

Chapter 12

DBMS Concepts

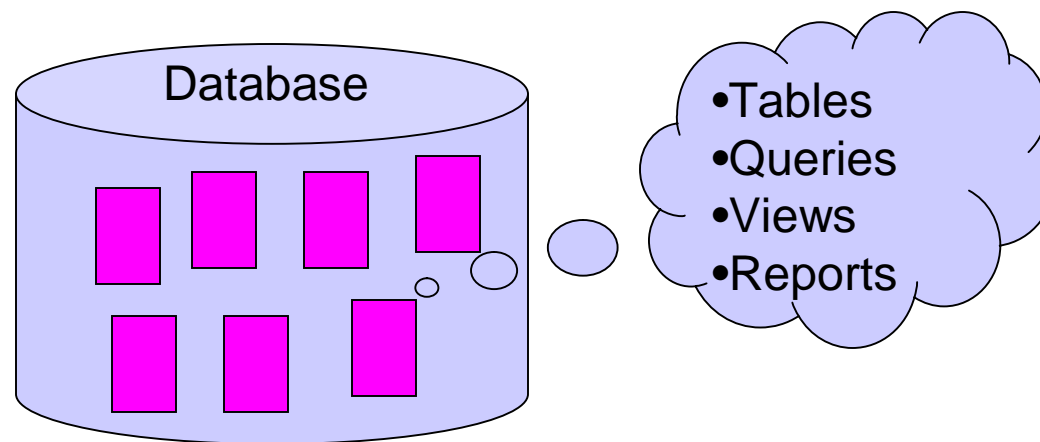
Manish Kumar Prajapati

PGT (Comp.Sc.) (First Shift)

Kendriya Vidyalaya Bailey Road Patna-14

What is the Database?

- A database is a collection of interrelated data stored together to serve multiple application.
- It work like a container which contains the various object like Tables, Queries, Reports, Procedures in organized way.



What is the Database Management System (DBMS)?

- A DBMS refers to a software that is responsible for storing, maintaining and utilizing database in an efficient way.
- A Database along with DBMS software is called Database System.
- Example of DBMS software are Oracle, MS SQL Server, MS Access, Paradox, DB2 and MySQL etc.
- MySQL is open source free ware DBMS.

Why Database System is used?

- **Databases reduces Redundancy**
It removes duplication of data because data are kept at one place and all the application refers to the centrally maintained database.
- **Database controls Inconsistency**
 - When two copies of the same data do not agree to each other, then it is called Inconsistency. By controlling redundancy, the inconsistency is also controlled.
- **Database facilitate Sharing of Data**
 - Data stored in the database can be shared among several users.
- **Database ensures Security**
 - Data are protected against accidental or intentional disclosure to unauthorized person or unauthorized modification.
- **Database maintains Integrity**
 - It enforces certain integrity rules to insure the validity or correctness of data. For ex. A date cant be like 25/25/2000.
- **Database enforces Standard**
 - Database is maintained in a standard format which help to data interchange or migration of data between two systems.

What is Database Abstraction?

A Database system being a collection of files and set of programs, allows access data in secure way. It provides the required information to various types of users like end users, Administrator but hides the technical details that how data is stored?

- A database is implemented through three general levels-
- **Internal Level (Physical Level)**
 - Describes how the data are actually stored on the storage media. At this level low-level data structure are described. It is more closure to Physical storage.
- **Conceptual Level (Logical Level)**
 - This level describes What data are actually stored in the database. In this level database is represented in logically in terms of simple data-structure.
- **External Level (View Level)**
 - This level is concerned with the way in which the data are viewed by individual users. This level is more closure to users. Since users are not concerned with the technical how and what Data is stored. Instead they need only relevant data through a query.

What is Data Independence?

As discussed earlier, a Database is organized in three levels of abstractions, any change in one level may affect other level's schema.

The Ability to modify a schema definition (structure) in one level without affecting a schema definition in the next higher level is called Data Independence.

- Two types of Data independence may be-

- **Physical Data Independence**

- This is the ability to change in Physical Level without affecting Logical or Conceptual level.

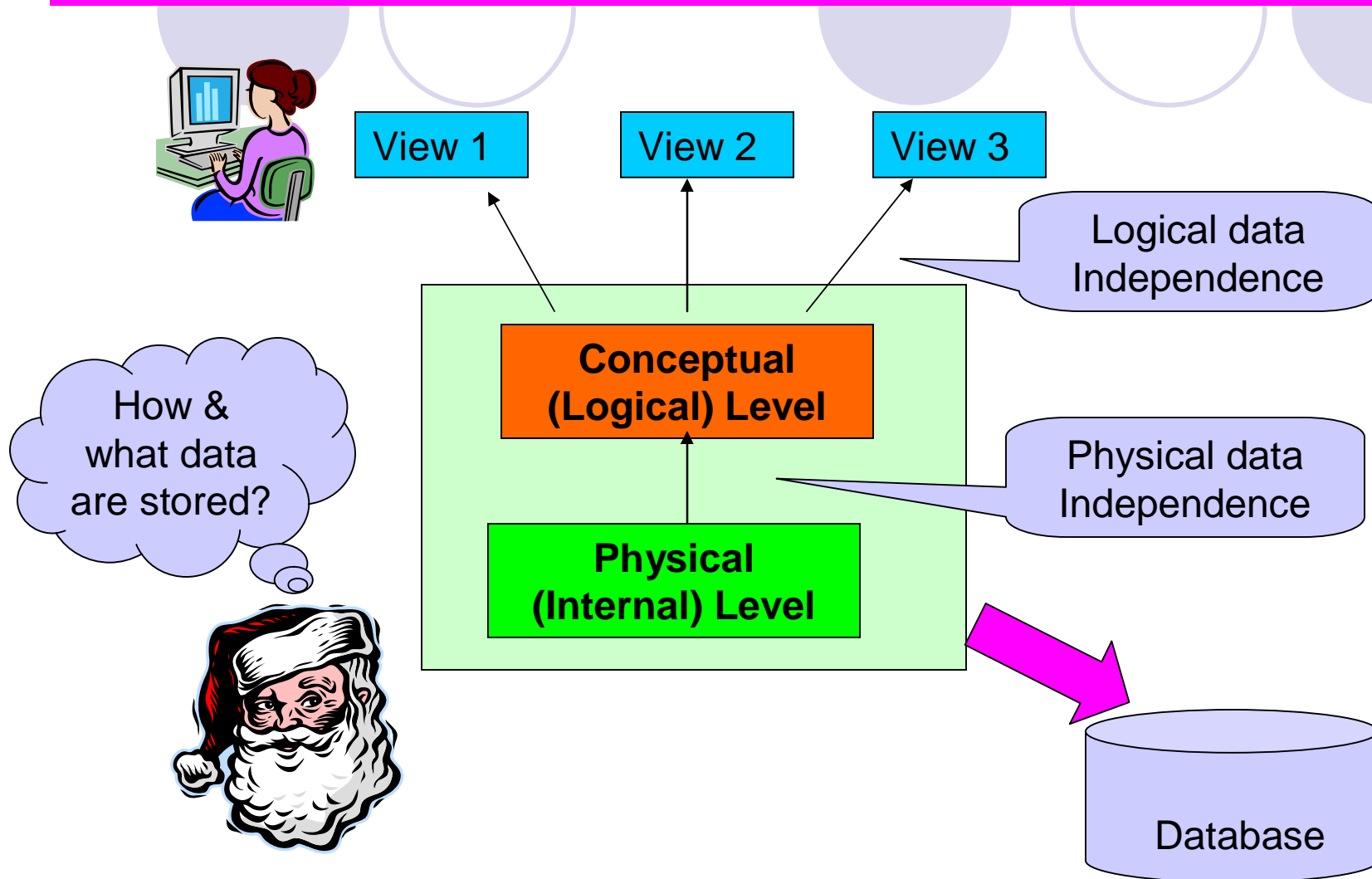
- This Independence refers the change of physical structure without affecting the Application programs and structure of the database.

- **Logical Data Independence**

- This is the ability to change in Logical level without affecting View or External level.

- This Independence refers the change of Logical structure without affecting the Application programs or User interface at User end.

Data Abstraction & Data Independence



What is Data Model?

At External Level or Conceptual level of Database system, certain data model (Presentations) are used which shows 'How data is organized or stored in the database. These Data Presentations are known as Data Models.

- A data is model may-
- **Relational Data Model**
 - In this model data is organized into **Relations** or **Tables** (i.e. Rows and Columns). A row in a table represents a relationship of data to each other and also called a **Tuple** or **Record**. A column is called **Attribute** or **Field**.
- **Network Data Model**
 - In this model, data is represented by collection of records and relationship among data is shown by **Links**.
- **Hierarchical Data Model**
 - In this model, Records are organized as **Trees**. Records at top level is called Root record and this may contains multiple directly linked children records.
- **Object Oriented Data Model**
 - In this model, records are represented as a objects. The collection of similar types of object is called class.

Data Models

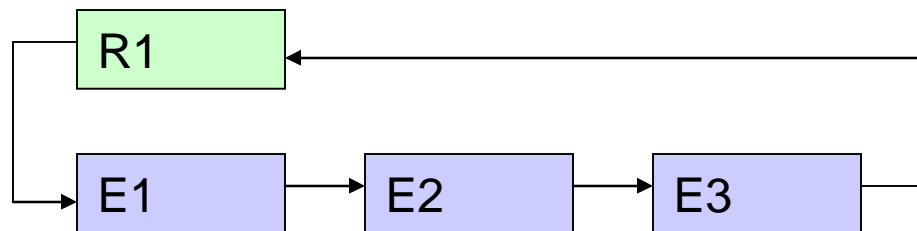
Attribute (Field)

Table (Relation)

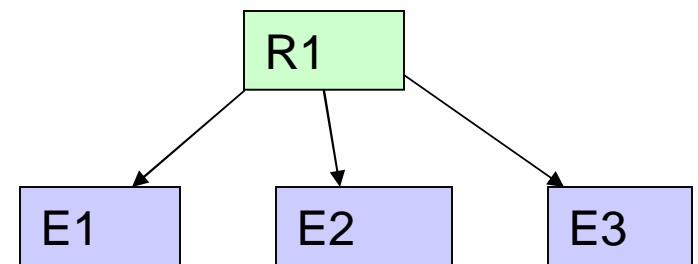
Name	Address	DOB	City	Phone
Amar	2/3 Chowk	01.04.1990	Kanpur	12345
Kailash	22 Katra	23.10.1992	Lucknow	67890

Entity (Record)

Relational Model



Network Model



Hierarchical Model

Representation of Records and Relationship in various Data Model

Various Terms used in the Relational Model

Relational Model was developed by E.F.codd of the IBM and used widely in the most of the DBMS.

● Relation (Table)

- A Relation or Table is Matrix like structure arranged in Rows or Columns. It has the following properties-
- ❖ All items in a column are homogeneous i.e. same data type.
- ❖ Each column assigned a unique name and must have atomic (indivisible) value.
- ❖ All rows of a relation are distinct i.e. no two identical rows (record) are present in the Relation.
- ❖ Ordering or Rows (Records) or Columns (fields) are immaterial.

● Domain

- It is collection of values from which the value is derived for a column.

● Tuple / Entity / Record

- Rows of a table is called Tuple or Record.

● Attribute/ Field

- Column of a table is called Attribute or Field.

● Degree

- Number of columns (attributes) in a table.

● Cardinality

- Number of Records in a table.

Concept of Keys

As discussed earlier, In a Relation each record must be unique i.e. no two identical records are allowed in the Database. A key attribute identifies the record and must have unique value.

- **Primary Key**

A set of one or more attribute that can identify a record uniquely in the relation is called Primary Key.

- **Candidate Key**

- All attribute combinations that can serve as primary key are called Candidate keys as they are candidate to become as primary key.

- **Alternate Key**

- A Candidate Key that is not a Primary key is called Alternate key.

- **Foreign Key**

- A non-key attribute whose values are derived from the primary key of some other table is called Foreign key.
- Foreign Key is used to implement **Referential Integrity** in the Database.

Chapter 13

Introduction to MySQL

Manish Kumar Prajapati

PGT (Comp.Sc) (First Shift)

Kendriya Vidyalaya Bailey Road Patna-14

Introduction

- MySQL is an open source, free and powerful Relational Database Management System (DBMS) that uses SQL.
- It was developed by Michael Widenius and AKA Monty. It was named after Monty's daughter *My*. The logo of MySQL – the dolphin, is named as *Sakila*.
- It is a fast, reliable, scalable alternative to many of the commercial RDBMS.
- MySQL is created and distributed by MySQL AB, a company based in Sweden, now part of the Sun Microsystems.
- It can be freely downloaded from www.mysql.org

Key features of MySQL

- **Speed**
 - It is faster than most of the commercial RDBMSs like Oracle, MS SQL Server.
- **Free of Cost**
 - It is available free of cost as Open Source database. It is part of LAMP (Linux, Apache, MySQL, PHP/ Perl/ Python)
- **Portability**
 - It can be installed and run on different types of Hardware and Operating System platform.
- **Security**
 - It offers privilege and password system for authorization.
- **Connectivity**
 - It may connect various types of client using different protocols.
- **Ease of Use**
 - It is simple database system and offers an interactive environment to work.
- **Query Language**
 - It uses SQL (Structured Query Language) as query language, which is standardized by ANSI.

MySQL & SQL

In order to access data from MySQL database, all program and user must use SQL (Structured Query Language). SQL is a set of commands that are recognized by all the RDBMSs and has become a standard.

SQL is a language that enables you to create and manage a relational database, in which all the information are kept in tables.

There are numerous version of SQL. The original version was developed at IBM's San Jose Research Laboratory with a name of Sequel, as a part of System R project in 1970s. It was standardized by ANSI in 1986 by the name of SQL.

Features of SQL- Capabilities

The processing capabilities of SQL are followings-

- **Data Definition Language (DDL)**
 - The SQL DDL provides commands to create, alter and delete database schema objects like table, views, index etc.
- **Data Manipulation Language (DML)**
 - These commands are used to insert, delete, update and retrieve the stored records from the table.
- **Embedded Data Manipulation Language**
 - The SQL DML command can be executed or used in General Programming Languages like Pascal, C,C++ and JAVA etc.
- **View Definition**
 - The SQL DDL command also used to create Views for simplicity and privacy.
- **Authorization**
 - The SQL DDL command also used for specifying access rights to the Relations (Table) and Views.
- **Integrity**
 - Various data validation rules (Integrity constraints) can also implemented by SQL to insure correctness of data.
- **Transaction control**
 - The SQL includes commands to the database transactions effectively.

Types of SQL Commands

The commands of SQL can be categorized in the followings-

- **Data Definition Language (DDL)**

- These SQL commands are used to create, alter and delete database objects like table, views, index etc.
- CREATE TABLE, CREATE VIEW, CREATE INDEX,
- ALTER TABLE, DROP TABLE, DROP INDEX etc.

- **Data Manipulation Language (DML)**

- These commands are used to insert, delete, update and retrieve the stored records from the table.
- SELECT...., INSERT..., DELETE..., UPDATE.... etc.

- **Transaction Control Language (TCL)**

- These commands are used to control the transaction.
- COMMIT, ROLLBACK, SAVEPOINT etc.

- **Data Control Language (DCL)**

- These commands are used to manipulate permissions or access rights to the tables etc.
- GRANT.. , REVOKE...

Chapter 14

Simple Queries in SQL

Manish Kumar Prajapati
PGT (Comp.Sc) (First Shift)
Kendriya Vidyalaya Bailey Road Patna-14

SQL Elements in MySQL

- Literals:
 - Literals refers to the fixed data value. It may be Numeric or Character. Numeric literals may be integer or real numbers and Character literals must be closed in single quotes like 'Hello'.
 - Null values:
 - If a column in a row has no value then it is said to NULL. The Null should not be used as a Zero value.
 - Comment:
 - /* ... */ (Multi line comment)
 - -- ... (single line comment)
 - # ... (single line comment from the appearance)
 - Data types
 - Numeric type
 - Date & Time type
 - String or Text type
-

Data Types in SQL

□ **Numeric Data Types:**

- **INTEGER** or **INT** – up to 11 digit number without decimal.
- **SMALLINT** – up to 5 digit number without decimal.
- **FLOAT (M,D)** – Real numbers up to M digit with D decimal places.
Ex. Float (10,2)
- **DECIMAL(M,D)** or **NUMERIC(M,D)** – Unpacked floating point up to M length and D decimal places.

□ **Date & Time Data Types:**

- **DATE** – A date in YYYY-MM-DD format.
- **DATETIME** – A date and time format like YYYY-MM-DD HH:MM:SS
- **TIME** – Stores time in HH:MM:SS format.

□ **String or Text Data Type:**

- **CHAR(M)** – A fixed length string up to 255 characters. (default is 1)
 - **VARCHAR(M)** – A variable length string up to 255 characters.
 - **BLOB** – Used to store image data or characters up to 65535.
 - **TEXT** – Same as BLOB but offers case insensitive search.
-

Some Data Management commands in MySQL

❑ **Creating a Database.**

The following command will create School database in MySQL.

Ex. mysql> **CREATE DATABASE School;**

❑ **Opening a database**

To open an existing database you can use the following command.

Ex. mysql> **USE school ;**

❑ **Creating a table in the database**

To create a table we can use Create Table.... command. The following command will create Student table with *Stname* and *Stcity* columns.

Ex. mysql> **CREATE TABLE student**
(Stname char(30),
Stcity char(20),
age Integer);

❑ **Inserting a Record in a table**

mysql> **INSERT INTO Student VALUES ('Amit', 'Suratgarh', 16);**

❑ **Getting listings of database and tables**

mysql> **SHOW DATABASES;**

mysql> **SHOW TABLES;**

❑ **Deleting a Table and database**

mysql> **DROP TABLE Student ;**

mysql> **DROP DATABASE School ;**

❑ **Viewing Table Structure**

mysql> **DESCRIBE Student ;**

Making Simple Queries

The SELECT command of SQL, empower you to make a request (queries) to retrieve records from the database.

The syntax of SQL is given below-

```
SELECT < * | column name(s) > FROM <table(s)>  
WHERE <condition>  
ORDER BY <column name> [ASC | DESC] ;
```

Consider the table ***Student*** as –

StID	Name	Fname	DOB	City	Class
S1	Amitabh	Harivansh Rai	1948-11-10	Allahabad	12
S2	Sharukh	Firoz	1970-05-10	Delhi	11
S3	Irphan	Akbar	1970-10-05	Jaipur	11
S4	Salman	Salim Javed	1972-04-10	Mumbai	10
S5	Abhishek	Amitabh	1975-03-12	Mumbai	10

Making Simple Queries – Cont..

□ Selecting all columns

If you want to view all columns of the student table, then you should give the following command-

```
mysql> SELECT * FROM Student ;
```

MySQL will display the all records with all columns in the Student table.

* Is used to represent all columns.

StID	Name	Fname	DOB	City	Class
S1	Amitabh	Harivansh Rai	1948-11-10	Allahabad	12
S2	Sharukh	Firoz	1970-05-10	Delhi	11
S3	Irphan	Akbar	1970-10-05	Jaipur	11
S4	Salman	Salim Javed	1972-04-10	Mumbai	10
S5	Abhishek	Amitabh	1975-03-12	Mumbai	10

Making Simple Queries – Cont..

❑ Selecting columns

If you want to view only **Name** and **City** columns of the student table

```
mysql> SELECT Name, City FROM Student ;
```

Name	City
Amitabh	Allahabad
Sharukh	Delhi
Irphan	Jaipur
Salman	Mumbai
Abhishek	Mumbai

But if you give

```
mysql> SELECT City, Name FROM Student ;
```

City	Name
Allahabad	Amitabh
Delhi	Sharukh
Jaipur	Irphan
Mumbai	Salman
Mumbai	Abhishek

Making Simple Queries – Cont..

❑ Eliminating Duplicate values in a column - **DISTINCT**

mysql> **SELECT City FROM Student ;**

City
Allahabad
Delhi
Jaipur
Mumbai
Mumbai

Mumbai is repeated

mysql> **SELECT DISTINCT City FROM Student ;**

City
Allahabad
Delhi
Jaipur
Mumbai

Only Unique Cities are displayed

Making Simple Queries – Cont..

□ Doing simple calculations

We can also perform simple calculations with SQL Select command. SQL provide a dummy table named DUAL, which can be used for this purpose.

```
mysql> SELECT 4*3 FROM DUAL ;
```

We can also extend this idea with a columns of the existing table.

```
mysql> SELECT Name, Sal *12 FROM EMP ;
```

□ Using Column Aliases

We can give a different name to a column or expression (Alias) in the output of a query.

Alias for Sal*12

```
mysql> SELECT Name, Sal*12 AS 'Annual Salary' FROM EMP;
```

```
mysql> SELECT Name, DOB AS 'Date of Birth' FROM Student;
```

```
mysql> SELECT 22/7 AS PI FROM Dual;
```



- ✓ When Alias name is a single word then ' ' is not required.
- ✓ In MySQL FROM DUAL is optional i.e *Select 4*3 ;* is valid

Making Simple Queries – Cont..

□ Displaying Text in the Query output

We can also display any text in the query output.

```
mysql> SELECT 'Mr.' , Name, City FROM Student ;
```

```
mysql> SELECT Name, 'Rs', Sal from Emp;
```

```
Mysql> SELECT Name, 'Rs', Sal*12 AS 'Annual Salary'  
FROM Emp;
```

□ Handling NULL Values

MySQL stores a special value called NULL for empty cells, which is displayed when data is not available or missing.

We can substitute this NULL by a text or message to get more readable output. **IFNULL(<column name>, <text>)** can be used for this purpose.

```
mysql> SELECT Name, IFNULL(DOB, 'Not Available' ) FROM Student;
```

```
mysql> SELECT Name, IFNULL(Class, 'Not allotted') FROM Student;
```

Selecting Specific Rows – WHERE clause

□ WHERE <Condition>

We can select specific records by specifying condition with WHERE clause.

```
mysql> SELECT * FROM Student WHERE City='Mumbai';
```

StID	Name	Fname	DOB	City	Class
S4	Salman	Salim Javed	1972-04-10	Mumbai	10
S5	Abhishek	Amitabh	1975-03-12	Mumbai	10

```
mysql> SELECT Name, Fname, City from Student  
WHERE Class >10;
```

Condition

Name	Fname	City	Class
Amitabh	Harivansh Rai	Allahabad	12
Sharukh	Firoz	Delhi	11
Irphan	Akbar	Jaipur	11

□ We can **operators** to make a complex conditions

Selecting Specific Rows – WHERE clause

□ Relational Operators

We can use the following Relational operators in condition.

=, >, <, >=, <=, <>, IS, LIKE, IN, BETWEEN

□ Logical Operators

We can use the following Logical Operators to connect two conditions.

OR (||), AND (&&), NOT (!)

```
mysql> SELECT Name, City from Student  
WHERE City <> 'Mumbai' AND Class>10;
```

```
mysql> SELECT * FROM Emp  
WHERE Sal >10000 OR Job ='Manager';
```

```
mysql> SELECT * FROM Student  
WHERE NOT Grade='A';
```

Selecting Specific Rows – WHERE clause

□ Specifying Range of Values – BETWEEN Operator

```
mysql> SELECT * FROM Emp  
      WHERE Sal BETWEEN 5000 AND 10000 ;
```

The same query can also be written as -

```
mysql> SELECT * FROM Emp  
      WHERE Sal >= 5000 AND Sal <= 10000 ;
```

Other Logical operators also can be applied-

```
mysql> SELECT * FROM Emp  
      WHERE NOT Sal BETWEEN 5000 AND 10000 ;
```

□ Specifying List – IN Operator

```
mysql> SELECT * FROM Emp  
      WHERE Sal IN (5000, 10000) ;
```

The same query can also be written as -

```
mysql> SELECT * FROM Emp  
      WHERE Sal = 5000 OR Sal = 10000 ;
```

```
mysql> SELECT * FROM Student  
      WHERE City IN ('Mumbai', 'Delhi', 'Kanpur') ;
```

Selecting Specific Rows – WHERE clause

□ Pattern Matching – LIKE Operator

A string pattern can be used in SQL using the following wild card

- % - Represents any substring
- _ - Any Character

Example.

- 'A%' – represents any string starting with 'A' character.
- '__A' - represents any 3 character string ending with 'A'.
- '_B%' - represents any string having second character 'B'
- '___' – represents any 3 letter string.

A pattern is case sensitive and can be used with LIKE operator.

```
mysql> SELECT * FROM Student WHERE Name LIKE 'A%';  
mysql> SELECT * FROM Student WHERE Name LIKE '%Singh%';  
mysql> SELECT Name, City FROM Student  
WHERE Class >= 9 AND Name LIKE '%Kumar%' ;
```

Selecting Specific Rows – WHERE clause

□ Searching NULL Values – IS Operator

```
mysql> SELECT * FROM Student WHERE City IS NULL ;
```

The NOT Operator can also be applied -

```
mysql> SELECT * FROM Student WHERE City IS NOT NULL;
```

□ Ordering Query Result – ORDER BY Clause

A query result can be orders in ascending (A-Z) or descending (Z-A) order as per any column. Default is Ascending

```
mysql> SELECT * FROM Student ORDER BY City;
```

To get descending order use DESC key word.

```
mysql> SELECT * FROM Student ORDER BY City DESC;
```

```
mysql> SELECT Name, Fname, City FROM Student  
Where Name LIKE 'R%'  
ORDER BY Class;
```

Chapter 15

MySQL Functions

Manish Kumar Prajapati
PGT (Comp.Sc) (First Shift)
Kendriya Vidyalaya Bailey Road Patna-14

What is Function?

□ Definition:

A function is a special types of command that performs some operation and returns a single value as a result.

It is similar to method or function in JAVA, which can be called by giving some argument.

□ Types of Functions:

- Numeric Functions
 - String Functions
 - Date & Time Function
 - Aggregate Functions
-

Numeric Functions

- These functions may accept some numeric values and performing required operation, returns numeric values as result.

Name	Purpose	Example
MOD (M, N)	Returns remainder of M divide by N	Select MOD(11,4) ; → 3
POWER (M, N) POW (M, N)	Returns M^N	Select POWER(3,2); → 9
ROUND (N [,M])	Returns a number rounded off up to M place. If M is -1, it rounds nearest 10.	Select ROUND(15.193,1); → 15.2
SIGN (N)	Returns sign of N. -1 if $N < 0$, 1 if $N > 0$ and 0 if $N = 0$	Select SIGN(-15); → -1
SQRT (N)	Returns square root of N	Select SQRT(25); → 5
TRUNCATE(N,M)	Returns number after truncating M decimal place.	Select TRUNCATE(15.79,1) → 15.7

String Functions

□ CHAR()

Returns a character/string made up of the ASCII representation of the given number.

CHAR(Num1 [,Num2 ,])

mysql> **SELECT CHAR(65) FROM DUAL;**

→ A

mysql> **SELECT CHAR(65,66,67) FROM DUAL;**

→ ABC

- If given number is fractional value then it is considered/rounded as integer.

mysql> **SELECT CHAR(66.3) FROM DUAL;** **→ B**

- If given number is in string format, it is also treated as number

mysql> **SELECT CHAR('66.3') FROM DUAL;** **→ B**

String Functions

cont...

□ **CONCAT()**

Concatenates (Adds) two string.

CONCAT(Str1 , Str2)

```
mysql> SELECT CONCAT('ab' , 'cd' ) FROM DUAL;
```

→ **abcd**

```
mysql> SELECT CONCAT('Mr', Name) FROM Student;
```

➤ Concat() can be nested.

```
mysql> SELECT CONCAT(CONCAT(Name,'son of '), Fname)
FROM Student;
```

□ **LENGTH()**

Returns length of given string.

LENGTH (Str)

```
mysql> SELECT LENGTH('abcd' ) FROM DUAL;
```

→ **4**

```
mysql> SELECT Name, LENGTH(Name) FROM Student;
```

String Functions

cont...

❑ LOWER() or LCASE()

Converts given string in lower case.

LOWER (Str)

```
mysql> SELECT LOWER('ABcD' ) FROM DUAL;
```

→ **abcd**

```
mysql> SELECT LOWER(Name) FROM Student;
```

```
mysql> SELECT LCASE(Fname) FROM Student;
```

❑ UPPER() or UCASE()

Converts given string in upper case.

UPPER (Str)

```
mysql> SELECT UPPER('abcD' ) FROM DUAL;
```

→ **ABCD**

```
mysql> SELECT UPPER(Name) FROM Student;
```

```
mysql> SELECT UCASE(Fname) FROM Student;
```

String Functions

cont...

□ LTRIM()

Returns string after removing leading spaces.

LTRIM (Str)

```
mysql> SELECT LTRIM('  abcd' ) FROM DUAL;
```

→ **abcd**

```
mysql> SELECT LTRIM(Name) FROM Student;
```

□ RTRIM()

Returns string after removing trailing spaces.

RTRIM (Str)

```
mysql> SELECT RTRIM('abcd  ') FROM DUAL;
```

→ **abcd**

```
mysql> SELECT RTRIM(Name) FROM Student;
```

String Functions

cont...

□ TRIM()

Returns string after removing leading and trailing spaces. It can also work as LTRIM() and RTRIM().

TRIM ([BOTH | LEADING | TRAILING][Char][FROM] Str)

- BOTH is default setting.
- Default Char is space character, if not given.

```
mysql> SELECT TRIM('  abcd  ') FROM DUAL;
```

➔ **abcd**

```
mysql> SELECT TRIM(BOTH ' ' FROM ' abc ');
```

➔ **abc**

```
mysql> SELECT TRIM(BOTH '*' FROM '***abc***');
```

➔ **abc**

```
mysql> SELECT TRIM(TRAILING '*' FROM '**abc***');
```

➔ ****abc**

String Functions

cont...

□ SUBSTR()

Returns a sub string of given length from specified position.

SUBSTR (Str, position [,length])

```
mysql> SELECT SUBSTR('MY COMPUTER', 4,3' ) → COM
```

➤ If position is negative then backward position is counted.

```
mysql> SELECT SUBSTR('ABCDEFGF' , -5, 4) FROM Student;
```

→ CDEF

➤ If Length is omitted then up to end of the string is considered.

```
mysql> SELECT SUBSTR('ABCDEFGF' , 3) FROM Student;
```

→ CDEFG

□ INSTR()

Searches a string in to another string and returns its position.

INSTR(Str1, Str2)

```
mysql> SELECT INSTR('CORPORATE FLOOR', 'OR'); → 2
```

```
mysql> SELECT Name, INSTR(Name,'a') FROM Student;
```

String Functions

cont...

□ LEFT()

Returns leftmost string up to given length.

LEFT (Str , length)

```
mysql> SELECT LEFT('MYSQL', 2 ) → MY
```

```
mysql> SELECT LEFT (Name, 4) FROM Student;
```

□ RIGHT()

Returns rightmost string up to given length.

RIGHT (Str , length)

```
mysql> SELECT RIGHT('MYSQL', 3 ) → SQL
```

```
mysql> SELECT RIGHT (Name, 4) FROM Student;
```

□ MID()

Returns a substring upto given length from given position.

MID (Str ,Pos, Length)

```
mysql> SELECT MID('COMPUTER', 4,3 ) → PUT
```

```
mysql> SELECT MID (Name, 4,3) FROM Student;
```

Date & Time Functions

□ CURDATE() or CURRENT_DATE()

Returns current date of the system in YYYY-MM-DD format.

```
mysql> SELECT CURDATE() FROM DUAL → 2010-01-30
```

```
mysql> SELECT CURDATE()+10 ; → 2010-02-09
```

□ SYSDATE()

Returns current date and time as YYYY-MM-DD HH:MM:SS

```
mysql> SELECT SYSDATE() FROM DUAL
```

```
→ 2010-01-30 10:30:20
```

□ NOW()

Returns current date and time as YYYY-MM-DD HH:MM:SS

```
mysql> SELECT NOW() FROM DUAL
```

```
→ 2010-01-30 10:30:20
```

Difference between SYSDATE() & NOW()

NOW() returns the time when command began to execute and does not change time during execution. Where as SYSDATE() changes its time continuously.

Date & Time Functions

cont...

□ DATE()

Returns date part of the given date-time expression.

DATE (Dt)

```
mysql> SELECT DATE('2008-12-31 01:02:03') FROM DUAL;
```

→ 2008-12-31

```
mysql> SELECT DATE( SYSDATE()) FROM DUAL;
```

□ YEAR()

Returns year of the given date expression.

YEAR (Dt)

```
mysql> SELECT YEAR('2008-12-31') FROM DUAL; → 2008
```

```
mysql> SELECT YEAR(DOB) FROM Student;
```

□ MONTH()

Returns month of the given date expression.

MONTH (Dt)

```
mysql> SELECT MONTH('2008-12-31') FROM DUAL; → 12
```

```
mysql> SELECT MONTH( CURDATE()) FROM DUAL;
```

Date & Time Functions

cont...

□ DAYOFMONTH()

Returns day of month of the given date expression.

DAYOFMONTH (Dt)

```
mysql> SELECT DAYOFMONTH('2008-12-31') FROM DUAL;
```

→ 31

```
mysql> SELECT DAYOFMONTH( CURDATE()) FROM DUAL;
```

```
mysql> SELECT DAYOFMONTH( DOB) FROM Student;
```

□ DAYNAME()

Returns the name of Week day of the given date expression.

DAYNAME (Dt)

```
mysql> SELECT DAYNAME('2008-12-31') FROM DUAL;
```

→ SUNDAY

```
mysql> SELECT DAYNAME( CURDATE()) FROM DUAL;
```

```
mysql> SELECT DAYNAME( DOB) FROM Student;
```

Date & Time Functions

cont...

□ DAYOFWEEK()

Returns day of week i.e. 1- Sunday, 2- Monday.. etc. of given date.

DAYOFWEEK (Dt)

```
mysql> SELECT DAYOFWEEK('2008-12-31') FROM DUAL;
```

→ 1

```
mysql> SELECT DAYOFWEEK(CURDATE()) FROM DUAL;
```

□ DAYOFYEAR()

Returns the day of year of the given date expression.

DAYOFYAER (Dt)

```
mysql> SELECT DAYOFYAER('2010-02-05') FROM DUAL;
```

→ 36

```
mysql> SELECT DAYOFYAER( CURDATE()) FROM DUAL;
```

```
mysql> SELECT DAYOFYEAR( DOB) FROM Student;
```

Aggregate Functions

□ **SUM()**

Returns sum of given column in the table.

SUM (<Field>)

```
mysql> SELECT SUM (Sal) FROM Emp;
```

```
mysql> SELECT SUM(Sal) FROM Emo WHERE City='Jaipur';
```

□ **MIN()**

Returns minimum value in the given column of table.

MIN (<Field>)

```
mysql> SELECT MIN (Sal) FROM Emp;
```

```
mysql> SELECT MIN(Sal) FROM Emo WHERE City='Jaipur';
```

□ **MAX()**

Returns maximum value in the given column of table.

MAX (<Field>)

```
mysql> SELECT MAX (Sal) FROM Emp;
```

```
mysql> SELECT MAX(Sal) FROM Emo WHERE City='Jaipur';
```

Aggregate Functions cont..

□ **AVG()**

Returns average value of given column in the table.

AVG (<Field>)

```
mysql> SELECT AVG (Sal) FROM Emp;
```

```
mysql> SELECT AVG(Sal) FROM Emo WHERE City='Jaipur';
```

□ **COUNT()**

Returns number of values in the given column of table. It also reflect the number of record in the table.

COUNT (<Field| * >)

```
mysql> SELECT COUNT (Name) FROM Emp;
```

```
mysql> SELECT COUNT(Name) FROM Emo  
WHERE City='Jaipur';
```

```
mysql> SELECT COUNT (*) FROM Emp;
```

→ Number of records in the Emp table

```
mysql> SELECT COUNT(*) FROM Emo  
WHERE City='Jaipur';
```

Chapter 16

Table Creation & Data Manipulation Commands in MySQL

Manish Kumar Prajapati

PGT (Comp.Sc) (First Shift)

Kendriya Vidyalaya Bailey Road Patna-14

Handling Database

❑ Creating Database:

You can create a database in MySQL by using the following DDL command.

CREATE DATABASE [IF NOT EXIST] <Database name>

```
mysql> CREATE DATABASE School;
```

```
mysql> CREATE DATABASE IF NOT EXIST School;
```

❑ Opening Database:

USE <database name>

```
mysql>USE School;
```

❑ Removing Database:

DROP <database name>

```
mysql>DROP School;
```

Handling Tables

□ Creating Tables:

CREATE TABLE < Table Name >

(<Col name> <data type> [(size)] [Constraints],)

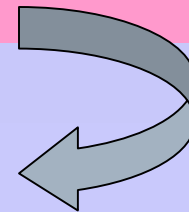
▪ Data types

Commonly used data types are-

INTEGER, DECIMAL(P,D), NUMERIC(P,D), CHAR(n),
VARCHAR(n), DATE etc.

Emp (empID, ename, sex, grade, gross)

```
mysql> CREATE TABLE Emp
      ( empID integer,
        ename char(20),
        sex char(1),
        grade char(2),
        gross decimal );
```



Handling Tables

Cont...

❑ Viewing Table Structure:

You can view structure of any table after using database as-

DESC[RIIBE] <table name>

```
mysql> DESC Student;
```

❑ Deleting Table:

You can delete an existing table as-

DROP TABLE [IF EXIST] <table name>

```
mysql> DROP TABLE Student;
```

❑ Creating Table from Existing Table:

CREATE TABLE <Table name>

AS (<Select Query>);

```
mysql> CREATE TABLE Staff  
      ( Select empID, ename, sex From Emp);
```

```
mysql> CREATE TABLE Staff  
      ( Select * From Emp);
```

It will create identical table as Emp

Constraints in the Table

A Constraint is a condition or check applicable to a column or table which ensures the integrity or validity of data.

Constraints are also called Integrity constraints. The following constraints are commonly used in MySQL.

- NOT NULL**
 - PRIMARY KEY**
 - UNIQUE**
 - DEFAULT**
 - CHECK**
 - FOREIGN KEY**
-

Constraints in the Table

Cont..

❑ NOT NULL

This constraint specifies that column must not contain NULL value i.e. value for the column must be given (mandatory)

❑ PRIMARY KEY

This constraint declares a column as the primary key. Since Primary key must not have NULL value, so it is used with NOT NULL constraints.

❑ UNIQUE

This constraint ensures that the value for the column should be Unique i.e. no two rows have the same (duplicate) value.

UNIQUE v/s PRIMARY KEY

- UNIQUE allows NULL values but PRIMARY KEY does not.
- Multiple columns may have UNIQUE constraints, but there is only one PRIMARY KEY constraint.

Constraints in the Table

Cont..

□ DEFAULT

A default value can be specified for a column, which can be used when user does not enter a value for the column at time of insertion of record.

□ CHECK <condition>

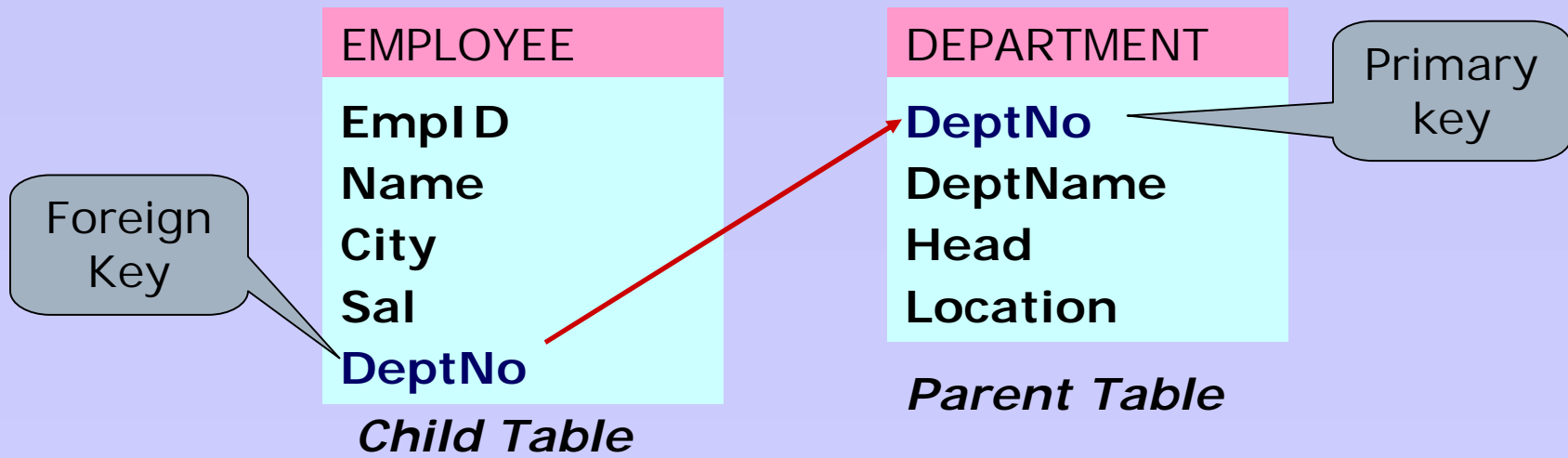
This constraints specifies a user defined condition for a column, which must be satisfied (true) for to be a valid record.

```
mysql> CREATE TABLE Student
( StCode char(3) NOT NULL PRIMARY KEY,
  Stname char(20) NOT NULL,
  StAdd varchar(40),
  AdmNo char(5) UNIQUE,
  StSex char(1) DEFAULT = 'M',
  StAge integer CHECK (StAge >= 10),
  Stream char(1) CHECK Stream IN ('S', 'C', 'A')
);
```

Constraints in the Table Cont..

□ FOREIGN KEY REFERENCES

- A Foreign key is non-key column in a table whose value is derived from Primary key of some other table.
- Each time when record is inserted/updated the other table is referenced. This constraints is also called Referential Integrity Constraints.
- This constraints requires two tables in which Reference table (having Primary key) called **Parent table** and table having Foreign key is called **Child table**.



Constraints in the Table **Cont..**

□ How to Implement FOREIGN KEY constraints

```
CREATE TABLE Department  
( DeptNo      char(2)  NOT NULL PRIMARY KEY,  
  DeptName    char(10) NOT NULL,  
  Head        char(30) );
```

Parent table should be created first.

```
CREATE TABLE Employee  
( EmpNo char(3) NOT NULL PRIMARY KEY,  
  Name  char(30) NOT NULL,  
  City  char(20),  
  Sal   decimal(8,2),  
  DeptNo char(2),  
  FOREIGN KEY (DeptNo) REFERENCES Department (DeptNo));
```

Foreign key as table level constraints... written after all columns.

➔ Alternatively, it can be applied in front of column itself... without using *Foreign Key* keyword.

```
DeptNo char(2) REFERENCES Department (DeptNo),
```

Constraints in the Table Cont..

❑ Other options with FOREIGN KEY

What will happen, when a primary key value is deleted or updated in parent table, which is being referred in child table?

We can set the following options with FOREIGN KEY.

ON DELETE CASCADE / RESTRICT/ SET NULL/ NO ACTION

ON UPDATE CASCADE / RESTRICT/ SET NULL/ NO ACTION

CASCADE:

The values of Foreign key in child table, will be updated or referenced record will be deleted automatically.

RESTRICT:

This option will reject any Delete or Update operation on Primary key of Parent table.

SET NULL:

This option will set NULL value in referenced records of Foreign key in child table.

NO ACTION:

No any action will be taken when any Delete/Update operation is carried in Primary key of Parent table.

```
CREATE TABLE Employee
```

```
( .....
```

```
  FOREIGN KEY DeptNo REFERENCE Department (DeptNo)
```

```
  ON DELETE CASCADE ON UPDATE CASCADE );
```

Modifying Table Structure

You can alter (modify) the structure of existing table by the using ALTER TABLE.... Command of MySQL.

You can do the following with the help of ALTER TABLE.. Command.

- **Add a new Column & Constraints**
- **Modifying existing column (name, data type, size etc.)**
- **Delete an existing column & Constraints**
- **Changing Column Name**

ALTER TABLE <Table Name>

ADD | MODIFY | DROP | CHANGE <Options>

Modifying Table Structure cont..

□ Adding new column

ALTER TABLE <Table Name>
ADD <Column><data type> <size> [<Constraints>]

```
mysql> ALTER TABLE Student ADD (TelNo Integer);
```

```
mysql> ALTER TABLE Student  
      ADD (Age Integer CHECK (Age >= 5) );
```

```
mysql> ALTER TABLE Emp  
      ADD (Sal Number(8,2) DEFAULT 5000 );
```

□ Modifying Existing Column

ALTER TABLE <Table Name>
MODIFY <Column><data type> <size> [<Constraints>]

```
mysql> ALTER TABLE Student  
      MODIFY (Name VARCHAR(40));
```

```
mysql> ALTER TABLE Emp MODIFY (Sal DEFAULT 4000 );
```

Modifying Table Structure cont..

□ Removing Column & Constraints

ALTER TABLE <Table Name>

DROP COLUMN <Column name> <Constraints>

```
mysql> ALTER TABLE Student  
        DROP COLUMN TelNo;
```

```
mysql> ALTER TABLE Student  
        DROP PRIMARY KEY;
```

```
mysql> ALTER TABLE Emp  
        DROP COLUMN JOB, DROP FOREIGN KEY ;
```

□ Changing Column Name of Existing Column

ALTER TABLE <Table Name>

CHANGE [COLUMN] <Old name> <New Name>

```
mysql> ALTER TABLE Student  
        CHANGE COLUMN Name Sname;
```

Using DML Commands

❑ Inserting Record into table:

You can insert record in the table by using by using the following DML command.

**INSERT INTO <Table Name> [<Column list>]
VALUES <list of values>**

Suppose a table named STUDENT has been created with the following structure.

StID	NAME	FNAME	DOB	CITY	CLASS
------	------	-------	-----	------	-------

We can insert a record as follows-

```
mysql> INSERT INTO Student VALUES  
('s1','Amitabh', 'Harivansh','1955-10-25', 'Mumbai', 12);  
mysql> INSERT INTO Student VALUES  
('s2','Sharukh Khan', NULL,'1972-5-25', 'Delhi', 10);  
mysql> INSERT INTO Student (StID, FName, Name, Class)  
VALUES ('s3','Amitabh', 'Abhishek', 10);
```

Using DML Commands cont..

❑ **Inserting Record from Another table:**

You can insert all or selected record(s) in the table from another table by using Select ... command in place of Values.

Suppose a table named NEWSTUDENT has been created and records to be inserted from OLDSTUDENT table having the same structure of columns.

```
mysql> INSERT INTO Newstudent
      VALUES ( SELECET * FROM Oldstudent);
mysql> INSERT INTO Newstudent
      VALUES ( SELECT * FROM Oldstudent
              WHERE City='Mumbai');
mysql> INSERT INTO Newstudent (StID, FName, Name, Class)
      VALUES (Select StID,FName,Name,Class
              FROM Oldstudent WHERE Class >=11);
```

Using DML Commands cont..

❑ **Deleting Record from the table:**

You can delete all or selected record(s) from the table by using the following DDL command.

DELETE FROM <Table Name> [WHERE <Condition>]

```
mysql> DELETE FROM Student ;
```

```
mysql> DELETE FROM Student WHERE City='Mumbai' ;
```

```
mysql> DELETE FROM Student WHERE Class >=11 ;
```

- You can recall (Undelete) records by giving ROLLBACK command.

```
mysql> ROLLBACK ;
```

- You can issue COMMIT command to record the changes permanently.

```
mysql> COMMIT ;
```

Using DML Commands cont..

❑ **Modifying Records in the table:**

You can modify the values of columns of all or selected records in the table by using the following DDL command.

**UPDATE <Table Name> SET <Column> = <Expression>
[WHERE <Condition>]**

```
mysql> UPDATE Student SET Class =10 ;
```

```
mysql> UPDATE Student
```

```
        SET FName= CONTACT('Mr.', FName) ;
```

```
mysql> UPDATE Emp SET Sal = Sal+(Sal*10/100);
```

```
mysql> UPDATE Emp SET Sal = Sal+(Sal*10/100)
```

```
        WHERE Sal <=10000;
```

```
mysql> UPDATE Emp SET City = 'Suratgarh'
```

```
        WHERE CITY IS NULL;
```
