# Data base

* What is sql ?

Stands for structured query language.

* What is dbms ?

Stands for database management system

* What is rdbms ?

Stands for relational database management system

# Mysql

* CREATE DATABASE `databaseName`; (CREATE DATABASE IF NOT EXISTS `databaseName`)
* DROP DATABASE `databaseName`; (DROP DATABASE IF EXISTS `databaseName`)
* CREATE TABLE `tableName` (  
  `columnName` DataType(length) Attributes Defaults Constraints,

)

* DROP TABLE `tableName`
* ALTER TABLE `tableName` ADD `columnName` DataType(length) Attributes Defaults Constraints
* ALTER TABLE `tableName` DROP `columnName`
* ALTER TABLE `tableName` CHANGE `oldColumnName` `newColumnName` DataType(length) Attributes Defaults Constraints
* ALTER TABLE `tableName` ADD CONSTRAINT `constraintName` FOREIGN KEY(`foreignKeyColumn`) REFERENCES `referenceTableName`(`primaryKey`) ON DELETE RESTRICT ON UPDATE CASCADE
* INSERT INTO `tableName` (`columnName1`,`columnName2`,…) VALUES (‘value1’,’value2’,…)  
  INSERT INTO `tableName` VALUES (value1,’value2’,….), (value1,’value2’,….)
* UPDATE `tableName` SET `columnName` = ‘value1’  
  UPDATE `tableName` SET `columnName` = ‘value1’ WHERE condition
* DELETE FROM `tableName`  
  DELETE FROM `tableName` WHERE conditon
* TRUNCATE TABLE `tableName`

## DAY2

* SELECT `columnName` AS `newColumnName`,`columnName2` FROM `tableName`
* SELECT \* FROM `tableName` WHERE condition
* Operators (athematic , logical , conditional)
* Built in functions
* WHERE `columnName` BETWEEN min AND max
* WHERE `columnName` IN(value1,’value2’,…)
* WHERE `columnName` IS NULL | IS NOT NULL
* WHERE `columnName` LIKE ‘%string\_’
* Aggregates functions (MAX,MIN,COUNT,AVG,SUM,GROUP\_CONCAT)
* GROUP BY `columnName`
* ORDER BY `columnName` ASC|DESC , `columnName2` ASC|DESC , …
* HAVING condition on alias or aggregate
* LIMIT number,offset
* JOIN (INNER-LEFT-RIGHT)
* JOIN `tableName` ON `parent`.`primary` = `child`.`foreign`
* CREATE VIEW `viewName` AS (SELECT Query)
* DROP VIEW `viewName`
* Examples:

1. SELECT

`id` ,

CONCAT(`first\_name`, ' ', `last\_name`) AS `full\_name`,

((`salary` + `bonus`) \* 2) / 0.5 AS `salary`

FROM

`users`

WHERE `gender` = 'f'

1. SELECT \* FROM `users` WHERE `id` >= 1 AND `id` <= 10;
2. SELECT \* FROM `users` WHERE `bonus` NOT BETWEEN 0 AND 100;
3. SELECT \* FROM `users` WHERE `bonus` > 100 OR `bonus` < 0 ;
4. SELECT \* FROM `users` WHERE `bonus` IN(100,200,300) ;
5. SELECT \* FROM `users` WHERE `bonus` = 100 OR `bonus` = 200 OR `bonus`= 300;
6. SELECT

MAX(`bonus`) AS `max\_bonus`,

MIN(`salary`) AS `min\_salary`,

SUM(`bonus`) AS `total\_bonus`,

# SUM(`bonus`) / COUNT(`id`) AS `avg\_bonus`

AVG(`bonus`) AS `avg\_bonus`

FROM

`users`

1. SELECT `gender` , AVG(`bonus`) as `BONUS\_AVG`

FROM `users`

GROUP BY `gender`

1. SELECT `gender` , COUNT(`gender`) AS `number\_users`

FROM `users`

GROUP BY `gender`

1. SELECT `last\_name` , COUNT(`id`) AS `number\_users`

FROM `users`

GROUP BY `last\_name`

HAVING `number\_users` > 1

ORDER BY `number\_users` ASC , `last\_name` ASC

1. SELECT \* FROM `users` ORDER BY `created\_at` DESC LIMIT 1
2. SELECT

`categories`.`name\_en` AS `category\_name\_en`,

`subcategories`.`name\_en` AS `subcategory\_name\_en`

FROM `subcategories`

LEFT JOIN `categories`

ON `categories`.`id` = `subcategories`.`category\_id`

1. SELECT

CONCAT(`users`.`first\_name` , ' ' , `users`.`last\_name`) AS `full\_name`,

COUNT(`carts`.`product\_id`) AS `number\_of\_products`

FROM

`users`

LEFT JOIN `carts`

ON `users`.`id` = `carts`.`user\_id`

GROUP BY `users`.`id`

ORDER BY `number\_of\_products` DESC , `full\_name` ASC

1. SELECT

`products`.\*,

`brands`.`name\_en` AS `brand\_name\_en`,

`subcategories`.`name\_en` AS `subcategory\_name\_en`,

`categories`.`id` AS `category\_id`,

`categories`.`name\_en` AS `category\_name\_en`

FROM

`products`

JOIN `brands`

ON `products`.`brand\_id` = `brands`.`id`

JOIN `subcategories`

ON `subcategories`.`id` = `products`.`subcategory\_id`

JOIN `categories`

ON `categories`.`id` = `subcategories`.`category\_id`

1. Ex::

**SELECT** `tableName`.`columnName` , aggregate() AS `newName`

**FROM** `tableName`

**JOIN** `tableName1**`**

**ON `child`.**`forgeinKey` = `parent`.`primaryKey`

**WHERE** condition

**GROUP BY** `columnName`

**HAVING** condition

**ORDER BY** `tableName`.`columnName` ASC | DESC , `tableName`.`columnName1` ASC | DESC

**LIMIT** value,offset

1. **Search on :** UNION , UNION ALL , GROUP\_CONCAT, limit value,offset , builtin functions , triggers , events , subqueries , join with subqueries , IF conditon , WHEN CASE , SELECT DISTINCT