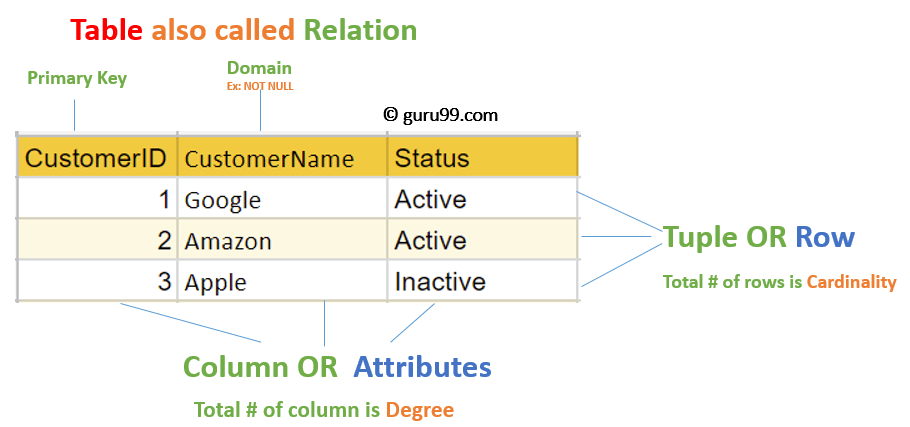
* **Relational Model (RM)** represents the database as a collection of relations. A relation is nothing but a table of values. Every row in the table represents a collection of related data values. These rows in the table denote a real-world entity or relationship.
* The table name and column names are helpful to interpret the meaning of values in each row. The data are represented as a set of relations. In the relational model, data are stored as tables. However, the physical storage of the data is independent of the way the data are logically organized.
* STRUCTURED QUERY LANGUAGE/SQL is the standard language for database management.
* **SQL** is a database language designed for the retrieval and management of data in a relational database.
* All the RDBMS systems like MySQL, MS Access, Oracle, Sybase, Postgres, and SQL Server use SQL as their standard database language. SQL programming language uses various commands for different operations. We will learn about the like DCL, TCL, DQL, DDL and DML commands in SQL with examples.

**Relational Model Terms**

1. **Attribute:** Each column in a Table. Attributes are the properties which define a relation. e.g., Student\_Rollno, NAME,etc.
2. **Tables** – In the Relational model the, relations are saved in the table format. It is stored along with its entities. A table has two properties rows and columns. Rows represent records and columns represent attributes.
3. **Tuple** – It is nothing but a single row of a table, which contains a single record.
4. **Relation Schema:** A relation schema represents the name of the relation with its attributes.
5. **Degree:** The total number of attributes which in the relation is called the degree of the relation.
6. **Cardinality:**Total number of rows present in the Table.

[](https://www.guru99.com/images/1/091318_0803_RelationalD1.png)

MySQL Relational Database is an assemblage of relational data that is structured or organized in the form of tables, columns, and rows, where tables represent the objects, columns represent the fields and rows represent the records.

**Relational Integrity Constraints**

Relational Integrity constraints in DBMS are referred to conditions which must be present for a valid relation. These Relational constraints in DBMS are derived from the rules in the mini-world that the database represents.

There are many types of Integrity Constraints in DBMS. Constraints on the Relational database management system is mostly divided into three main categories are:

1. Domain Constraints
2. Key Constraints
3. Referential Integrity Constraints

**Domain Constraints**

Domain constraints can be violated if an attribute value is not appearing in the corresponding domain or it is not of the appropriate data type.

Domain constraints specify that within each tuple, and the value of each attribute must be unique. This is specified as data types which include standard data types integers, real numbers, characters, Booleans, variable length strings, etc.

**Example:**

Create DOMAIN CustomerName

CHECK (value not NULL)

The example shown demonstrates creating a domain constraint such that CustomerName is not NULL

**Key Constraints**

An attribute that can uniquely identify a tuple in a relation is called the key of the table. The value of the attribute for different tuples in the relation has to be unique.

**Example:**

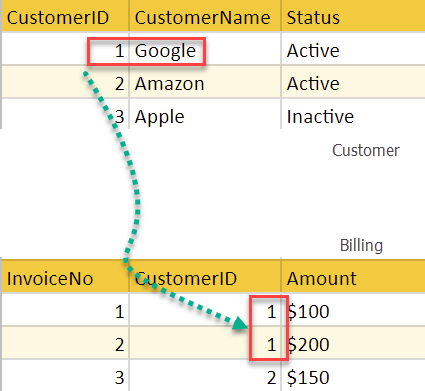
In the given table, CustomerID is a key attribute of Customer Table. It is most likely to have a single key for one customer, CustomerID =1 is only for the CustomerName =" Google".

|  |  |  |
| --- | --- | --- |
| CustomerID | CustomerName | Status |
| 1 | Google | Active |
| 2 | Amazon | Active |
| 3 | Apple | Inactive |
|  |  |  |

**Referential Integrity Constraints**

Referential Integrity constraints in DBMS are based on the concept of Foreign Keys. A foreign key is an important attribute of a relation which should be referred to in other relationships. Referential integrity constraint state happens where relation refers to a key attribute of a different or same relation. However, that key element must exist in the table.

**Example:**

[](https://www.guru99.com/images/1/091318_0803_RelationalD2.png)

In the above example, we have 2 relations, Customer and Billing.

Tuple for CustomerID =1 is referenced twice in the relation Billing. So we know CustomerName=Google has billing amount $300

**Operations in Relational Model**

Four basic update operations performed on relational database model are

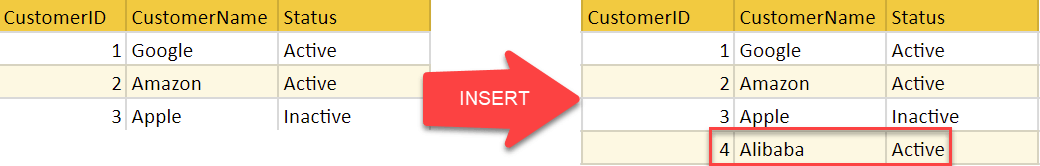
Insert, update, delete and select.

* Insert is used to insert data into the relation
* Delete is used to delete tuples from the table.
* Modify allows you to change the values of some attributes in existing tuples.
* Select allows you to choose a specific range of data.

Whenever one of these operations are applied, integrity constraints specified on the relational database schema must never be violated.

**Insert Operation**

The insert operation gives values of the attribute for a new tuple which should be inserted into a relation.

[](https://www.guru99.com/images/1/091318_0803_RelationalD3.png)

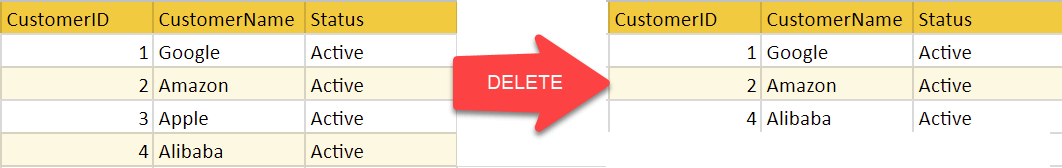
**Update Operation**

You can see that in the below-given relation table CustomerName= 'Apple' is updated from Inactive to Active.

[](https://www.guru99.com/images/1/091318_0803_RelationalD4.png)

**Delete Operation**

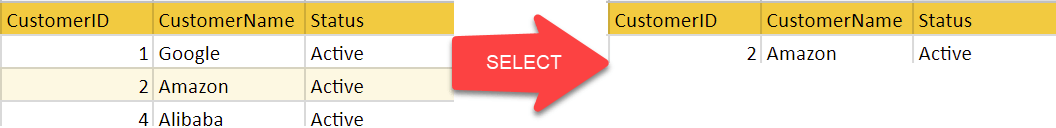
To specify deletion, a condition on the attributes of the relation selects the tuple to be deleted.

[](https://www.guru99.com/images/1/091318_0803_RelationalD5.png)

In the above-given example, CustomerName= "Apple" is deleted from the table.

The Delete operation could violate referential integrity if the tuple which is deleted is referenced by foreign keys from other tuples in the same database.

**Select Operation**

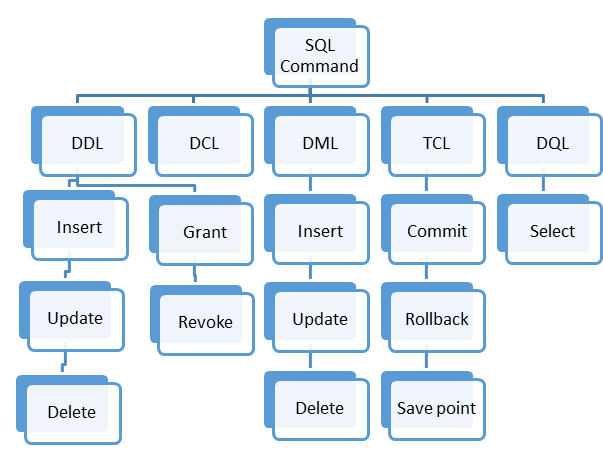
[](https://www.guru99.com/images/1/091318_0803_RelationalD6.png)

In the above-given example, CustomerName="Amazon" is selected

**Types of SQL**

Here are five types of widely used SQL queries.

* Data Definition Language (DDL)
* Data Manipulation Language (DML)
* Data Control Language(DCL)
* Transaction Control Language(TCL)
* Data Query Language (DQL)

[](https://www.guru99.com/images/2/041720_1130_SQLCommands1.png)

Types of SQL

Data Definition Language helps you to define the database structure or schema. Let's learn about DDL commands with syntax.

Five types of DDL commands in SQL are:

**CREATE**

CREATE statements is used to define the database structure schema:

**Syntax:**

CREATE TABLE TABLE\_NAME (COLUMN\_NAME DATATYPES[,....]);

**For example**:

Create database university;

Create table students;

Create view for\_students;

**DROP**

Drops commands remove tables and databases from RDBMS.

Syntax

DROP TABLE ;

**For example:**

Drop object\_type object\_name;

Drop database university;

Drop table student;

**ALTER**

Alters command allows you to alter the structure of the database.

**Syntax:**

To add a new column in the table

ALTER TABLE table\_name ADD column\_name COLUMN-definition;

To modify an existing column in the table:

ALTER TABLE MODIFY(COLUMN DEFINITION....);

**For example:**

Alter table guru99 add subject varchar;

**TRUNCATE:**

This command used to delete all the rows from the table and free the space containing the table.

**Syntax:**

TRUNCATE TABLE table\_name;

**Example:**

TRUNCATE table students;

**What is Data Manipulation Language?**

Data Manipulation Language (DML) allows you to modify the database instance by inserting, modifying, and deleting its data. It is responsible for performing all types of data modification in a database.

There are three basic constructs which allow database program and user to enter data and information are:

Here are some important DML commands in SQL:

* INSERT
* UPDATE
* DELETE

**INSERT:**

This is a statement is a SQL query. This command is used to insert data into the row of a table.

**Syntax:**

INSERT INTO TABLE\_NAME (col1, col2, col3,.... col N)

VALUES (value1, value2, value3, .... valueN);

Or

INSERT INTO TABLE\_NAME

VALUES (value1, value2, value3, .... valueN);

**For example:**

INSERT INTO students (RollNo, FIrstName, LastName) VALUES ('60', 'Tom', Erichsen');

**UPDATE:**

This command is used to update or modify the value of a column in the table.

**Syntax:**

UPDATE table\_name SET [column\_name1= value1,...column\_nameN = valueN] [WHERE CONDITION]

**For example:**

UPDATE students

SET FirstName = 'Jhon', LastName= 'Wick'

WHERE StudID = 3;

**DELETE:**

This command is used to remove one or more rows from a table.

**Syntax:**

DELETE FROM table\_name [WHERE condition];

**For example:**

DELETE FROM students

WHERE FirstName = 'Jhon';

**What is DCL?**

DCL (Data Control Language) includes commands like GRANT and REVOKE, which are useful to give "rights & permissions." Other permission controls parameters of the database system.

**Examples of DCL commands:**

Commands that come under DCL:

* Grant
* Revoke

**Grant:**

This command is use to give user access privileges to a database.

**Syntax:**

GRANT SELECT, UPDATE ON MY\_TABLE TO SOME\_USER, ANOTHER\_USER;

**For example:**

GRANT SELECT ON Users TO'Tom'@'localhost;

**Revoke:**

It is useful to back permissions from the user.

**Syntax:**

REVOKE privilege\_nameON object\_nameFROM {user\_name |PUBLIC |role\_name}

**For example:**

REVOKE SELECT, UPDATE ON student FROM BCA, MCA;

**What is TCL?**

Transaction control language or TCL commands deal with the transaction within the database.

**Commit**

This command is used to save all the transactions to the database.

**Syntax:**

Commit;

**For example:**

DELETE FROM Students

WHERE RollNo =25;

COMMIT;

**Rollback**

Rollback command allows you to undo transactions that have not already been saved to the database.

**Syntax:**

ROLLBACK;

**Example:**

DELETE FROM Students

WHERE RollNo =25;

**SAVEPOINT**

This command helps you to sets a savepoint within a transaction.

**Syntax:**

SAVEPOINT SAVEPOINT\_NAME;

**Example:**

SAVEPOINT RollNo;

**What is DQL?**

Data Query Language (DQL) is used to fetch the data from the database. It uses only one command:

**SELECT:**

This command helps you to select the attribute based on the condition described by the WHERE clause.

**Syntax:**

SELECT expressions

FROM TABLES

WHERE conditions;

**For example:**

SELECT FirstName

FROM Student

WHERE RollNo > 15;

**Summary:**

* SQL is a database language designed for the retrieval and management of data in a relational database.
* It helps users to access data in the RDBMS system
* In the year 1974, the term Structured Query Language appeared
* Five types of SQL queries are 1) Data Definition Language (DDL) 2) Data Manipulation Language (DML) 3) Data Control Language(DCL) 4) Transaction Control Language(TCL) and, 5) Data Query Language (DQL)
* Data Definition Language(DDL) helps you to define the database structure or schema.
* Data Manipulation Language (DML) allows you to modify the database instance by inserting, modifying, and deleting its data.
* DCL (Data Control Language) includes commands like GRANT and REVOKE, which are useful to give "rights & permissions."
* Transaction control language or TCL commands deal with the transaction within the database.
* Data Query Language (DQL) is used to fetch the data from the [database](https://www.guru99.com/introduction-to-database-sql.html).

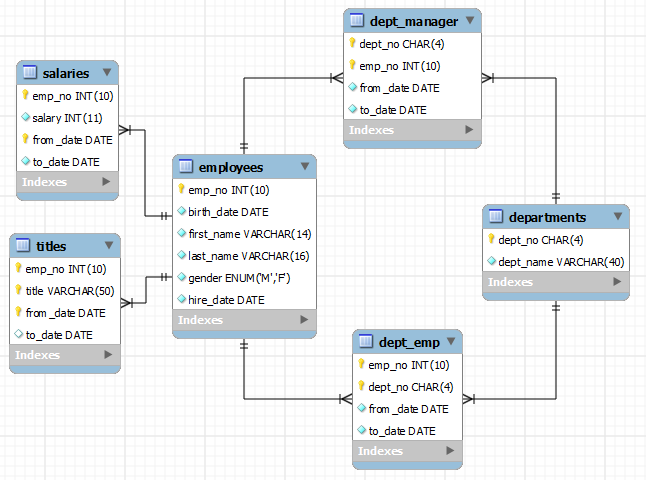
### MySQL's Sample Employee Database

**Reference:** MySQL's Sample Employees Database @ <http://dev.mysql.com/doc/employee/en/index.html>.

This is a rather simple database with 6 tables but with millions of records.

#### 1.1  Database and Tables

There are 6 tables as follows:



##### Table "employees"

CREATE TABLE **employees** (

emp\_no INT NOT NULL, -- UNSIGNED AUTO\_INCREMENT??

birth\_date DATE NOT NULL,

first\_name VARCHAR(14) NOT NULL,

last\_name VARCHAR(16) NOT NULL,

gender ENUM ('M','F') NOT NULL, -- Enumeration of either 'M' or 'F'

hire\_date DATE NOT NULL,

PRIMARY KEY (emp\_no) -- Index built automatically on primary-key column

-- INDEX (first\_name)

-- INDEX (last\_name)

);

There are 300,024 records for this table.

##### Table "departments"

CREATE TABLE **departments** (

dept\_no CHAR(4) NOT NULL, -- in the form of 'dxxx'

dept\_name VARCHAR(40) NOT NULL,

PRIMARY KEY (dept\_no), -- Index built automatically

UNIQUE KEY (dept\_name) -- Build INDEX on this unique-value column

);

The keyword KEY is synonym to INDEX. An INDEX can be built on unique-value column (UNIQUE KEY or UNIQUE INDEX) or non-unique-value column (KEY or INDEX). Indexes greatly facilitates fast search. However, they deplete the performance in INSERT, UPDATE and DELETE. Generally, relational databases are optimized for retrievals, and NOT for modifications.

There are 9 records for this table.

##### Table "dept\_emp"

Junction table to support between many-to-many relationship between employees and departments. A department has many employees. An employee can belong to different department at different dates, and possibly concurrently.

CREATE TABLE **dept\_emp** (

emp\_no INT NOT NULL,

dept\_no CHAR(4) NOT NULL,

from\_date DATE NOT NULL,

to\_date DATE NOT NULL,

KEY (emp\_no), -- Build INDEX on this non-unique-value column

KEY (dept\_no), -- Build INDEX on this non-unique-value column

FOREIGN KEY (emp\_no) REFERENCES employees (emp\_no) ON DELETE CASCADE,

-- Cascade DELETE from parent table 'employee' to this child table

-- If an emp\_no is deleted from parent 'employee', all records

-- involving this emp\_no in this child table are also deleted

-- ON UPDATE CASCADE??

FOREIGN KEY (dept\_no) REFERENCES departments (dept\_no) ON DELETE CASCADE,

-- ON UPDATE CASCADE??

PRIMARY KEY (emp\_no, dept\_no)

-- Might not be unique?? Need to include from\_date

);

The foreign keys have ON DELETE reference action of CASCADE. If a record having a particular key-value from the parent table (employees and departments) is deleted, all the records in this child table having the same key-value are also deleted. Take note that the default ON DELETE reference action of is RESTRICTED, which disallows DELETE on the parent record, if there are matching records in the child table.

There are two reference actions: ON DELETE and ON UPDATE. The ON UPDATE reference action of is defaulted to RESTRICT (or disallow). It is more meaningful to set ON UPDATE to CASCADE, so that changes in parent table (e.g., change in emp\_no and dept\_no) can be cascaded down to the child table(s).

There are 331,603 records for this table.

##### Table "dept\_manager"

join table to support between many-to-many relationship between employees and departments. Same structure as dept\_emp.

CREATE TABLE **dept\_manager** (

dept\_no CHAR(4) NOT NULL,

emp\_no INT NOT NULL,

from\_date DATE NOT NULL,

to\_date DATE NOT NULL,

KEY (emp\_no),

KEY (dept\_no),

FOREIGN KEY (emp\_no) REFERENCES employees (emp\_no) ON DELETE CASCADE,

-- ON UPDATE CASCADE??

FOREIGN KEY (dept\_no) REFERENCES departments (dept\_no) ON DELETE CASCADE,

PRIMARY KEY (emp\_no, dept\_no) -- might not be unique?? Need from\_date

);

There are 24 records for this table.

##### Table "titles"

There is a one-to-many relationship between employees and titles. One employee has many titles (concurrently or at different dates). A titles record refers to one employee (via emp\_no).

CREATE TABLE **titles** (

emp\_no INT NOT NULL,

title VARCHAR(50) NOT NULL,

from\_date DATE NOT NULL,

to\_date DATE,

KEY (emp\_no),

FOREIGN KEY (emp\_no) REFERENCES employees (emp\_no) ON DELETE CASCADE,

-- ON UPDATE CASCADE??

PRIMARY KEY (emp\_no, title, from\_date)

-- This ensures unique combination.

-- An employee may hold the same title but at different period

);

There are 443,308 records for this table.

##### Table "salaries"

Similar structure to titles table. One-to-many relationship between employees and salaries.

CREATE TABLE **salaries** (

emp\_no INT NOT NULL,

salary INT NOT NULL,

from\_date DATE NOT NULL,

to\_date DATE NOT NULL,

KEY (emp\_no),

FOREIGN KEY (emp\_no) REFERENCES employees (emp\_no) ON DELETE CASCADE,

PRIMARY KEY (emp\_no, from\_date)

);

There are 2,844,047 records for this table.

#### 1.2