330en23

23

DATABASE MANAGEMENT SYSTEMS

In the previous lesson you have learnt that data processing concepts. Now you know that computers are used to process data. You can feed data into the computers and give the instructions (in the form of programs) to process the data. Computers can process the data by following the specified instructions. Computers are able to process data at extremely high speeds. There are lots and lots of data which are processed by the computers. For easy and efficient processing of data it is necessary that the data is stored in an organized way. There are specialized software, called Database Management Systems, which facilitate organized storage of data. In this lesson you will learn about Database Management Systems.

OBJECTIVES

After reading this lesson, you will be able to:

- define database;
- design simple databases;
- create databases and tables within them using MySQL;
- write and execute SQL commands to perform various database operations.

23.1 TERMINOLOGY

To understand the concepts of Database Management System, you must be familiar with a few terms which are given below:

Database: A database is an organized collection of data. From daily life we can take many examples of simple databases. A few are: An index of a book,

telephone directory, classified ads in a newspaper etc. More complex examples may include registers containing data of students in a school, data of employees in a big organization etc.

Need of computerized Database: Databases can be organized manually or using computers. When the databases are organized manually, some problems creep in frequently. These problems are:

- **Redundancy**: Having multiple copies of the same data. For example a student's name is written in office records as well as in class teacher's record. Redundancy leads to inconsistency.
- **Inconsistency**: Having multiple mismatching copies of the same data. For example if a student's section changes from A to B, this change is recorded in class teachers' registers but not in office register.
- **Loss of Integrity**: Integrity means having consistent valid data all the time. Data consistency is also a kind of data integrity. Data integrity also includes having valid data only. For example if there are four sections (A, B, C, D) of a class but a student's section is recorded as something other than these, then it is an example of loss of integrity.
- **Data Insecurity**: Manual databases can be secured physically but cannot be secured against unauthentic data alterations.

Computerized databases provide centralized control of its operational data. Thus, with the help of a computerized database:

1. Redundancy is controlled.
2. Inconsistence can be avoided.
3. Integrity of data is maintained.
4. Data can be shared.
5. Security restrictions can be applied.

Data Base Management System (DBMS): A DBMS is a software that allows us to manage computerized databases.

Relational Database Management System (RDBMS): It is a DBMS based on relational model introduced by E.F.Codd.

Relational Database: A relational database is a database in which data is organized in two-dimensional tables (also called relations). A Table consists of rows (also called tuples or records) and columns (also called attributes or fields).

The following table consists of three fields : Name, Telephone, Address; and two rows.

Name	Telephone	Address
Abhishek	555227822,	Prem Nagar.
Puneet	5438763	A-15, Mohan Nagar.

Schema – It refers to the organization of data or skeleton structure that represents the logical flow of the entire database.

Entity: An entity is a person or object (physical or logical) for which we need to store data in a database. In a school it can be a student, a teacher, a course etc. In a library an entity can be a book, a member, a book supplier etc.

Relation: A relation (or a table) is a collection of data corresponding to the same kinds of entities in a database. A relation may contain data of all the books in a library, or data of all the items in a shop etc. A database contains one or more than one relations.

Tuple/Row/Record: A tuple corresponds to a row of a relation. A tuple contains data corresponding to an entity. The number of tuples in a relation is called the **cardinality** of the relation.

Attribute: An attribute corresponds to a column in a relation. The number of attributes (columns) in a relation is called its **degree**.

Domain: A domain (of a column) is a pool of values from which that column draws its actual values. For example, domain of the column RollNumber may be integers from 1 to 50.

Key: A column or a combination of columns which can be used to identify a row (tuple) in a table is called its key. In general, any column or any combination of columns in a table is a key.

Primary Key: The group of one or more columns used to uniquely identify each row of a relation is called its Primary Key. For example, in a students table AdmissionNumber can be the primary key. A table cannot have more than one primary key.

Candidate Key: A field or combination of fields which can be used as a primary key of a relation is called a candidate key because it is one of the candidates available to be the primary key of the relation. A table may have multiple candidate keys. For example, in a students table *AdmisionNumber* as well as the combination *Class+Section+RollNumber* are the candidate keys.

Alternate Key: A table may have multiple candidate keys. One of these keys become the primary key of the table. All the remaining candidate keys are called alternate keys of the relation.

Foreign Key: A group of one or more columns in a table which is used to set relationship of this table with some other table.

Cartesian Product (of two tables): It is a new table which contains the columns of both the given tables. The rows of the new table are obtained by pairing each row of the first table with each row of the second table.

Normalization: Normalization is the process of analyzing the data to be represented and breaking it down into separate tables in accordance with the principles of relational structure.

Need for Normalization: Normalization is carried out for four reasons:

1. To structure the data
2. To permit the simple retrieval of data in response to queries or reports.
3. Simplify maintenance of data through update, delete and insert operations.
4. To reduce the need to restructure or reorganize the data when new application requirement arises.

23.2 DATA MODELS

To computerize a database, first the database has to be designed. If database is not designed properly then it definitely leads to failure of any software based on it. Proper attention must be paid to study the database to be designed. The developer should know and very well understand all the minute details of the database then only he/she can develop a good and sustainable database. For this purpose the developer has to design three models of the database – First the **Conceptual Data Model**, then the **Logical Data Model**, and then the **Physical Data Model** – in this sequence only.

To understand this concept of data models assume the simplified version of School. This school offers various courses (vocational, academic, short term, long term etc.) to its students. Due to increasing number of students and a large number of courses, it is becoming difficult for the school to manage this database (containing the data of students, courses, and results) manually. So, the school decides to hire a software consultant, say Mr. SoftCon, to computerize this database. To computerize the database Mr. SoftCon designs the conceptual data model, logical data model and physical data model.

Conceptual Data Model: Conceptual Data Model can be thought of as a bird's eye-view of the database. It identifies various entities and the highest-level relationship among them.

In our example of School, the entities are: Students, Courses, Marks, and Results. Results are prepared by taking data of students, courses, and marks. This can be displayed as follows:

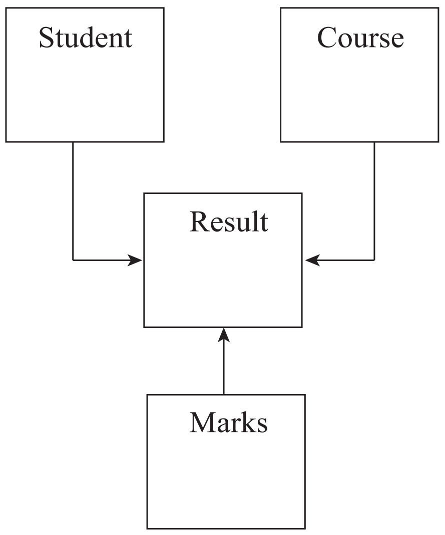


Fig. 23.1: Conceptual data model

Conceptual data model has the following features:

- It includes important entities and relationships among them.
- No attribute is specified.
- No primary key is specified.

Logical data Model: Logical data model can be thought of as a closer view of the database. In the conceptual model we look at the entities and their relationships only. In logical data model we describe the attributes of these entities in as much details as possible.

In our example of school, the entities are: Students, Courses, Marks, and Results.

In the logical data model we may consider their attributes as follows:

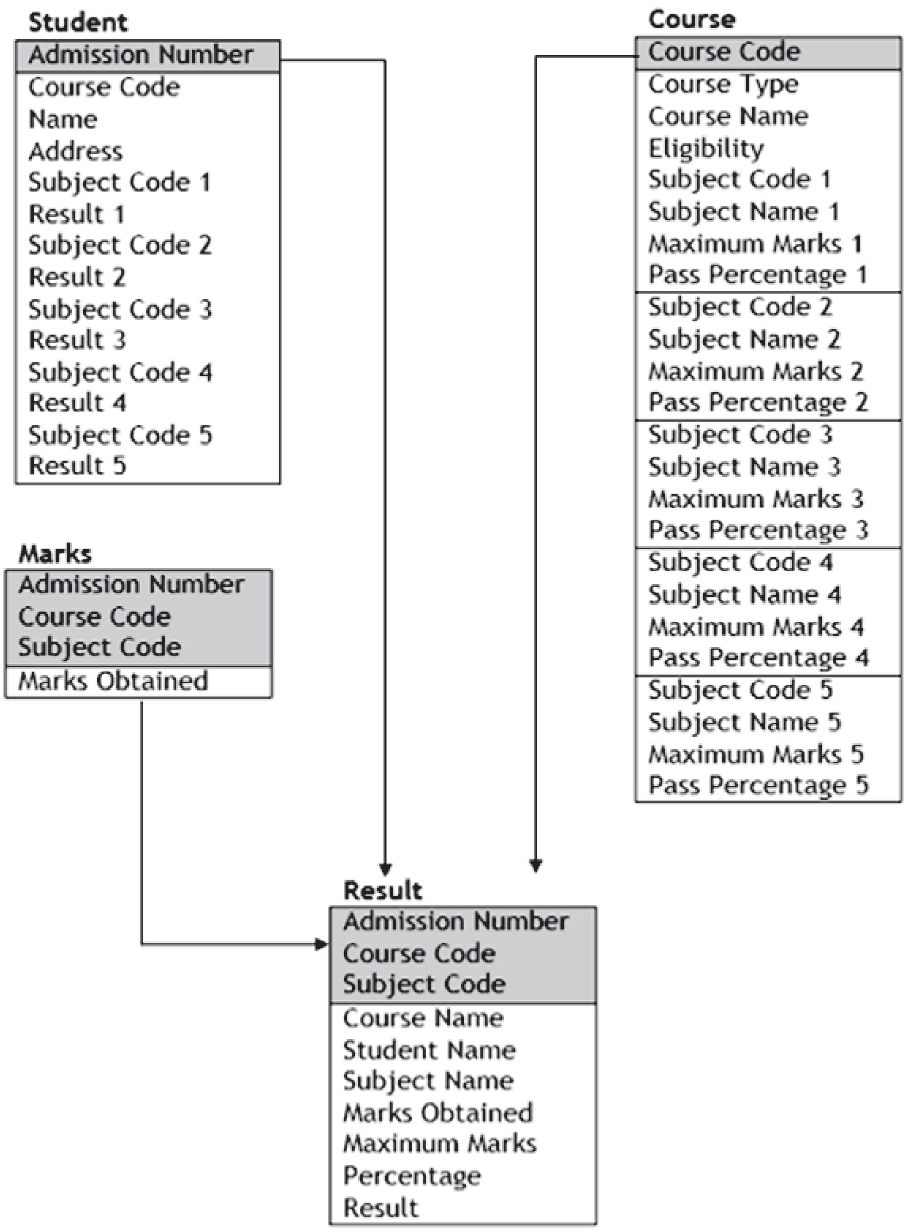


Fig. 23.2: logical data model

Foreign Keys:

Table	Foreign Key(s)
Course	None
Student	Course Code, Subject Codes
Marks	Admission Number, Course Code, Subject Code
Result	Admission Number, Course Code, Subject Code

Logical data model has the following features:

- It includes all entities and relationships among them.
- Normalization occurs at this level.
- All attributes for each entity are specified.
- The primary key for each entity is specified. In our example primary keys have been highlighted.
- Foreign keys (keys identifying the relationship between different entities) are specified.

Physical data Model: Physical data model represents the actual implementation of the Logical Data Model. A physical data model shows all the table structures including column names, column data types, column constraints, primary key, foreign key and relationship between tables.

In our example of School, the physical data model can be represented as follows:

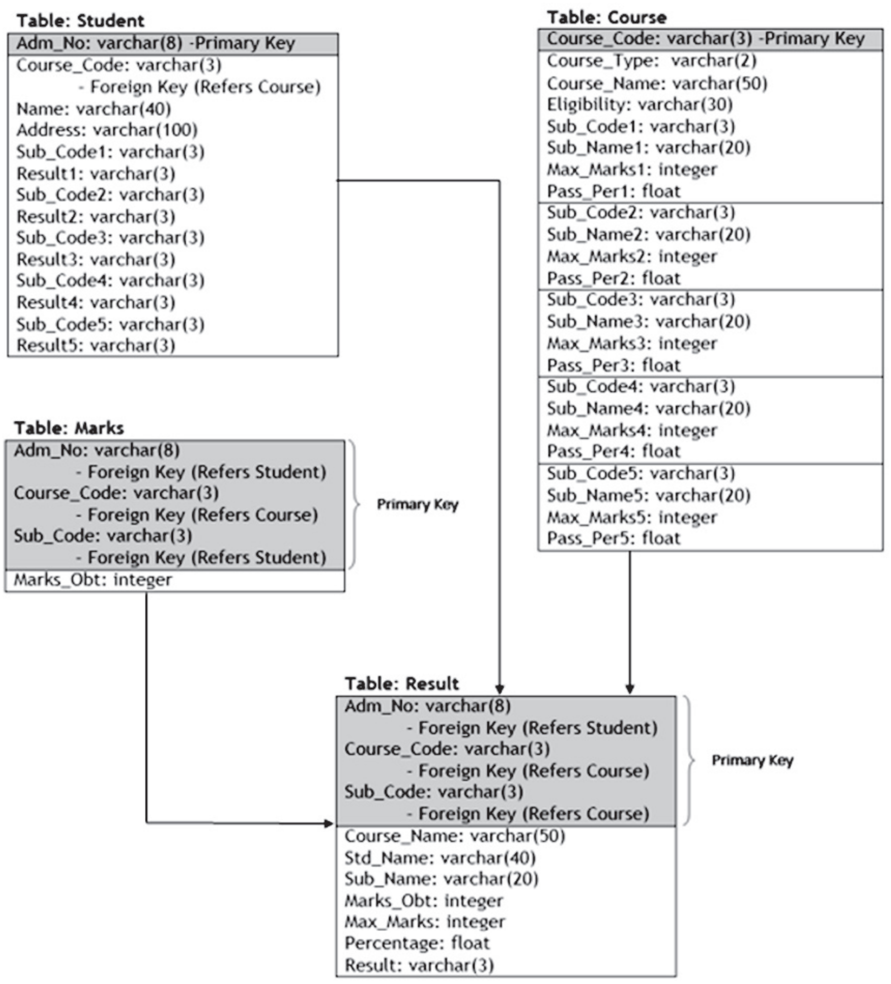


Table: Student
Adm_No: varchar(8) -Primary Key
Course_Code: varchar(3) - Foreign Key (Refers Course)
Name: varchar(40)
Address: varchar(100)
Sub_Code1: varchar(3)
Result1: varchar(3)
Sub_Code2: varchar(3)
Result2: varchar(3)
Sub_Code3: varchar(3)
Result3: varchar(3)
Sub_Code4: varchar(3)
Result4: varchar(3)
Sub_Code5: varchar(3)
Result5: varchar(3)

Table: Course
Course_Code: varchar(3) -Primary Key
Course_Type: varchar(2)
Course_Name: varchar(50)
Eligibility: varchar(30)
Sub_Code1: varchar(3)
Sub_Name1: varchar(20)
Max_Marks1: integer
Pass_Per1: float
Sub_Code2: varchar(3)
Sub_Name2: varchar(20)
Max_Marks2: integer
Pass_Per2: float
Sub_Code3: varchar(3)
Sub_Name3: varchar(20)
Max_Marks3: integer
Pass_Per3: float
Sub_Code4: varchar(3)
Sub_Name4: varchar(20)
Max_Marks4: integer
Pass_Per4: float
Sub_Code5: varchar(3)
Sub_Name5: varchar(20)
Max_Marks5: integer
Pass_Per5: float

Table: Marks
Adm_No: varchar(8) - Foreign Key (Refers Student)
Course_Code: varchar(3) - Foreign Key (Refers Course)
Sub_Code: varchar(3) - Foreign Key (Refers Student)
Marks_Obt: integer

Table: Result
Adm_No: varchar(8) - Foreign Key (Refers Student)
Course_Code: varchar(3) - Foreign Key (Refers Course)
Sub_Code: varchar(3) - Foreign Key (Refers Course)
Course_Name: varchar(50)
Std_Name: varchar(40)
Sub_Name: varchar(20)
Marks_Obt: integer
Max_Marks: integer
Percentage: float
Result: varchar(3)

Fig. 23.3 Physical data model

INTEXT QUESTIONS 23.1

1. Fill in the blanks:
   a. A .... is an organized collection of data.
   b. Having multiple copies of the same data is called ....
   c. In a ....database the data is organized in two-dimensional tables.
   d. A .... key is a group of one or more columns in a table which is used to set relationship of a table with some other table.
2. State whether the following statements are true or false:
   a. A table can have more than one primary keys.
   b. A relation can have more than one foreign keys.
   c. By normalization we combine all the data into one big table.
   d. Conceptual data model and logical data model are the same.
   e. In the physical data model we specify the data types of the attributes.

23.3 MYSQL

Once we have designed the physical data model, it is time to actually create the database in the computer and store data in it. To create the database we have to have some RDBMS (Relational DataBase Management System) software. There are many RDBMS software available like Oracle, MySQL, MS SQL Server, PostgreSQL etc.

MySQL is an open source free software. It can be downloaded free of cost from **http://dev.mysql.com/downloads/mysql** and requires no cost for its usage. After downloading MySQL software, you can easily install it. After installing MySQL you have to locate the file.

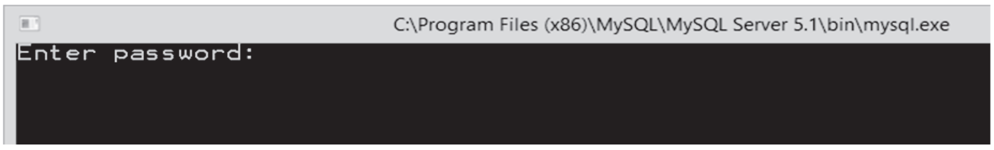
MySQL Command Line Client

Most probably you will find this file in the folder (Assuming that MySQL is installed in C:\ drive)

C:\Program Files\MySQL\MySQL Server 5.1\bin

If you do not find the file here then you can search for it. Once you find it, you should create its shortcut on the desktop so that you can easily run it.

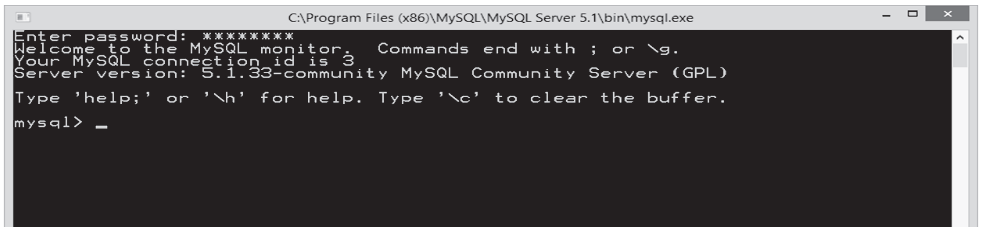
When you run this file, MySQL window will open with a prompt to provide password:



```
Enter password:
```

Fig. 23.4: MySQL Window

Here you should enter the same password which you entered during installation. If you enter the correct password, MySQL will welcome you with a MySQL prompt (`mysql> _`) in the window:



```
Enter password: ********
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3
Server version: 5.1.33-community MySQL Community Server (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> _
```

Fig. 23.5: MySQL command prompt

Now you can use MySQL to create and manage databases. To exit from MySQL you have to give the command QUIT or EXIT at the MySQL prompt.

23.3.1 Structured Query Language:

After installing MySQL you are ready to use it. Now you can create a database and tables within it. You can also enter data in the tables and manipulate this data. But the question is: How do we do this? To do all this we have to use a language called SQL (Structured Query Language).

SQL is a language specifically oriented around relational databases. It is a non-procedural language and requires us to specify what is to be done as opposed to how to do it. SQL offers us various commands using which you create and manage relational databases. These commands are classified into following classes depending on their purpose:

1. **Data Definition Language (DDL)**: It consists of those commands that create, modify, and remove objects (tables, indexes, views and so on) in the database.

2. **Data Manipulation Language (DML)**: It consists of those commands that are used to retrieve, manipulate and update the data present in tables.

3. **Data Control Language (DCL)**: It consists of those commands that determine whether a user is permitted to perform a particular action.

Before we start learning SQL, let us note few points:

- SQL is a non case-sensitive language. It means that commands can be entered using uppercase letter, lowercase letters, or a combination of these two. For SQL all these mean the same.
- Each command in SQL is terminated by a semi-colon(;).
- A statement may spread into multiple lines. It will be terminated only when a semi-colon appears.

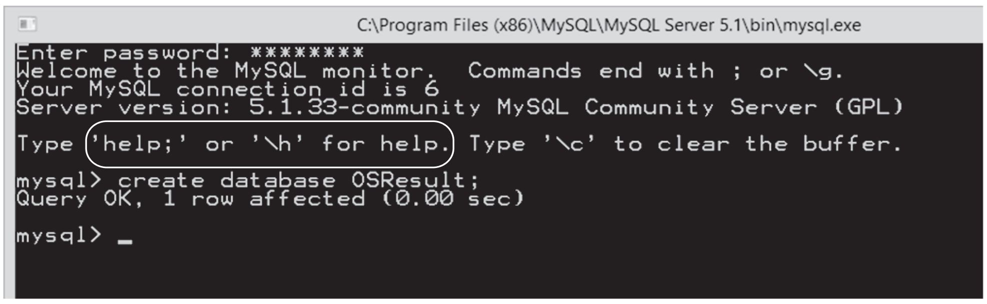
First of all we have to create the database using the SQL command CREATE DATABASE as follows:

CREATE DATABASE <DatabaseName>;

We can create multiple databases in this way. To work on any database and to switch from one database to another. We use the command: USE DB_NAME; where DB_Name is the name of the database we want to use.

Suppose you want to name the database as OSResult, then the command will be given as follows:

CREATE DATABSE OSResult;


C:\Program Files (x86)\MySQL\MySQL Server 5.1\bin\mysql.exe

```
Enter password: ********
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 6
Server version: 5.1.33-community MySQL Community Server (GPL)
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
mysql> create database OSResult;
Query OK, 1 row affected (0.00 sec)
mysql> _
```

If a given command is executed successfully then MySQL gives an OK message, otherwise it gives an error message stating why the command failed.

First the database has to be made active using USE command.

mysql> use OSResult

After creating and using the database you have to create tables. A table is created using the CREATE TABLE command as follows:

```
CREATE TABLE <TableName>
(<ColumnName1><Data Type1> NOT NULL,
<ColumnName2><Data Type2> default NULL,
 .
 .
<ColumnNameN><Data TypeN> default <some value>
);
```

Notes

A data type specifies what kind of values can be stored in a column. In MySQL various data types available are:

Data Type	Syntax	Usage	Example
Char	CHAR(n) – where n is a natural number.	For a fixed length column which can store a maximum of 255 character.	CHAR(5) specifies that a column can store a string of upto 5 characters.
Varchar	VARCHAR(n)–where n is a natural number	For a variable length column which can store a maximum of 255 character.	VARCHAR(5) specifies that a column can store a string of upto 5 characters.
Integer	INT or INTEGER	For columns storing integer values in the range -2147483648 to 2147483647.	
Decimal	DECIMAL(m,d)– where m and d are natural numbers.	For columns storing numerical values with fractional part also.	DECIMAL (6,4) specifies that a column can store values in the range -99.9999 to 99.9999
Date	DATE	For columns storing dates	For Date of Birth, Date of Joining etc.

NULL / NOT NULL is used to specify if we want to allow blank values in a column or not. The keyword 'default' is used to add a default value for any column. For e.g., default NULL or default TIMESTAMP. Let us create a table that has some information about the items sold at a local shop. Let us call the table as "ITEMS" table.

To create the ITEMS table, we will give the following command:

```
create table ITEMS (item_code char(5), item_desc varchar(20) NOT NULL;
                    CP decimal(6,2), SP decimal(6, 2));
```

Here item-code, item-desc, CP, SP are column names. Column SP will not allow blank entries. You can view the structure of the table by using the command:

```
DESCRIBE <TableName>;
            Or
DESC <TableName>;
```

For example, the command

```
desc items;
```

will display the structure of ITEMS table as follows:

Field	Type	Null	Key	Default	Extra
item_code	char(5)	YES		NULL	
item_desc	varchar(20)	YES		NULL	
CP	decimal(6,2)	YES		NULL	
sp	decimal(6,2)	NO		NULL	

Similarly, you can use the command to create the BILLS table and then display its structure.

After creating a table you have to enter data in the table. The process of entering data in a table is also called populating the table. You enter data in a table row-wise. To enter a row in a table you use the command INSERT INTO as follows:

```
INSERT INTO <TableName>
VALUES (<Value1>,<Value2>,… ,<ValueN>);
```

For example, to enter a record (row) in the ITEMS table, you use the command:

```
insert into items values (“A001”, “Curd - 1Kg Pack”, 50,60);
```

Here you should notice that the character (CHAR and VARCHAR) values are placed in double quotation marks. They can also be placed in single quotation marks. Similarly date (DATE) values must also be placed within quotation marks. Numeric values (INT and DECIMAL) must not be placed in quotation marks.

Let us insert some more rows in the ITEMS table by the commands:

```
insert into items values (‘A002’, “Curd - 0.5Kg Pack”, 27,35);
insert into items values (‘B010’, “Wheat Flour”, 18,20);
insert into items values (‘B011’, “Basmati Rice”, 35,42);
```

Now if you want to check the data in the table you use the SELECT command. The following command

```
select * from items;
```

will display the contents of the table as follows:

item_code	item_desc	cp	sp
A001	Curd - 1Kg Pack	50.00	60.00
A002	Curd - 0.5Kg Pack	27.00	35.00
B010	Wheat Flour	18.00	20.00
B011	Basmati Rice	35.00	42.00

SELECT is the most widely used command in SQL and has many variations. Now we will discuss this in detail.

SELECT

SELECT command is used to retrieve data from table(s). In its simplest form SELECT command is used to retrieve complete data from a table as follows:

```
SELECT * FROM <TableName>;
```

For example, to view all the data from ITEMS table we use the command:

```
select * from items;
```

If you want to view only specific columns of a table, you can specify the column names instead of asterisk (*) in the SELECT command as follows:

```
SELECT <ColumnName1>,<ColumnName2>. . <ColumnName n>
FROM <table name>;
```

For example, to view only the item description and selling price of items you use the following command:

```
SELECT Item_Desc, sp FROM ITEMS;
```

It will display the following:

Item_Desc	sp
Curd - 1Kg Pack	60.00
Curd - 0.5Kg Pack	35.00
Wheat Flour	20.00
Basmati Rice	42.00

The column names can be specified in any order. For example, to view SP, followed by Item_Desc, followed by CP, we use the command:

SELECT SP, Item_Desc, cp FROM ITEMS;

SELECT command can also be used to display calculated fields. For example, to view the profit on each item, we use the command:

Select item_desc, sp-cp from items;

This will display the following result:

item_desc	sp-cp
Curd - 1Kg Pack	10.00
Curd - 0.5Kg Pack	8.00
Wheat Flour	2.00
Basmati Rice	7.00

Data can be fetched from tables by using SELECT command. Conditioned data can be fetched by using WHERE clause. For example, if we want to fetch all the items that cost more than 30, we can use:-

Select * from items where CP > 30;

This will display the following result:

item_code	item_dese	CP	SP
A001	curd 1Kg Pack	50.00	60.00
B011	Basmati Rice	35.00	42.00

We can specify more than one condition in where clause to get more specified data. e.g., if we need to fetch the items whose CP is more than 30 but SP is less than 50, we can use:-

Select * from item where CP > 30 and SP <50;

As shown, we have to use 'and' or 'or' a combination of both to get any type of data that we want. Another example:-

Select * from item where (CP > 30 or SP > 30) and SP < 50;

It is important to use the braces for correct order of execution. It is interesting to see that we can fetch data from more than one table simultaneously by using these conditions in where clause and mentioning table names in from clause. For example:-

Select field1, field2, ..., fieldN from table1, table2 where condition1 AND/OR condition2 AND/OR condition....

Please notice, if the number of conditins are more then more time will be taken by the query for processing and fetching the results.

If we want to extract only unique values from any column we can use DISTINCT.

For example,

Select DISTINCT (CP), SP from ITEMS;

To sort the output based on any column in ascending or descending order, use ORDER BY

For example,

Select * from ITEMS ORDER BY SP DESC;

By default the order by will display the values in ascending order.

Modifying the structure of MySQL Tables

It is a command defined under Data definition language as it alters the structure of an existing table. It has different variations according to the operation to be performed. To add a new column to your existing table or remove a column you use alter command.

To add new columns in a table you use alter command as follows:

ALTER TABLE *table_name* **ADD** *(newcolumnname datatype (size), newcolumnname datatype (size),...)***;**

For example, to add 'profit' column in the ITEMS table you use the following command:

ALTER TABLE ITEMS **ADD** *(profit decimal (6, 2)***;**

The above command will add new column 'profit' to the 'ITEMS' table.

To remove a column in a table you use alter command as follows:

ALTER TABLE ITEMS **Drop** *profit***;**

The above command will drop a column 'profit', from the ITEMS table.

You can change the column definition by using alter command.

ALTER TABLE ITEMS **MODIFY** item_desc varchar(30);

This command will modify the existing structure of the table. In the above command, the column size of the column *item_desc* of a table items have been increased to 30 from 20.

UPDATE Query

You can update your data values of the table by using Update command as follows:

UPDATE *table_name* **SET** *field1=new-value1, field2=new-value2;*

Or

UPDATE *table_name* **SET** *field1=new-value1, field2=new-value2*

[WHERE *Clause***];**

Update command is use to update either all (without **where** clause) or selected data values (By using the **where** clause) of the table. One can update one or more fields of a same table under a single command by specifying the WHERE clause. It is a data manipulation command.

For example, to update the SP of the item B010 i.e. wheat flour from 20 to 22, you use the update command as follows:

UPDATE *ITEMS* **SET** *SP = 22*

WHERE *item_code = B010;*

This query will update the record whose id is B010. Its SP will be set to new value 22.

You can delete the data of a table by using Delete command as follows:

DELETE FROM *tablename***;**

Or

DELETE FROM *tablename* **WHERE** *search_condition***;**

Delete command is used to remove rows from the table. It can be used to delete either all the rows from the table or selected set of rows from the tables using where clause.

For example, to delete all the rows from the ITEMS table you can use the Delete command as follows:

DELETE FROM ITEMS;

The above command will delete all the rows from the table 'ITEMS' table.

DELETE FROM ITEMS WHERE *CP > 40***;**

The above command will delete the rows from the table 'ITEMS' that have CP greater than 40.

Drop command is used to remove the structure of the table from the memory. You can use drop command as follows:

Syntax:

DROP TABLE *table_name***;**

For example, to drop the structure of ITEMS table, you can use the drop command as follows:

DROP TABLE ITEMS;

The above command will drop the structure of 'ITEMS' table.

INTEXT QUESTIONS 23.2

1. Fill in the blanks:

   a. Oracle, MySQL, and PostgreSQL are examples of .... software.

   b. SQL is an abbreviation for ....

   c. .... is a language specially oriented around .... databases.

   d. .... consists of the commands which are used to create, modify, and remove objects in a database.

   e. Each command in SQL is terminated by a ....

2. State whether the following statements are true or false:

   a. MySQL is an open source free software.

   b. To run MySQL we can enter any password.

   c. DDL, DML, and DCL are three different languages.

   d. SQL is a case-sensitive language.

   e. In SQL, a statement may spread into multiple lines.

WHAT YOU HAVE LEARNT

- Database is an organized collection of data.
- Database management system is a software that allows us to manage computerized databases.

- Relation is a collection of data corresponding to the same kinds of entities in a database.
- An attribute corresponds to a column in a relation.
- The group of one or more columns used to uniquely identify each row of a relation is called primary key.
- A field or combination of fields which can be used as a primary key of relation is called a candidate key.
- Normalization is the process of analyzing the data to be represented and breaking it down into separate tables in accordance with the principles of relational structure.
- Data Definition Language consists of those commands that create, modify and remove objects in the database.
- Data Manipulation Language consists of those commands that are used to retrieve, manipulate and update data present in tables.
- Data Control Language consists of those commands that determine whether a user is permitted to perform a particular action.

TERMINAL EXERCISE

1. Create a following table:

a) Supplier

Field	Type	Null	Key
SNO	varchar(4)	No	Primary
SName	varchar(20)	Yes	
Status	number(4)	Yes	
City	varchar(20)	Yes	

b) Parts

Field	Type	Null	Key
PNO	varchar(4)	No	Primary
PName	varchar(20)	Yes	
Color	varchar(10)	Yes	
Weight	number(4)	Yes	
City	varchar(20)	Yes	

c) Project

Field	Type	Null	Key
JNO	varchar(4)	No	Primary
JName	varchar(20)	Yes	
City	varchar(20)	Yes	

d) SPJ

Field	Type	Null	Key	
SNO	varchar(4)	No		Foreign
PNO	varchar(4)	No	Primary	Foreign
JNO	varchar(4)	No		Foreign
QTY	number(4)	Yes		

2. Insert the following data in the respective tables:

a) Supplier table

SNO	SName	Status	City
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

b) Parts table

PNO	PName	Color	Weight	City
P1	Nut	Red	12	London
P2	Bolt	Green	17	Paris
P3	Screw	Blue	17	Rome
P4	Screw	Red	14	London
P5	Cam	Blue	12	Paris
P6	Cog	Red	19	London

c) Project table

JNO	JName	City
J1	Sorter	Paris
J2	Display	Rome
J3	OCR	Athens
J4	Console	Athens
J5	RAID	London
J6	Eds	Oslo
J7	Tape	London

d) SPJ table

SNO	PNO	PNO	QTY
S1	P1	J1	200
S1	P1	J4	700
S2	P3	J1	400
S2	P3	J2	200
S2	P3	J6	400
S2	P3	J7	800
S3	P3	J1	200
S3	P4	J2	500
S4	P2	J2	200
S4	P6	J3	300
S4	P6	J7	300
S5	P1	J4	100
S5	P2	J4	100
S5	P3	J4	200
S5	P4	J4	800

3. Perform the following queries on the above data:

a) Find the supplier numbers of the supplies who supplies to project 'J1'.

b) Find the supplier numbers of the supplies who supplies to project 'J1' with part 'P1'.

c) Retrieve the project name where supplier 'S1' is supplying the parts.

d) Retrieve the color values for the parts supplied by supplier 'S1'.

e) Retrieve the part numbers supplied to any project in 'London'.

f) Retrieve the supplier number who supplies to London or Paris with Red color part.

g) Find the total quantity of part P1 supplied by supplier S1.

h) Find the total number of projects supplied by supplier 'S3'.

i) Change the color of all Red parts to Orange.

ANSWERS TO INTEXT QUESTIONS

23.1

1. (a) Database
   (b) Redundancy
   (c) Relational
   (d) Foreign

2	(a) False	(b) True
	(c) False	(d) False
	(e) True	

23.2

1. (a) RDBMS
   (b) Structured Query Language
   (c) SQL, relational
   (d) DDL (Data Definition Language)
   (e) ; (or) semicolon

2.	(a) True	(b) False
	(c) False	(d) False
	(e) True	