23 July 2012

**Database Basics**

**Aim:**

To study a relational database and study Data Definition Language (DDL) and Data Manipulation Language (DML)

**Theory:**

Data Manipulation Language (DML) is a language that enables users to access or manipulate data as organized by the appropriate data model.

The typed of access are:

1. Retrieval of information stored in the database.
2. Insertion of new information into the database.
3. Deletion of information from the database.
4. Modification of information stored in the database.

SQL supports following clauses:

* INSERT adds rows (formally tuples) to an existing table.
* UPDATE modifies a set of existing table rows.
* DELETE removes existing rows from a table.
* MERGE is used to combine the data of multiple tables. It combines the INSERT and UPDATE elements.

There are basically two types:

1. Procedural DMLs require a user to specify what data are needed and how to get those data.
2. Declarative DMLs (also referred to as non procedural DMLs) require a user to specify what data are needed without specifying how to get those data.

Data Definition Language (DDL) is a language that expresses the database schema by a set of definitions. The DDL is also used to specify additional properties of the data.

* CREATE creates an object (a table, for example) in the database.
* ALTER modifies the structure of an existing object in various ways, for example, adding a column to an existing table or a constraint.
* DROP deletes an object in the database.

The storage structure and access methods used by the database system are specified by a set of statements in a special type of DDL called a data storage and definition language. These statements define the implementation detail of the database schemas, which are usually hidden from the user.

A query includes a list of columns to be included in the final result immediately following the SELECT keyword. An asterisk ("\*") can also be used to specify that the query should return all columns of the queried tables. SELECT is the most complex statement in SQL, with optional keywords and clauses that include:

* The FROM clause which indicates the table(s) from which data is to be retrieved. The FROM clause can include optional JOIN sub clauses to specify the rules for joining tables.
* The WHERE clause includes a comparison predicate, which restricts the rows returned by the query.
* The GROUP BY clause is used to project rows having common values into a smaller set of rows. GROUP BY is often used in conjunction with SQL aggregation functions or to eliminate duplicate rows from a result set. The WHERE clause is applied before the GROUP BY clause.
* The HAVING clause includes a predicate used to filter rows resulting from the GROUP BY clause.
* The ORDER BY clause identifies which columns are used to sort the resulting data, and in which direction they should be sorted (options are ascending or descending).

Enter password:

Welcome to the MySQL monitor. Commands end with ; or \g.

Your MySQL connection id is 3

Server version: 5.5.24-log MySQL Community Server (GPL)

Copyright (c) 2000, 2011, Oracle and/or its affiliates. All rights reserved.

mysql> create database db1;

mysql> use db1;

mysql> create table account (accno int, branchname varchar(32), balance int);

mysql> create table branch (branchname varchar(32), branchcity varchar(32), assets int);

mysql> create table customer (custname varchar(32), custaddress varchar(32), custcity varchar(32));

mysql> create table depositor (custname varchar(32), accno int);

mysql> create table loan (loanno int, branchname varchar(32), amount int);

mysql> create table borrower (custname varchar(32), loanno int);

mysql> insert into account values(101,'madgaon',10000);

mysql> insert into account values(102,'ponda',12000);

mysql> insert into account values(103,'farmagudi',1500);

mysql> insert into depositor values('mr. pqr',101);

mysql> insert into depositor values('mr. xyz',102);

mysql> insert into depositor values('mr. ijk',103);

mysql> insert into loan values(401,'panaji',5000);

mysql> insert into loan values(402,'farmagudi',4600);

mysql> insert into borrower values ('mr. abc',401);

mysql> insert into borrower values ('mr. ijk',402);

mysql> insert into branch values('madgaon','goa',300000);

mysql> insert into branch values('ponda','goa',450000);

mysql> insert into branch values('panaji','goa',550000);

mysql> insert into branch values('farmagudi','goa',250000);

mysql> insert into customer values('mr. xyz','xyz address','ponda');

mysql> insert into customer values('mr. pqr','pqr address','madgaon');

mysql> insert into customer values('mr. abc','abc house','panaji');

mysql> insert into customer values('mr. ijk','ijk address','farmagudi');

mysql> select \* from account;

+-------+------------+---------+

| accno | branchname | balance |

+-------+------------+---------+

| 101 | madgaon | 10000 |

| 102 | ponda | 12000 |

| 103 | farmagudi | 1500 |

+-------+------------+---------+

mysql> select \* from branch;

+------------+------------+--------+

| branchname | branchcity | assets |

+------------+------------+--------+

| madgaon | goa | 300000 |

| ponda | goa | 450000 |

| panaji | goa | 550000 |

| farmagudi | goa | 250000 |

+------------+------------+--------+

mysql> select \* from customer;

+----------+-------------+-----------+

| custname | custaddress | custcity |

+----------+-------------+-----------+

| mr. xyz | xyz address | ponda |

| mr. pqr | pqr address | madgaon |

| mr. abc | abc house | panaji |

| mr. ijk | ijk address | farmagudi |

+----------+-------------+-----------+

mysql> select \* from depositor;

+----------+-------+

| custname | accno |

+----------+-------+

| mr. pqr | 101 |

| mr. xyz | 102 |

| mr. ijk | 103 |

+----------+-------+

mysql> select \* from loan;

+--------+------------+--------+

| loanno | branchname | amount |

+--------+------------+--------+

| 401 | panaji | 5000 |

| 402 | farmagudi | 4600 |

+--------+------------+--------+

mysql> select \* from borrower;

+----------+--------+

| custname | loanno |

+----------+--------+

| mr. abc | 401 |

| mr. ijk | 402 |

+----------+--------+

***Find loan no, loan amt and customer names for all loans at Farmagudi branch***

mysql> select loan.loanno, loan.amount, custname from loan,borrower where loan.loanno=borrower.loanno and branchname='farmagudi';

+--------+--------+----------+

| loanno | amount | custname |

+--------+--------+----------+

| 402 | 4600 | mr. ijk |

+--------+--------+----------+

***Find names of all branches that have assets greater than at least one branch located in Ponda city***

mysql> select branchname from branch where assets > some(select min(assets) from branch where branchname='ponda');

+------------+

| branchname |

+------------+

| panaji |

+------------+

***Find names of all customer s whose address includes substring ‘house’***

mysql> select custname from customer where custaddress like '%house%';

+----------+

| custname |

+----------+

| mr. abc |

+----------+

***List in alphabetical order all customers who have an account at farmagudi branch***

mysql> select custname from depositor,account where account.accno=depositor.accno and branchname='farmagudi' order by custname;

+----------+

| custname |

+----------+

| mr. ijk |

+----------+

***Find average account balance at ponda branch***

mysql> select avg(balance)as avg1 from account where branchname='ponda';

+------------+

| avg1 |

+------------+

| 12000.0000 |

+------------+

***Find all customers who have both loan and account at bank***

mysql> select depositor.custname from depositor,borrower where borrower.custname=depositor.custname;

+----------+

| custname |

+----------+

| mr. ijk |

+----------+

***Find max balance at each branch***

mysql> select branchname, max(balance) from account group by branchname;

+------------+--------------+

| branchname | max(balance) |

+------------+--------------+

| farmagudi | 1500 |

| madgaon | 10000 |

| ponda | 12000 |

+------------+--------------+

***Find number of depositors for each branch***

mysql> select branchname,count(\*) from account, depositor where account.accno=depositor.accno group by branchname;

+------------+----------+

| branchname | count(\*) |

+------------+----------+

| farmagudi | 1 |

| madgaon | 1 |

| ponda | 1 |

+------------+----------+

***Retrieve average account balance for each branch provided it is greater than 5,000***

mysql> select branchname,avg(balance) as avg1 from account group by branchname having avg(balance)>5000;

+------------+------------+

| branchname | avg1 |

+------------+------------+

| madgaon | 10000.0000 |

| ponda | 12000.0000 |

+------------+------------+