MySQL is RDBMS program that use SQL offed by Oracle.

JMS

Unit-V MySQL and SQL

The SQL stands for Structured Query Language. It is non-procedural database computer language for retrieving and managing data in relational database.

The SQL ~~begin~~started in IBM Laboratory in California and was developed in late 1970. It become ANSI (American National Standards Institute) in 1986 and ISO (International Standard Organisation) in 1987. The initial name was 'SEQUEL' (Structured English Query Language) and later named to SQL.

It is called structured query language because it organises the data in structured way like a table so that the data becomes easy to retrieve and manipulate.

We use SQL for CRUD operation:-

i> CREATE:- To create database, tables etc.
ii> READ:- To read data present in database.
iii> UPDATE:- To modify already inserted data.
iv> DELETE:- To delete database, tables etc.

It is comprehensive language for controlling and interacting with DBMS.

Advantages of SQL

i> Standard Independent Language :- ANSI and ISO established universal rules so, it is open-standard language.

ii> Speed :- It is fast language so database made easily and fastly.

Disad.
i Complex Query
ii Scalability
iii Limited for unstructured data
iv Expensive
v Slow performance
vi. Concurrency

iii> Easy to learn and Use :- It resembles simple english sentences so it makes simple, easy to learn and use.

iv> Less Programming :- SQL allows organisation and manipulation of data with less programming.

v> Data Integrity

Dr. Codd's 12 Rules for Relational Model

Relational database model is widest used method to organised data. It is written by DR E.F Codd in 1970. He written 13 rules oddly known as Codd's 12 rule

0> Relational DBMS must be able to manage database.

i> Information Rule :- All the information in relational database is represented explicitly.
(clearly)

ii> Guaranteed Access:- Every value in relational database is guaranteed to access by using combination of table name, column name.

iii> Systematic Null value support:- The DBMS provides systematic support for null values, distinct from default values.

iv> Active, online relational catalog:- The description of database and its content represented at logical level as tables & then queried using DBMS language.

v> Comprehensive Data Sublanguage:- At least one language must have well-defined syntax and be comprehensive for access, manipulate.

vi> View updating Rule:- All the view are updated through system.

vii> Set level insertion, update & deletion:- It supports set level facilities.

viii> Physical Data Independance:- Application program are logically unaffected when physical access methods are changed.

ix> Logical Data Independance:- Application program are logically unaffected when changes made to table structure.

x) Integrity Independance :- The database language must be capable of defining integrity rules.

xi) Distribution Independance :- Application program are logically unaffected, when data is first distributed.

xii) Non sub-version :- It is not possible to bypass integrity rules through database language by using lower-level language.

SQL Data Types

Data types are used to represent the nature of data that can be stored in database table. There are mainly three types of datatypes for database :-

i) String data types
ii) Numeric Data types
iii) Date & Time Data types

i) MySQL string data types :-

1) Char(size) :- It is used to specify fixed length string and size ranges from 0 to 255 charaters & default is 1.

ii) Varchar(size) :- It is used to specify varible length string and size ranges from 0 to 65535 characters.

yyyy-MM-dd hh:mm:ss

Comments = (--) double hyphen

iii> Text(size) :- It holds a string that contains maximum length of 255 character.

iv> Binary(size) :- It is equal to char() but store binary type string.

v> Varbinary(size) :- It is equal to varchar() but store binary type string.

ii> MySQL numeric data types :-

i> Bit(size) :- It is used for bit-value type and its size is 1 to 64.

ii> Int(size) :- It is used for integer value and the size is 255.

iii> Integer(size) :- It is equal to int(size).

iv> float(size,d) :- It is used to specify a floating point number. The number of digit is specified by 'd' parameter.

iii> MySQL date & time data types :-

i> Date :- It is used to specify date format in YYYY-MM-DD.

ii> Datetime(fsp) :- It is used to specify date & time combination in YYYY-MM-DD hh:mm:ss.

iii> Time(fsp) :- It is used to specify time format in hh:mm:ss.

iv> Year :- It is used to specify a year in four-digit format YYYY.

SQL Commands/Components

i> Data Definition Language (DDL):-

The data definition language is a subset of SQL responsible for defining and managing the structure of database. DDL commands are used to create, modify and delete database like tables, indexes etc.

The commands are:

i>Create Table: It is used to create new table in database. It specifies the table name, column name, data types etc.

Example:

```
CREATE TABLE employee(id INT PRIMARY
KEY, name VARCHAR(50), salary DECIMAL
(10,2);
```

ii>Alter Table:- It is used to modify the structure of existing table.

Example:

```
ALTER TABLE employee
ADD COLUMN email
VARCHAR(100);
```

iii>Drop Table:- It is used to delete exiting table with its data and structure

Example:

```
DROP TABLE employees
```

iv> Create Index: It is used to create an index on one or more columns. It improves query performance.

Example:

```
CREATE INDEX idx_employee_name
ON employees(name);
```

v> Drop Index :- It is used to remove existing index from a table.

Example:-

```
DROP INDEX idx_employee_name;
```

vi> Trauncate Table :- It is used to delete the data inside a table but not table.

Syntax:

```
TRAUNCATE TABLE table_name
```

ii> Data Query/Retrieval language (DQL or DRL) :- It is a subset of SQL focus on retrieving data from database. The SELECT statement is foundation of DQL and allows to extract specific columns.

i> SELECT :- It is used to select data from a database:

syntax:

```
SELECT column1....., FROM table-name;
```

ii> Where :- It is used to filter records.

syntax:

```
SELECT column1....., FROM table-name WHERE condition
```

(=), (>), (<), (>=), (<=) (<>) ← Not equal to etc are operator.

in some version

iii> Distinct :- It removes duplicate rows from query result.

Syntax:

SELECT DISTINCT column-... , FROM table-name;

iv> In :- It filters result based on a list in the WHERE clause.

SELECT के साथ WHERE, GROUP BY के साथ HAVING — Same function

~~SELECT * FROM orders WHERE order_date BETWEEN '2023-01-01'~~

Example:

SELECT * FROM products WHERE category_id IN (1,2,3);

v> Between :- It filters result within a specified range in WHERE clause.

Example:

SELECT * FROM orders WHERE order_date BETWEEN '2023-01-01' AND '2023-06-30';

vi> Is Null :- It checks for Null values in WHERE clause.

Example:

SELECT * FROM customers WHERE email IS NULL;

vii> As :- It renames columns in query results

Example:

SELECT first_names AS "First Name", last_name AS "Last Name" FROM employees

Encompasso - अंतर्गत करें

viii) Order By :- It allows to sort result of a query based one one or more columns.

syntax:

```
SELECT column1... FROM table_name
    ORDER BY column1 [ASC|DESC];
```

ix) Group By :- It is used to group rows from a table onone or more column.

syntax:

```
SELECT column1, aggregate_function(column2)
    FROM table_name
    GROUP BY column1;
```

iii) Data Manipulation Language (DML):-

It encomepasses the commands that manipulate data within database. It allows to select, insert, update and delete records.

i) Select :- It allows to retrieve records from one or more table in database.

syntax:

```
SELECT columns FROM tables WHERE predial;
```

ii) Insert :- It is used to add new records.

syntax:

```
INSERT INTO employe VALUES value1...;
```

iii> Update:- It modifies existing records in table

Syntax:

```
UPDATE table-name SET column1= value1...
WHERE condition;
```

iv> DELETE:- It is used to delete records in table

Syntax:

```
DELETE FROM table-name WHERE condition;
```

iv> Data Control Language (DCL):- It focuses on manggement of access rights, permission and security related of database. It is used to control who can access the data, modify data etc within database.

i> GRANT:- It is used to provide specific permission to users. Permission can include ability to perform various action on tables.

Syntax:

```
GRANT privelege_type
  ON object_name
  TO user_name;
```

Examples:

```
GRANT select
  ON employees
  TO Analyst;
```

ii> REVOKE :- It is used to remove specific previlege that have been granted prevesuly.

syntax:

```
REVOKE privilege_name
  ON object_name
  FROM user_name;
```

Example:

```
REVOKE SELECT
  ON Employees
  FROM Analyst;
```

v> Transaction Control Language (TCL) :- It deals with management of transaction within a database. These commands are used to control the initiation, execution and termination of transaction. It ensures data consistency, integrity in database either committing or rolling back changes.

i> COMMIT :- It is used to permanently saved the changes made during transaction. Once, COMMIT executed, the transaction is considered successful.

Example:

```
UPDATE employee
  SET salary = salary * 1.10
  WHERE department = 'sales'

COMMIT;
```

To maintain database
- Maintenance tasks
- Maintenance plan used

ii> ROLLBACK:- It is used to undo changes made during transaction. It reverses the changes applied to database. It is used when an error occur during transaction.

Example:

```
BEGIN;

UPDATE Inventory
SET Quantity = Quantity - 10
WHERE productID = 101;
--An error occurs

ROLLBACK;
```

Aggregate Functions

These are used to perform calculations on group of rows. These functions in SQL provide insights into data and return a single scalar value.

Syntax:

```
SELECT <aggregate function>(<column name>)
FROM <table name>
WHERE <condition>;
```

There are different types of aggrete function given below:-

i> AVG
ii> COUNT
iii> ROUND
iv> LISTAGG
v> SUM
vi> MIN
vii> MAX
viii> COUNT(*)

i> AVG :- This function calculates the average of a set values.

syntax:

```
SELECT AVG(<columnname>)
FROM <table name>
WHERE <condition>;
```

ii> COUNT :- It is used to count rows in table

syntax:

```
SELECT COUNT (<columnname>)
FROM <tablename>
WHERE <condition>;
```

iii> SUM :- It is used to calculate sum of values.

syntax:

```
SELECT SUM (<column name>)
FROM <tablename>
WHERE <condition>;
```

iv> MIN :- It is used to set minimum value in a set of value

syntax:

```
SELECT MIN (<columnname>)
FROM (table name>
WHERE <condition>;
```

v> MAX :- It is used to get maximum values in a set of values.

Syntax:

```
SELECT MAX (<column name>)
FROM (table name)
WHERE <condition>;
```

Joins in Database

SQL

It is an operation that combine rows from two or more tables in FROM clause based on related column between them.

It is used to retrieve data from multiple tables linking them using a key.

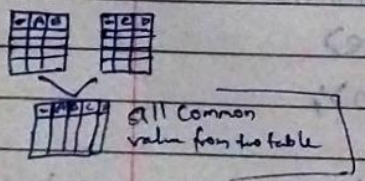
There are mainly four types of joins in database SQL:-

i> Inner Join (simple join)
ii> Outer Join
iii> Cross Join
iv> Self Join

i> Inner Join:- It combines data from two or more tables based on specific condition. The results include only rows where join is met. It is also called simple join.

Syntax:

```
SELECT columns
FROM table1
INNER JOIN table2
ON table1.column = table2.column;
```



ii> Outer Join:- It combines data from two or more tables based on specific condition. The result include row but do not have matching values.

There are three types of outer joins-
i> Left Outer, ii> Right Outer and iii> Full Outer

1> Left Outer Join:- It returns all the rows from the left table and matching rows from right table.

Example:

```
SELECT customers; name, order
  FROM customers
LEFT JOIN Orders ON customer. costomerID
= Order. customerID;
```

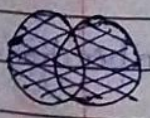
ii> Right Outer Join:- It returns all records from right table and match from left table.

Example:

```
SELECT customers. customerName, orders. Produt
  FROM customers
  RIGHT JOIN Orders ON customer. customerID
    = Orders. customer ID;
```

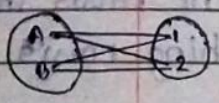
iii> Full Outer Join:- It returns all records when there is match in either left or right table.

Example:



```
SELECT customer. customerName, orders. Produt
  FROM customers
  FULL OUTER JOIN orders ON customer.
  customerID = Orders. customer ID;
```

iii> Cross Join :- It is also called Cartesian Product, it is type of join operation that combines every row from one table. It doesn't require specific condition between table



Syntax:

```
SELECT columns
FROM table1
CROSS JOIN table2;
```

iv> Self Join :- It involves joining a table itself. It is used to find relationships or hierarchies within table.

Syntax:

```
SELECT columns
FROM table1 AS alias1
JOIN table AS alias2 ON alias1.colamn
= alias2.column;
```

Set Operation in SQL

It is used to combine or manipulate the results set of multiple SELECT queries. It allows to perform operations like union, intersect etc on data from tables.

There are four primary set operations:-

i> UNION
ii> INTERSECT
iii> EXCEPT (MINUS)
iiii> UNION ALL

i) UNION :- It combines the result set of two or more SELECT queries into single result set. It removes duplicate rows between SELECT statements.

syntax:

```
SELECT field1....
FROM tables
UNION
SELECT field1---
FRom tables;
```

ii) INTERSECT :- It results the common rows exist in result sets of two or more SELECT queries. It only return distinct row in all sets.

syntax:

```
SELECT field1....
FROM tables
INTERSECT
SELECT field1---
FROM tables;
```

iii) EXCEPT(MINUS) :- It returns all rows in first query that are not return in second query.

syntax:

```
SELECT field1---
FROM Tables
MINUS
SELECT field1---
FROM tables;
```

Privileges in SQL

The access rights on a database provided to user, is called privilege. When multiple user can access database then authorization can be controlled. These privilege are granted or revoked by the administrator.

There are two types of privileges:-

i) System privilege
ii) Object privilege

i) System Prèvilege:- The right to perform a particular action to any object, called system privilege. The object may be table, view, index etc. It allows user to create, alter or drop database objects. Various system privileges are ADMIN, ALTER TABLE, ALTER VIEW INDEX, CREATE TABLE etc.

ii) Object Privileges:- The right to perform a particular action on an object, is called object privilege. A user with ADMIN privilege can grant or revoke these privelege to other user. It allows the user to execute, select, insert, update or delete data from database objects. Various object privileges are DELETE, EXECUTE, INDEX, Flush, Insert Load, Refresh, Select, Update etc.

SubQuery or Nested Query

A subquery is a query within a query. It is a form of statement that appears inside another SQL statement. These subqueries can reside in WHERE clause, FROM clause or SELECT clause.

must return 1 value

Uses:- i> to create view
ii> To insert records in a table
iii> To provide values for condition
iv> To create and insert table in a table.

i> WHERE Clause:- A subquery can be in WHERE clause.

Exple:-

255 level Subquery

```
SELECT * FROM all_tables tabs
WHERE tabs.table_name in
FROM all_tab_columns cols
WHERE cols.column_name = 'SUPPLIER_ID';
```

ii> FROM clause:- A subquery can also be in FROM clause.

Example:

Unlimited Subquery

```
SELECT suppliers.name, subquery1.total_amt
  FROM supplier
(SELECT supplier_id, SUM(orders.amount) as
 total_amt
  FROM order
  GROUP BY supplier_id) subquery1,
  WHERE subquery1.supplier_id = suplier_id;
```

iii> SELECT clause:- A subquery can also be in SELECT clause.

Example: SELECT ---

```
FROM ---
WHERE ---
AND ---
```

Keys in SQL

A key is an attribute which identify every record uniquely and establish relationship between table.

There are various types of key in SQL:-

i> Primary Key
ii> Unique key
iii> Foreign key
iv> Normal Key

i> Primary Key:- It is a single field that uniquely identifies a ~~relations~~ records. A table can have only one primary key.

i> Using a CREATE TABLE statement

syntax:

```
CREATE TABLE table_name
(column1 datatype null/not null,
CONSTRANT constraint_name
PRIMARY KEY (column1,....));
```

ii> Using an ALTER TABLE statement

syntax:

```
ALTER TABLE table_name
ADD constraint constraint name
PRIMARY KEY (column1,---);
```

iii> Dropping a Primary Key:

Syntax:

```
ALTER TABLE table_name
DROP CONSTRAINT constraint_name;
```

iv> Enable Primary Key

Syntax:

```
ALTER TABLE table_name
ENABLE CONSTRAINT constraint_name;
```

v> Disable Primary Key

Syntax:

```
ALTER TABLE table_name
DISABLE CONSTRAINT constraint_name;
```

ii> Unique Key:- It is a single field that uniquely identifies record. Some fields may have null values.

i> Using a CREATE TABLE statement

Syntax:

```
CREATE TABLE table_name
(column1 datatype null/not null,
:CONSTRAINT constraint_name
UNIQUE (column1,...));
```

ii> Using an ALTER TABLE statement

Syntax:

```
ALTER TABLE table_name
ADD CONSTRAINT constraint_name
UNIQUE (column1,...);
```

iii> Drop a Unique Constraint

syntax:

```
ALTER TABLE table-name
DROP CONSTRAINT constraint-name;
```

iv> Enable Unique Constraint

syntax:

```
ALTER TABLE table_name
ENABLE CONSTRAINT constraint-name;
```

v> Disable Unique Constraint

syntax:

```
ALTER TABLE table_name
DISABLE CONSTRAINT constraint-name;
```

iii> Foreign Key:- It is a column which serves as primary key of another table. The value in one table must appear in another table.

i> Using a CREATE TABLE statement

syntax:

```
CREATE TABLE name_name
(column datatype null/not null,
CONSTRAINT fk-column
FOREIGN KEY (column1... )
REFRENCE parent_table (column1...));
```

ii> Using ALTER TABLE statement

syntax:

```
ALTER TABLE table_name
ADD CONSTRAINT constraint_name
FOREIGN KEY(column1---)
REFRENCES parent_table(column1--);
```

Lokesh