Mastek

RDBMS

using MySQL Database

MASTEK INSTITUTE
FOR LEARNING & DEVELOPMENT

Session Objectives for Database Concepts

- Introduction to Database
- Introduction to Structured Query Language
- Creating and Managing Database Objects
- Managing Data in Database objects
- Fetching Data from Database Objects

RDBMS

- What is Data?
- What is Database?
- What is DBMS?
- What is RDBMS?

RAM 20 60 70 AB 99

- Data represents unorganized and unprocessed facts.

What is Database?

- A database is a collection of information that is organized so that it can easily be accessed, managed, and updated.

- In one view, databases can be classified according to types of content

What is DBMS?

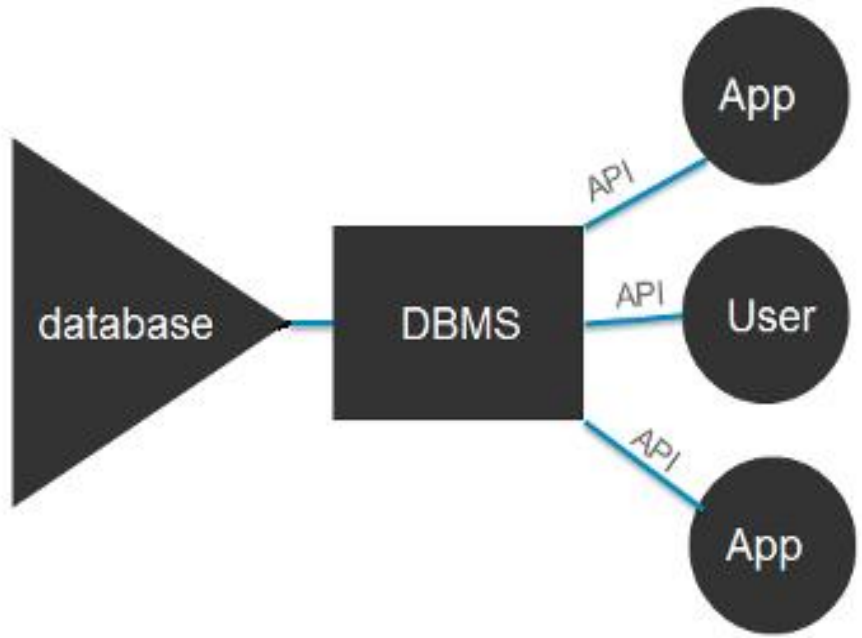
- A database management system (DBMS) is a collection of programs that enables you to store, modify, and extract information from a database.

Relational Database

- A relational database is a set of tables containing data fitted into predefined categories.
- Each table contains one or more data categories in columns.
- Each row contains a unique instance of data for the categories defined by the columns.

Relational Database

- The relational database was invented by E. F. Codd at IBM in 1970.


App
API
database
DBMS
API
User
API
App

Advantages of RDBMS

- Data abstraction and independence
- Data security
- A locking mechanism for concurrent access
- Robust data integrity capabilities
- Logging and auditing of activity
- Simple access using a standard application programming interface (API)
- Backup and recovery

MySQL Introduction

- MySQL is a database management system
- SQL stands for the Structured Query Language. It defines how to insert, retrieve, modify and delete data
- Free from www.mysql.com
- Reference sites
  - NASA, Yahoo!, Compaq, Motorola

How MySQL stores data (by default)

- A MySQL server can store several databases
- Databases are stored as directories
  - Default is at /usr/local/mysql/var/
- Tables are stored as files inside each database (directory)
- For each table, it has three files:
  - table.FRM file containing information about the table structure
  - table.MYD file containing the row data
  - table.MYI containing any indexes belonging with this table, as well as some statistics about the table.

Basic MySQL Operations

- **Data Definition Language**: Create, Alter, Drop Table
- **Data Manipulation Language**: Insert, Update, Delete records
- **Data Selection**: Select, Join records, Count, Like, Order by, Group by
- Advanced Fetch using sub-queries, Managing Views

Basic Syntax

```
CREATE TABLE [schema.]table
        ( { column datatype [DEFAULT expr] [column_constraint] ...
          | table_constraint}
        [, { column datatype [DEFAULT expr] [column_constraint] ...
          | table_constraint} ]...)
        [AS subquery]
```

Constraints

- Constraints are the set of rules defined in Oracle tables to ensure data integrity.
- These rules are enforced placed for each column or set of columns.
- Whenever the table participates in data action, these rules are validated and raise exception upon violation.

Constraints

- The available constraint types are
- NOT NULL,
- Primary Key,
- Unique,
- Foreign Key.

Foreign Key

Syntax :

COLUMN [data type] [CONSTRAINT] [constraint name] [REFERENCES] [table name (column name)]

Ex:

```
CREATE TABLE EMP
(
.........
DEPARTMENT_ID NUMBER
CONSTRAINT DEPT_ID_FK REFERENCES DEPARTMENT(DEPARTMENT_ID),
.....
);
```

ON DELETE CASCADE

- If a foreign key is defined on the column in child table then Oracle does not allow the parent row to be deleted,if it contains any child rows.
- However,if ON DELETE CASCADE option is given at the time of defining foreign key,Oracle deletes all child rows while parent row is being deleted.
- Similarly,ON DELETE SET NULL indicates that when a row in the parent table is deleted, the foreign key values are set to null.

ON DELETE CASCADE - Example

```
CREATE TABLE EMP
(
.........
DEPARTMENT_ID NUMBER
CONSTRAINT DEPT_ID_FK REFERENCES
DEPARTMENT(DEPARTMENT_ID)
    ON DELETE CASCADE,
.....
);
```

ALTER TABLE statement

- We can change the table definition after creating it.
- A DBA can make changes to the table structure or column definitions.
- RENAME Table or Column
- ADD Column
- Drop Column
- Add Constraints
- Drop Constraints

DROP TABLE statement

It remove a table from the database.

Syntax :

DROP TABLE [TABLE NAME]

Data Manipulation Language (DML) Statements

- SELECT
- INSERT
- UPDATE
- DELETE

Transaction Control Statements

- COMMIT
- ROLLBACK
- SAVEPOINT

Insert Record

- INSERT INTO table_name SET col_name1=value1, col_name2=value2, col_name3=value3, …
- Example

Student_ID	Name	Major	Grade
101	Shannon	BCB	A

- mysql> INSERT INTO student SET student_ID=101, name='Shannon', major='BCB', grade='A';
- Query OK, 1 row affected (0.00 sec)

Insert Record

- INSERT INTO table_name (Col1,Col2,Col3) Values(value1,value2,value3, …
- Example

- mysql> INSERT INTO student (student_ID,name,major,grade) values(101,’Shannon’,’BCB’,’A’);

Student_ID	Name	Major	Grade
101	Shannon	BCB	A

Update Record

UPDATE table_name

SET which columns to change

WHERE condition

- Example

```
mysql> UPDATE student SET grade='B' WHERE name='Shannon';
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
mysql> SELECT * FROM student WHERE name='Shannon';
+-----------+--------------+--------+--------+
| name      | student_ID | major | grade |
+-----------+--------------+--------+--------+
| Shannon |       101     | BCB   | B       |
+-----------+--------------+--------+--------+
1 row in set (0.00 sec)
```

Delete Record

DELETE FROM table_name WHERE condition

- Example

DELETE FROM student WHERE name='Shannon';

Query OK, 1 row affected (0.00 sec)

DELETE FROM student;

Will delete ALL student records!

Basics of SELECT statement

- SELECT *Column_List*
- FROM *Table_Name*
- WHERE *Filter_Condition*
- ORDER BY *Column_List*
- LIMIT *Row_Limit*

WHERE clause

- **SELECT * FROM TableName**
- **SELECT Col1, Col2, Col3 FROM TableName**
- **SELECT Col1, Col2, Col3 FROM TableName WHERE Col1 = ‘SomeValue’**

WHERE clause –Comparison & Logical Operators

- **WHERE clause comparison operators**Equal (=)
  - Less than (<)
  - Greater than (>)
  - Less than or Equal to (<=)
  - Greater than or Equal to (>=)
  - Not equal (<> or !=)
- **WHERE clause logical operators**
  - AND
  - OR
  - NOT

WHERE clause –Other Operator

- **WHERE clause other operators**
  - IN (and NOT IN)
  - BETWEEN (and NOT BETWEEN)
  - LIKE (and NOT LIKE)Wildcard -%
  - Wildcard -_
- **NULL clause**
  - IS NULL clause
  - IS NOT NULL clause

More Table Retrieval

- OR
  - mysql> select name from student where major = 'BCB' OR major = 'CS';
- COUNT (Count query results)
  - mysql> select count(name) from student where major = 'BCB' OR major = 'CS';
- ORDER BY (Sort query results)
  - mysql> select name from student where major = 'BCB' OR major = 'CS‘ ORDER BY name;
  - mysql> select name from student where major = 'BCB' OR major = 'CS‘ ORDER BY name DESC;
  - mysql> select * from student where major = 'BCB' OR major = 'CS‘ ORDER BY student_id ASC, name DESC
- LIKE (Pattern matching)
  - mysql> select name from student where name LIKE "J%";
- DISTINCT (Remove duplicates)
  - mysql> select major from student;
  - mysql> select DISTINCT major from student;

Group By

Cluster query results based on different groups
Example

- mysql> select major, count(*) from student GROUP BY major;
- +---------+----------+
- | major | count(*) |
- +---------+----------+
- | BBMB | 3 |
- | BCB | 3 |
- | Stat | 2 |
- +---------+-----------+

NULL

No Value

Can not use the usual comparison operators (>, =, != …)

Use IS or IS NOT operators to compare with

- Example

- mysql> select name from student where project_ID = NULL;

- mysql> select name from student where project_ID IS NULL;

Table Join

- Retrieve information from multiple tables
- Example
  - Which BCB students chose level-4 project?

```
mysql> select s.name from student s, project p
    -> where s.project_ID = p.project_ID
    -> and s.major='BCB' and p.level=4;
+------------+
| name       |
+------------+
| Stephen |
+------------+
1 row in set (0.00 sec)
```

THANK YOU