(Free) My SQL Full Course

SQL → Structured Query Language.

DBMS → Database Management System.

Chapters:

1. Databases (create, delete)
2. Tables.
3. Insert rows
4. Select
5. update & delete
6. Auto commit, commit, roll back
7. current_date(), current_time(), now()
8. unique
9. not null
10. check
11. default
12. primary keys
13. auto_increment
14. foreign keys
15. joins
16. functions
17. AND, OR, NOT
18. wild cards
19. order by
20. limit
21. unions
22. self joins
23. views
24. Indexes.
25. subqueries
26. group by
27. roll up
28. on delete
29. stored procedures
30. triggers.

create database <database_name>;

The above command is used to create a database.

---

use <database_name>;

used to change the current database to that of <database_name>.

---

drop <database_name>;

used to delete the database

---

alter database <db_name> read only = 1;

makes the database read only - can't modify the existing data but can be fetched from. Also cannot be deted. To change re run the command using = 0.

---

create table <table_name> (

// fields with their datatypes

);

used to create a table

---

select * from <table_name>;

used to display the data stored in the table.

rename table <before> to <after>;

changes the table name from <before> to <after>

---

drop table <table_name>;

deletes the table from the database.

---

alter table <table_name> add col_name data_type;

used to add a new column to the table.

---

alter table <table_name> rename column <before>

to <after>;

used to change the column name from <before>

to <after>.

---

alter table <tablename> modify column columnn

<new_ datatype>;

changes the datatype of the named column.

---

alter table <table_name > modify column_name

datatype after <column-name>;

changes the position of the column in the

table. To get it first replace after with first.

---

alter table <table_name> drop column <column_name>

deletes or removes the column from the

table.

insert into <table_name> values (// values in order);

adds the data into the table. To add multiple values
add multiple ( ).

insert into <table_name> (column_names) values (values);

used to add data to only few columns in the table.

---

desc <table_name>;

used to display the structure of the data.

---

select col_name1, col_name2 from <table_name>;

displays the data stored in both the columns from the table only.

---

select * from <table_name> where <condition>;

displays the data matching the condition only.

The condition clauses can be

= → equal to

!= → not equal to

> → greater than

< → not greater than

>= → greater than or equal to

<= → lesser than or equal to

is NULL → will display the fields with no data.

is NOT NULL → will display data with fields only with data that is present.

---

update <table_name> set column_name = value where <condition>;
changes the data of the table for the column matching the condition(s).

---

delete from <table_name>;
removes all the data present in the table.

---

delete from <table_name where <condition>;
conditionally removes the data from the table.

---

set autocommit = off;
will not change the data or will not save the operations into the database automatically.

---

commit;
will save the data / sync the data into the database when autocommit is off.

---

rollback;
returns to the previously commited state of the database.

---

current_date() → saves the current date as a value

current_time() → saves the current time as a value.

now() → saves both the date & time as a value.

---

unique → this keyword with any column name and datatype makes the column value in the table unique to be saved.

---

alter table <table_name> add constraint unique (column_name);

makes the values or add the unique constraint for the particular column in the table.

NOT NULL → constraint added with the colname and datatype used to specify that the value of the column must be added & can't be left empty

---

check (condition) returns true or false value to be stored or executed. This check condition is validated i.e, the data is entered only if the condition is not violated.

Default value → saves the default value when no value is added.

Primary key → unique identifies in the table.
The table can have only one primary key.
The primary key value can not be NULL.
There can be only one primary key for a table

Autoincrement → constraint that is like index.

---

foreign key → used to connect another table.

Syntax as in adding column,

foreign key (column_name) references <tablename(column_name)>

foreign key is used to link two tables.

Joins is a clause used to combine rows from two or more tables.

---

select * from <table 1> inner join <table 2> on condition;

displays the common rows that meets the conditions.

for left joins it displays all rows from <table1>
for right joint it displays all rows from <table2>

functions in sql are short programs that can pass parameters and return a value.

count() → returns the count of the rows matching the condition.

max() → returns the maximum of the rows matching the condition.

as → can show different names for column name without changing while displaying.

avg() → returns average

min() → returns minimum

sum() → returns the accumulated sum.

concat() → can add/concat different col name values. with n number of parameters

---

Logical operators.

AND → used to match both the conditions

OR → used to match either the conditions

NOT → used to match the negate of the condition.

Between → used to define range.

in → present in the array.

wild card characters → %, _

used to substitute one or more characters in a string.

% → represents any number of wild card characters but is accepted only when the condition is (LIKE)

_ → used to represent one wild card character.

---

order by → sorting while displaying based on the particular column, asc or desc.

Default → asc.

LIMIT clause is used to limit the number of records. Useful if working with a lot of data. Can be used to display a large data or pages.

select * from <table_name> limit number;

displays only the number of values.

limit number, number → offset.

UNION combines the results of two or more select statements.

UNION works only when both the tables have same number of columns while displaying. Can select specific number of columns too.

self join

Join another copy of a table to itself
used to compare rows of the same table
helps to display a hierarchy of data.

update <column_name> set column_name = value
<where condition>;
changes the field value of the condition
met column. in the database table.

We can also join the same table.

Views

Virtual table based on the result-set of an SQL statement.
The fields in a view are fields from one or more real tables in the database.
They are not real tables, but can be interacted with as if they were

Syntax: create view view_name ~~col_names~~ as
select column_names from table_name;

creates a view ~~that~~ is not saved as a table but is saved as a temporary view derived from a specific real table.

Indexes (BTree data structure)

Indexes are used to find values within a specific column more quickly.

MySQL normally searches sequentially through a column. The bigger the column, the more expensive the operation is.

The UPDATE takes more time, SELECT takes less time.

---

show indexes from <table_name>;

shows all the indexes from the table. The fastest way to access any table data.

---

create index index_name on table_name (column_name)

creates an index for the table using the column

Subquery

a query within a query.

query written as query (sub query).

Distinct → unique while displaying (constraint)

Group by

aggregate all rows by a specific column

often used with aggregate functions. as in

SUM(), MAX(), MIN(), AVG(), COUNT().

ROLLUP, extension of the GROUP BY clause.

Produces another row & shows the GRAND TOTAL (supersagged value)

The [with ROLLUP] will return the grand total below all the rows.

ON DELETE SET NULL = When a Foreign key is deleted, replace foreign key with NULL

ON DELETE CASCADE = When a Foreign key is deleted, delete row.

ON DELETE clause to be added when creating the table or to be added in the structure.

Stored procedure: is a prepared SQL code that can save great if there is a query that written after. Procedure is created as.

CREATE procedure () begin ⟷ end;

Delimiter is changed i.e the call for execution

The stored procedure is used as call procedure

Trigger: When an event happens, do something. checks data, handles errors, auditing tablog,

create trigger trigger_name

before/after ~~condition~~ create/update/delete on table_name for each row

statements;