**My sql Commands:**

-> **first commands** :

**USE** database\_name; -> to use the specified database

**SHOW DATABASES** ; -> to show all databases you have on your pc

**DROP** database\_name; -> to drop the specified database

-> **Operations On Table** :

**CREATE TABLE IF NOT EXISTS** table\_name( here we write its FEATURES ) ;

-> to define and create new table

**SHOW TABLES** ;

-> to show all tables you have

**DESCRIBE** table\_name ;

-> to see the fields of the table

**SHOW COLUMNS FROM** table\_name ;

-> to see the fields of the table

**SHOW TABLE STATUS**;

-> to see the status of the table

**SHOW CREATE TABLE** ;

-> to see the code of the table created

**RENAME TABLE** table\_name **TO** new\_name ;

-> to rename the table

-> **ALter table** :

**ALTER TABLE** table\_name **ENGINE** = engine\_name ;

-> to change the engine name

different engines : CVS , MRG\_MYISAM , MEMORY , ANIA , MYISAM , SEQUENCE , INNODB , PERFORMANCE\_SHEMA

**ALTER TABLE** tab\_name **ADD** col\_name **AFTER** col\_name;

-> to add new field after another field , ! if we don't write the word after automatically will add the field in end of the table

**ALTER TABLE** tab\_name **ADD** col\_name **FIRST**;

-> to add the field as first field on the table

**ALTER TABLE** tab\_name **DROP** col\_name ;

-> to drop or delete the column specified

**ALTER TABLE** tab\_name **CHANGE** col\_name **NEW**\_col\_name TYPE ;

-> to move and change the name of a field we have

**ALTER TABLE** tab\_name col\_name New\_col\_name **NEW**\_data type;

-> to change the name or the type of a field we have

**ALTER TABLE** tab\_name **MODIFY** col\_name **NEW**\_DATA\_TYPE ;

-> to change the type of the the field

**ALTER TABLE** tab\_name **CONVERT TO CHARACTERS SET** name\_of\_the\_set ;

-> to change the type of the group of the characters

-> **Constraint table** :

**- NOT NULL , UNIQUE**

**CREATE TABLE** tab\_name (id **NOT NULL UNIQUE**) ;

-> to create table , **NOT NULL** = the values shouldn’t accept the null , **UNIQUE** = the value shouldn’t repeated

**ALTER TABLE** tab\_name **DROP** **INDEX** unique\_field\_name ;

-> to delete the unique field in the table

- **PRIMARY KEY** :

**CREATE TABLE** tab\_name (id **INT NOT NULL PRIMARY KEY**) ;

-> as unique , but primary key it used as the key of the table , 1 table must be as primary key , no greather than 1 field

- **FOREIGN KEY** :

**CREATE TABLE** tab\_name (

details ,

**FOREIGN KEY** (field\_name) **REFERENCES** tab\_name (field\_name)

);

-> the foreign key it used to make a relation between 2 tables or more

- **UPDATE , DELETE** :

**ALTER TABLE** tab\_name **CONSTRAINT** cons\_name **FORIGN** **KEY** (filed\_name) **REFERENCES** tab\_name(field)

**ON UPDATE SET NULL** , **ON DELETE** **SET NULL**

-> if we delete the data of user from the parent\_table , it will be rest in the child\_table

- we can alse use this method in the update and delete (**NO ACTION** , **RESTRINCT**)

-> **String functions** :

**LEFT** (string, length) -> to get the characters from the left

**SELECT** \* , **LEFT**(string , length) **FROM** tab\_name ;

**RIGHT**(string , length) -> to get the characters from the right

**SELECT** \* , **RIGHT**(string, length) **FROM** tab\_name ;

**MID**(string, position , length) -> to get the characters from the middle

**SELECT** \* , **MID**(string, position , length) **AS** field\_name **FROM** tab\_name ;

**AS** -> it used to name the field with the **MID** function

**LENGTH**(string) -> to get the number of characters on the field

**SELECT** field , **LENGTH**(field) **FROM** tab\_name ;

**CHAR\_LENGTH**(string) -> to get the number of characters , it’s more used than LENGTH() function

**LCASE** = **LOWER**(string) -> to set all the field with lower characters

**UPDATE** tab\_name **SET** field\_name = **LOWER**(field\_name) ;

**UCASE** = **UPPER**(string) -> to set all the field with upper characters

**UPDATE** tab\_name **SET** field\_name = **UCASE**(field\_name) ;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**REPEAT**(string, number o repeats) -> the repeat the string

**SELECT** \* , **REPEAT**(field , 2) **FROM** tab\_name ;

**REPLACE**(string, string from , string to) -> to replace the characters with others

**UPDATE** tab\_name **SET** field= **REPLCAE**(string , from , to) ;

**REVERSE**(string) -> to reverse the string

**UPDATE** tab\_name **SET** field = **REVERSE**(field\_name) ;

**CONCAT**(str , str , str....) -> to concatenate strings

**SELECT** \* , **CONCAT**(‘hello’ , filed\_name ) **FROM** tab\_name ;

**CONCAT\_WS**(separator , str , str ,str...) -> to concate with separator

**UPDATE** tab\_name **SET** field = **CONCAT**\_**WS**(‘@’ , field\_name , ‘helo’ ) ;

**INSERT** (string, position , length , new\_string) -> to insert into field , it’s different than the insert command

SELECT \* , INSERT(field , 2, 7 , ‘medo’ ) FROM tab\_name ;

**TRIM**(leading |trailing |both **REMOVE**\_**STRING FROM** string) -> to remove characters

**SELECT** \* , **TRIM**(leading ‘me’ **FROM** field\_name) **FROM** tab\_name ;

- methods optional if not written , BOTH will used

- remove string if not written SPACE will removed

**LTRIM**(text ) -> to remove characters from left

**SELECT** \* , **LTRIM**(text) **FROM** tab\_name;

**RTRIM**(text) -> to remove characters from right

**SELECT** \*, **RTRIM**(text) **FROM** tab\_name;

-> **Numeric functions** :

**CEIL**(number) -> approach the number DESCENDING

**UPDATE** tab\_name **SET** field = **CEIL**(number) ;

**FLOOR**(number) -> approach the number ASCENDING

**SELECT** \*, **FLOOR**(number) **FROM** tab\_name ;

**ROUND**(number , decimal) -> approach the number with the good number exactly

**UPDATE** tab\_name **SET** field = **ROUND**(number , 2 ) ;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**TRUNCATE**(number , decimal) -> to get the number with the number of digits after comma

**SELECT** \* , **TRUNCATE**(number , 3) **FROM** tab\_name ;

**POWER**(number , powered) -> to multiple the number with his power

**SELECT** \* , **POWER**(number , 2) **FROM** tab\_name ;

**MOD**(number , modules ) -> to get the modules

**UPDATE** tab\_name **SET** field= MOD(number , 3) ;

-> **Date functions** :

**CURTIME**() , **CURRENT\_TIME**() , **CURRENT\_TIME** -> to get the current time (h : m :s )

**UPDATE** tab\_name **SET** field = **CURRENT**\_**TIME**() ;

**CURDATE**() , **CURRENT\_DATE**() , **CURRENT\_DATE** -> to get the current date (y - m - d )

SELECT **CURRENT\_DATE**() ;

**NOW**() , **CURRENT\_TIMESTAMP**() -> to get the the current date and time ( y - m - d h : m : s )

SELECT **NOW**() ;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**DAYNAME**(date) ; -> to get the day name o the date

**SELECT DAY\_NAME**(‘2022 - 6 -9’) ;

**DAYOFWEEK**(date) -> to get the day of the week

**SELECT DAYOFWEEK**(‘2022 - 6 -9’ ) ;

**DAYOFMONTH**(date) -> to get the date of the month

**SELECT DAYOFMONTH**(‘2022 - 6 -9’) ;

**DAYOFYEAR**(date) -> to get the day of the year

**SELECT DAYOFYEAR**(‘2022 - 6 -9’) ;

**MONTH**(date ) -> to get the month

**SELECT MONTH**(‘2022 - 6 -9’) ;

**MONTHNAME**(date) -> to get the name of the month

**SELECT MONTHNAME**(‘2022 - 6 -9’) ;

**HOUR**(time) -> to get the hour

**SELECT HOUR**(‘23:45:21’) ;

**MINUTE**(time) -> to get the minute

**SELECT MINUTE**(‘23:45:21)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**DATEDIFF**(date2 , date2) -> to compare days and return the number

**SELECT** \* , **DATEDIFF**(curdate() , date ) **FROM** tab\_name ;

**LAST**\_**DAY**(date) -> to get the last day of the date

**SELECT** \* , **LAST**\_**DAY**(date) **FROM** tab\_name ;

**DATE**\_**ADD**(date , **INTERVAL** expressing **UNITE**\_**TIME** ) -> to add time

**UPDATE** tab **SET** filed = **DATE**\_**ADD**(date , **INTERVAL** 10 **DAY**) ;

**DAY**\_**SUB**(date , **INTERVAL** expression **UNITE**\_**TIME**) -> to sub days

**UPDATE** tab **SET** field = **DAY**\_**SUB**(date , **INTERVEL** 4 MONTH) ;

-> **Comparaison functions** :

- **BETWEEN** , **AND** | **NOT BETWEEN** , **AND**

**BETWEEN** expression **AND** expression

**SELECT** id **FROM** tab\_name **WHERE** id **BETWEEN** 1 **AND** 4 ;

**NOT BETWEEN** expression **AND** expression

**SELECT** id **FROM** tab\_name **WHERE** id NOT **BETWEEN** 1 **AND** 3 ;

- **IN** | **NOT IN**

**SELECT** \* **FROM** tab\_name **WHERE** field **IN** (values)

**SELECT** \* **FROM** tab\_name **WHERE** field **NOT IN** (values)

- **LIKE | NOT LIKE** :

**LIKE** :

- **%** : [ **EMPTY OR COLLECTION CHARACTERS** ] -> we van use this in first , middle or end

- **\_** : [ **ONE CHARACTER** ] -> the same , we can use it in first , middle or end

-> **ALSEO** we **CAN** use **THEM** all **IN** the same **COMMAND**

**SELECT** \* **FROM** tab\_name **WHERE** field **LIKE** '**%**values'

**SELECT** \* **FROM** tab\_name **WHERE** field **LIKE** '**\_**values'

**SELECT** \* **FROM** tab\_name **WHERE** field **LIKE** '**%**values**\_%**'

**NOT LIKE** : -> the same , it's just the opposite . ! Also we can use with it the % and \_

-> **COMPARISON OPERATORS :**

**=** [ EQUAL] | **!=** or **<>** [ NOT EQUAL]

**>=** [ GREATHER TAHN OR EQUAL ]

**<=** [ LESS THAN OR EQUAL ]

--------------------------------------------------------------------------------------------------------------------------------------

-> **LOGICAL OPERATORS** :

**AND** -> **&&** [ condition + condition ]

SELECT \* FROM tab\_name WHERE field condition AND field condition ;

**NOT** -> **!** [ negative ]

**SELECT** \* **FROM** tab\_name **WHERE NOT** field **CONDITION** ;

**||** -> **OR** [ A condition OR B condition ] -> one of the 2 condition must verified , if the 2 are verifies its ok

**SELECT** \* **FROM** tab\_name **WHERE** field condition **OR** field **CONDITION** ;

**XOR** [ A condition AND not B condition ] -> should one condition must be verified

**SELECT** \* **FROM** tab\_name **WHERE** field **condition XOR** field **condition** ;

-> **CONTROL FLOW FUNCTIONS if** :

**IF** ( **CONDITION** , **TRUE** , **FALSE** )

**SELECT** id , name , **IF** (number < 1 , 'HARD LUCK' , 'CONGRATS' ) **FROM** users;

**UPDATE** users **SET** number = **IF** (number < 1 , number + 10 , number ) ;

-> **CONTROL FLOW FUNCTIONS case** :

**FIRST syntax** of **CASE** :

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**CASE SELECT** id , name ,

**WHEN** expression = value **THEN** result **CASE**

**WHEN** expression = value **THEN** result WHEN number = 10 THEN "not bad" ,

**WHEN** expression = value **THEN** result WHEN number = 15 THEN "good"

**ELSE** result ELSE "unkown"

**END END**

--------------------------------------------------------------------------------------------------

**SECOND syntax** of **CASE** :

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**CASE** expression **CASE** number

**WHEN** value **THEN** result **WHEN** 10 **THEN** "not bad" ,

**WHEN** value **THEN** result **WHEN** 15 **THEN** 15+ 15

**WHEN** value **THEN** result

**ELSE** result **ELSE** number

**END END**

-> **ARITHMETIC OPERATIONS** :

**+** : **ADDITION** | **-** : **MINUS** | **/** : **DEVISION** | **\*** : **MULTIPLICATION** | **%** : **MODULOS**

**SELECT** name ,

days\_work ,

Days\_Salary ,

( days **\*** day\_Salary) AS result\_money ,

( days **+** day\_Salary ) **-** ( 100 **%** 8) AS full\_result

**FROM** users;

-> **INFORMATION FUNCTIONS** :

**USER**() , **SESSION\_USER(**) , **SYSTEM\_USER**() -> return the name of the user and the name of the local host

**VERSION**() -> return the name and the version of the system management that you works on it

**CHAR\_SET**(name of the field or the function ) -> return the engine

**DATABASE**() -> to know the name of the database

**CONNECTION\_ID** () -> the id connector with database

-> **GROUP , ORDER BY and HAVING** :

**SELECT** \* **FROM** tab **ORDER BY** field\_name , -> ordering the data by specific field

**SELECT** \* **FROM** tab **GROUP BY** field -> grouping data with specific field

**SELECT** \* , **SUM**(field\_name) **FROM** tab\_name -> get the sum of values on the table

**SELECT** \* , **COUNT**(field\_name) **FROM** tab\_name -> count the data

**SELECT** field **FROM** tab\_name **GROUP BY** field HAVING condition -> select by the condition

-> **STIMULATION OF JOIN** :

we use **JOIN** when we have two tables an we want to select from , we can do what **JOIN** do just from select **BY** :

**SELECT** \* **FROM** tab , tab2 **WHERE** tab1.field = tab2.field ;

-> **ALIAS IN DEEP** :

**SELECT** tab\_name.field **FROM** tab1, tab2

-> we use this command to specifies the field from the table what we want to select from if we have 2 tables have the same field

**SELECT**

**u**.user\_id ,

**u**.user\_name ,

**b**.barber\_name

**FROM**

tab\_name ; **u** or **b** THEY ARE **alias** to WIN TIME and HARD work

-> **INNER JOIN** :

syntax :

**SELECT**

column

**FROM**

table1

**INNER JOIN**

Table2

**ON** -> we can replace **ON** by **USING** and write the FIELD like this (field\_name)

tabl1.col = tab2.col

**exemple** :

**SELECT**

s.id ,

c.course\_id

c.course\_name

**FROM**

students s

**INNER JOIN**

courses c

**ON**

c.course\_id = s.course\_id

- **LEFT , RIGHT JOIN** :

their syntax is like the **INNER JOIN** , just we need o replace **INNER** by **LEFT** or **RIGHT**