SQL

STRUCTURED QUERY LANGUAGE

RAJEEV SRIVASTAVA (RAJEEVS@CDAC.IN)

SQL

- Stands for "Structured Query Language"
- Also pronounced as "SEQUEL" (Structured English QUEry Language)
- Standard access mechanism to every RDBMS.
- Case Insensitive
- 4th Generation Language
- Standard Based; SQL Standards are defined by ANSI (American National Standards Institute)
- First Standard Published in 1989; Latest is 2016

COMPONENTS (CATEGORIES) OF SQL STATEMENTS

- DDL: Data Definition Language(CREATE/ALTER/DROP/TRUNCATE)
- DML: Data Manipulation Language(INSERT/UPDATE/DELETE)
- DCL: Data Control Language (GRANT/REVOKE)
- DTL: Data Transaction Language OR
  TCL: Transaction Control Language (COMMIT/ROLLBACK)
- DRL: Data Retrieval Language OR
  DQ: Data Query Language (SELECT)

CREATING A TABLE

- Data in a data base is stored in the form of tables
- The table is a collection of related data entries and it consists of columns and rows.

SYNTEX -

- CREATE TABLE <table name> (
  <col name> <data type>,
  <col name> <data type>
-);

CONSIDERATIONS FOR CREATING TABLE

Points to be considered before creating a table -

- What are the attributes of the tuples to be stored?
- What are the data types of the attributes? Should varchar be used instead of char ?
- Which columns build the primary key?
- Which columns do (not) allow null values? Which columns do (not) allow duplicates ?
- Are there default values for certain columns?

MYSQL: DATA TYPES

- Data Type defines what kind of values can be stored in a column.
- Data Type also defines the way data will be stored in the system and the space required in disk.
- Data Type also impact database performance.
- Ex- Char, Varchar, Text, Integer, Float, Double, Date, Timestamp, Enum, Blob etc.
- More on SQL Datatypes : https://www.w3schools.com/sql/sql_datatypes.asp

INSERT COMMAND

- Used to insert data into a table
- Insert command always inserts values as new row

Syntax -

INSERT INTO <table name> VALUES (<val 1>, <val 2>);

- Insert data into specific columns of a table -

INSERT INTO <table name> (<col1>) VALUES (<value>);

- Define an insertion order -

INSERT INTO <table name> (<col 2>, <col 1>) VALUES (<val 2>, <val 1>);

- Missing attribute → NULL.
- May drop attribute names if give them in order

SELECT COMMAND

- Used to Retrieve/Fetch information from the database.

Syntax :

SELECT <col name> FROM <table name> [WHERE <condition>];

SELECT <col1>, <col2> FROM <table name> [WHERE <condition>];

An asterisk symbol (*) Represents all columns/attributes.

SELECTION & PROJECTION

- SELECTION – limiting rows selection (by using WHERE clause)
  - SELECT * FROM <table name> WHERE <col1> = <val1> ;
- PROJECTION – limiting columns selection (by using SELECT clause)
  - SELECT <col1>, <col2> FROM <table name>;

ALTER TABLE

- Syntax
  - To modify/change an existing column

    ALTER TABLE <tablename> MODIFY COLUMN <col1> <new data type>;
  - To add new column

    ALTER TABLE <tablename> ADD COLUMN <col> <data type>;
  - To rename an existing column

    ALTER TABLE <tablename> RENAME COLUMN <col> TO <new col name>;
  - To drop/remove an existing column

    ALTER TABLE <tablename> DROP COLUMN <col>;

NULL VALUE

- When you do not insert data into a column of a table for a specific row then by default a NULL value will be inserted into that column by the database.

  INSERT INTO dept (deptno, deptname) VALUES (40, 'BIOM');

- NULL value does not occupy space in memory
- NULL value is independent of a data type
- A NULL value is not a zero (0) OR an empty string (‘ ’), rather it represents an Unknown or Not applicable value.

UPDATE COMMAND

- This command inserts or modifies values in the cells in existing rows
- This command can also be used for deleting values from a cell of a table without the need for deleting a row or a column

Syntax

- UPDATE <table name> SET <col name> = <new value> [WHERE <condition>];

DELETE COMMAND

- This command is used for deleting specific or all the rows from a table

Syntax

DELETE FROM <table name> [WHERE <condition>];

TRUNCATE COMMAND

- This command can also be used for deleting all the rows from a table

Syntax

TRUNCATE TABLE <table name>;

- Truncate command cannot be rolled back because it is a AUTO COMMIT operation, i.e. changes committed cannot be rolled back. But DELETE is not a AUTO COMMIT operation. Hence DELETE can be ROLLED BACK.
- Truncate is a DDL Command.

DROP COMMAND

- DROP command can be used for permanently deleting database objects like table, view, function etc. from a database

Syntax

DROP TABLE <table name>;

CONSTRAINTS

- PRIMARY KEY
- FOREIGN KEY
- UNIQUE
- NOT NULL
- DEFAULT
- CHECK (NOT SUPPORTED BY MySQL)

RELATIONAL OPERATORS

Operator	Description	Example
=	Checks if the values of two operands are equal or not, if yes condition becomes true.	(A=A) is true (A = B) is not true.
!= <>	Checks if the values of two operands are equal or not, if not equal then condition becomes true.	(A != B) is true.
>	Checks if the value of left operand is greater than the value operand, if yes then condition becomes true.	(A > B) is not true.
<	Checks if the value of left operand is less than the value of operand, if yes then condition becomes true.	(A < B) is true.
>=	Checks if the value of left operand is greater than or equal value of right operand, if yes then condition becomes	(A >= B) is not true.
<=	Checks if the value of left operand is less than or equal to value of right operand, if yes then condition becomes	(A <= B) is true

BETWEEN … AND… OPERATOR

- This operator is used as a replacement for relational (>, <) and logical operators (AND, OR)
- In this operator lower and upper values are inclusive if they are of number or date types
- The lower limit must be <= upper limit
  - SELECT * FROM emp WHERE sal BETWEEN 10000 AND 20000;

IS NULL OPERATORS

- Used for testing for the existence of NULL values in any column
- Ex. Display the details of those employees who are not having any job

  SELECT * FROM emp WHERE job IS NULL:

  IS NOT NULL

  SELECT * FROM emp WHERE job IS NOT NULL:
- Null value cannot be compared. Hence you cannot use relational operators for comparing NULL value with a column
- Therefore IS NULL operator has to be used for the purpose

IN OPERATOR

- This operator is used to compare multiple values with a single column
- In this operator, values supplied must be of the same type and they should belong to only 1 column
- This operator is a replacement for multiple OR operators
  - SELECT * FROM emp WHERE job IN (‘PE', ‘TO‘, ‘SE’);

LIKE OPERATOR - % AND _

- This operator is used for comparing characters/numbers in a value from a specific position
  - % ignores variable number of characters
  - _ ignores only 1 char
- Display the details of those emps whose name is starting with ‘r’

```
SELECT * FROM emp WHERE ename LIKE 'r%‘;
```

- Display the details of those emps who have ‘h’ as second character in their name -

```
SELECT * FROM emp WHERE ename LIKE ‘_h%‘;
```

In MySQL, for character values case is ignored by Like operator.

DISTINCT - ELIMINATING DUPLICATES

- DISTINCT command is used to select only unique values from a column.
- i.e. only single occurrence of a value will be returned.

```
SELECT DISTINCT job FROM EMP;
SELECT DISTINCT loc FROM dept;
```

FUNCTIONS

- Function is a Sub Program which performs a specific task
- Every function returns only 1 value
- Functions in MySQL database can be used or defined for
  - Performing arithmetic calculations which are not possible/easy using arithmetic operators
  - Formatting text/numbers/dates
  - Type casting i.e. converting one type of data into another
  - To fetch information from system schema. Eg. VERSION()

FUNCTION – TYPES AND USES

- Types of Functions
  - System defined functions
  - User defined functions
- Syntax -

  SELECT <function name(args)> [FROM <table name>];

FUNCTIONS – SYSTEM DEFINED

- Numeric functions
- String functions
- Date and Time functions
- Conversion functions
- Aggregate functions

More Functions -

  - https://www.w3schools.com/sql/sql_ref_mysql.asp

FEW FUNCTIONS

Aggregate Functions

- COUNT() – Gives count of occurrences; Works on All data types.
- AVG() – Average. Works only on Numeric values
- SUM() – Gives total/sum. Works only on Numeric values
- MAX() – Maximum Value. All data types
- MIN() – Minimum Value. All data types

Concatenation Function

CONCAT(<col/value1>, <col/value2>, …) – Used to combine/merge two or more values

OTHER IMP FUNCTIONS

- FORMAT
- LEFT
- LENGTH
- LOWER
- LOCATE
- LPAD
- LTRIM
- REPLACE
- REVERSE
- RIGHT
- RPAD
- RTRIM
- SUBSTR
- TRIM
- UCASE
- UPPER
- ABC
- CEIL
- EXP
- FLOOR
- MOD
- ROUND
- SQRT
- DATE
- DAY
- SECOND
- MINUTE
- HOUR
- DAY
- MONTH
- YEAR
- NOW
- SYSDATE
- EXTRACT
- LAST_DAY
- DATE_FORMAT

REAL TIME USE OF FUNCTIONS

SELECT UPPER(dname) FROM dept;

SELECT MAX(sal) FROM emp;

SELECT SUM(sal), AVG(SAL) FROM emp;

JOINS

- Joins is a technique of retrieving data from multiple tables
- Display the ename AND dname from emp and dept tables

  SELECT ename, dname FROM emp, dept WHERE emp.deptno = dept.deptno;
- Display the ename and dname of those emps whose sal is > 20000

  SELECT ename, dname, sal FROM emp, dept WHERE emp.deptno = dept.deptno AND sal > 20000;

SUBQUERIES

- It is a query within some other query

- Display the details of those employees who are getting a sal > Aditya

  SELECT * FROM emp WHERE sal > (SELECT sal FROM emp WHERE ename = 'Aditya';

- Display details of all employees who work in department 'SENG';

  SELECT * FROM emp WHERE DEPTNO = (SELECT deptno FROM dept WHERE dname = 'SENG');

MORE SUBQUERIES EXAMPLES

- Display the maximum salary from every department and also the name of that employee who is getting the maximum Salary.

```
SELECT ename, sal, deptno FROM emp WHERE (deptno, sal) IN
(SELECT deptno, max(sal) FROM emp GROUP BY deptno);
```

MORE SQL CLAUSES

- ORDER BY (if used, ORDER BY must by last clause of a query, except when you have used LIMIT Clause)
  - SELECT * FROM EMP ORDER BY SAL;
  - SELECT * FROM EMP ORDER BY SAL DESC;
  - SELECT * FROM EMP ORDER BY SAL DESC, ENAME;
- GROUP BY
  - SELECT DEPTNO, SUM(SAL) FROM EMP GROUP BY DEPTNO;
- GROUP BY….HAVING
  - SELECT DEPTNO, SUM(SAL) FROM EMP GROUP BY DEPTNO HAVING SUM(SAL) < 50000;

LIMITING NO OF RECORDS

- LIMIT clause is used to limit number of records return in a SELECT query.
- Its not part of SQL standard, works in MySQL, and few other RDBMS.
- Unexceptionally, must be the last clause of a query.

```
SELECT * FROM emp LIMIT 2;
SELECT * FROM emp ORDER BY sal DESC LIMIT 3;
```

COPYING TABLE/DATA

- A copy of the table (with, without or selected data)can be created using SELECT command

  CREATE table dept_copy AS SELECT * FROM dept;

  CREATE table emp_pe AS SELECT * FROM emp WHERE job = 'PE';

  CREATE table emp_new AS SELECT * FROM emp WHERE 1=2

- To copy only data of a table to another table

  INSERT INTO emp_seng SELECT * FROM emp WHERE deptno = (SELECT deptno FROM dept WHERE dname = 'SENG');

  INSERT INTO emp_seng SELECT * FROM emp WHERE deptno = 10;

COMMIT, SAVEPOINT & ROLLBACK

- By default autocommit parameter is ON in MySQL and can be reconfigured by MySQL Administrator. A user can set autocommit parameter to ON & OFF for herself using SET command
- To check the status of autocommit for the user –

  SHOW LOCAL VARIABLES LIKE 'autocommit';
- To set the autocommit ON (1) /OFF (0)

  SET autocommit=[0 | 1]
- Savepoint and Rollback commands will be effective only when autocommit is OFF;
- System (MySQL) autocommits all uncommitted operations before executing any DDL command.

  COMMIT;

  SAVEPOINT <variable name>;

  ROLLBACK [TO <variable name>];

GRANT & REVOKE

- GRANT commond is used to give selected/all DML and DDL command privileges to other database user;

  GRANT [PRIVILEGES] ON <dbname>.<tablename> TO '<username>';

  GRANT [PRIVILEGES] ON <dbname>.<tablename> TO '<username>' WITH GRANT OPTION;

- REVOKE command is used to REVOKE command privileges given to other database user using GRANT Command;

  REVOKE [PRIVILEGES] ON <dbname>.<tablename> FROM '<username>';

- TO Check the current privileges

  SHOW GRANTS FOR <user name>;

ADDING A CONSTRAINT AFTER CREATION OF A TABLE

- Any type of constraint can be added or dropped even after creation of table.

PRIMARY KEY

ALTER TABLE <tablename> ADD CONSTRAINT <constraintname> PRIMARY KEY (<coliumn name>);
ALTER TABLE <tablename> ADD PRIMARY KEY (<columnname>);

ALTER TABLE <tablename> DROP PRIMARY KEY;

UNIQUE

ALTER TABLE <tablename> ADD CONSTRAINT <constraintname> <UNIQUE>;
ALTER TABLE <tablename> DROP INDEX <constraintname>;

DEFAULT

ALTER TABLE <tablename> ALTER <column name> SET DEFAULT <default value>;

ALTER TABLE <tablename> ALTER <column name> DROP DEFAULT;

JOIN TYPES : SQL STANDARD SYNTAX

- **(INNER) JOIN**: Returns records that have matching values in both tables
- **LEFT (OUTER) JOIN**: Return all records from the left table, and the matched records from the right table
- **RIGHT (OUTER) JOIN**: Return all records from the right table, and the matched records from the left table
- **FULL (OUTER) JOIN**: *(NOT Supported by MySQL)* Return all records when there is a match in either left or right table

SELECT <column names> FROM <table name1> [LEFT |RIGHT] OUTER JOIN <tablename2> ON <tablename1>.<column name> = <tablename2>.<column name>;

INNER JOIN: EXAMPLES

```
SELECT empno, ename, sal, dname
FROM dept JOIN emp
ON dept.deptno = emp.deptno;
```

```
SELECT *
FROM dept JOIN emp
ON dept.deptno = emp.deptno;
```

LEFT OUTER JOIN: EXAMPLE

```
SELECT empno, ename, sal, dname
FROM dept LEFT OUTER JOIN emp
ON dept.deptno = emp.deptno;

SELECT *
FROM dept LEFT OUTER JOIN emp
ON dept.deptno = emp.deptno;
```

RIGHT OUTER JOIN: EXAMPLE

```
SELECT empno, ename, sal, dname
FROM dept RIGHT OUTER JOIN emp
ON dept.deptno = emp.deptno;

SELECT *
FROM dept RIGHT OUTER JOIN emp
ON dept.deptno = emp.deptno;
```

SET OPERATIONS: UNION & UNION ALL

- Set operations treat the tables as sets and are the usual set operators of union, intersection, and difference
- MySQL Supports only UNION [ALL] operators

```
SELECT * FROM <table name 1>
UNION [ALL]
SELECT * FROM <table name 2>;
```

- SET operations must fulfil following two conditions –
  - Number of columns in the SELECT clause of both[all] queries must be same.
  - Data type of respective columns in both[all] queries must be same or compatible.

VIEWS

- Views are ‘Virtual Tables’.
- Views does not store data but fetches the data from underlying tables[s] dynamically at runtime.

  CREATE VIEW <view name> AS <Select Query> ;
- All DML operations can be performed on a view and it affects underlying table.

INDEXES

- Indexes are database objects used by DBMS to retrieve the data from table in a faster manner.
- Invisible to end user
- Speed up SELECT queries but slow down INSERTS –

  CREATE [UNIQUE] INDEX <index name>

  ON <table name> (<column name> [, <column name2>],.....';
- Displaying all indexes existing on a table

  SHOW INDEX IN <table name>;
- Dropping and Index

  ALTER TABLE <table name> DROP INDEX <index name>;

SEQUENCES/AUTO INCREMENT

- SEQUENCES used to insert sequential values in a key column.
- SEQUENCES are implemented through AUTO_INCREMENT in MySQL

  ALTER TABLE <tablename> MODIFY <column name> <data type> PRIMARY KEY AUTO_INCREMENT;

  ALTER TABLE dept MODIFY deptno integer(10) PRIMARY KEY AUTO_INCREMENT;

- Resetting value of AUTO_INCREMENT -

  ALTER TABLE <table name> AUTO_INCREMENT = <increment value>;