|  |  |  |
| --- | --- | --- |
| Colour | Meaning | Example |
| COLOUR | Words With Special Meaning | Data Type |
| COLOUR | Key Words / Special Character or Symbol Used With the Key Words | INT |
| COLOUR | Things Should Be Filled | INT(‘SIZE’) |
| **COLOUR** | Actual Variable / Actual Data / Actual Number / Actual String | INT(100) |

Contents

[1. Data Definition Language 2](#_Toc188674107)

[2. Data Manipulation Language 5](#_Toc188674108)

# Data Definition Language

1. **Create / Use / Delete Database :**

|  |  |  |
| --- | --- | --- |
| **No.** | **Apply** | **Explain** |
| 1. | SHOW DATABASES; | * To show all exist databases. |
|  | | |
| 2. | USE ‘DatabaseName’; | * To use a specific database. * The first step should be done before peform any operation related table.   If not, the system will show error.   * All of the operation related the table will save inside the current using database. |
|  | | |
| 3a. | CREATE DATABASE ‘DatabaseName’; | * To create a database. |
| 3b. | CREATE DATABASE IF NOT EXISTS ‘DatabaseName’; | * To create a database and avoid errors if the database already exists. |
|  | | |
| 4a. | DROP DATABASE ‘DatabaseName’; | * To delete a database. * Delete database is permanent and cannot be undone.   Make sure you have backed up any necessary data before running this command |
| 4b. | DROP DATABASE IF EXISTS ‘DatabaseName’; | * To delete a database and avoid errors if the database does not exists. |

1. **Data Type :**

* If not assign the ‘SIZE’, the ‘SIZE’ of the Data Type will be automatic set as the highest size of it can be hold.

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Data Type** | **Spec** | **Apply** |
| 1. | CHAR | String (0 - 255) |  |
| 2. | VARCHAR | String (0 - 255) | VARCHAR(‘SIZE’) |
| 3. | TINY TEXT | String (0 - 255) |  |
| 4. | TEXT | String (0 - 65535) |  |
| 5. | BLOB | String (0 - 65535) |  |
| 6. | MEDIUMTEXT | String (0 - 16777215) |  |
| 7. | MEDIUMBLOB | String (0 - 16777215) |  |
| 8. | LONGTEXT | String (0 - 4294967295) |  |
| 9. | LONGBLOB | String (0 - 4294967295) |  |
|  | | | |
| 10. | TINYINT | Integer (-128 to 127) |  |
| 11. | SMALLINT | Integer (-32768 to 32767) |  |
| 12. | MEDIUMINT | Integer (-8388608 to 8388607) |  |
| 13. | INT | Integer (-2147483648 to 2147483647) | INT(‘SIZE’) |
| 14. | BIGINT | Integer (-9223372036854775808 to 9223372036854775807) |  |
| 15. | BOOLEAN | TINYINT(1) |  |
|  | | | |
| 16. | FLOAT | Decimal (Precise to 23 Digits) |  |
| 17. | DOUBLE | Decimal (24 to 53 Digits) | DOUBLE(‘TOTAL NUMBER OF DIGIT’,‘SIZE FRACTION PART’) |
| 18. | DECIMAL | DOUBLE Stored As String | DECIMAL(‘TOTAL NUMBER OF DIGIT’,‘SIZE FRACTION PART’) |
|  | | | |
| 19. | DATE | YYYY-MM-DD | DATE |
| 20. | DATETIME | YYYY-MM-DD HH:MM:SS | DATETIME |
| 21. | TIMESTAMP | YYYYMMDDHHMMSS | TIMESTAMP |
| 22. | TIME | HH:MM:SS | TIME |
|  | | | |
| 23. | ENUM | One Of Preset Options |  |
| 24. | SET | Selection Of Preset Options |  |

1. **Constraint :**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Constraint** | **Explain** | **Apply** |
| 1. | PRIMARY KEY | * Creates a Primary Key for the table. * Only 1 Primary Key can be created for each table. * The PRIMARY KEY constraint is a column or a set of columns that uniquely identifies each row in a table. * No column that is part of the Primary Key can contain a NULL value. * A Composite Primary Key must be created at the Table Level. | * **Column Level :**   ColumnName’ ‘DataType’ CONSTRAINT ‘ConstraintName’ PRIMARY KEY   * **Example :**   deptNo INT(5) CONSTRAINT dept\_deptNo\_pk PRIMARY KEY |
| * **Table Level :**   CONSTRAINT ‘ConstraintName’ PRIMARY KEY (‘ColumnName’)   * **Example :**   CONSTRAINT dept\_deptNo\_pk PRIMARY KEY (deptNo) |
| 2. | UNIQUE | * Every value in a column or a set of columns be unique. * If the UNIQUE constraint has more than one column, that group of columns is called a Composite Unique Key. * UNIQUE constraints enable the input of NULLs. * A NULL in a column always satisfies a UNIQUE constraint. | * **Column Level :**   ‘ColumnName’ ‘DataType’ CONSTRAINT ‘ConstraintName’ UNIQUE   * **Example :**   email VARCHAR(30) CONSTRAINT emp\_email\_uk UNIQUE |
| * **Table Level :**   CONSTRAINT ‘ConstraintName’ UNIQUE (‘ColumnName’)   * **Example :**   CONSTRAINT emp\_email\_uk UNIQUE (email) |
| 3. | CHECK | * Defines a condition that each row must be satisfy. * It cannot reference columns from other tables. | * **Column Level :**   ‘ColumnName’ ‘DataType’ CONSTRAINT ‘ConstraintName’ CHECK (‘Condition’)   * **Example :**   salary DECIMAL(10,2) CONSTRAINT emp\_chk\_salary\_min CHECK (salary >= 1700) |
| * **Table Level :**   CONSTRAINT ‘ConstraintName’ CHECK (‘Condition’)   * **Example :**   CONSTRAINT emp\_chk\_salary\_min CHECK (salary >= 1700) |
| 4. | FOREIGN KEY | * Designates a column or a combination of columns as Foreign Key. * Establishes a relationship with a Primary Key in the same table or a different table. | * **Column Level :**   ‘ColumnName’ ‘DataType’ CONSTRAINT ‘ConstraintName’ FOREIGN KEY (‘ColumnName’) REFERENCES ‘TableNameOtherTable’(‘ColumnNameOtherTable’)   * **Example :**   deptNo INT(5) CONSTRAINT emp\_fk\_deptNo FOREIGN KEY (deptNo) REFERENCES Department(deptNo) |
| * **Table Level :**   CONSTRAINT ‘ConstraintName’ FOREIGN KEY (‘ColumnName’) REFERENCES ‘TableNameOtherTable’(‘ColumnNameOtherTable’)   * **Example :**   CONSTRAINT emp\_fk\_deptNo FOREIGN KEY (deptNo) REFERENCES Department(deptNo) |
| 5. | NOT NULL | * Every value in a column cannot be NULL. * It only can be defined at the Column Level. | * **Column Level :**   ‘ColumnName’ ‘DataType’ CONSTRAINT NOT NULL   * **Example :**   fName VARCHAR(10) CONSTRAINT NOT NULL |

1. **Create Table :**

* Column Level Constraint : ‘ColumnName’ ‘DataType’ CONSTRAINT ‘ConstraintName’ ‘ConstraintType’
* Table Level Constraint : CONSTRAINT ‘ConstraintName’ ‘ConstraintType’
* Last statement inside the CREATE TABLE no need to put comma (,).
* CONSTRAINT at Column Level can be removed if no require adding ‘ConstraintName’.
* More prefer adding ‘ConstraintName’ at Table Level. (mySQL Not Support Put CONSTRAINT at Column Level)

|  |  |
| --- | --- |
| **Create Table** | **Example** |
| CREATE TABLE ‘TableName’ (  ‘ColumnName’ ‘DataType’ CONSTRAINT ‘ConstraintName’ ‘ConstraintType’,  ‘ColumnName’ ‘DataType’ CONSTRAINT ‘ConstraintName’ ‘ConstraintType’,  ↓  ‘ColumnName’ ‘DataType’ CONSTRAINT ‘ConstraintName’ ‘ConstraintType’,  CONSTRAINT ‘ConstraintName’ ‘ConstraintType’,  ↓  CONSTRAINT ‘ConstraintName’ ‘ConstraintType’  ); | OR  OR |

1. **Create Table (Copy) :**

|  |  |  |
| --- | --- | --- |
| **No.** | **Apply** | **Explain** |
| 1. | CREATE TABLE ‘TableName’  AS (SELECT \* FROM ‘TableName’); | * Create table from existing database tables. * All data will copy but it will not copy constraints. (Except Null) |
| 2. | CREATE TABLE ‘TableName’  LIKE ‘TableName’; | * Create table an empty table based on the definition of an existing table. * All data will not copy but it will copy constraints. |

1. **Delete / View / Rename Table :**

|  |  |  |
| --- | --- | --- |
| **No.** | **Apply** | **Explain** |
| 1. | SHOW TABLES; | * To show all exist tables in database. |
|  | | |
| 2. | DROP TABLE ‘TableName’; | * To delete a table. |
|  | | |
| 3. | DESCRIBE ‘TableName’;  OR  DESC ‘TableName’; | * Viewing table structures. |
|  | | |
| 4. | RENAME TABLE ‘TableName’ TO ‘NewTableName’; | * To rename a table. |

1. **Alter Table :**

|  |  |  |
| --- | --- | --- |
| **No.** | **Apply** | **Explain** |
| 1. | ADD ‘ColumnName’ ‘DataType’; | * Add new column. |
| 2. | ADD CONSTRAINT ‘ConstraintName’ ‘ConstraintType’; | * Add new constraint. |
| 3. | MODIFY ‘ColumnName’ ‘NewDataType / Size / DefaultValue’; | * Change data type, data size, default values and constraints. |
| 4. | DROP COLUMN ‘ColumnName’; | * Delete Column. |
| **Alter Table** | | **Example** |
| ALTER TABLE ‘TableName’  ADD ‘ColumnName’ ‘DataType’,  ADD CONSTRAINT ‘ConstraintName’ ‘ConstraintType’,  MODIFY ‘ColumnName’ ‘NewDataType / Size / DefaultValue’,  DROP COLUMN ‘ColumnName’; | | --- |

* Do not ADD and MODIFY same column in a statement. (Error)
* Do not MODIFY more than one column in a statement. (Error)

# Data Manipulation Language

1. **Insert / Update / Delete Data :**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Apply** | **Explain** | **Example** |
| 1a. | INSERT INTO ‘TableName’ (‘Column1’, ‘Column2’, … , ‘Column\_n’)  VALUES (‘Value1’, ‘Value2’, … , ‘Value\_n’); | **Insert Data**   * Insert a row of data. * The string must put inside a Single Quotes (‘ ’). * The date must put inside a Double Quotes (“ “). * SYSDATE() can be used to insert current date or time. * NULL can be used to insert a null value in a specific cell but we must make sure the column of the specific cell belong allow null value. |  |
| 1b. | INSERT INTO ‘TableName1’ (‘Column1\_Table1’, ‘Column2\_Table1’, … , ‘Column\_n\_Table1’)  SELECT ‘Column1\_Table2’, ‘Column2\_Table2’, … , ‘Column\_n\_Table2’  FROM ‘TableName2’; | **Insert Data (Copy)**   * Copy all row of data from ‘TableName2’ to ‘TableName1’. |  |
| 1c. | INSERT INTO ‘TableName1’ (‘Column1\_Table1’, ‘Column2\_Table1’, … , ‘Column\_n\_Table1’)  SELECT ‘Column1\_Table2’, ‘Column2\_Table2’, … , ‘Column\_n\_Table2’  FROM ‘TableName2’  WHERE ‘Column\_n\_Table1’ = ‘Value\_n’; | **Insert Data (Copy & With Condition)**   * Copy all row of data from ‘TableName2’ to ‘TableName1’ where ‘Column\_n\_Table1’ = ‘Value\_n’. * WHERE ‘Column\_n\_Table1’ = ‘Value\_n’ is the search condition and only copy the data which satisfy the search condition. |  |
|  | | | |
| 2a. | UPDATE ‘TableName’  SET ‘Column1’ = ‘UpdateValue1’, ‘Column2’ = ‘UpdateValue2’, … , ‘Column\_n’ = ‘UpdateValue\_n’  WHERE ‘Column’ = ‘Value’; | **Update Data**   * To modify existing values in a table. * WHERE ‘Column’ = ‘Value’ is the search condition and only update the data which satisfy the search condition. (Usually Will Choose Primary Key of the Row We Need To Update As Search Condition) |  |
| 2b. | UPDATE ‘TableName1’  SET (‘Column1\_Table1’, ‘Column2\_Table1’, … , ‘Column3\_Table1’) =  (SELECT ‘Column1\_Table2’, ‘Column2\_Table2’, … , ‘Column3\_Table2’  FROM ‘TableName2’  WHERE ‘Column\_Table2’ = ‘Value\_Table2’)  WHERE ‘Column\_Table1’ = ‘Value\_Table1’; | **Update Data (Copy)**   * To modify existing values in a table by copy from others. * WHERE ‘Column\_Table1’ = ‘Value\_Table1’ is the search condition and only update the data which satisfy the search condition. * Copy From :   SELECT ‘Column1\_Table2’, ‘Column2\_Table2’, … , ‘Column3\_Table2’  FROM ‘TableName2’  WHERE ‘Column\_Table2’ = ‘Value\_Table2’ |  |
|  | | | |
| 3a. | DELETE FROM ‘TableName’  WHERE ‘Column’ = ‘Value’; | **Delete Data (Certain Row)**   * To delete existing rows from table. * WHERE ‘Column’ = ‘Value’ is the search condition and only delete the row of data which satisfy the search condition. |  |
| 3b. | DELETE FROM ‘TableName’; | **Delete Data (All Row)**   * To delete all existing rows from table. |

1. **Select Data :**

* SELECT / WHERE / GROUP BY / HAVING / ORDER BY can be used by combining with other code.

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Apply** | **Explain** | **Example** |
| 1. | SELECT \* FROM ‘TableName’; | **Select Data (All Row)**   * List all rows of the table. * \* is Many (All). |  |
| 2a. | SELECT ‘Column1’, ‘Column2’, … , ‘Column\_n’  FROM ‘TableName’;  WHERE ‘Column’ = ‘Value’; | **Select Data (Certain Row)**   * List all rows of the table where ‘Column’ = ‘Value’. * WHERE ‘Column’ = ‘Value’ is the search condition and only list the row which satisfy the search condition. |
| 2b. | SELECT …  FROM …  WHERE … BETWEEN … AND … ; |  | --- |
| 2c. | SELECT …  FROM …  WHERE … IN … ; |  | --- |
| 2d. | SELECT …  FROM …  WHERE … LIKE … ; |  | --- |
| 2e. | SELECT …  FROM …  WHERE … IS NULL; |  | --- |
| 2f. | SELECT …  FROM …  WHERE … IS NOT NULL; |  | --- |
| 2g. | SELECT …  FROM …  WHERE … AND … ; |  | --- |
| 2h. | SELECT …  FROM …  WHERE … OR … ; |  | --- |
| 2i. | SELECT …  FROM …  WHERE … NOT … ; |  | --- |
|  | | | |
| 3. | SELECT DISTINCT …  FROM … ; |  | --- |
|  | | | |
| 4a. | SELECT …  FROM …  ORDER BY … ASC; | * 顺序 | --- |
| 4b. | SELECT …  FROM …  ORDER BY … DESC; | * 逆序 * SELECT \* FROM staff ORDER BY DoorplatNo, Postcode DESC;   会让DoorplatNo先顺序(Default)排列先， 然后重复的 DoorplatNo 会让Postcode根据逆序排列 | --- |
|  | TRUNCATE() |  |  |
|  | ROUND() |  |  |
|  | MOD() |  |  |
|  | | | |
| 5. | SELECT …  FROM …  HAVING … ; |  |  |
|  | MAX() |  |  |
|  | MIN() |  |  |
|  | AVG() |  |  |
|  | SUM() |  |  |
|  | COUNT() |  |  |
|  |  |  |  |
|  | IFNULL(‘expr1’, ‘expr2’) | * Return expr2 if expr1 is NULL. |  |
|  | IF(‘expr1’, ‘expr2’, ‘expr3’) | * Return expr2 if expr1 is true. * Return expr3 if expr1 is false. |  |
|  | NULLIF(‘expr1’, ‘expr2’) | * Return NULL to expr1 if expr1 equal to expr2. |  |
|  | COALESCE(‘expr1’, ‘expr2’, … , ‘expr\_n’) | * Return the first NOT NULL expr in the list. |  |
|  | | | |
| 6. | SELECT …  FROM …  ORDER BY … ; |  |  |

1. **Order By :**

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Apply** | **Explain** | **Example** |
| 1. |  |  |  |