SQL

Tuesday, July 2, 2024 1:58 PM

MySQL is a sodtware which uses SQL language to fetch data from a database
RDBMS 1.MySQl
2.Oracle
3.MS SQL Server
4.IBM
MySQL is Open Source used by Adobe ,Meta etc

Data types

1. int
2. varchar variable datatype (0-255)
3. char fixed size (0-255) can lead to wastage of memory
,but in case of varchar it is dynamic even though we can give it 25 bytes it will store according to the need.

4. tinytext 0-65535
5 .BLOB (0-65535) audio or video files
6. MEdiumText (0- 16777215)
7. MEDIUMBLOB (0-16777215)
8.LONGTEXT (0-4294967295)
9.LONGBLOB (0-4294967295)
10.TINYINT (-128 TO 127)
11.SMALLINT (-32768 TO 32767)

MEDIUMINT	integer(-8388608 to 8388607)
INT	integer(-2147483648 to 2147483647)
BIGINT	integer (-9223372036854775808 to 9223372036854775807)
FLOAT	Decimal with precision to 23 digits
DOUBLE	Decimal with 24 to 53 digits
DECIMAL	Double stored as string
DATE	YYYY-MM-DD
DATETIME	YYYY-MM-DD HH:MM:SS
TIMESTAMP	YYYYMMDDHHMMSS
TIME	HH:MM:SS
ENUM	One of the preset values
SET	One or many of the preset values
BOOLEAN	0/1
BIT	e.g., BIT(n), n upto 64, store values in bits.

SIGNED AND UNSIGNED
TINYINT ---> -128 to 127
UNSIGNED TINYINT (0-255)

CREATE TABLE NAME(COL1 INT,COL2 INT UNSIGNED)

Advance data types
JSON
Example:

```
CREATE TABLE NAME (
        COL1 JSON
        );
```

SQL : types of commands

1. DDL (DATA DEFINITION LANGUAGE)
    1. **CREATE** : create table,db,view
    2. **ALTER TABL**E modification in table structure
    3. **DROP** delete table ,Db ,view
    4. **TRUNCATE** : only remove tuples while maintaining the schema
    5. **RENAME** : rename DB name , table name, column name
2. DRL (Data Retrieval Language)/DQL
    1. **SELECT**
3. DML (data modification language)
    1. **INSERT** : insert data into relation
    2. **UPDAT**E : update relation data
    3. **DELETE** :delete rows from the relation
4. DCL(data control language)
    1. **GRANT** :access privileges to the DB
    2. **REVOKE**: revoke user access privileges
5. TCL(transaction Control language)

1. **START TRANSACTION** :begin a transaction
2. **COMMIT** : apply all the changes and end transaction
3. **ROLLBACK** : discard changes and end transaction
4. **SAVEPOINT** : checkout within the group of transactions in which to rollback

Managing DB (DDL)

1. Creation of database
    1. **CREATE DATABASE IF NOT EXISTS** db- name;
    2. **USE** db-name;
    3. **DROP DATABASES;**
    4. **SHOW DATABASES;**
    5. **SHOW TABLES;**

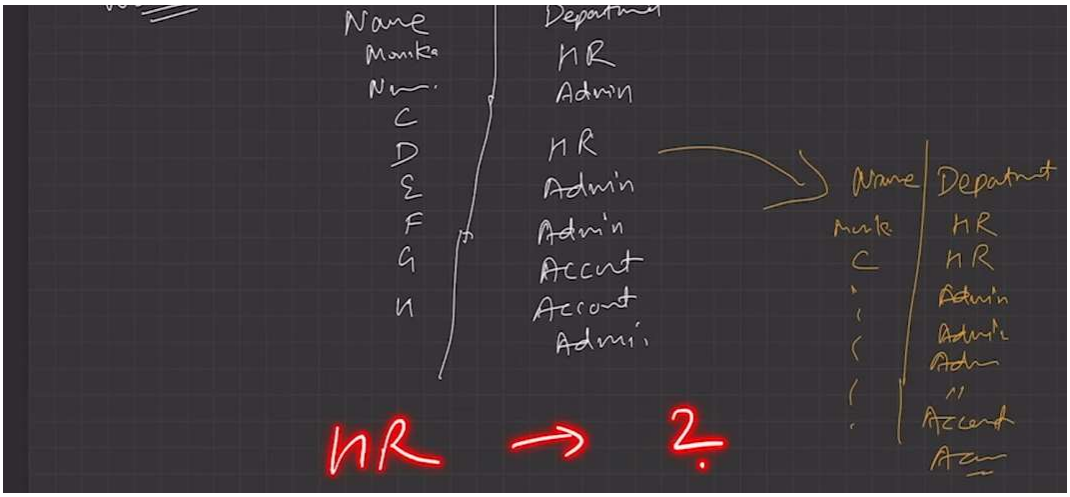
10 row(s) affected, 10 warning(s): 4095 Delimiter '.' in position 11 in datetime value '14-02-20 09.00.00' at row 1 is deprecated. Prefer the standard ':'. 4095 Delimiter '.' in position 11 in datetime value '14-02-20 09.00.00' at row 2 is deprecated. Prefer the standard ':'. 4095 Delimiter '.' in position 11 in datetime value '14-02-20 09.00.00' at row 3 is deprecated. Prefer the standard ':'. 4095 Delimiter '.' in position 11 in datetime value '14-02-20 09.00.00' at row 4 is deprecated. Prefer the standard ':'. 4095 Delimiter '.' in position 11 in datetime value '14-02-20 09.00.00' at row 5 is deprecated. Prefer the standard ':'. 4095 Delimiter '.' in position 11 in datetime value '14-02-20 09.00.00' at row 6 is deprecated. Prefer the standard ':'. 4095 Delimiter '.' in position 11 in datetime value '14-02-20 09.00.00' at row 7 is deprecated. Prefer the standard ':'. 4095 Delimiter '.' in position 11 in datetime value '14-02-20 09.00.00' at row 8 is deprecated. Prefer the standard ':'. 4095 Delimiter '.' in position 11 in datetime value '14-02-20 09.00.00' at row 9 is deprecated. Prefer the standard ':'. 4095 Delimiter '.' in position 11 in datetime value '14-02-20 09.00.00' at row 10 is deprecated. Prefer the standard ':'.
Records: 10 Duplicates: 0 Warnings: 10

DATA RETRIEVAL LANGUAGE (DRL)

1. Syntax: SELECT <set of column names> FROM <table_name>;
2. Order of execution from RIGHT to LEFT.
3. Q. Can we use SELECT keyword without using FROM clause?
    1. Yes, using DUAL Tables.
    2. Dual tables are dummy tables created by MySQL, help users to do certain obvious actions without referring to user defined tables.
    3. e.g., SELECT 55 + 11;
       SELECT now();
       SELECT ucase(); etc.
4. **WHERE**
    1. Reduce rows based on given conditions.
    2. E.g., SELECT * FROM customer WHERE age > 18;
5. **BETWEEN**
    1. SELECT * FROM customer WHERE age between 0 **AND** 100;
    2. In the above e.g., 0 and 100 are inclusive.
6. **IN**
    1. Reduces **OR** conditions;
    2. e.g., SELECT * FROM officers WHERE officer_name IN ('Lakshay', 'Maharana Pratap', 'Deepika');
7. **AND/OR/NOT**
    1. AND: WHERE cond1 AND cond2
    2. OR: WHERE cond1 OR cond2
    3. NOT: WHERE col_name NOT IN (1,2,3,4);
8. **IS NULL**
    1. e.g., SELECT * FROM customer WHERE prime_status is NULL;
9. **Pattern Searching / Wildcard ('%', '_')**
    1. '%', any number of character from 0 to n. Similar to '*' asterisk in regex.
    2. '_', only one character.
    3. SELECT * FROM customer WHERE name **LIKE** '%p_';
10. **ORDER BY**
    1. Sorting the data retrieved using **WHERE** clause.
    2. ORDER BY <column-name> DESC;
    3. DESC = Descending and ASC = Ascending
    4. e.g., SELECT * FROM customer ORDER BY name DESC;
11. **GROUP BY**
    1. GROUP BY Clause is used to collect data from multiple records and group the result by one or more column. It is generally used in a SELECT statement.
    2. Groups into category based on column given.
    3. SELECT c1, c2, c3 FROM sample_table WHERE cond GROUP BY c1, c2, c3.
    4. All the column names mentioned after SELECT statement shall be repeated in GROUP BY, in order to successfully execute the query.
    5. Used with aggregation functions to perform various actions.
        1. COUNT()
        2. SUM()
        3. AVG()
        4. MIN()
        5. MAX()

6. **IN**
    1. Reduces **OR** conditions;
    2. e.g., SELECT * FROM officers WHERE officer_name IN ('Lakshay', 'Maharana Pratap', 'Deepika');
7. **AND/OR/NOT**
    1. AND: WHERE cond1 AND cond2
    2. OR: WHERE cond1 OR cond2
    3. NOT: WHERE col_name NOT IN (1,2,3,4);
8. **IS NULL**
    1. e.g., SELECT * FROM customer WHERE prime_status is NULL;
9. **Pattern Searching / Wildcard ('%', '_')**
    1. '%', any number of character from 0 to n. Similar to '*' asterisk in regex.
    2. '_', only one character.
    3. SELECT * FROM customer WHERE name **LIKE** '%p_';
10. **ORDER BY**
    1. Sorting the data retrieved using **WHERE** clause.
    2. ORDER BY <column-name> DESC;
    3. DESC = Descending and ASC = Ascending
    4. e.g., SELECT * FROM customer ORDER BY name DESC;
11. **GROUP BY**
    1. GROUP BY Clause is used to collect data from multiple records and group the result by one or more column. It is generally used in a SELECT statement.
    2. Groups into category based on column given.
    3. SELECT c1, c2, c3 FROM sample_table WHERE cond GROUP BY c1, c2, c3.
    4. All the column names mentioned after SELECT statement shall be repeated in GROUP BY, in order to successfully execute the query.
    5. Used with aggregation functions to perform various actions.
        1. COUNT()
        2. SUM()
        3. AVG()
        4. MIN()
        5. MAX()

Working of group by



4. e.g., SELECT * FROM customer ORDER BY name DESC;

11. **GROUP BY**
    1. GROUP BY Clause is used to collect data from multiple records and group the result by one or more column. It is generally used in a SELECT statement.
    2. Groups into category based on column given.
    3. SELECT c1, c2, c3 FROM sample_table WHERE cond GROUP BY c1, c2, c3.
    4. All the column names mentioned after SELECT statement shall be repeated in GROUP BY, in order to successfully execute the query.
    5. Used with aggregation functions to perform various actions.
        1. COUNT()
        2. SUM()
        3. AVG()
        4. MIN()
        5. MAX()

12. **DISTINCT**
    1. Find distinct values in the table.
    2. SELECT DISTINCT(col_name) FROM table_name;
    3. GROUP BY can also be used for the same
        1. "Select col_name from table GROUP BY col_name;" same output as above DISTINCT query.