Data Manipulation Language (DML) Statements

1. **SELECT**

**SELECT** column1, column2, ... **FROM** table\_name **WHERE** condition;

* 1. **COUNT**

**SELECT** COUNT(\*) **FROM** table\_name;

* 1. **DISTINCT**

**SELECT DISTINCT** Title **FROM** table\_name;

* 1. **LIMIT**

**SELECT** \* **FROM** table\_name **LIMIT** 25;

* 1. **LIKE %**

**SELECT** column\_name **FROM** table\_name **WHERE** condition **LIKE % %**;

* 1. **BETWEEN**

**Select \* FROM** table\_name **WHERE** condition **BETWEEN** value1 AND value2;

* 1. **ORDER BY, DESC**

**Select** column\_name **FROM** table\_name **ORDER BY** column\_name;

**Select** column\_name **FROM** table\_name **ORDER BY** column\_name **DESC**;

* 1. **GROUP BY**

**Select AVG(**column\_name) **FROM** table\_name **GROUP BY** column\_name;

* 1. **AS**

**Select A.**column\_name, **B.**column\_name

**FROM** table\_name **AS** A**,** table\_name **AS** B;

1. **INSERT is used to insert values into tables**

**INSERT** INTO table\_name (column1, column2, ...) **VALUES** (value1, value2, ...);

1. **UPDATE**

**UPDATE** table\_name  
**SET** column1 = value1, column2 = value2, ... **WHERE** condition;

1. **DELETE**

**DELETE FROM** table\_name **WHERE** condition;

Data Definition Language (DDL) Statements

1. **CREATE** (to create a new table in a database).

**CREATE TABLE** table\_name(column1 datatype, column2 datatype, ...);

1. **DROP (**to delete an existing table in a database).

**DROP TABLE** table\_name;

1. **TRUNCATE** (to remove all rows from an existing table without deleting the table itself).

**TRUNCATE TABLE** table\_name;

1. **ALTER** (to add, delete, or modify columns in an existing table)
   1. **ADD COLUMN**

**ALTER TABLE** table\_name

**ADD COLUMN** column\_name data\_type column\_constraint;

* 1. **DROP COLUMN**

**ALTER TABLE** table\_name

**DROP COLUMN** column\_name;

* 1. **ALTER COLUMN**

**ALTER TABLE** table\_name

**ALTER COLUMN** column\_name SET DATA TYPE data\_type;

* 1. **RENAME COLUMN**

**ALTER TABLE** table\_name

**RENAME COLUMN** current\_column\_name TO new\_column\_name;

Built-in Database Functions

1. SUM

SUM (column\_name);

1. MIN / MAX

MIN (column\_name);

MAX (column\_name);

1. **AVG**

**SELECT** AVG (column\_name) FROM table\_name;

1. ROUND

SELECT ROUND (column\_name) FROM table\_name;

1. LENGTH (To find the length of each value in the column).

SELECT LENGTH (column\_name) FROM table\_name;

1. UCASE / LCASE

SELECT UCASE (column\_name) FROM table\_name;

SELECT LCASE (column\_name) FROM table\_name;

Date and Time Built-in Functions

DATE: YYYYMMDD

TIME: HHMMSS

TIMESTAMP: YYYYXXDDHHMMSSZZZZZZ (WHERE Zs REPRESENTS THE MICRO SECONDS)

YEAR(), MONTH(), DAY(), DAYOFMONTH(), DAYOFWEEK(), DAYOFYEAR(), WEEK(),

HOUR(), MINUTE(), SECOND(), CURRENTDATE(), CURRENTTIME()

Sub-Queries and Nested Selects

These queries can be used in the:

* WHERE Clause.
* List of Columns.
* From Clause.

SELECT column\_name [, column\_name ]

FROM table1 [, table2 ]

WHERE column\_name OPERATOR

(SELECT column\_name [, column\_name ]

FROM table1 [, table2 ]

WHERE condition);

Working with Multiple Tables

An Implicit version of CROSS JOIN (also known as Cartesian Join) statement syntax:

**SELECT** column\_name(s) **FROM** table1, table2;

An Implicit version of INNER JOIN statement syntax:

**SELECT** column\_name(s) **FROM** table1, table2

**WHERE** table1.column\_name = table2.column\_name;