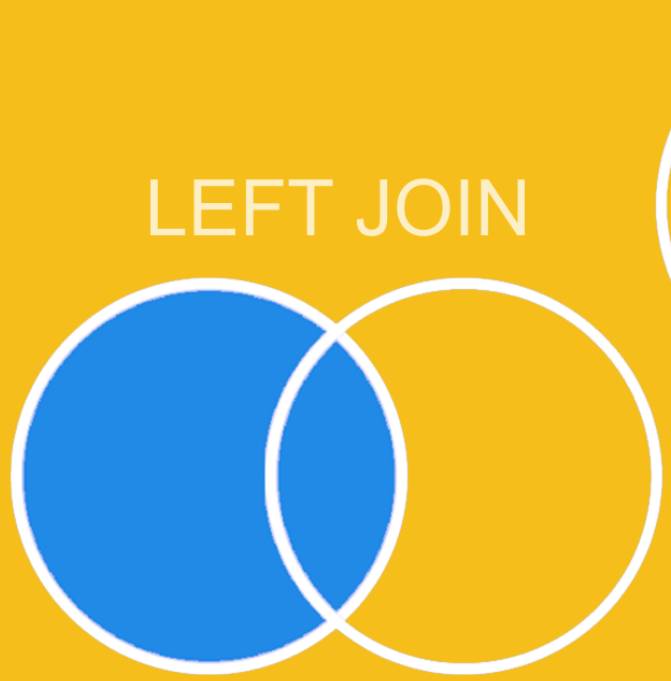
*Աղյուակի անվան պարագայում < **as** > հրամանը կարելի է

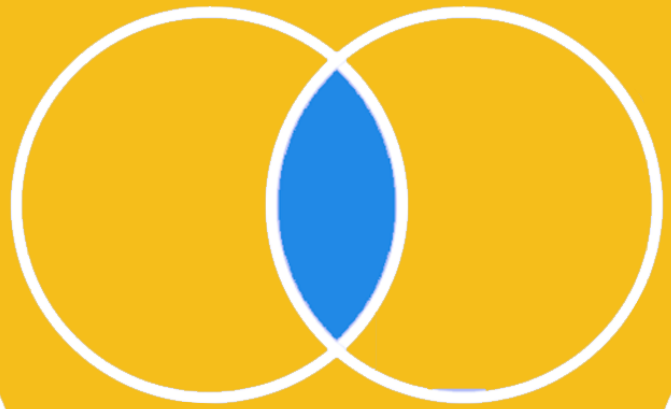
MySQL JOINS (MySQL միացումներ)

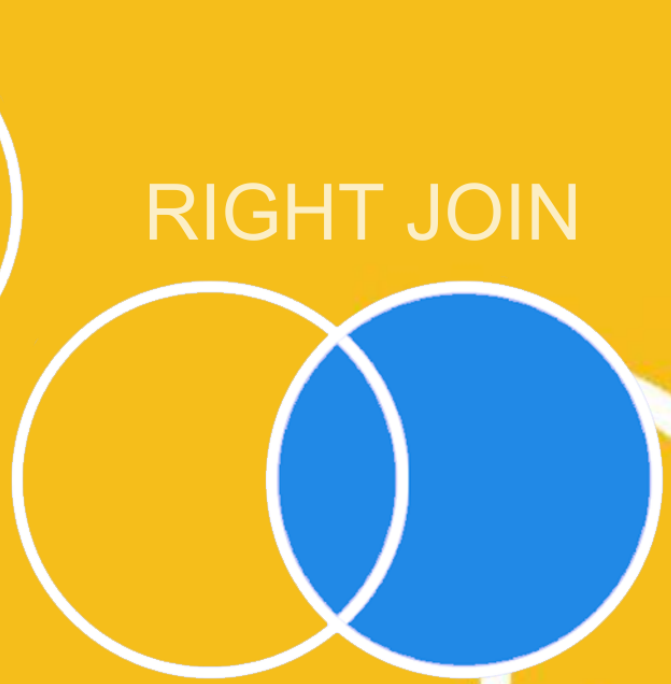
MySQL միացումները(JOINS) հնարավորություն են տալիս մի քանի աղյուսակներից ստանալ ինֆորմացիա միաժամանակ, այսինքն մի SELECT հրամանով ստանալ անհրաժեշտ տվյալները երկու և ավելի աղյուսակներից:

Կախված պահանջվող արդյունքից MySQL-ը թույլ է տալիս միացման 3 եղանակ՝

- **INNER JOIN,**
- **LEFT JOIN,**
- **RIGHT JOIN**







MySQL JOINS (MySQL միացումներ)

INNER JOIN

```
mysql> SELECT p.*, d.* FROM `products` p
       INNER JOIN `descriptions` d ON p.`id` = d .`id`;
```

< **ON** > հրամանը թույլ է տալիս աղյուսակները <կապել> իրար, այսինքն՝ հրամանը կատարվելիս աղյուսակները կհամեմատվի ըստ այդ պայմանի:

INNER JOIN-ին համարժեք են նաև < **CROSS JOIN** >-ը, < **JOIN** >-ը և ստորակետով միացումը:

INNER JOIN

<**ON**>-ի փոխարեն կարող ենք օգտագործել նաև < **USING()** > ֆունկցիան, միայն, եթե երկու աղյուսակների համեմատվող սյուների անունները նույնն են, հակառակ դեպքում օգտագործվում է ON-ը:

```
mysql>  SELECT * FROM `products` JOIN `descriptions` USING(`id`);
```

LEFT **JOIN**

LEFT JOIN-ը հնարավորություն է տալիս ստանալ տվյալներ աղյուսակից և այն լրացնել այլ աղյուսակի տվյալներով:

```
mysql>  SELECT * FROM `products` LEFT JOIN `descriptions` USING(`id`);
```

Ի տարբերություն < **INNER JOIN** >–ի, եթե աղյուսակին կապած մյուս աղյուսակում համապատասխան տողը բացակայում է, ապա այդ տողի բացակայող արժեքի փոխարեն կվերադարձնի NULL արժեքը

RIGHT JOIN

RIGHT JOIN-ը գործնականում չի տարբերվում LEFT JOIN-ից, բացառությամբ այն բանի, որ տվյալները վերցվում են երկրորդ աղյուսակից (JOIN-ից հետո գրված) և համեմատվում առաջին աղյուսակի հետ:

```
mysql>  SELECT * FROM `products` RIGHT JOIN `descriptions` USING(`id`);
```

Նաև հնարավոր է նաև միմյանց միացնել 2-ից ավել աղյուսակներ

```
mysql> SELECT `products`.`product_name`,
       `descriptions`.`product_description`,
       `sale`.`sum`
       FROM `products`
            INNER JOIN `descriptions` USING (`id`)
             LEFT JOIN `sale` ON `sale`.`p_id` = `products`.`id`
Query OK, 0 rows affected (0.09 sec)
```

Self JOIN

<Self JOIN >-ը օգտագործվում է այն դեպքում երբ անհրաժեշտ է աղյուսակը կապել ինքն իրեն

```
mysql>  SELECT A.CustomerName AS cm1,
        B.CustomerName AS cm2,  A.City as c
                FROM Customers  A, Customers  B
        WHERE A.CustomerID <> B.CustomerID
        AND A.City = B.City ;
```

UNION

< **UNION** >-ը օտգագործվում է մի քանի աղյուսակներից ստացված արդյունքը մի տեղ միավորելու համար

```
mysql>  SELECT  name FROM customers
            UNION
        SELECT name FROM users;
```

* Կարևեր է, որ ստացվու արդյունքների սյուների անունները, քանակությունը և հեևթականությունը նունը լինեն

```
mysql>  SELECT  last_name as name FROM customers
            UNION
        SELECT name FROM users;
```

Աղյուսակների տեսակները

- **MyISAM**
- **InnoDB**
- MARGE
- MEMORY
- BDB(BerkeleyDB)
- EXAMPLE
- FEDERATED
- ARCHIVE
- BLACKHOLE
- CSV

FOREIGN KEY

< **FOREIGN KEY** >-ը օգտագործվում է երկու աղյուսակների դաշտեր միյանց կապելու համար

< **FOREIGN KEY** >-ով կապվում է աղյուսակի մեկ դաշտը մյուս աղյուսակի < **PRIMARY KEY** >-ով դաշտի հետ

Պարտադիր է որ երկու դաշտերն էլ լինեն միանման (տիպը, չափը. . .)

```
mysql>  CREATE TABLE `users` (
        `id` INT(11) AUTO_INCREMENT,
        `first_name` VARCHAR(255),
        `last_name` VARCHAR(255),
        `birthday` DATE,
        `phone_number` INT(11),
        `e_mail` VARCHAR(255),
        `city_id` INT UNSIGNED,
        PRIMARY KEY (`id`),
        FOREIGN KEY (`city_id`) REFERENCES `cities`(`id`)
)ENGINE=INNODB CHARSET=utf8;
```

< **FOREIGN KEY** >-ով աղյուսակները միմյանց հետ կապելիս կարելի է նշել նաև կապվող աղյուսակի համապատասխան տողի փոփոխության կամ ջնջվելու դեպքում ինչ պետք է տեղի ունենա

Հնարավոր տարբերակները՝

- Cascade –
- No action –
- Set null –
- Restrict -

```
mysql>  USE `blog`;
        CREATE TABLE `users` (
         `id` INT(11) AUTO_INCREMENT,
         `first_name` VARCHAR(255),
         `last_name` VARCHAR(255),
         `birthday` DATE,
         `phone_number` INT(11),
         `e_mail` VARCHAR(255),
         `city_id` INT unsigned,
         PRIMARY KEY (`id`)
        FOREIGN KEY (`city_id`) REFERENCES `cities`(`id`) ON UPDATE CASCADE ON DELETE SET NULL
)ENGINE=INNODB CHARSET=utf8;
```

Աղյուսակի կրկնօրինակում

```
mysql > CREATE TABLE `newtable` LIKE `oldtable`;
mysql > INSERT `newtable` SELECT * FROM `oldtable`;
```

```
mysql > CREATE TABLE `newtable` AS SELECT * FROM ` oldtable `;
```

Mysql պայմաններ

```
mysql > select id, ( CASE
        WHEN qty_1 <= '23' THEN price
        WHEN '23' > qty_1 && qty_2 <= '23' THEN price_2
        WHEN '23' > qty_2 && qty_3 <= '23' THEN price_3
        WHEN '23' > qty_3 THEN price_4
        ELSE 1 END)
    AS total from product;
```

Mysql պայմաններ

```
mysql > SELECT col1, col2, IF( action = 2 AND state = 0, 1, 0 ) AS state from tbl1;
```

```
mysql > SELECT IF( id = 1,  99) AS state from tbl1;
```

MySQL ընթացակարգեր (procedures) և ֆունկցիաներ (functions)

CREATE PROCEDURE	Ընթացակարգի ստեղծում
CREATE FUNCTION	Ֆունկցիայի ստեղծում
ALTER PROCEDURE	Ընթացակարգի փոփոխում
ALTER FUNCTION	Ֆունկցիայի փոփոխում
DROP PROCEDURE	Ընթացակարգի ջնջում
DROP FUNCTION	Ֆունկցիայի ջնջում
CALL proc_name()	proc_name ընթացակարգի կանչ
SELECT func_name()	func_name() ֆունկցիայի կանչ
DECLARE	Լոկալ փոփոխականի հայտարարում
SET	Լոկալ կամ գլոբալ փոփոխականի արժեքի փոփոխում
SELECT ... INTO	Նշված սյունակի արժեքի պահում փոփոխականի մեջ
RETURNS	Ֆունկցիայի կողմից վերադարձվող արժեքներ

Functions

Ֆունկցիայի ստեղծելը

```
mysql > CREATE FUNCTION hello (str CHAR(20))
        RETURNS CHAR(50)
        RETURN CONCAT('Hello, ', str, '! ' );
```

Ֆունկցիայի օգտագործումը

```
mysql > SELECT hello('World');
```

Procedures

```
mysql >
DELIMITER //
    CREATE PROCEDURE `insert_proc`(IN f_name VARCHAR(255), IN l_name VARCHAR(255),
    IN b_day CHAR(10), IN phone INT(111), IN e_mail VARCHAR(255), IN city_id INT(10))
    BEGIN
            DECLARE count_users INT DEFAULT 0;
            INSERT INTO `users` (`first_name`,`last_name`,`birthday`, `phone_number`, `e_mail`, city_id)
            VALUES (f_name, l_name, b_day, phone, e_mail, city_id);
            SELECT COUNT(`id`) INTO count_users FROM `users`;
            SET @count = count_users;
    END//
DELIMITER ;
```

Procedures

```
mysql > CALL insert_proc('Հայկ', 'Հակոբյան', '2017-09-14', '65656565', 'hayk@gmail.com', '55');
```

- mysql > SELECT @count;