INDEX

S.No	Topic	Page no	Date	Sign
1	Introduction			
2	ER Diagrams for Roadway Travels Using Software Tool			
3	MYSQL Installation			
4	DDL Commands			
5	DML Commands			
6	KEY Constraints			
7	Aggregate Functions and Mathematical Functions			
8	Nested Queries & Correlated Queries			
9	Views			
10	Joins			
11	Triggers			
12	Stored Procedures,TCL Commands			
13	PL/SQL			
14	DCL- Grant and Revoke.			

INTRODUCTION

Hierarchical Model

This model is like a hierarchical tree structure, used to construct a hierarchy of records in the form of nodes and branches. The data elements present in the structure have Parent-Child relationship. Closely related information in the parent-child structure is stored together as a logical unit. A parent unit may have many child units, but a child is restricted to have only one parent.

The drawbacks of this model are:

The hierarchical structure is not flexible to represent all the relationship proportions,

which occur in the real world.

It cannot demonstrate the overall data model for the enterprise because of the non-availability of actual data at the time of designing the data model.

It cannot represent the Many-to-Many relationship.

Network Model

It supports the One-To-One and One-To-Many types only. The basic objects in this model are Data Items, Data Aggregates, Records and Sets.

It is an improvement on the Hierarchical Model. Here multiple parent-child relationships are used. Rapid and easy access to data is possible in this model due to multiple access paths to the data elements.

Relational Model

Does not maintain physical connection between relations
Data is organized in terms of rows and columns in a table

The position of a row and/or column in a table is of no importance
The intersection of a row and column must give a single value

Features of an RDBMS

The ability to create multiple relations and enter data into them
An attractive query language

Retrieval of information stored in more than one table

An RDBMS product has to satisfy at least Seven of the 12 rules of Codd to be accepted as a full- fledged RDBMS.

Relational Database Management System

RDBMS is acronym for Relation Database Management System. Dr. E. F. Codd first introduced the Relational Database Model in 1970. The Relational model allows data to be represented in a simple row- column. Each data field is considered as a column and each record is considered as a row. Relational Database is more or less similar to Database Management S ystem. In relational model there is relation between their data elements. Data is stored in tables. Tables have columns, rows and names. Tables can be related to each other if each has a column with a common type of information. The most famous RDBMS packages are Oracle, Sybase and Informix.

Simple example of Relational model is as follows :

Student Details Table

Roll_no	Sname	S_Address
1	Rahul	Satelite
2	Sachin	Ambawadi
3	Saurav	Naranpura

Student Marksheet Table

Rollno	Sub1	Sub2	Sub3
1	78	89	94
2	54	65	77
3	23	78	46

Here, both tables are based on students details. Common field in both tables is Rollno. So we can say both tables are related with each other through Rollno column.

Degree of Relationship

One to One (1:1)

One to Many or Many to One (1:M / M: 1)

Many to Many (M: M)

The Degree of Relationship indicates the link between two entities for a specified occurrence of each.

One to One Relationship: (1:1)

1: 1

Student Has Roll No.

One student has only one Rollno. For one occurrence of the first entity, there can be, at the most one related occurrence of the second entity, and vice-versa.

One to Many or Many to One Relationship: (1:M/M: 1)

1 :M

Course Contains Students

As per the Institutions Norm, One student can enroll in one course at a time however, in one course, there can be more than one student.

For one occurrence of the first entity there can exist many related occurrences of the second entity and for every occurrence of the second entity there exists only one associated occurrence of the first.

Many to Many Relationship: (M:M)

M: M

Students Appears Tests

The major disadvantage of the relational model is that a clear-cut interface cannot be determined. Reusability of a structure is not possible. The Relational Database now accepted model on which major database system are built.

Oracle has introduced added functionality to this by incorporated object-oriented capabilities. Now it is known is as Object Relational Database Management System (ORDBMS). Object-oriented concept is added in Oracle8.

Some basic rules have to be followed for a DBMS to be relational. They are known as Codd's rules, designed in such a way that when the database is ready for use it encapsulates the relational theory to its full potential. These twelve rules are as follows.

E. F. Codd Rules

1. **The Information Rule**

   All information must be store in table as data values.

2. **The Rule of Guaranteed Access**

   Every item in a table must be logically addressable with the help of a table name.

3. **The Systematic Treatment of Null Values**

   The RDBMS must be taken care of null values to represent missing or inapplicable information.

4. **The Database Description Rule**

   A description of database is maintained using the same logical structures with which data was defined by the RDBMS.

5. **Comprehensive Data Sub Language**

   According to the rule the system must support data definition, view definition, data manipulation, integrity constraints, authorization and transaction management operations.

6. **The View Updating Rule**

   All views that are theoretically updatable are also updatable by the system.

7. **The Insert and Update Rule**

   This rule indicates that all the data manipulation commands must be operational on sets of rows having a relation rather than on a single row.

8. **The Physical Independence Rule**

   Application programs must remain unimpaired when any changes are made in storage representation or access methods.

9. **The Logical Data Independence Rule**

   The changes that are made should not affect the user's ability to work with the data.The change can be splitting table into many more tables.

10. **The Integrity Independence Rule**

    The integrity constraints should store in the system catalog or in the database.

11. **The Distribution Rule**

    The system must be access or manipulate the data that is distributed in other systems.

12. The Non-subversion Rule

If a RDBMS supports a lower level language then it should not bypass any integrity constraints defined in the higher level.

Object Relational Database Management System

Oracle8 and later versions are supported object-oriented concepts. A structure once created can be reused is the fundamental of the OOP's concept. So we can say Oracle8 is supported Object Relational model, Object - oriented model both. Oracle products are based on a concept known as a client-server technology. This concept involves segregating the processing of an application between two systems. One performs all activities related to the database (server) and the other performs activities that help the user to interact with the application (client). A client or front-end database application also interacts with the database by requesting and receiving information from database server. It acts as an interface between the user and the database.

The database server or back end is used to manage the database tables and also respond to client requests.

Introduction to ORACLE

ORACLE is a powerful RDBMS product that provides efficient and effective solutions for major database features. This includes:

Large databases and space management control
Many concurrent database users

High transaction processing performance
High availability

Controlled availability

Industry accepted standards
Manageable security

Database enforced integrity
Client/Server environment

Distributed database systems
Portability

Compatibility

Connectivity

An ORACLE database system can easily take advantage of distributed processing by using its Client/ Server architecture. In this architecture, the database system is divided into two parts:

A front-end or a client portion

The client executes the database application that accesses database information and interacts with the user.

A back-end or a server portion

The server executes the ORACLE software and handles the functions required for concurrent, shared data access to ORACLE database.

ILLUSTRATION: ROADWAY TRAVELS

"Roadway Travels" is in business since 1977 with several buses connecting different places in India. Its main office is located in Hyderabad.

The company wants to computerize its operations in the following areas:

Reservations
Ticketing
Cancellations

Reservations :

Reservations are directly handeled by booking office.reservations can be made 60 days in advance in either cash or credit. In case the ticket is not available,a wait listed ticket is issued to the customer. This ticket is confirmed against the cancellation.

Cancellation and modification:

Cancellations are also directly handed at the booking office. Cancellation charges will be charged.

Wait listed tickets that do not get confirmed are fully refunded.

WEEK-1

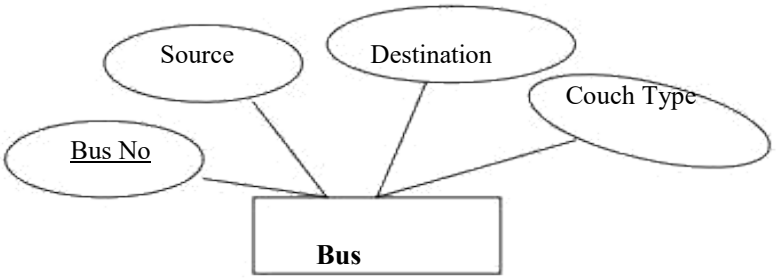
AIM: Analyze the problem and come with the entities in it. Identify what Data has to be persisted in the databases.

The Following are the entities:

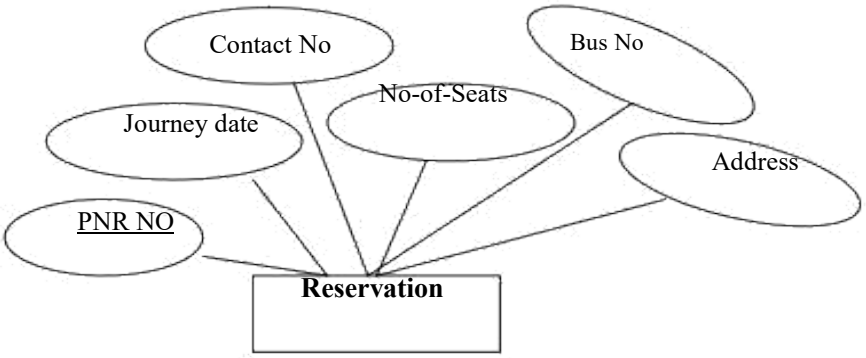
1 .Bus
2. Reservation
3. Ticket
4. Passenger
5. Cancellation

The attributes in the Entities:

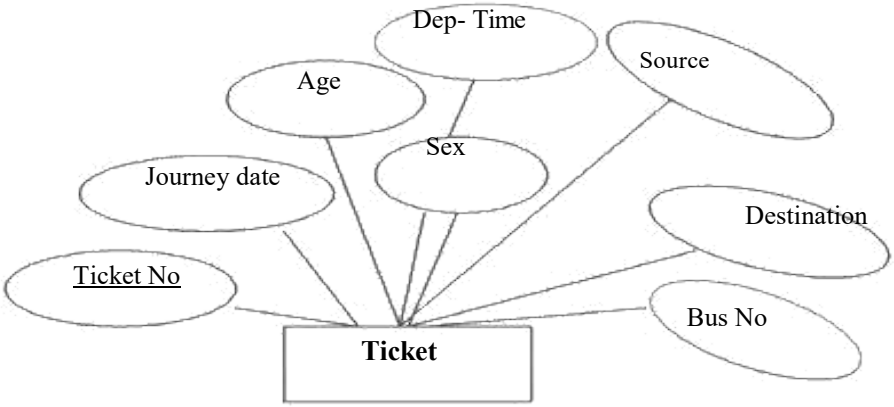
Bus:(Entity)



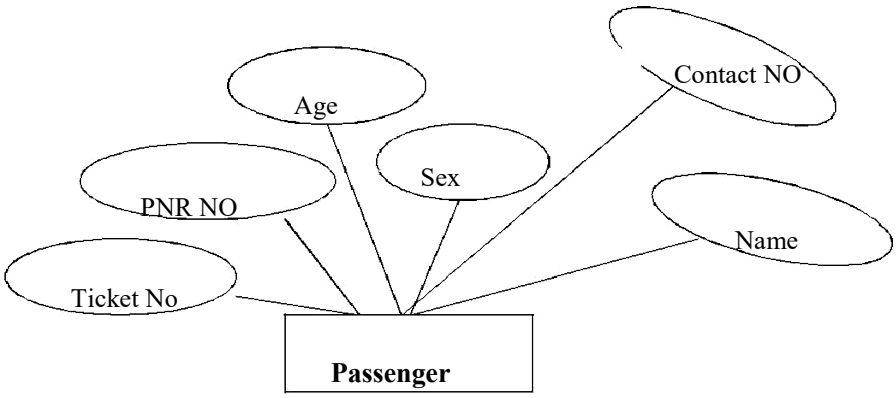
Reservation (Entity)



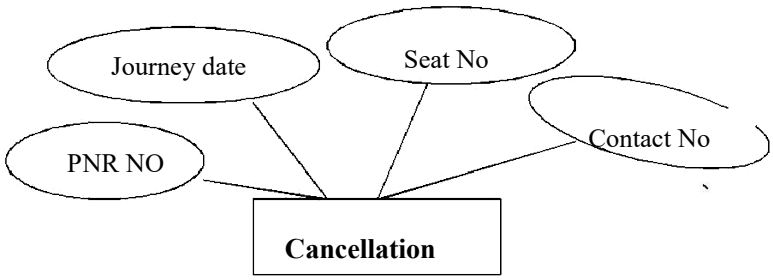
Ticket :(Entity)



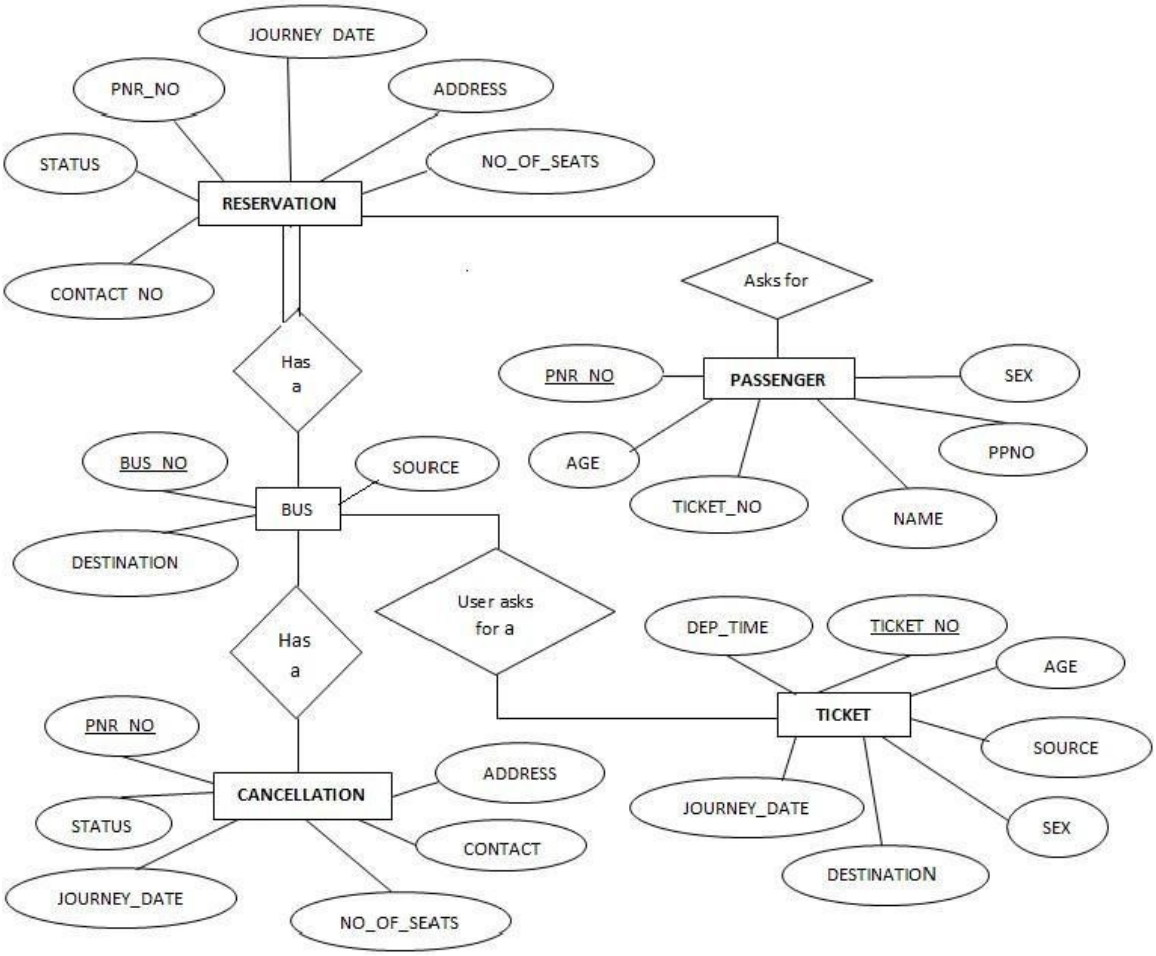
Passenger:



Cancellation (Entity)



Concept design with E-R Model:



Excersize 1:

Consider the following information about a university database:

- Professors have an SSN, a name, an age, a rank, and a research specialty.
- Projects have a project number, a sponsor name (e.g., NSF), a starting date, an ending date, and a budget.
- Graduate students have an SSN, a name, an age, and a degree program (e.g., M.S. or Ph.D.).
- Each project is managed by one professor (known as the project's principal investigator).
- Each project is worked on by one or more professors (known as the project's co-investigators).
- Professors can manage and/or work on multiple projects.
- Each project is worked on by one or more graduate students (known as the project's research assistants).
- When graduate students work on a project, a professor must supervise their work on the project. Graduate students can work on multiple projects, in which case they will have a (potentially di_erent) supervisor for each one.
- Departments have a department number, a department name, anda main office.

- Departments have a professor (known as the chairman) who runs the department.
- Professors work in one or more departments, and for each department that they work in, a time percentage is associated with their job.
- Graduate students have one major department in which they are working on their degree.
- Each graduate student has another, more senior graduate student (known as a student advisor) who advises him or her on what courses to take.

WORKSHEET

What is SQL and SQL*Plus

Oracle was the first company to release a product that used the English-based Structured Query Language or SQL. This language allows end users to manipulate information of table(primary database object). To use SQL you need not to require any programming experience. SQL is a standard language common to all relational databases. SQL is database language used for storing and retrieving data from the database. Most Relational Database Management Systems provide extension to SQL to make it easier for application developer. A table is a primary object of database used to store data. It stores data in form of rows and columns.

SQL*Plus is an Oracle tool (specific program) which accepts SQL commands and PL/SQL blocks and executes them. SQL *Plus enables manipulations of SQL commands and PL/SQL blocks. It also performs additional tasks such as calculations, store and print query results in the form of reports, list column definitions of any table, access and copy data between SQL databases and send messages to and accept responses from the user. SQL *Plus is a character based interactive tool, that runs in a GUI environment. It is loaded on the client machine.

To communicate with Oracle, SQL supports the following categories of commands:

1)Data Definition Language

Create, Alter, Drop and Truncate

2)Data Manipulation Language

Insert, Update, Delete and Select

3)Transaction Control Language

Commit, Rollback and Save point

4)Data Control Language

Grant and Revoke

The following are tabular representation of the above entities and relationships

BUS:

COLUMN NAME	**DATA TYPE**	**CONSTRAINT**
Bus No	varchar2(10)	**Primary Key**
Source	varchar2(20)	
Destination	varchar2(20)	
Couch Type	varchar2(20)	

Reservation:

COLUMN NAME	**DATA TYPE**	**CONSTRAINT**
PNRNo	number(9)	**Primary Key**
Journey date	Date	
No-of-seats	integer(8)	
Address	varchar2(50)	
Contact No	Number(9)	Should be equal to 10 numbers and not allow other than numeric
BusNo	varchar2(10)	**Foreign key**
Seat no	Number	

Ticket:

COLUMN NAME	**DATA TYPE**	**CONSTRAINT**
Ticket_No	number(9)	**Primary Key**
Journey date	Date	
Age	int(4)	
Sex	Char(10)	
Source	varchar2(10)	
Destination	varchar2(10)	
Dep-time	varchar2(10)	
Bus No	Number2(10)	

Passenger:

COLUMN NAME	**DATA TYPE**	**CONSTRAINT**
PNR No	Number(9)	**Primary Key**
Ticket No	Number(9)	Foreign key
Name	varchar2(15)	
Age	integer(4)	
Sex	char(10)	(Male/Female)
Contact no	Number(9)	Should be equal to 10 numbers and not allow other than numeric

Cancellation:

COLUMN NAME	**DATA TYPE**	**CONSTRAINT**
PNR No	Number(9)	Foriegn-key
Journey-date	Date	
Seat no	Integer(9)	
Contact_No	Number(9)	Should be equal to 10 numbers and not allow other than numeric

Record - Notes

WEEK-2

AIM: Installation of MySQL and practicing DDL & DML commands.

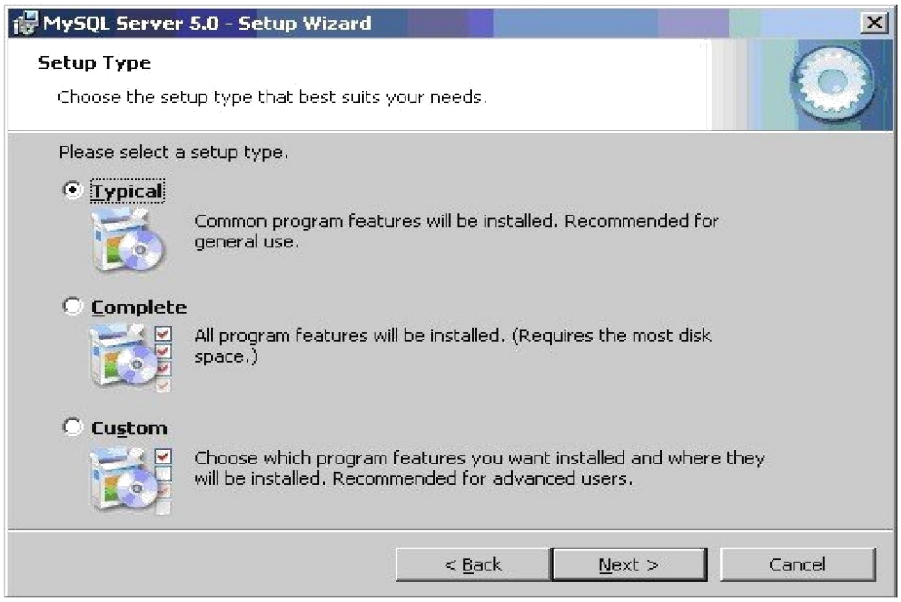
1. Steps for installing MySQL

Step1

Make sure you already downloaded the **MySQL essential 5.0.45 win32.msi file**. Double click on the .msi file.

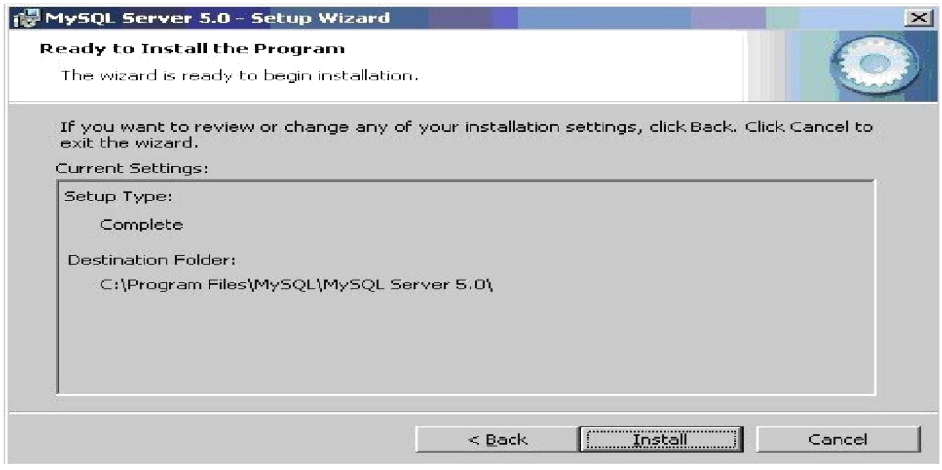
Step2

This is MySQL Server 5.0 setup wizard. The setup wizard will install MySQL Server 5.0 release 5.0.45 on your computer. To continue, click **next.**



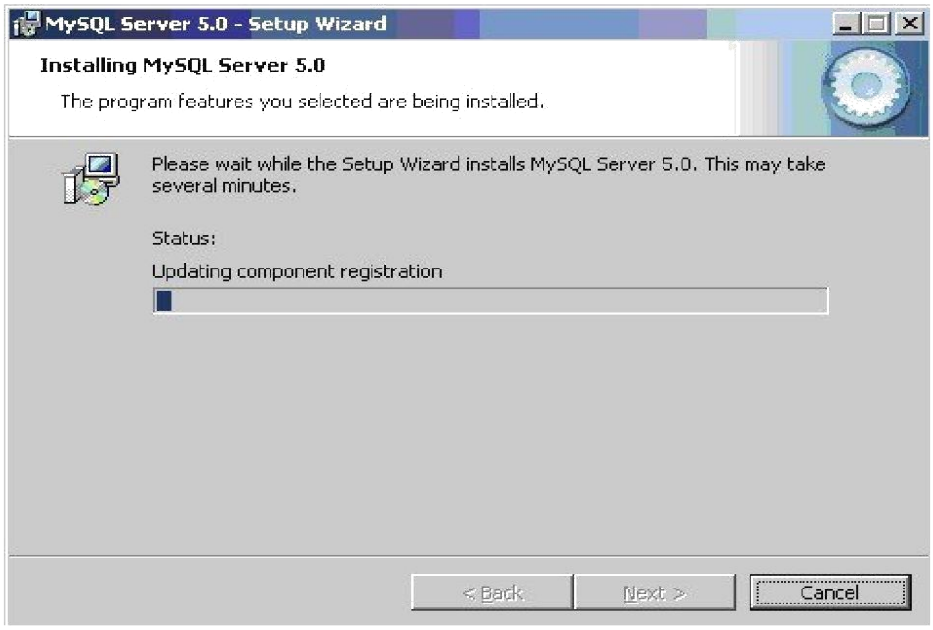
Step3

Choose the setup type that best suits your needs. For common program features select ***Typical*** and it's recommended for general use. To continue, click **next**.



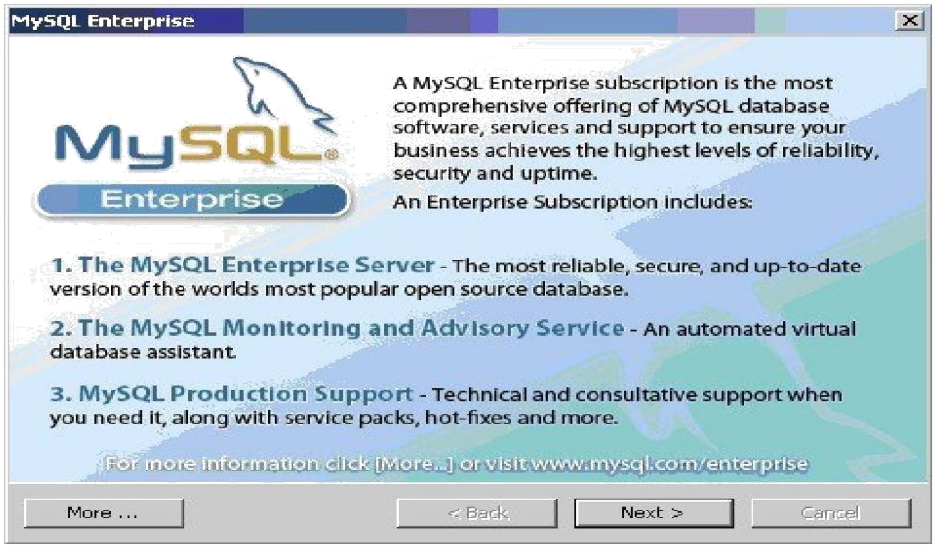
Step4

This wizard is ready to begin installation. Destination folder will be in **C:\Program Files\MySQL\MySQL Server 5.0**. To continue, click **next**.

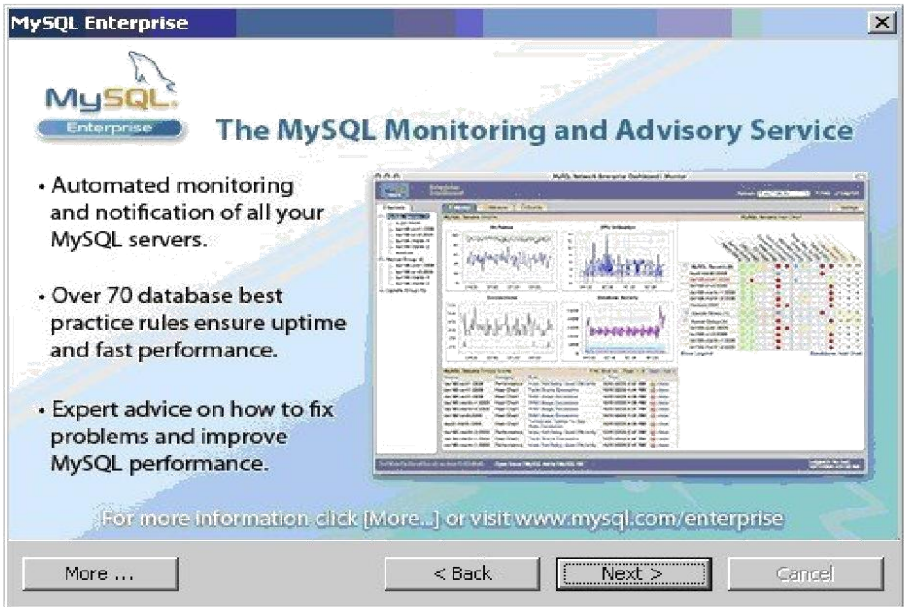


step5

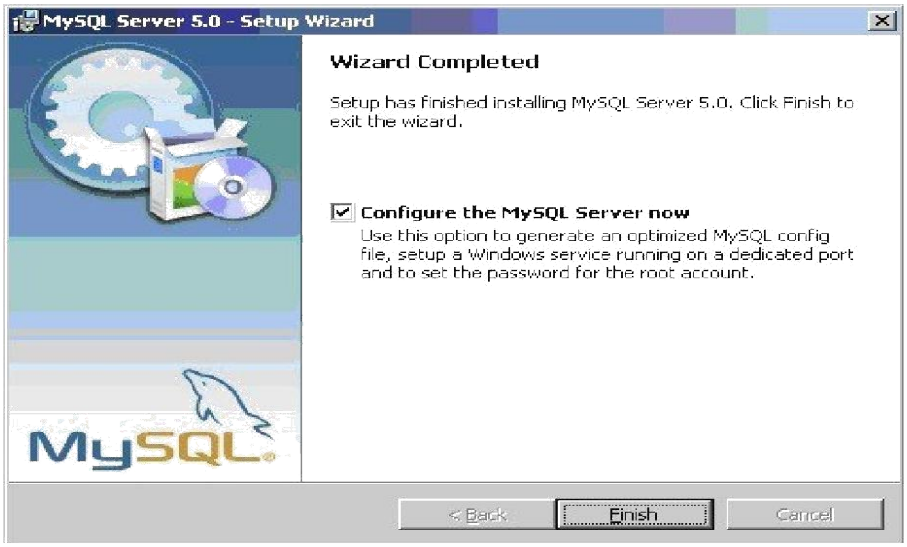
The program features you selected are being installed. Please wait while the setup wizard installs MySQL 5.0. This may take several minutes.



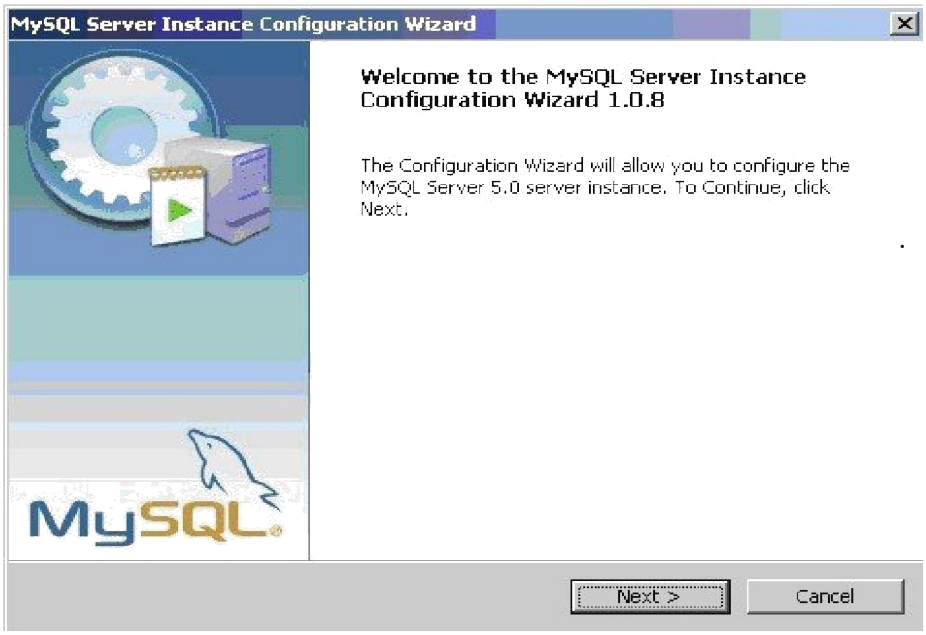
Step6
To continue, click **next**.



Step7
To continue, click **next**.

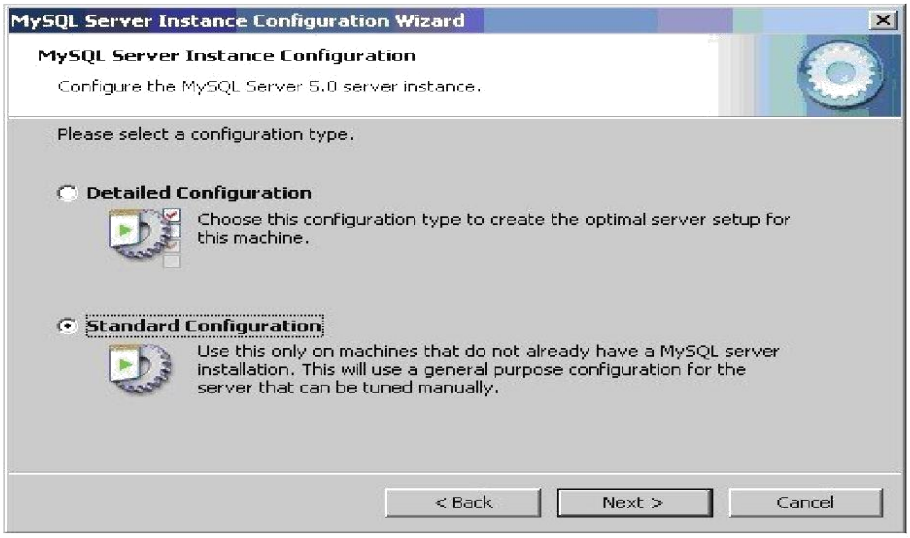


Step8
Wizard Completed. Setup has finished installing MySQL 5.0. **Check** the configure the MySQL server now to continue. Click **Finish** to exit the wizard



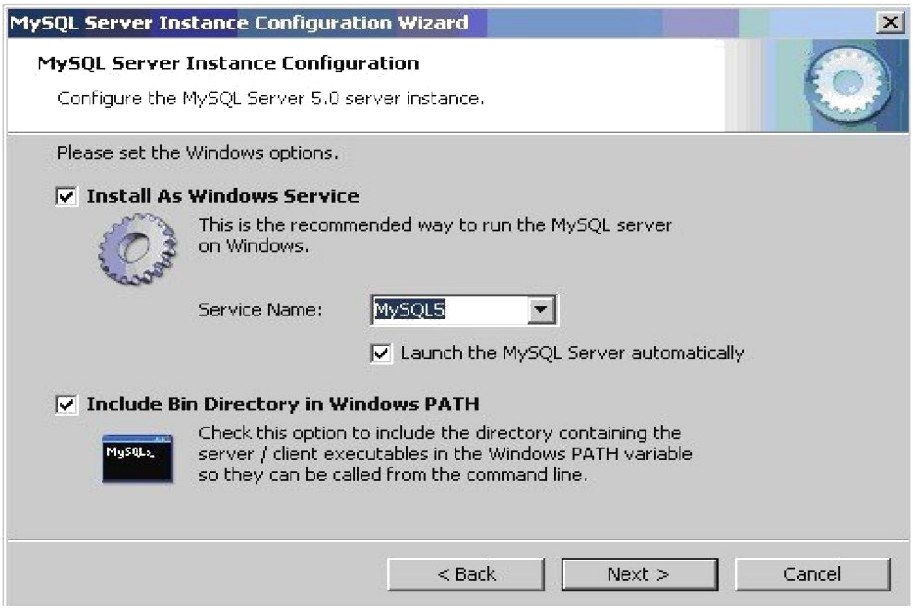
Step9

The configuration wizard will allow you to configure the MySQL Server 5.0 server instance. To continue, click **next**.



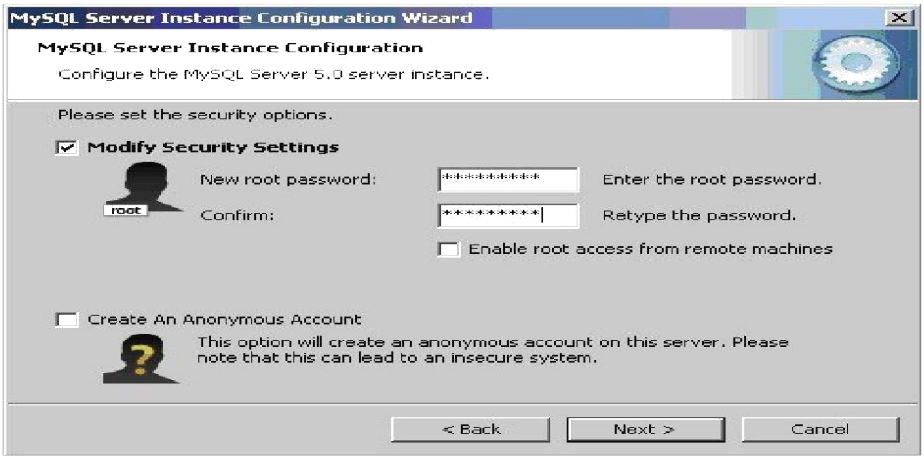
Step10

Select a **standard configuration** and this will use a general purpose configuration for the server that can be tuned manually. To continue, click **next**.



Step11

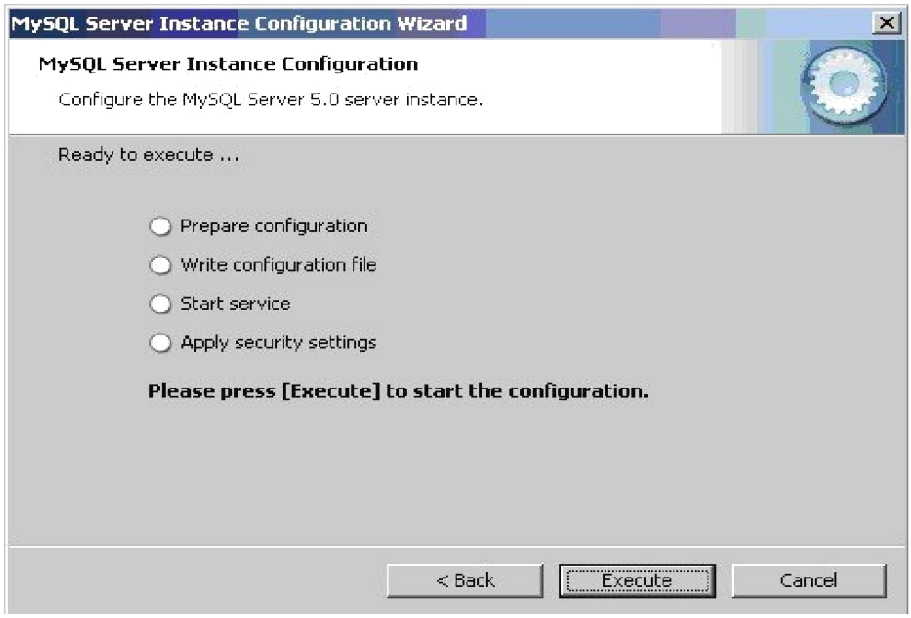
Check on the **install as windows service** and **include bin directory in windows path**. To continue, click **next**.



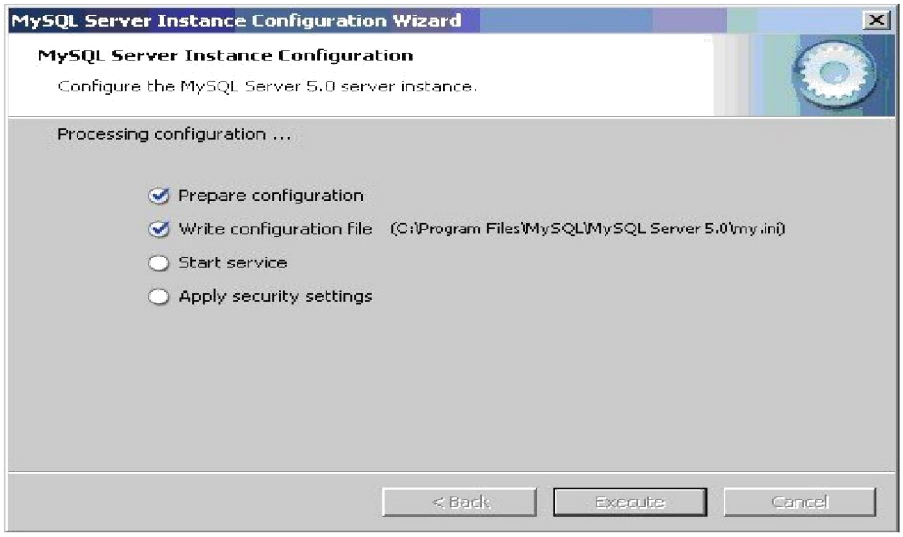
Step12

Please set the security options by entering the root password and confirm retype the password.

continue, click next.



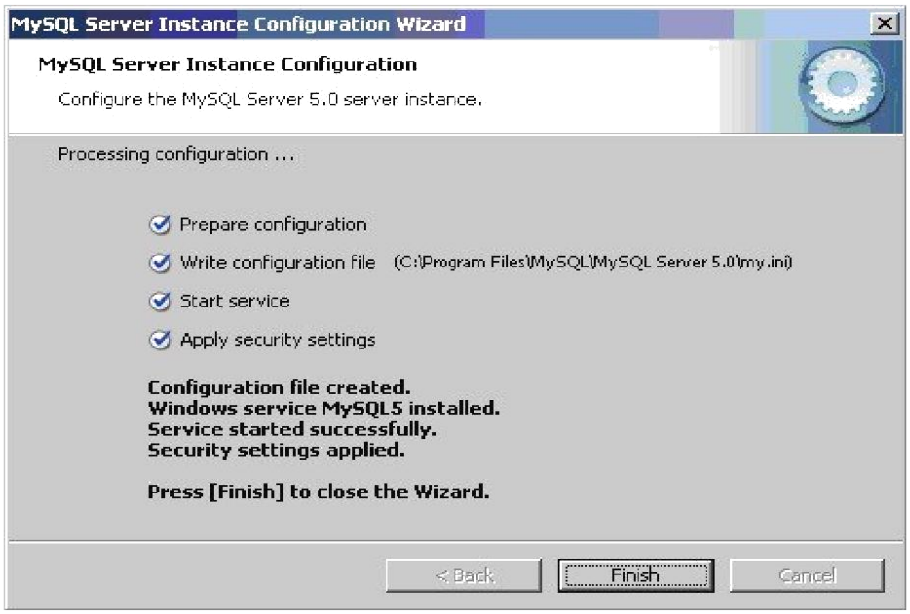
Step13
Ready to execute? Clicks **execute** to continue.



Step14
Processing configuration in progress.

Step15

Configuration file created. Windows service MySQL5 installed. Press **finish** to close the wizard.



Record - Notes

PRACTISING DDL & DML COMMANDS

Data Definition Language

The data definition language is used to create an object, alter the structure of an object and also drop already created object. The Data Definition Languages used for table definition can be classified into following:

- Create table command
- Alter table command
- Truncate table command
- Drop table command

WEEK-3

1. CREATION OF TABLES:

SQL - CREATE TABLE:

Table is a primary object of database, used to store data in form of rows and columns. It is created using following command:

Syntax: CREATE TABLE tablename (column_name data_ type constraints, …)

mysql>CREATE TABLE SAILORS ((SID int(10) PRIMARY KEY,
SNAME VARCHAR (10), RATING int (10), AGE int (10));

Table Created.

Desc command

The DESCRIBE command is used to view the structure of a table as follows.

mysql>DESC SAILORS;

TEST RESULT

Example 1: Create an RESERVES table with fields (SID , BID ,DAY) and display using DESCRIBE command.

Example 2:Create a BOATS table with Fields(BID,BNAME,COLOR)and display using DESCRIBE command

2)ALTER TABLE :

To ADD a column:

SYNTAX: ALTER TABLE <TABLE NAME>ADD (<NEW COLUMN NAME><DATA TYPE>(<SIZE>), <NEW COLUMN NAME><DATA TYPE>(<SIZE>) .....);

EX: (Write your own Query)

TEST OUTPUT

To DROP a column:

SYNTAX: ALTER TABLE <TABLE NAME>DROP COLUMN <COLUMN NAME>;.

EX: (Write your own Query)

TEST OUTPUT

To MODIFY a column:

SYNTAX: ALTER TABLE <TABLE NAME>MODIFY(<COLUMN NAME> <NEW DATATYPE>(<NEW SIZE>));

EX: (Write your own Query)

TEST OUTPUT

Example1:

mysql>ALTER TABLE SAILOR ADD (SNO NUMBER(10));

TEST OUTPUT

2. **RENAME A TABLE**

Rename command is used to give new names for existing tables.

mysql> **RENAME** oldtablename TO

newtablename; EX: (Write your own Query)

TEST OUTPUT

3. **TRUNCATE A TABLE**

Truncate command is used to delete all records from a table.

mysql> TRUNCATE TABLE tablename;

EX: (Write your own Query)

TEST OUTPUT

4. **DROP A TABLE**

Drop command is used to remove an existing table permanently from database.

mysql> DROP TABLE tablename;

EX: (Write your own Query)

TEST OUTPUT

Record - Notes

WEEK-4

DML COMMANDS

1. To Retrieve / Display Data from Tables:

a. Select command is used to select values or data from table

SYNTAX

SELECT * FROM TABLENAME;

Example:

mysql>SELECT * FROM

SAILORS; TEST OUTPUT:

b. The retrieving of specific columns from a table

mysql> **SELECT** columnname 1, columnname 2,…. columnname n FROM

tablename; EX: (Write your own Query)

TEST OUTPUT

mysql> SELECT DISTINCT **columnname 1, columnname 2,…. columnname n FROM tablename;**

EX: (Write your own Query)

TEST OUTPUT

c. Selecting a data set from table data

mysql> **SELECT** columnname 1, columnname 2,…. columnname n FROM

tablename WHERE searchcondition;

EX: (Write your own Query)

TEST OUTPUT

Example1: Display Data From RESERVES Table

Example2: Display Data From BOATS Table

2. INSERTING DATA IN TO TABLE

Insert command is used to insert rows into the table.

SYNTAX:
INSERT INTO tablename (columnname1, columnname2,….columnname n)

Example:

mysql>INSERT INTO SAILORS VALUES (22,'DUSTIN', 7, 45.0);
1 row created
mysql>INSERT INTO SAILORS VALUES (29,'BRUTUS', 1, 33.0);
1 row created

INSERTION of Data can also be done by the following Syntax:

SYNTAX

INSERT IN TO tablename (columnname1, columnname2,….columnname n)
VALUES(Value1,Value2,..Value n);

Example:
mysql>INSERT INTO SAILORS(SID,SNAME,RATING,AGE)
VALUES (31,'LUBBER', 8, 55.5);
1 row created

Example1: INSERT data into RESERVES table:

TEST OUTPUT:

Example 2: INSERT data into BOATS table:

TEST OUTPUT:

UPDATE

This SQL command is used to modify the values in an existing table.

mysql>**UPDATE** tablename

SET column1= expression1, column2= expression 2,...

WHERE somecolumn=somevalue;

An expression consists of either a constant (new value), an arithmetic or string operation or an SQL query. Note that the new value to assign to <column> must matching data type.

An update statement used without a where clause results in changing respective attributes of all tuples in the specified table.

Example1: UPDATE SAILORS S
SET S.age=S.age+1,S.rating=S.rating-1
Where S.sid=34546;
TEST OUTPUT

Example2: (Write your own Query)

TEST OUTPUT

DELETE :

In order to delete rows from a table we use this command

mysql>**DELETE** FROM tablename WHERE condition;

Based on the condition specified the rows gets fetched from the table and gets deleted in table. Here the WHERE clause is optional.

Example1: DELETE S.AGE FROM SAILORS S where S.Sname='Smith';

TEST OUTPUT

Example2: DELETE FROMSAILORS;

TEST OUTPUT

Record - Notes

WEEK-5

KEY CONSTRAINTS

Domain Integrity constraints
Entity Integrity constraints
Referential Integrity constraints

1. PRIMARY KEY & NOT NULL

Example:

```
CREATE TABLE sailors ( sid integer,
                       sname varchar(32) ,
                       rating integer NOT
                       NULL, age real,
                        PRIMARY KEY (sid) );
```

Table created.

Test Output:

Example: Practice with your own Query:

Test Output:

Imposing IC using ALTER

Example: Alter Table Sailors MODIFY sname varchar(32) NOT NULL;

Test Output

Example:Practice with your own Query:

Test Output:

DEFAULT:

```
CREATE TABLE sailors ( sid integer,
                       sname varchar(32) ,
                       rating integer NOT
                       NULL, age real
                       DEFAULT 25,
                       PRIMARY KEY (sid)
                       );
```

Example:Practice with your own Query:

Test Output:

2. UNIQUE:

```
CREATE TABLE sailors ( sid integer,
                       sname varchar(32)
                       UNIQUE , rating integer,
                        age real DEFAULT
                        25, PRIMARY
                        KEY (sid) );
```

Test Output:

Example: Practice with your own Query:

Test Output:

3. FOREIGN KEY

```
CREATE TABLE reserves ( sid integer not null, bid integer not null, day datetime not null,
PRIMARY KEY (sid, bid, day), FOREIGN KEY (sid) REFERENCES sailors(sid) );
```

Example: Practice with your own Query:

Test Output

Adding Foreign Key to an existing Table

Alter table reserves ADD Foreign Key(sid REFERENCES Sailors(sid));

Example: Practice with your own Query:

Test Output

Record - Notes

WEEK-6

I) AGGREGATE FUNCTIONS and NUMERIC FUNCTIONS

1. COUNT:

SYNTAX:

Select count ([<distinct>/<ALL]<expr>)

INPUT mysql>

Count number of different sailor names?

Select COUNT (distinct s.sname) from sailors

TEST RESULT:

2. SUM:

SYNTAX:

Select SUM ([<distinct>/<ALL]<n>)

INPUT mysql>

Find the sum of ages of all sailors?

Select Sum(S.age)from sailors S;

TEST RESULT:

3. AVG:

SYNTAX:

Select AVG ([<distinct>/<ALL]<n>)

INPUT mysql>

Find average of rating of all sailors?

Select avg(S.rating)from sailors S;

TEST RESULT:

4. MINIMUM(MIN):

SYNTAX:

Select MIN ([<distinct>/<ALL]<expr>)

INPUT mysql>

Find youngest sailor from sailors?

Select min(S.age)from sailors S;

TEST RESULT:

5. MAXIMUM(MAX):

SYNTAX:

Select MAX ([<distinct>/<ALL]<expr>)

INPUT mysql>

Find sid of the oldest sailors?

Select max(s.sid)from sailor

TEST RESULT:

II) NUMERIC FUNCTIONS

Select abs(-9);

1) SYNTAX: Ceil()
   Ex: Select ceil(9.5);
   test output

2) SYNTAX :Floor()
   Ex: Select floor(10.5);
   test output

3) SYNTAX mod()
   Ex: select mod(17,5);
   test output

4) SYNTAX :power(n,m)
   Ex: select power(2,2);
   test output

5) SYNTAX : round(n,m)
   Ex: select round(10.586,2);
   test output

6) SYNTAX : truncate(n,m)
   Ex: select truncate(1.223,1);
   test output

7) SYNTAX : sign()
   Ex: select sign(-5);
   test output

8) SYNTAX : sqrt()
   Ex: select sqrt(25);
   test output

III) COMPARISON OPERATORS:

BETWEEN & AND
Example:

mysql> select * from emp_master where salary BETWEEN 5000 AND

8000; Test Output:

IN Operator:

mysql>Select * from emp_master where deptno IN(10,30);

Test Output:

LIKE Operator:

mysql>select*From emp_master where job like 'M%';

Test Output:

Logical operator:
mysql>select*From emp_master where job like „_lerk";
Test Output:

:
AND Operator:
mysql> select * from emp_master where salary > 5000 and comm <
750;
Test Output:

OR Operator:

mysql>select * from emp_master where salary > 5000 or comm < 750;

Test Output:

:

NOT Operator:

mysql>select*from emp_master where not salary=10000; Test Output:

IV) SINGLE ROW FUNCTIONS (SCALAR FUNCTIONS):

String Functions:

1) **Initcap (Initial Capital):** This String function is used to capitalize first character of the input string.

Syntax: initcap(string)

Example:

mysql> select initcap('azure') from dual;

Test Output

2) Lower: This String function will convert input string in to lower case.

Syntax: Lower(string)

Example:

mysql> select lower(‘AZURE’) from

dual; Test Output:

3) Upper: This string function will convert input string in to upper case.

Syntax: Upper(string)

Example:

mysql> select upper(‘azure’) from dual;

Test Output:

4) Ltrim (Left Trim):

Syntax: Ltrim(string,set)

Example:

mysql>select ltrim(‘azuretech’,’azure’) from dual;
Test Output:

5) Rtrim (Right Trim):

Syntax: Rtrim(string,set)

Example:

mysql>select rtrim(‘azuretrim’,’trim’) from dual;

Test Output:

6) Translate:

Syntax: Translate(string1, string2, string3)

Example:

mysql>select translate('abcde','xaybzcxdye','tanzmulrye') from

dual; Test Output:

7) Replace:

Syntax: Replace(string, searchstring, replacestring)

Example:

mysql> select replace('jack and jue','j','bl') from
dual; Test Output:

8) Substr:

Syntax: Substr (string, starts [, count])

Example:

mysql>select substr ('azuretechnology',4,6) from dual;

Test Output:

9) Char:

Syntax: Char(number)

Test Output:

Example:

mysql>select char(65) from dual;

Test Output:

10) Lpad (Left Pad):

Syntax: Lpad(String,length,pattern)

Example:

mysql > select lpad('Welcome',15,'*') from dual;

Test Output

11) Rpad (Right Pad):

Syntax: Lpad(String,length,pattern)

Example:

mysql> select rpad('Welcome',15,'*') from dual;

Test Output:

12) Length:

Syntax:Length(string)

Example:

mysql>select length('azure') from

dual; Test Output:

13) Concatenation (||) Operator:

Syntax: Concat(string1,string2)

mysql> select concat('Azure',' Technology') from

dual; Test Output.

:mysql> select 'ename is '||ename from
emp_master; Test Output:

Record - Notes

WEEK-7

NESTED QUERIES & CORRELATED QUERIES

NESTED QUERIES

Nested Query is a query that has another query embedded within it.

I) IN- is an operator which allows us to check whether a value is present in a given set of elements

Example1: Find the Names of Sailors who have reserved boats no:103

```
Select S.name
        From Sailors S
        Where S.sid IN (select R.sid
                        from reserves R
                        where R.bid= 103);
```

Test Output

Example: Practice with your own Query using NOT IN:

Test Output

Example2: Find the names of Sailors who have reserved a red boat using a nested query.

```
Select S.name
        From Sailors S
        Where S.sid IN (select R.sid
                        from reserves R
                        where R.bid IN( select R.bid
                                        from Boats B
                                        where B.color='red'))
```

Test Output

Example3: Practice with your own Query:

Test Output

CORRELATED QUERIES

I) EXISTS:

The EXISTS operator is another set comparison operator such as IN.It allows us to test whether the set is non empty and will retrieve the Data.

Example1 :Find the names of Sailors who have reserved boat no.103

```
Select S.name
        From Sailors
        Where EXISTS(Select *
                                from Reserves R
                                where R.bid=103 AND R.sid=S.sid);
```

Test Output

Example2: Find the sailors whose rating is better than some Sailor called 'Horatio'

```
Select S.sid
        From Sailors S
        Where S.rating>ANY (select S2.rating
                                        From Sailors.S2
                                        Where S2.name='Horatio');
```

Test Output

Example3: Practice with your own Query:

Test Output

group by and having Clause

SYNTAX:

Select [DISTINCT] select list FROM fromlist WHERE qualification
Group by Groupinglist having group-qualification

Example1: select S.rating MIN(S.age) From Sailors S Group by S.rating;

Test Output

Example 2:Select sum(E.sal) From Employee Group by E.dept;

Test Output

HAVING CLAUSE:

Example1: Find the Average age of all Sailors for each rating level that has atleast two Sailors

SELECT AVG (S.age) FROM Sailors S GROUP By S.rating HAVING

Count(*)>1;

Test Output

Example 2: Find the age of the youngest sailor who is eligible to vote.

Test Output

Order By Clause

Select<column(s)>from<Table Name>where[condition(s)][order by<column name>[asc /] desc];

Example:

mysql> select empno,ename,salary from emp_master order by

salary; Test Output:

WEEK-8

VIEWS

After a table is created and populated with data, it may become necessary to prevent all users from accessing all columns of a table, for data security reasons. This would mean creating several tables having the appropriate number of columns and assigning specific users to each table as required. This will achieve the security requirements but will rise to a great deal of redundant data being resident in tables, in the database. To reduce redundant data to the minimum possible, oracle allows the creation of an object called a view.

A view is a virtual table or logical representation of another table or combination of tables. A view consists of rows and columns just like a table. The difference between a view and a table is that views are definitions built on top of other tables (or views), and do not hold data themselves. If data is changing in the underlying table, the same change is reflected in the view. A view can be built on top of a single table or multiple tables. It can also be built on top of another view. A view derives its data from the tables on which it is based. These tables are called base tables. Base tables might in turn be actual tables or might be views themselves. All operations performed on a view actually affect the base table of the view. We can use views in almost the same way as tables. Also can query, update, insert into and delete from views, just as in standard tables.

Views are essentially saved SELECT queries that can themselves be queried. They are used to provide easier access to normalized data. For example, the Orders table has information about an order. Although it references the employee and customer involved in each order, the Orders doesn't itself contain any valuable information about the employee and customer. We have seen how to use joins to output valuable data from different tables. Creating a view is a way of saving these types of more complicated queries. Views offer the following advantages:

1. Ease of use: A view hides the complexity of the database tables from end users. Essentially we can think of views as a layer of abstraction on top of the database tables.

2. Space savings: Views takes very little space to store, since they do not store actual data.

3. Additional data security: Views can include only certain columns in the table so that only the non-sensitive columns are included and exposed to the end user. In addition, some databases allow views to have different security settings, thus hiding sensitive data from prying eyes.

AIM : Implement Views:

Syntax: Create View <View_Name> As Select statement;

Example:
mysql>Create View EmpView As Select * from Employee;
View created.

Syntax: Select columnname,columnname from <View_Name>;
Example:

mysql>Select Empno,Ename,Salary from EmpView where Deptno in(10,30); Test Output:

UPDATABLE VIEWS:
Syntax for creating an Updatable View:

Create View Emp_vw As
Select Empno,Ename,Deptno from Employee; View

created.

mysql>Insert into Emp_vw values(1126,'Brijesh',20);

mysql>Update Emp_vw set Deptno=30 where

Empno=1125; 1 row updated.

mysql>Delete from Emp_vw where Empno=1122;

Test Output:

mysql>Update EmpDept_Vw set salary=4300 where

Empno=1125; Test Output:

mysql>Delete From EmpDept_Vw where
Empno=1123; Test Output

DESTROYING A VIEW:
Syntax: Drop View <View_Name>;
Example:
mysql>Drop View

Emp_Vw; Test Output:

Record - Notes

WEEK-9

JOINS

MYSQL joins are used to query data from two or more tables, based on a relationship between certain columns in these tables.

ORDER TABLE

OrderID	CustomerID	OrderDate
10308	2	1996-09-18
10309	37	1996-09-19
10310	77	1996-09-20

CUSTOMER TABLE

CustomerID	CustomerName	ContactName	Country
1	Alfreds Futterkiste	Maria Anders	Germany
2	Ana Trujillo Emparedados	Ana Trujillo	Mexico
3	Antonio Moreno Taquería	Antonio Moreno	Mexico

INNER JOIN

The INNER JOIN keyword return rows when there is at least one match in both tables.

1) SELECT Orders.OrderID, Customers.CustomerName, Orders.OrderDate

FROM Orders INNER JOIN Customers ON Orders.CustomerID=Customers.CustomerID;

Test Output

2) SELECT Customers.CustomerName, Orders.OrderID
FROM Customers INNER JOIN Orders ON Customers.CustomerID=Orders.CustomerID

ORDER BY Customers.CustomerName;

TEST OUTPUT

LEFT JOIN

The LEFT JOIN keyword returns all rows from the left
table (table_name1), even if there are no matches in the right table (table_name2).

SELECT Customers.CustomerName, Orders.OrderID FROM Customers

LEFT JOIN Orders ON Customers.CustomerID=Orders.CustomerID

ORDER BY Customers.CustomerName;

TEST OUTPUT

RIGHT JOIN

The RIGHT JOIN keyword Return all rows from the right
table (table_name2), even if there are no matches in the left table (table_name1).

This is another table named employee.here they have to give field accordingly.

SELECT Orders.OrderID, Employees.FirstName

FROM Orders

RIGHT JOIN Employees

ON Orders.EmployeeID=Employees.EmployeeID

ORDER BY Orders.OrderID;

TEST OUTPUT

OUTER JOIN

SELECT Customers.CustomerName, Orders.OrderID
FROM Customers
FULL OUTER JOIN Orders
ON Customers.CustomerID=Orders.CustomerID
ORDER BY Customers.CustomerName;

TEST OUTPUT

UNION

Syntax

Select attribute from table_name union select attribute from table_name;

Example: Select name from sailors union select bname from boats;

TEST OUTPUT

Example 2: SELECT City FROM Customers UNION

SELECT City FROM Suppliers ORDER BY City;

TEST OUTPUT

UNION ALL

Select rsid from reserve union all select sid from sailors;

TEST OUTPUT

Record - Notes

WEEK-10

TRIGGERS

In MySQL, a trigger is a set of SQL statements that is invoked automatically when a change is made to the data on the associated table. A trigger can be defined to be invoked either before or after the data is changed by INSERT, UPDATE or DELETE statement.

- BEFORE INSERT – activated before data is inserted into the table.
- AFTER INSERT – activated after data is inserted into the table.
- BEFORE UPDATE – activated before data in the table is updated.
- AFTER UPDAT – activated after data in the table is updated.
- BEFORE DELETE – activated before data is removed from the table.
- AFTER DELETE – activated after data is removed from the table.

A database trigger is procedural code that is automatically executed in response to certain events on a particular table or view in a database. The trigger is mostly used for maintaining the integrity of the information on the database.

The events that fire a trigger include the following:
1)DML statements that modify data in a table (INSERT , UPDATE , or DELETE)
2)DDL statements.
3) System events such as startup, shutdown, and error messages.
4) User events such as logon and logoff. Note: Oracle Forms can define, store, and run triggers of a different sort.

To View list of triggers;

Show triggers;

To remove a trigger for Database

drop trigger trigger_name;

ex: drop trigger ins_sal;

Types of Triggers:-
1. **Row Triggers** :-A row trigger is fired each time the table is affected by the triggering statement. For example, if an UPDATE statement updates multiple rows of a table, a row trigger is fired once for each row affected by the UPDATE statement. If a triggering statement affects no rows, a row trigger is not executed at all.

Row triggers are useful if the code in the trigger action depends on data provided by the triggering statement or rows that are affected. For example, Figure 15 - 3 illustrates a row trigger that uses the values of each row affected by the triggering statement.

2. Statement Triggers :A statement trigger is fired once on behalf of the triggering statement, regardless of the number of rows in the table that the triggering statement affects (even if no rows are affected). For example, if a DELETE statement deletes several rows from a table, a statement-level DELETE trigger is fired only once, regardless of how many rows are deleted from the table.

Statement triggers are useful if the code in the trigger action does not depend on the data provided by the triggering statement or the rows affected. For example, if a trigger makes a complex security check on the current time or user, or if a trigger generates a single audit record based on the type of triggering statement, a statement trigger is used.

When defining a trigger, specify the trigger timing. That is, specify whether the trigger action is to be executed before or after the triggering statement. BEFORE and AFTER apply to both statement and row triggers

Sample:

```
CREATE TRIGGER trigger_name      trigger_time      trigger_event
 ON table_name
 FOR EACH ROW
 BEGIN
 ...
 END;

trigger_time=before/after
 trigger_event=insert/delete/update
```

Example:

```
CREATE TRIGGER sal_sum after insert ON emp
    FOR EACH ROW SET @sal = @sal + NEW.sal;
```

Firing a trigger:

Question: Find the sum of salaries of all employees

1)First create a table **emp** with following columns

Field	**Type**
empid	int(11)
ename	varchar(50)
sal	int(11)

Write a Query:

TEST OUTPUT

2) create variable/parameter **sal** as below at mysql prompt

mysql> **set @sal=0;**

3) now create trigger on emp

```
CREATE TRIGGER sal_sum after insert ON emp
    FOR EACH ROW SET @sal = @sal + NEW.sal;
```

Test output:

4) insert the values into table emp;

```
mysql> insert into emp values(1001,'suhaas',10000);
mysql> insert into emp values(1002,'Dilraj',15000);
mysql> insert into emp values(1003,'Riyanshi',25000);
```

Note:trigger is fired on after insert

5)check values in the table emp;

mysql> select * from emp;

Test output:

6) checking value in the parameter sal

mysql> select @sal as TotalSalary;

TEST OUTPUT

Note:when ever there is insert operation that value in the **sal** variable increases

Record - Notes

WEEK-11

PROCEDURES

TCL (TRANSACTION CONTROL): Transaction control statements are used to manage the changes made by DML statements. It allows statements to be grouped together into logical transactions. TCL Commands are Commit, Rollback and Save point.

COMMIT - Saves the work done

SAVEPOINT - Identify a point in a transaction to which you can later roll back

ROLLBACK - Restore database to original since the last COMMIT

Procedure

By default, mysql recognizes the semicolon as a statement delimiter, so you must redefine the delimiter temporarily to cause mysql to pass the entire stored program definition to the server. To redefine the mysql delimiter, use the delimiter command.

syntax

```
delimiter //
create procedure procedure name(in parameter)
begin
select * from tablename;
end //
```

Example:

create table named emp with two fields name and salary;

Write a Query

TEST OUTPUT

delimiter //

```
 create procedure emp(in name_p varchar(20))
begin
select * from emp where name=name_p;
end //
call sailor("smith");
```

with in out parameter:

```
delimiter//
create procedure counterset(inout count in(4),in inc int(4))
begin
set count=count+inc;
end //

set @counter=1;

call counterset(@counter,4);

select @counter;
```

TEST OUTPUT

Transaction(commit & roll back)

```
create table account(name varchar(20),sal int);

insert into account values('mani',2000);
insert into account values('raju',3000);
select * from account;
```

TEST OUTPUT

```
set autocommit=0;
//
start transaction;
//
update account set sal=30000 where name='mani';
//
select * from account;
```

TEST OUTPUT

```
rollback;
select * from account;
//
update account set sal=5000 where name='raju';
//
select * from account;
```

commit;
changes are now made permanent
rollback;
select * from account;
/same output will get/

TEST OUTPUT

Note: when autocommit=1 the the database commits every single update and rollback is not possible.

Record - Notes

WEEK-12

PL/SQL

PL/SQL programs are written as lines of text using a specific set of characters:

- Upper- and lower-case letters A .. Z and a .. z
- Numerals 0 .. 9
- Symbols ( ) + - * / < > = ! ~ ^ ; : . ' @ % , " # $ & _ | { } ? [ ]
- Tabs, spaces, and carriage returns

PL/SQL keywords are not case-sensitive, so lower-case letters are equivalent to corresponding upper-case letters except within string and character literals.

A line of PL/SQL text contains groups of characters known as lexical units:

- Delimiters (simple and compound symbols)
- Identifiers, which include reserved words
- Literals
- Comments

To improve readability, you can separate lexical units by spaces. In fact, you must separate adjacent identifiers by a space or punctuation. The following line is not allowed because the reserved words END andIF are joined:

IF x > y THEN high := x; ENDIF; -- not allowed, must be END IF

You cannot embed spaces inside lexical units except for string literals and comments. For example, the following line is not allowed because the compound symbol for assignment (:=) is split:

count : = count + 1; -- not allowed, must be :=

To show structure, you can split lines using carriage returns, and indent lines using spaces or tabs. This formatting makes the first IF statement more readable.

IF x>y THEN max:=x;ELSE max:=y;END IF;

1) WRITE A PROGRAM TO PRINT HELLO WORLD

```
BEGIN
DBMS_OUTPUT.PUT_LINE ('HELLO WORLD');
END;
```

TEST OUTPUT

2) WRITE A PROGRAM TO PRINT EVEN NUMBERS FROM 1 TO 100

```
DECLARE
N NUMBER(3) :=0;
BEGIN
WHILE N<=100
LOOP
N :=N+2;
DBMS_OUTPUT.PUT_LINE(N);
END LOOP;
END;
```

TEST OUTPUT

3) WRITE A PROGRAM TO ACCEPT A NUMBER AND FIND SUM OF THE DIGITS

```
DECLARE
N NUMBER(5):=&N;
S NUMBER:=0;
R NUMBER(2):=0;
BEGIN
WHILE N !=0
LOOP
R:=MOD(N,10);
S:=S+R;
```



```
N:=TRUNC(N/10);
END LOOP;
DBMS_OUTPUT.PUT_LINE('SUM OF DIGITS OF GIVEN NUMBER IS '||S);
END;
```

TEST OUTPUT

4) Write a program to accept a number and print it in reverse order

```
DECLARE
N NUMBER(5):=&N;
REV NUMBER(5):=0;
R NUMBER(5):=0;
BEGIN
WHILE N !=0
LOOP
R:=MOD(N,10);
REV:=REV*10+R;
N:=TRUNC(N/10);
END LOOP;
DBMS_OUTPUT.PUT_LINE('THE REVERSE OF A GIVEN NUMBER IS '||REV);
END;
```

TEST OUTPUT

5) Write a program accept the value of A,B&C display which is greater

```
DECLARE
A NUMBER(4,2):=&A;
B NUMBER(4,2):=&B;
C NUMBER(4,2):=&C;
BEGIN
IF (A>B AND A>C) THEN
DBMS_OUTPUT.PUT_LINE('A IS GREATER '||"||A);
ELSIF B>C THEN
DBMS_OUTPUT.PUT_LINE('B IS GREATE '||"||B);
ELSE
DBMS_OUTPUT.PUT_LINE('C IS GREATER '||"||C);
END IF;
END;
```

Record - Notes

WEEK-13

DCL (DATA CONTROL LANGUAGE): Data Control Language statements are used to create roles, permissions, and referential integrity as well it is used to control access to database by securing it. DCL Commands are Grant and Revoke

GRANT - gives user's access privileges to database

REVOKE - withdraw access privileges given with the GRANT command

Checking of User Privileges, Grants etc

mysql> create user mru_user;
Query OK, 0 rows affected (0.30 sec)
*To Check where is the created user i.e location in our database
mysql> select user();
TEST OUTPUT

***To check what are the grants that the location is having/**
mysql> show grants**;**

TEST OUTPUT

***To Check what are the GRANTS having for created user**
mysql> show grants for mru_user;

TEST OUTPUT

mysql> show tables;

TEST OUTPUT

***To Flush (RE-FRESH) the privileges**
mysql> flush privileges;
Query OK, 0 rows affected (0.08 sec)

*** Explanation:** *To check where is the user i.e in case if we created user (Ex: mrcet_cse) it will be displayed as "%". Root user is by default so it will be available in "Localhost"*

mysql> select host, user from mysql.user;
TEST OUTPUT