J.B. INSTITUTE OF ENGINEERING AND TECHNOLOGY

(UGC AUTONOMOUS)
Bhaskar Nagar, Moinabad Mandal, R.R. District, Hyderabad -500075

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

OBSERVATION

DATABASE MANAGEMENT SYSTEMS LAB

(R20)

Name of the Student:

Roll No:

Class & Section:

Index

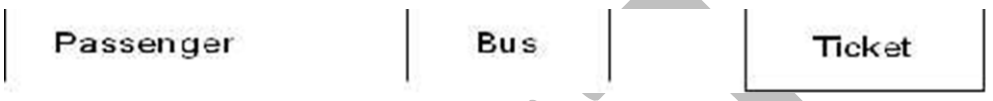
Experiment- 1
E-R Model

Analyze the problem carefully and come up with entities in it. Identify what date has to be persisted in the database. This contains the entities, attributes etc.
Identify the primary keys for all the entities. Identify the other keys like candidate keys, partial keys, if any.

Definitions:

Entity: the object in the **ER** Model represents is an entity which is thing in the real world with an Independent existence.

Eg:

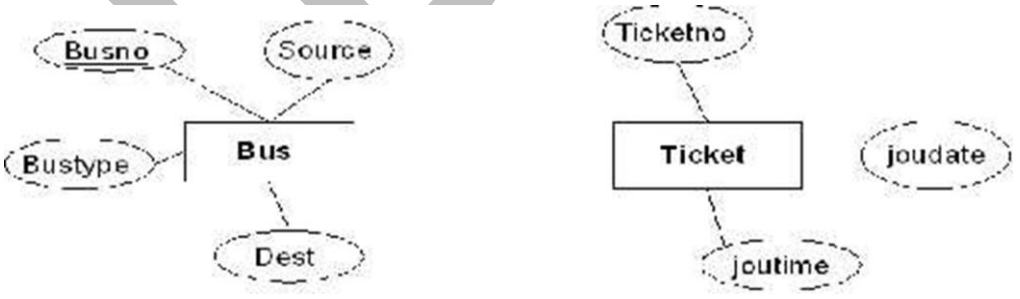


ER-Model:

Describes data as entities, relationships and attributes .The ER-Model is important preliminary for its role in database design. ER Model is usually shown pictorially using entity relationship diagrams.

Attributes:

The properties that characterize an entity set are called its attributes. An attribute is referred to by the terms data items, data element, data field item.
Ex: attributes for bus entity and ticket entity.



Candidate key:

It can be defined as minimal super key or irreducible super key. In other words an attribute or combination of attributes that identifies the record uniquely but none of its proper subsets can identify the record uniquely.

Busno	serviceno	source	destination	deptime	retime	bustype	noofseats

Busno,serviceno---------->candidate key

Primary key:
A candidate key that is used by the database designer for unique identification of each row in a table is known as primary key. A primary key consists of one or more attributes of the table.

Partial key:
A weak entity type normally has a partial key which is the set of attributes that can uniquely identify weak entity that are related to the same owner entity.

The entities in the "Roadway travels" is

1) Bus 2) Ticket 3) Passenger

Bus entity:
Attributes for the bus entity are

Busno, serviceno, source, destination, deptime, retime, bustype, noofseats

Bus schema:

Busno	**serviceno**	source	destination	deptime	retime	bustype	noofseats

Busno,serviceno,source -----> super key
Busno,serviceno,bustype-----> super key
Busno,serviceno---------->candidate key Busno,serviceno----------> primary key

Ticket entity:
Attributes for the ticket entity are

Ticketno, joudate, joutime, source, destination, seatno, amount, catcard

Ticket schema:

Ticketno	Joudate	Joutime	Source	Destination	Seatno	Amount	Catcard

Ticketno, source, destination ------- >Super key
Ticketno, source, seatno ----------> Super key Ticketno, destination, seatno --------->
Super key Ticketno----------> candidate key
Ticketno---------- >primary key

Passenger entity:
Attributes for the Passenger entity are

Pnrno, pname, age, sex, ticketno, address, phno, catno

Passenger schema:

Pnrno	pname	age	sex	ticketno	address	phno	catno

Pnrno,pname---------- super key
Pnrno,ticketno---------- super key
Pnrno,phno---------- super key
Pnrno ---------- candidate key
Pnrno ----------primary key

Sample data for *Bus* entity:

Busno	serviceno	source	destination	deptime	retime	bustype	Noofseats
Ap555	3889	Srpt	Hyd	9:00:00	19:15:00	Ac	36
Ap501	3891	Srpt	Hyd	10:00:15	20:15:00	Ac	36
Ap444	3601	Hyd	Srpt	9:00:00	19:30:00	Nonac	52
Ap891	3555	Hyd	Srpt	9:30:00	20:30:00	Nonac	52
Ap8830	3239	Hyd	Vij	9:00:00	22:30:00	Metro	45

Sample data for *Ticket* entity:

Ticketno	Joudate	Joutime	Source	Destination	Seatno	Amount	Catcard
1111	2010-8-5	9:00:00	Srpt	Hyd	5	96	No
2222	2010-8-5	10:00:15	Srpt	Hyd	10	88	Yes
3333	2010-8-15	9:00:00	Hyd	Srpt	15	88	Yes
4444	2010-8-18	9:30:00	Hyd	Srpt	20	96	No
5555	2010-8-6	9:00:00	Hyd	Vij	18	172	Yes

Sample data for *Passenger* entity:

Pnrno	pname	age	sex	ticketno	address	phno	Catno
1001	Subbu	31	M	1111	5-4,srpt	9492506282	Cap5112
1002	Achaith	22	M	2222	6-8,hyd	9949060540	
1003	Padma	25	F	3333	h/7,vij	9704054050	Cap5772
1004	Ravi	23	M	4444	8-9,hyd	9704613151	Cap6132
1005	Satyam	42	F	5555	9-11,hyd	9848354941	Cap6732

Experiment- 2

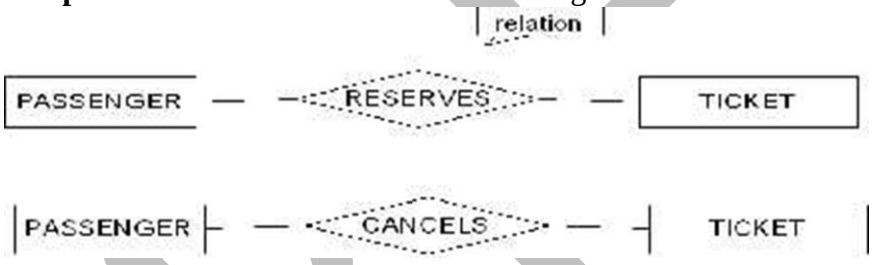
Concept design with E–R model

Relate the entities appropriatelyy. Apply cardinalities for each relationship. Identify strong
Entities and weak entities (if any). Indicate the type of relationship (total/partial). Try to incorporate generalization, aggregation, specialization etc wherever required.

Definitions:

The cardinality ratio: - for a binary relationship specifies the maximum number of relationships that an entity can participate in.

Relationship: - it is defined as an association among two or more entities.



Weak and strong entity: - an entity set may not have sufficient attributes to form a primary key. Such an entity set is termed a weak entity set. An entity set that has primary key is termed a strong entity set.

Total participation:-
Ex: - if a travel agency states that every passenger must make reservation then every passenger Travels in bus. Than a passenger's entity can exist only if it participates in at least one travels relationship instances. Thus the participation of passenger in travel is called total participation meaning that every entity in the "total set" passenger entities must be related to bus via travels relationship.

All passengers travel in one bus so it is total participation

Partial participation: a participation that is not total is called as partial participation.

Some passengers cancel ticket so it is partial participation

Generalization: consists of identifying some common characteristics of a collection of entity set and creating new entity set that contains entities possessing these common characteristics.

Aggregation: allows us to indicate that a relationship set participates in another relationship set.

Specialization: in the process of identifying subsets of an entity set (the super set) that share some distinguishing characteristics. This entity type is called the super class of the specialization.

Relationship between different entities:

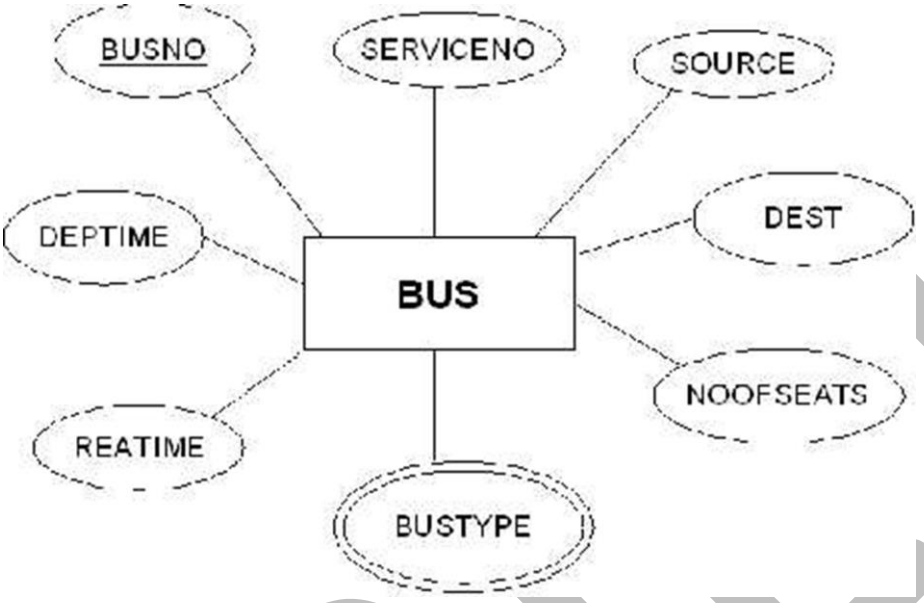
Relationship between Bus and Ticket entities
1:M binary relationship

Relationship between Passenger and Bus entities
M:1 binary relationship

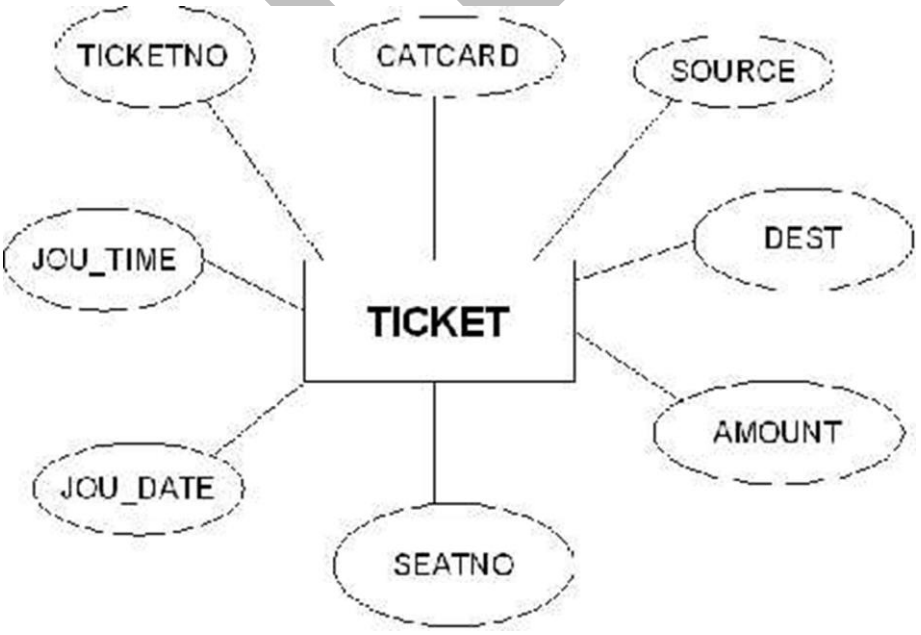
Relationship between Passenger and Ticket entities

M:N binary relationship

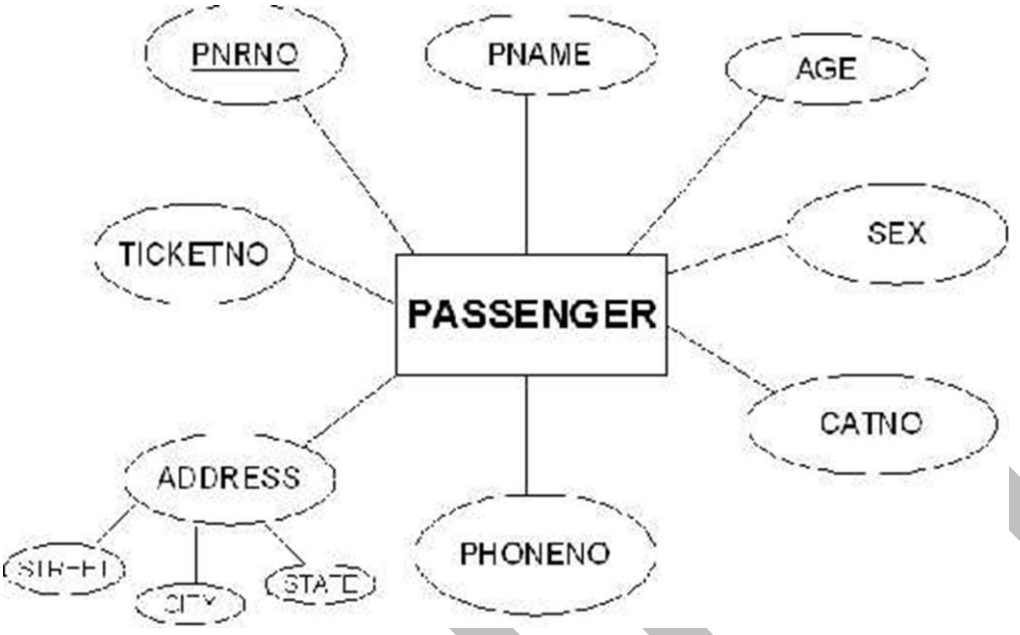
Entity diagram for *BUS*



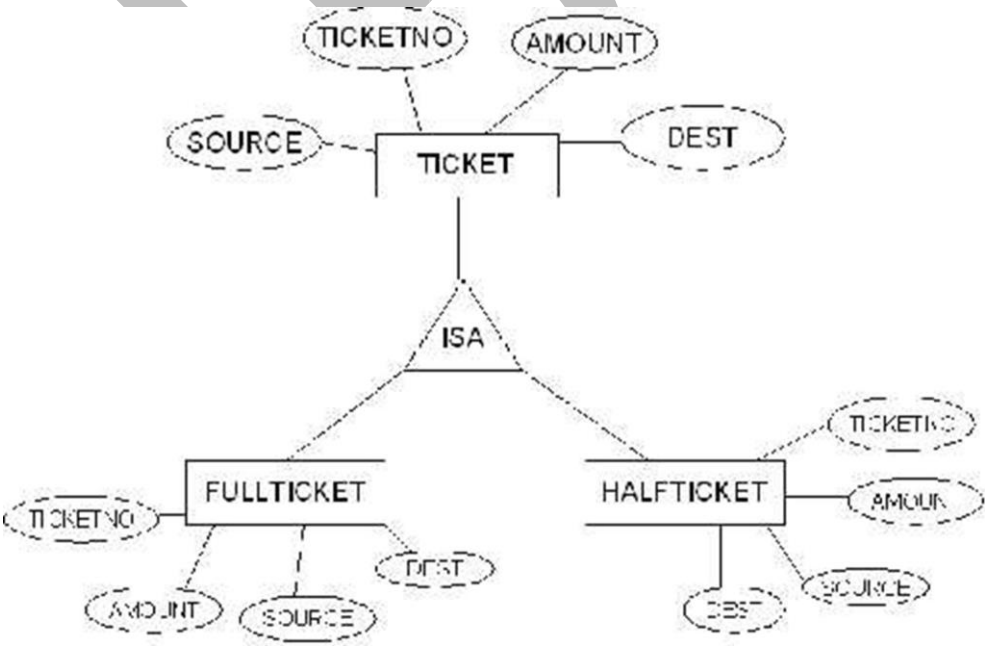
Entity diagram for *Ticket*



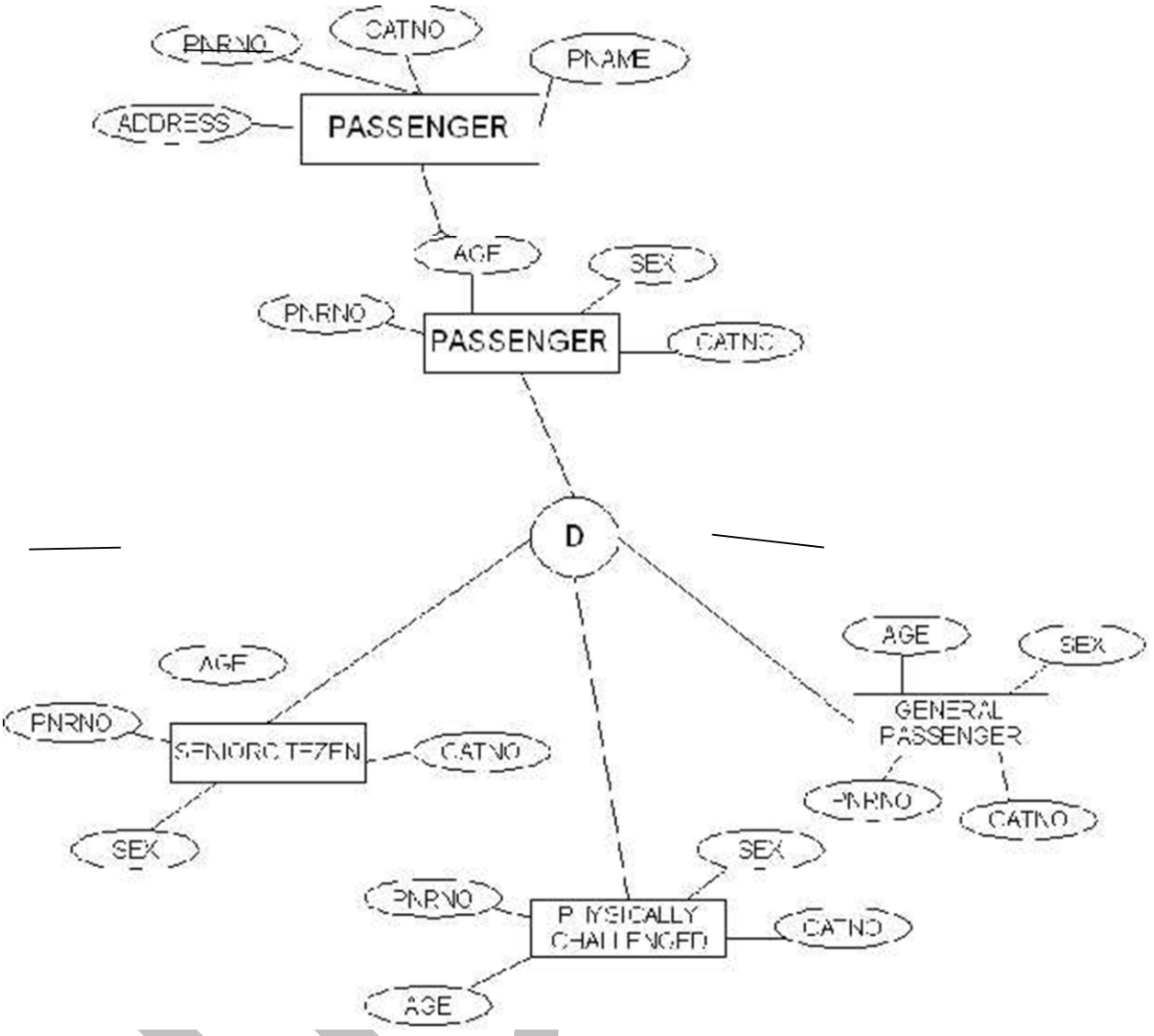
Entity diagram for *Passenger*



Generalization:

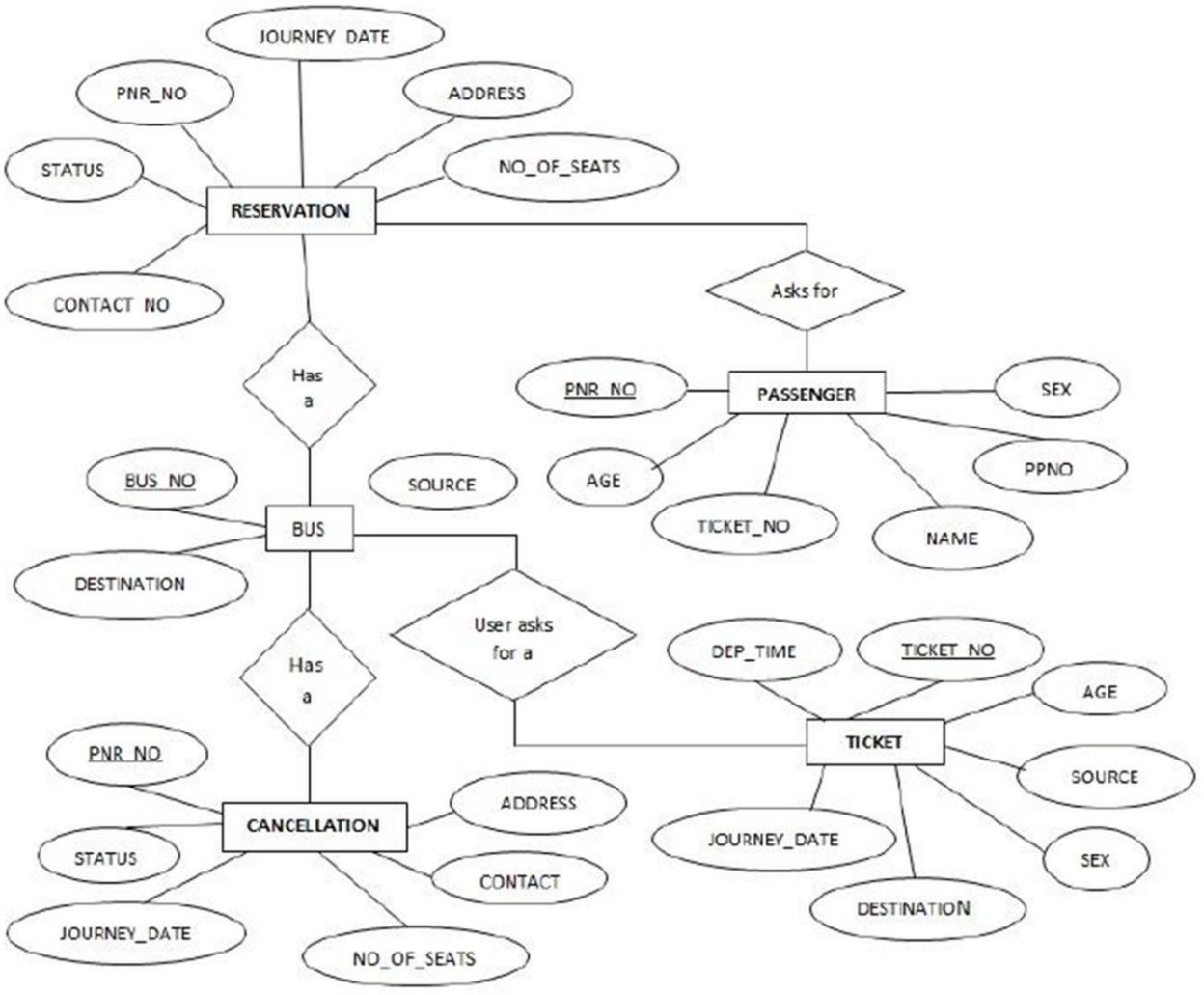


Aggregation:



Specialization:

E-R


JOURNEY DATE
PNR_NO
ADDRESS
STATUS
NO_OF_SEATS
RESERVATION
CONTACT NO
Asks for
Has a
PNR_NO
PASSENGER
SEX
AGE
PPNO
TICKET_NO
NAME
BUS_NO
SOURCE
BUS
DESTINATION
User asks for a
Has a
DEP_TIME
TICKET_NO
AGE
TICKET
PNR_NO
SOURCE
ADDRESS
CANCELLATION
JOURNEY_DATE
SEX
STATUS
CONTACT
DESTINATION
JOURNEY_DATE
NO_OF_SEATS

Experiment-3

Relational Model

Represent all entities (strong, week) in tabular fashion. Represent relationships in a tabular Fashion. There are different ways of representing as tables based on the cardinality. Represent attributes as columns in the tables or as tables based on the requirement. Different types of attributes (composite, multivalued and derived).

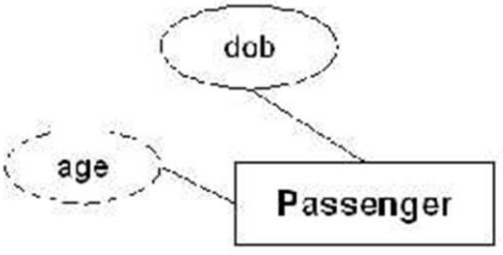
Definitions:

Composite attributes: can be divided into smaller sub parts which represent more basic Attributes with independent meaning.

Multivalued attributes: for ex the attribute in the Bus entity Bus type can have different types of buses according that the Bus type attribute contains the values as Garuda, Luxury, Express, and Ordinary. This type of attribute is called multivalued attribute and may have lower and upper bounds to constrain the number of values allowed for each individual entity.

Derived attributes:

In some cases, two or more attribute values are related. With the help of one attribute we get the value of another attribute. Age and DOB attributes. With the DOB we get the age of the person to the current date.



Entity sets to tables:

Relational schema for Bus relation:

Bus**(**Busno:numeric, serviceno:numeric, source:varchar(10), destination:varchar(10), deptime:time, retime:time, bustype:varchar(10), noofseats:int)

Relational shema for Ticket relation:

Ticket**(**Ticketno:numeric, joudate:date, joutime:time, source:varchar(10), destination:varchar(10), seatno:int(4), amount:decimal(10,3), catcard:char(3))

Relational shema for Passenger relation:

Passenger**(**Pnrno:numeric, pname:varchar(15), age:int(4), sex:char(3), ticketno:numeric, address:varchar(50), phno:numeric(10), catno:varchar(10))

Relationship sets to tables:

Relational shema for reserve relation:

Rserves(pnrno:numeric,joudate:date,noofseats:int(4),address:varchar(50),contact_no:numeric(10),status:char(3))

Relational shema for Cancels relation:

Cancels(pnrno:numeric,joudate:date,noofseats:int(4),address:varchar(50),contact_no:numeric(10),status:char(3))

TABLES:

Reserves:

Pnrno	Joudate	Noofseats	Address	Contact_no	Status
1001	2010-8-5	10	5-4,srpt	9492506282	Yes
1001	2010-8-15	5	5-4,srpt	9492506282	Yes
1002	2010-8-5	5	6-8,hyd	9949060540	Yes
1003	2010-8-15	6	h/7,vij	9704054050	Yes
1004	2010-8-18	8	8-9,hyd	9704613151	Yes
1005	2010-8-5	9	9-11,hyd	9848354941	No

Cancels:

Pnrno	Joudate	Noofseats	Address	Contact_no	Status
1001	2010-8-5	5	5-4,srpt	9492506282	Yes
1002	2010-8-5	2	6-8,hyd	9949060540	No
1003	2010-8-15	2	h/7,vij	9704054050	Yes
1004	2010-8-18	5	8-9,hyd	9704613151	Yes
1005	2010-8-6	4	9-11,hyd	9848354941	Yes

Bus:

Busno	serviceno	source	destination	deptime	retime	bustype	Noofseats
Ap555	3889	Srpt	Hyd	9:00:00	19:15:00	Ac	36
Ap501	3891	Srpt	Hyd	10:00:15	20:15:00	Ac	36
Ap444	3601	Hyd	Srpt	9:00:00	19:30:00	Nonac	52
Ap891	3555	Hyd	Srpt	9:30:00	20:30:00	Nonac	52
Ap8830	3239	Hyd	Vij	9:00:00	22:30:00	Metro	45

Ticket:

Ticketno	Joudate	Joutime	Source	Destination	Seatno	Amount	Catcard
1111	8/15/2010	9:00:00	Srpt	Hyd	5	96	no
2222	8/5/2010	10:00:15	Srpt	Hyd	10	88	yes
3333	8/15/2010	9:00:00	Hyd	Srpt	15	88	yes
4444	8/18/2010	9:30:00	Hyd	Srpt	20	96	no
5555	8/6/2010	9:00:00	Hyd	Vij	18	172	yes

Passenger:

Pnrno	pname	age	sex	ticketno	address	phno	Catno
1001	Subbu	31	M	1111	5-4,srpt	9492506282	Cap5112
1002	Achaith	22	M	2222	6-8,hyd	9949060540	Cap6900
1003	Padma	25	F	3333	h/7,vij	9704054050	Cap5772
1004	Ravi	23	M	4444	8-9,hyd	9704613151	Cap6132
1005	Satyam	42	F	5555	9-11,hyd	9848354941	Cap6732

Experiment-4

Normalization

Database normalization is a technique for designing relational database tables to minimize duplication of information and, in doing so , to safeguard the database against certain types of logical or structural problems namely data anomalies.

The normalization forms are:

1. **First Normal Form**: 1NF requires that the values in each column of a table are atomic. By atomic we mean that there are no sets of values within a column.
2. **Second Normal Form**: where the 1NF deals with atomicity of data, the 2NF deals with relationships between composite key columns and non-key columns. To achieve 2NF the tables should be in 1NF. The 2NF any non-key columns must depend on the entire primary key. In case of a composite primary key, this means that non-key column can't depend on only part of the composite key.
3. **Third Normal Form**: 3NF requires that all columns depend directly on the primary key. Tables violate the third normal form when one column depends an another column, which in turn depends on the primary key(transitive dependency). One way to identify transitive dependency is to look at your tables and see if any columns would require updating if another column in the table was updated. If such a column exists, it probably violates 3NF.

Let's normalize our entities:

Normalization of <u>Passenger</u> entity:

In the passenger entity there exists a passenger with two phone numbers, but atomic values should be there. So we normalize the relation as follows.

Passenger:

Pnrno	pname	age	sex	ticketno	address	phno	Catno
1001	subbu	21	M	1111	5-4,srp	9492506282, 9876655343	Cap5112
1002	Achaith	22	M	2222	6-8,hyd	9949060540	Cap6900
1003	Padma	25	F	3333	h/7,vij	9704054050	Cap5772
1004	Ravi	23	M	4444	8-9,hyd	9704613151	Cap6132
1005	Satyam	42	F	5555	9-11,hyd	9848354941	Cap6732

Pnrno	pname	age	sex	ticketno	address	phno	Catno
1001	subbu	21	M	1111	5-4,srp	9492506282,	Cap5112
1002	Achaith	22	M	2222	6-8,hyd	9949060540	Cap6900
1003	Padma	25	F	3333	h/7,vij	9704054050	Cap5772
1004	Ravi	23	M	4444	8-9,hyd	9704613151	Cap6132
1005	Satyam	42	F	5555	9-11,hyd	9848354941	Cap6732

The above relation is now in 1NF and the relation is 2NF as there are no partial functional dependencies and the relation is also in 3NF as there are no transitive dependencies.

Normalization of Bus entity:

Bus:

Busno	serviceno	source	destination	deptime	retime	bustype	Noofseats
Ap555	3889	Srpt	Hyd	9:00:00	19:15:00	Ac	36
Ap501	3891	Srpt	Hyd	10:00:15	20:15:00	Ac	36
Ap444	3601	Hyd	Srpt	9:00:00	19:30:00	Nonac	52
Ap891	3555	Hyd	Srpt	9:30:00	20:30:00	Nonac	52
Ap8830	3239	Hyd	Vij	9:00:00	22:30:00	Metro	45

In this relation the values in each column are atomic so it is already in 1NF.
In the Bus entity **Busno+serviceno** is the primary key.
There exists following partial dependencies.

Busno ----> Bustype,Noofseats
Serviceno---->Source,Dest

So the relation will be in 2NF as follows.

Busno	serviceno	deptime	retime
Ap555	3889	9:00:00	19:15:00
Ap501	3891	10:00:15	20:15:00
Ap444	3601	9:00:00	19:30:00
Ap891	3555	9:30:00	20:30:00
Ap8830	3239	9:00:00	22:30:00

Busno	bustype	Noofseats
Ap555	Ac	36
Ap501	Ac	36
Ap444	Nonac	52
Ap891	Nonac	52
Ap8830	Metro	45

serviceno	source	destination
3889	Srpt	Hyd
3891	Srpt	Hyd
3601	Hyd	Srpt
3555	Hyd	Srpt
3239	Hyd	Vij

The above relation is 2NF. And all columns directly depend on primary key. So there is no transitive dependency and the relation is 3NF.

Normalization of Ticket entity:

Ticketno	Joudate	Joutime	Source	Destination	Seatno	Amount	Catcard
1111	2010-8-5	9:00:00	Srpt	Hyd	5	96	No
2222	2010-8-5	10:00:15	Srpt	Hyd	10	88	Yes
3333	2010-8-15	9:00:00	Hyd	Srpt	15	88	Yes
4444	2010-8-18	9:30:00	Hyd	Srpt	20	96	No
5555	2010-8-6	9:00:00	Hyd	Vij	18	172	Yes

In this relation the values in each column are atomic so it is already in 1NF.
In the above relation there are no partial functional dependencies so the relation is in 2NF.
The ticket entity might face the following transitive dependency

Ticketno-------> catcard
Catcard--------->amount

So the relation is in 3NF.

Ticketno	Joudate	Joutime	Source	Destination	Seatno	Catcard
1111	2010-8-5	9:00:00	Srpt	Hyd	5	No
2222	2010-8-5	10:00:15	Srpt	Hyd	10	Yes
3333	2010-8-15	9:00:00	Hyd	Srpt	15	Yes
4444	2010-8-18	9:30:00	Hyd	Srpt	20	No
5555	2010-8-6	9:00:00	Hyd	Vij	18	Yes

Put the catcard and amount attributes in a separate table. Then the relation should be in 3NF.

Catcard	Amount
No	96
Yes	88
Yes	88
No	96
Yes	172

The above relation is 3NF as we have eliminated the transitive dependencies.

Finally all the tables are normalized and free from data redundancy, partial functional dependencies and transitive dependencies.

Experimen-5

Installation of Mysql and Practicing DDL commands and DML Commands

How to install MySQL server 5.1 on Windows

This article provides step-by-step installation guide for MySQL 5.1 (or higher) on Windows as a development machine. Other releases of MySQL should have similar installation process.

Step #1: Download MySQL 5.1 for Windows

You can download version 5.1 from this page Download MySQL 5.1

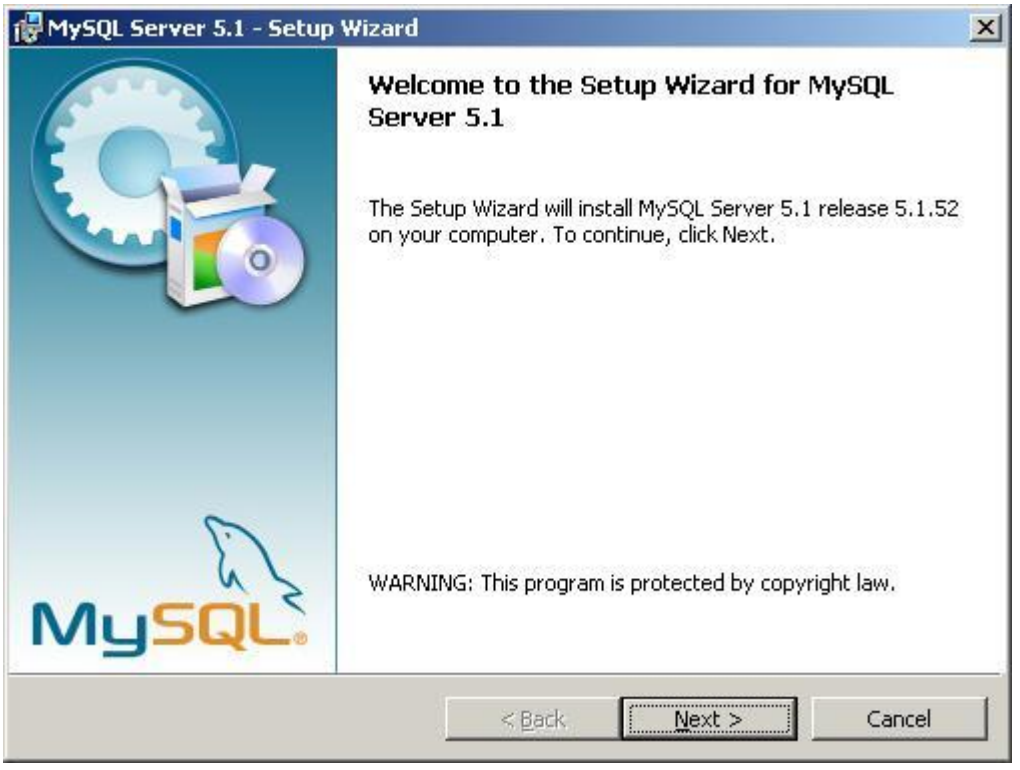
Or download the latest MySQL version here.

For archives of various MySQL versions, click here.

Step #2: Install MySQL

Double click the MSI installer to start installing MySQL. You will go through a setup wizard so it's fairly simple. Just follow the installation instructions step by step.

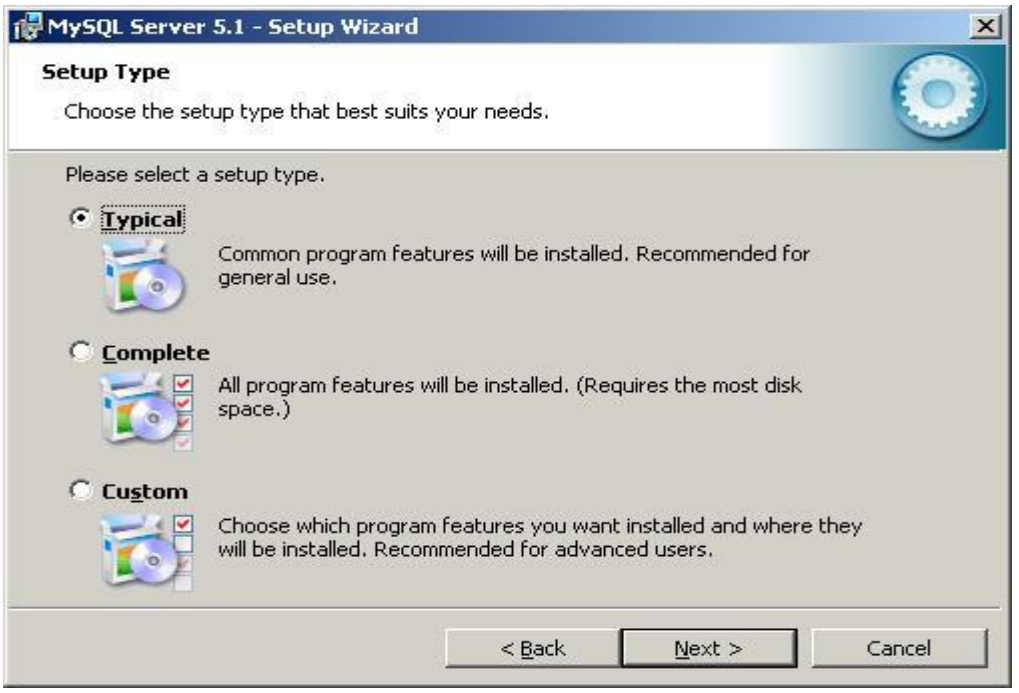
(1) Welcome to the Setup Wizard for MySQL Server 5.1. Click Next to continue.



(2) Select a setup type - Typical, Complete, Custom.

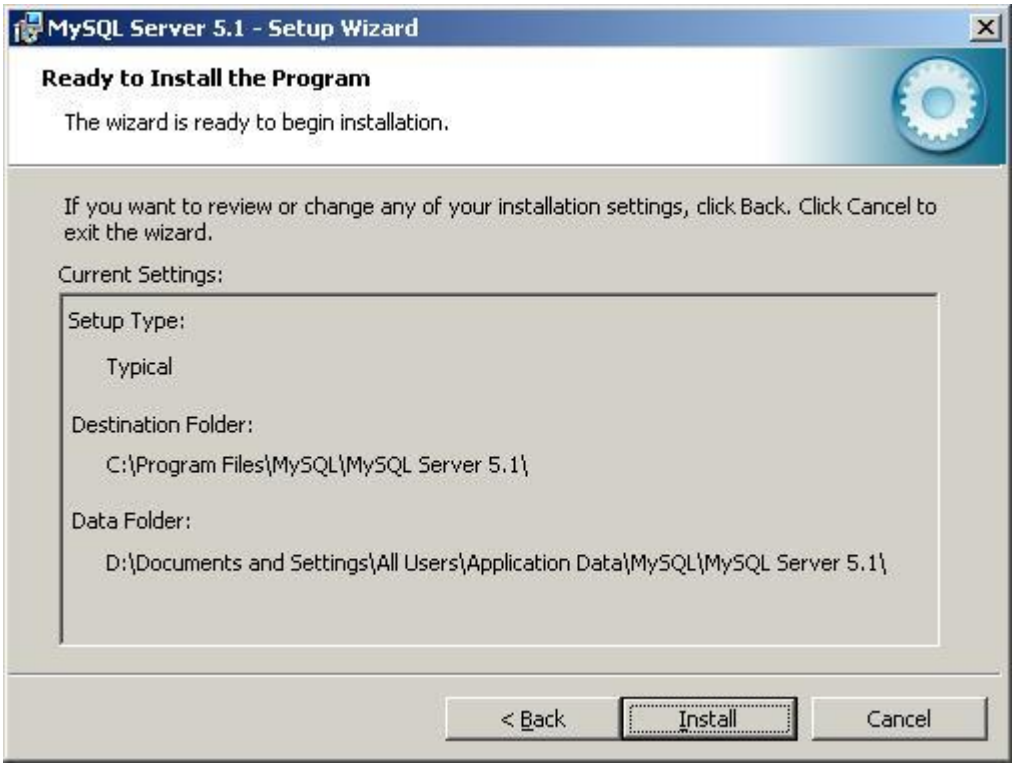
Select Typical and click Next.

The default insllation directory will be C:\Programs Files\MySQL\MySQL Server 5.0\

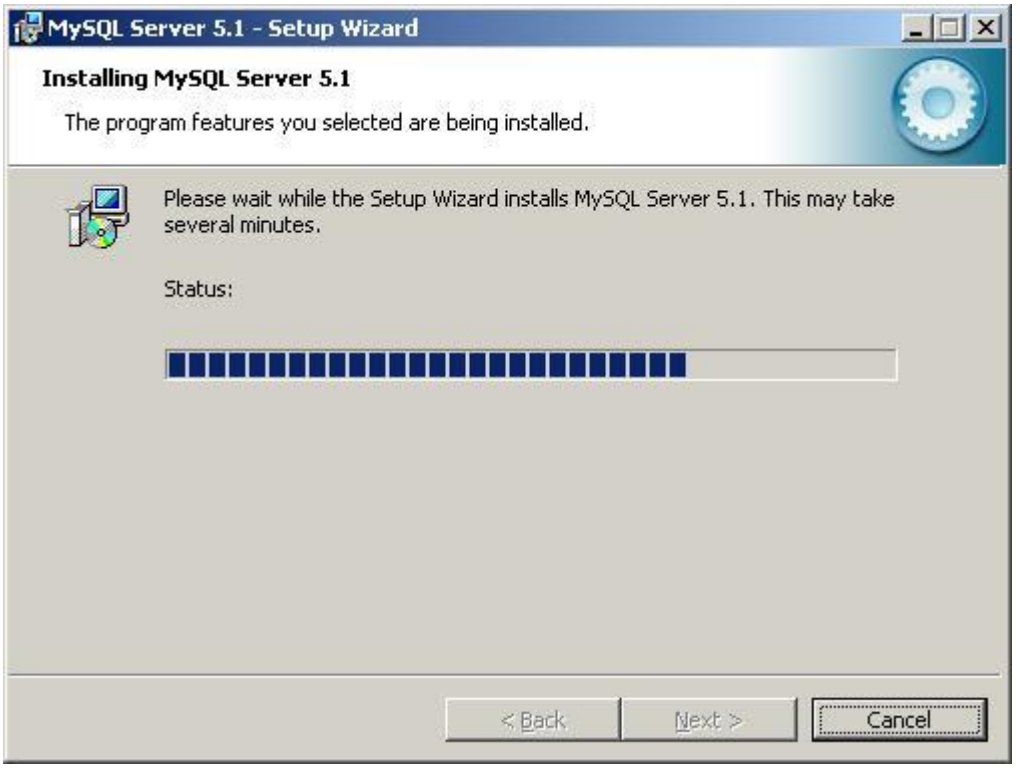


(3) Ready to install MySQL.

After review the settings, click Install. If you want to change any settings, click [Back] button.

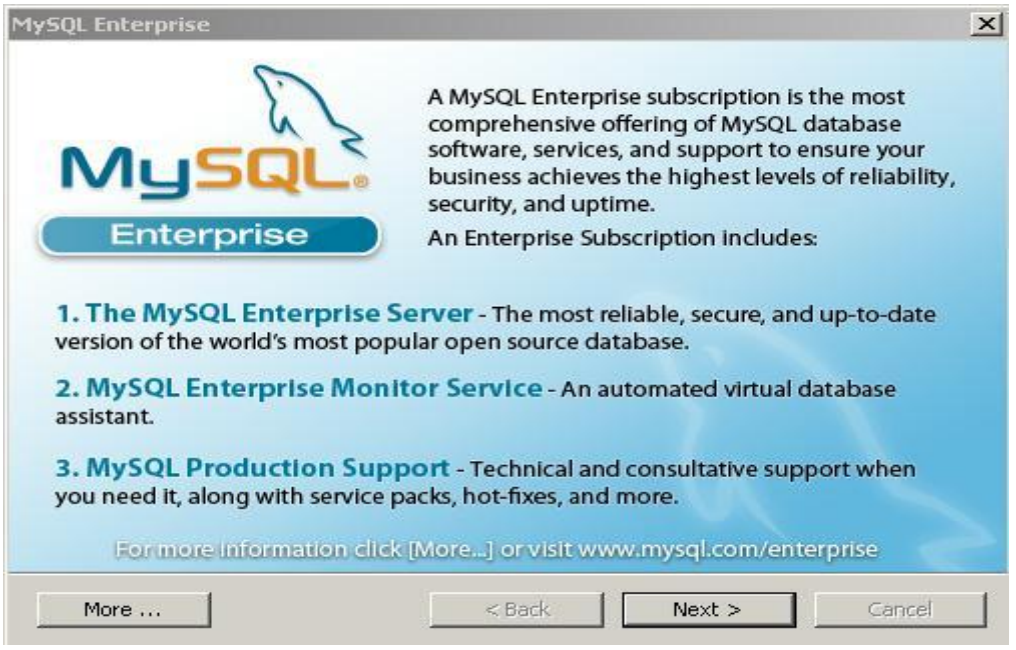


(4) Installation in progress.



(5) More information about MySQL Enterprise Subscription.

This step is just an informational step for MySQL Enterprise Subscription. Click the [More...] button if you want to know more. Otherwise, click [Next] to go to next step.



(6) Setup Wizard Completed.

Make sure you have selected 'Configure the MySQL Server now' checkbox if you want to configure it after clicking Finish button.

What's next

MySQL 5.1 provides an easy-to-use wizard to configure MySQL server instance. Next, we are going to go through detailed steps about MySQL server configuration. See How to configure MySQL server 5.1 on Windows.

How to configure MySQL server 5.1 on Windows

After installing MySQL 5.1 on Windows, we will need to configure it. This article provides step-by-step MySQL server 5.1 configuration guide on Windows as a development box.

If you want to install multiple versions of MySQL on the same box, read this article How to install two different versions of MySQL on the same computer for more info.

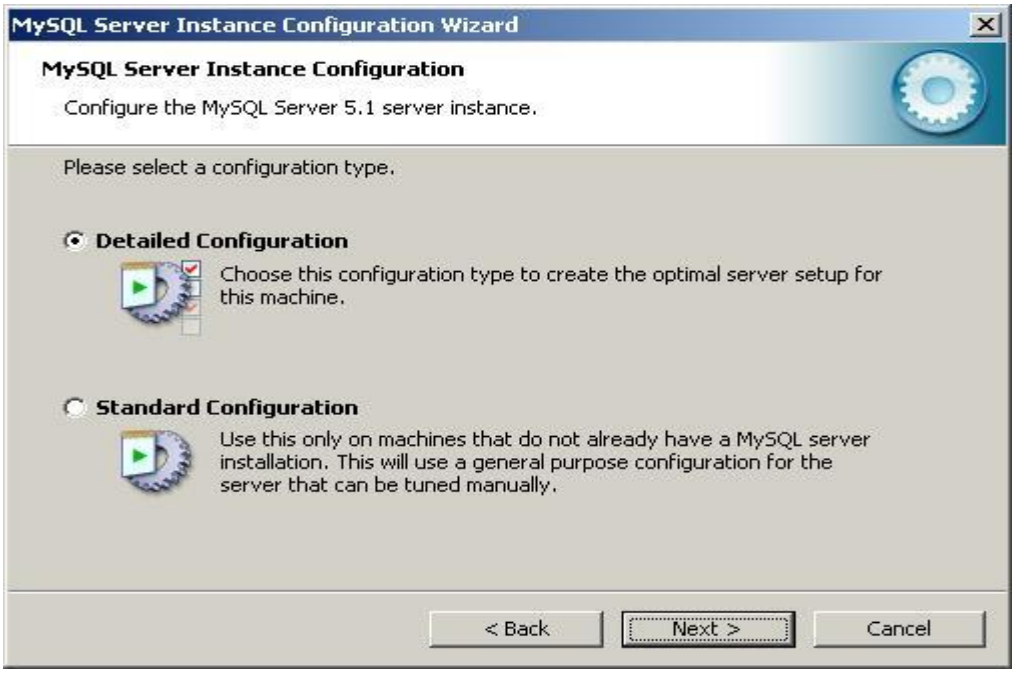
1. Start MySQL Server Instance Configuration Wizard.

If you have selected the checkbox 'Configure the MySQL Server now' in the last step when installing MySQL 5.1, the configuration wizard should start automatically when you click the Finish button.

Alternatively, you can launch the wizard from Start menu. Start -> Programs -> MySQL -> MySQL Server 5.1 -> MySQL Server Instance Config Wizard.

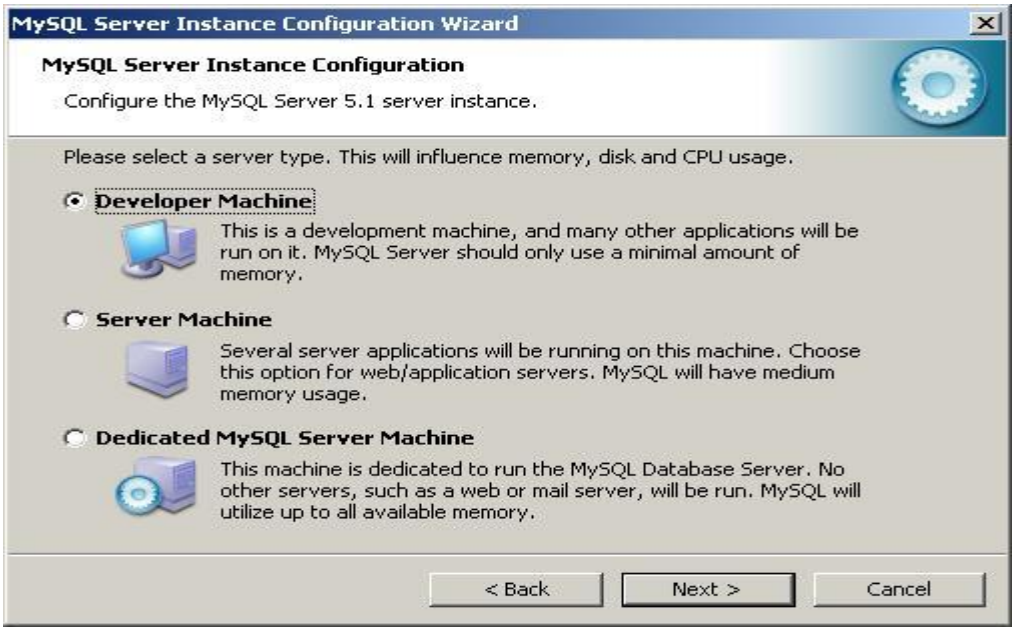
2. Select configuration type.

Select Detailed Configuration here. If this is the only MySQL server installed on your computer, you can select Standard Configuration.



3. Select a server type.

As this MySQL server is running on a development box, select Developer Machine as the server type.



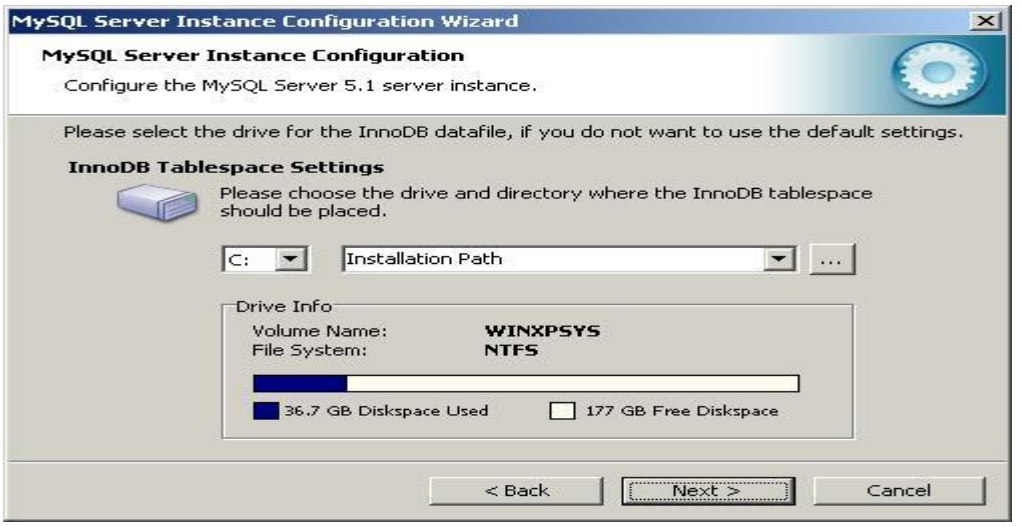
4. Select the database usage.

Depends on the purpose of your development box, you can select either Multifunctional Database or Transactional Database Only.

Here we selected Transactional Database Only.

5. Set InnoDB table space settings.

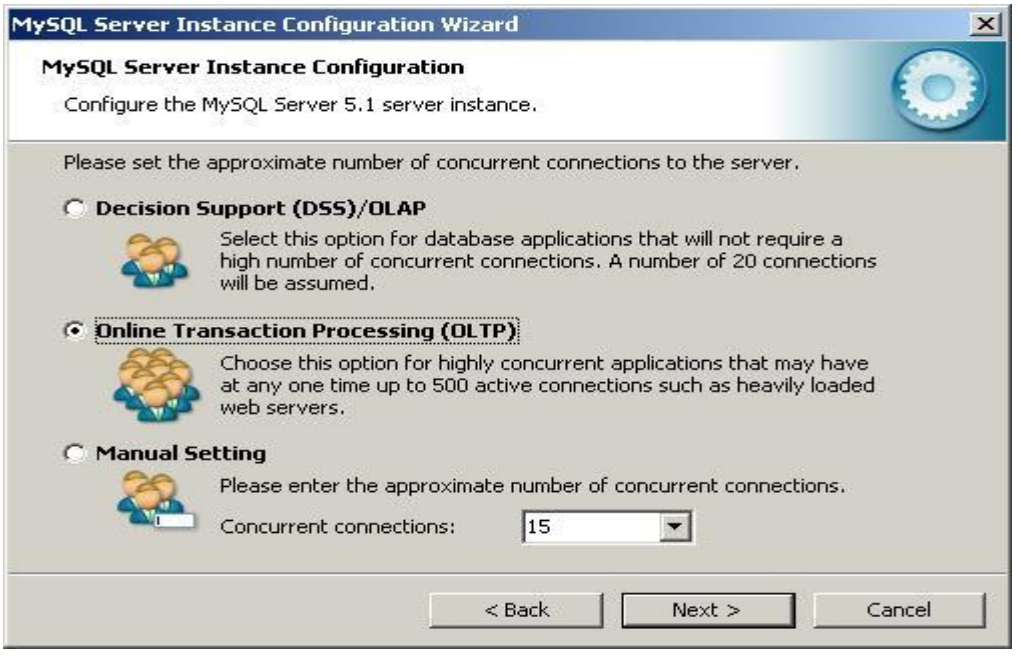
InnoDB table type is the storage engine for a transactional database. Use the default settings here.



6. Set the database engine's concurrent connections option.

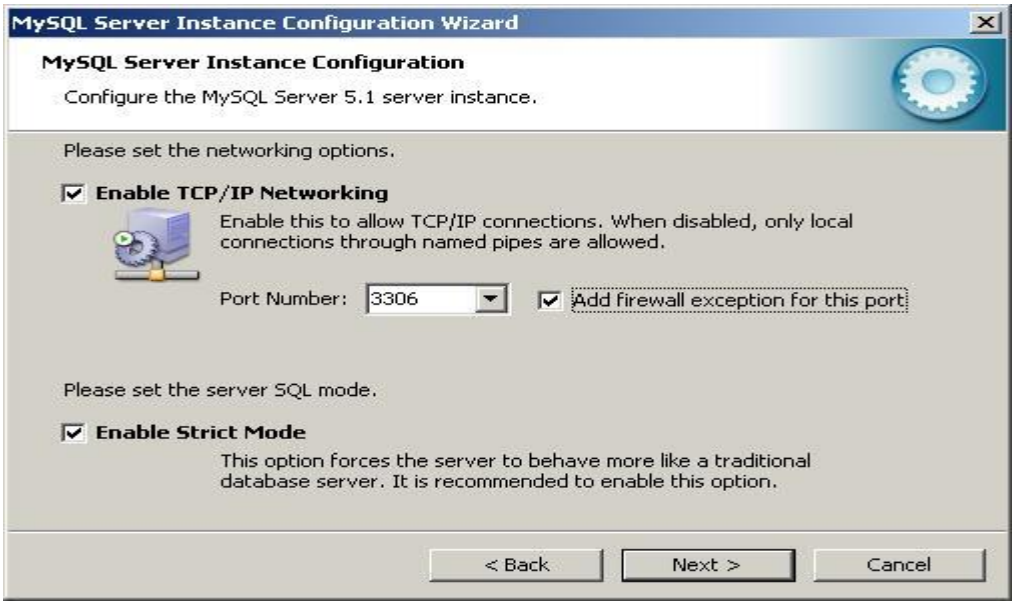
Here we selected Online Transaction Processing (OLTP) as it is the most common use of MySQL server.

If the MySQL server on your development machine is used for decision support such as data warehousing or data mining, select the first option.



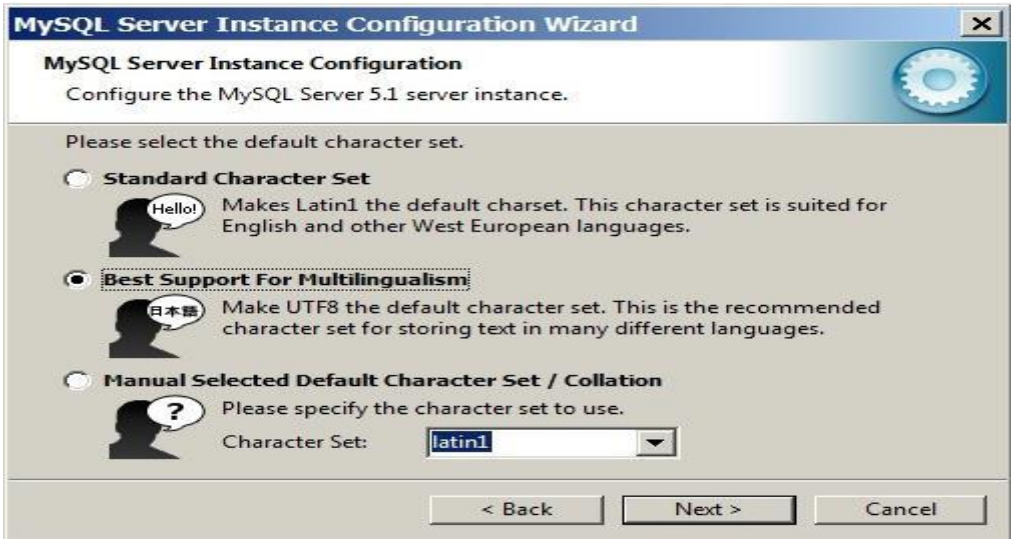
7. Set networking options.

By default, make your selection the same as screenshot below. If port 3306 has been used by another instance of MySQL server, you can select port 3307 or a different port. This will allow two instances of MySQL server to be accessed via different ports on the same box.



8. Select the default character set.

By default, the Standard Character Set is selected, but you may want to select the second option - Best Support For Multilingualism. This allows our database to store text in many different languages.



9. Install MySQL server as a Windows service.Below is the recommended way to run MySQL server on Windows. Making the service name as MySQL51 clearly identifies the service as a MySQL server version 5.1 database engine because you might install other versions of MySQL server on the same machine.

You might also want to check **Include Bin Directory in Windows PATH** if you want to operate MySQL from command line.

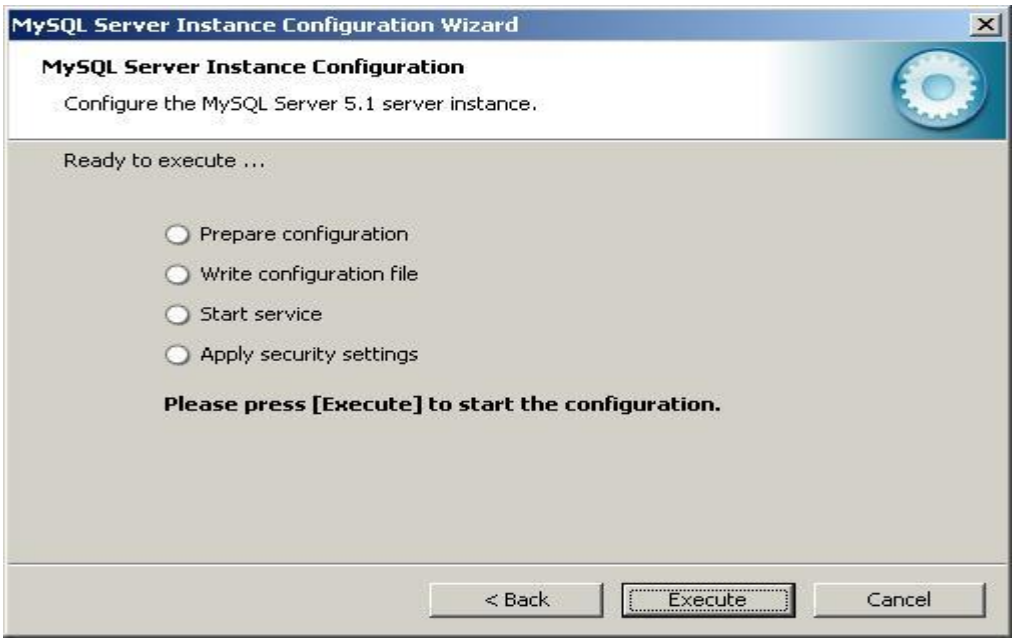
10. Set the root password.

Set a new password to the root account. Enter the same password to all three boxes. See below.

Don't select the Create An Anonymous Account checkbox. This can lead to an insecure system.

11. Ready to execute the configuration.

Now everything is ready to execute. Click [Execute] button.



12. Configuration successfully completed.

The configuration file has been created successfully. Click [Finish] to close the wizard.

Happy Configuring!

Creation of databases:

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| test               |
+--------------------+
3 rows in set (0.09 sec)
mysql> create database groupa;
Query OK, 1 row affected (0.01 sec)
mysql> use groupa;
Database changed
```

Creation of tables:

```
mysql> create table groupamem(rollno numeric(10),name varchar(15),phone numeric(
10),branch varchar(10));
Query OK, 0 rows affected (0.42 sec)

mysql> desc groupamem;
+--------+--------------+------+-----+---------+-------+
| Field | Type          | Null | Key | Default | Extra |
+--------+--------------+------+-----+---------+-------+
| rollno | decimal(10,0) | YES |     | NULL    |       |
| name   | varchar(15)   | YES |     | NULL    |       |
| phone  | decimal(10,0) | YES |     | NULL    |       |
| branch | varchar(10)   | YES |     | NULL    |       |
+--------+--------------+------+-----+---------+-------+
4 rows in set (0.03 sec)
```

Altering the table:

```
mysql> alter table groupamem add gender char(3);
Query OK, 0 rows affected (0.36 sec)
Records: 0 Duplicates: 0 Warnings: 0

mysql> desc groupamem;
+--------+--------------+------+-----+---------+-------+
| Field | Type          | Null | Key | Default | Extra |
+--------+--------------+------+-----+---------+-------+
| rollno | decimal(10,0) | YES |     | NULL    |       |
| Name   | varchar(15)   | YES |     | NULL    |       |
| phone | decimal(10,0) | YES |     | NULL    |       |
| Branch | varchar(10)   | YES |     | NULL    |       |
| Gender | char (3)      | YES |     | NULL    |       |
+--------+--------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

Dropping the table:

```
mysql> create table dd(name varchar(10));
Query OK, 0 rows affected (0.09 sec)

mysql> show tables;
+-----------------+
| Tables_in_groupa |
+-----------------+
| dd              |
| groupamem       |
+-----------------+
2 rows in set (0.00 sec)

mysql> drop table dd;
Query OK, 0 rows affected (0.06 sec)
```

Dropping the database:

```
mysql> create database dbl;
Query OK, 1 row affected (0.01 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| dbl                |
| groupa             |
| mysql              |
| Test               |
+--------------------+
6 rows in set (0.01 sec)

mysql> drop database dbl;
Query OK, 0 rows affected (0.00 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| groupa             |
| mysql              |
| test               |
+--------------------+
5 rows in set (0.00 sec)
```

Rename the tables:

```
mysql> rename table groupamem to ga;
Query OK, 0 rows affected (0.03 sec)

mysql> show tables;
+-----------------+
| Tables_in_groupa |
+-----------------+
| ga              |
+-----------------+
1 row in set (0.00 sec)
```

Truncate the table:

```
mysql> insert into ga values(1111,'ram',9885321456,'mbbs','m');
Query OK, 1 row affected (0.06 sec)

mysql> select * from ga;
+--------+------+------------+--------+--------+
| rollno | name | phone      | branch | gender |
+--------+------+------------+--------+--------+
|  1111 | ram  | 9885321456 | mbbs   | m      |
+--------+------+------------+--------+--------+
1 row in set (0.01 sec)

mysql> truncate table ga;
Query OK, 1 row affected (0.09 sec)
mysql> select * from ga;
Empty set (0.00 sec)
```

Creation of tables for Roadway Travels:

Bus

```
mysql> create table bus555(busno varchar(10),bustype varchar(10),primary key(bus
no));
Query OK, 0 rows affected (0.17 sec)
```

Ticket

```
mysql> create table ticket555(tic_no numeric(10),joudate date,source varchar(10)
,dest varchar(10),deptime time,reatime time,busnumber varchar(10),primary key(ti
c_no));
Query OK, 0 rows affected (0.08 sec)

mysql> alter table ticket555 add constraint tic_fk foreign key(busnumber) refere
nces bus555(busno);
Query OK, 0 rows affected (0.16 sec)
Records: 0  Duplicates: 0  Warnings: 0
);
```

Passenger

```
mysql> create table passenger(pnrno numeric(10),ticnumber numeric(10),pname varc
har(15),age int(4),sex char(10),ppno varchar(15),primary key(pnrno));
Query OK, 0 rows affected (0.06 sec)

mysql> alter table passenger add constraint pas_fk foreign key(ticnumber) refere
nces ticket555(tic_no);


Query OK, 0 rows affected (0.14 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Reserve

```
mysql> create table reserve(pnrnumber numeric(10),noofseats int(8),address varch
ar(50),phno numeric(10),status char(3));
Query OK, 0 rows affected (0.16 sec)

mysql> alter table reserve add constraint res_fk foreign key(pnrnumber) referenc
es passenger(pnrno);
Query OK, 0 rows affected (0.17 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Cancel

```
mysql> create table cancel(pnrnumber numeric(10),noofseats int(8),address varcha
r(50),phno numeric(10),status char(3));
Query OK, 0 rows affected (0.06 sec)

mysql> alter table cancel add constraint can_fk foreign key(pnrnumber) reference
s passenger(pnrno);
Query OK, 0 rows affected (0.14 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

Practicing DML commands:

DML commands are used to for managing data within the schema objects.

Use of insert command:

Inserting values into *Bus* table:

mysql> **insert into bus555 values('ap555','ac');**
Query OK, 1 row affected (0.03 sec)
mysql> insert into bus555 values('ap501','ac');
Query OK, 1 row affected (0.03 sec)
mysql> insert into bus555 values('ap444','nonac');
Query OK, 1 row affected (0.03 sec)
mysql> insert into bus555 values('ap891','nonac');
Query OK, 1 row affected (0.03 sec)
mysql> insert into bus555 values('ap8830','metro');
Query OK, 1 row affected (0.03 sec)

Inserting values into *Ticket* table:

mysql> **insert into ticket555 values(1111,'2010-08-05','srpt','hyd','09:00:05','19:15:00','ap555');**
Query OK, 1 row affected (0.03 sec)

mysql> insert into ticket555 values(2222,'2010-08-05','srpt','hyd','10:00:05','20:15:00','ap501');
Query OK, 1 row affected (0.03 sec)

mysql> insert into ticket555 values(3333,'2010-08-15','hyd','srpt','09:00:05','20:15:00','ap444');
Query OK, 1 row affected (0.03 sec)

mysql> insert into ticket555 values(4444,'2010-08-18','hyd','srpt','09:30:05','20:15:00','ap891');
Query OK, 1 row affected (0.03 sec)

mysql> insert into ticket555 values(5555,'2010-08-8','hyd','vij','09:10:05','22:15:00','ap8830');
Query OK, 1 row affected (0.03 sec)

Inserting values into *Passenger* table:

mysql> insert into passenger values(1001,1111,'subbu',31,'m','pp555');
Query OK, 1 row affected (0.05 sec)
mysql> insert into passenger values(1002,2222,'achaith',22,'m','pp8830');
Query OK, 1 row affected (0.03 sec)
mysql> insert into passenger values(1003,3333,'padma',25,'f','pp333');
Query OK, 1 row affected (0.05 sec)
mysql> insert into passenger values(1004,4444,'ravi',23,'m','pp444');
Query OK, 1 row affected (0.01 sec)
mysql> insert into passenger values(1005,5555,'nirma',42,'f','pp666');
Query OK, 1 row affected (0.03 sec)

Inserting values into *reserve* table:

mysql> **insert into reserve values(1001,10,'hno:5-4,srpt,nlg',9492506282,'yes');**
Query OK, 1 row affected (0.05 sec)
mysql> **insert into reserve values(1001,5,'hno:5-4,srpt,nlg',9492506282,'yes');**
Query OK, 1 row affected (0.03 sec)
mysql> **insert into reserve values(1002,5,'hno:15-4,lbngr,hyd',9491653714,'yes');**
Query OK, 1 row affected (0.01 sec)
mysql**> insert into reserve values(1003,6,'hno:151-4,dsnr,hyd',9704613151,'yes');**
Query OK, 1 row affected (0.03 sec)
mysql> **insert into reserve values(1004,8,'hno:11-4,dsnr,hyd',9704613111,'yes');**
Query OK, 1 row affected (0.02 sec)
mysql> **insert into reserve values(1005,8,'hno:41-4,dsnr,hyd',9989503111,'no');**
Query OK, 1 row affected (0.03 sec)
mysql> **insert into reserve values(1005,5,'hno:41-4,dsnr,hyd',9989503111,'yes');**
Query OK, 1 row affected (0.02 sec)
mysql> **insert into reserve values(1002,5,'hno:15-4,lbngr,hyd',9491653714,'yes');**
Query OK, 1 row affected (0.01 sec)
mysql> **insert into reserve values(1003,6,'hno:151-4,dsnr,hyd',9704613151,'yes');**
Query OK, 1 row affected (0.03 sec)
mysql**> insert into reserve values(1004,8,'hno:11-4,dsnr,hyd',9704613111,'yes');**
Query OK, 1 row affected (0.02 sec)
mysql> **insert into reserve values(1005,8,'hno:41-4,dsnr,hyd',9989503111,'no');**
Query OK, 1 row affected (0.03 sec)
mysql> **insert into reserve values(1005,5,'hno:41-4,dsnr,hyd',9989503111,'yes');**
Query OK, 1 row affected (0.02 sec)

Inserting values into *cancel* table:

mysql> insert into cancel values(1001,5,'hno:5-4,srpt,nlg',9492506282,'yes');
Query OK, 1 row affected (0.05 sec)

mysql> insert into cancel values(1001,2,'hno:5-4,srpt,nlg',9492506282,'yes');
Query OK, 1 row affected (0.03 sec)

mysql> insert into cancel values(1002,2,'hno:15-4,lbngr,hyd',9491653714,'no');
Query OK, 1 row affected (0.05 sec)

mysql> insert into cancel values(1003,2,'hno:151-4,dsnr,hyd',9704613151,'yes');
Query OK, 1 row affected (0.01 sec)

mysql> insert into cancel values(1004,5,'hno:11-4,dsnr,hyd',9704613111,'yes');
Query OK, 1 row affected (0.03 sec)

mysql> insert into cancel values(1005,4,'hno:41-4,dsnr,hyd',9989503111,'yes');
Query OK, 1 row affected (0.03 sec)

Use of select command:

mysql> select *from bus555;

```
+--------+---------+
| busno  | bustype |
+--------+---------+
| ap444  | nonac   |
| ap501  | ac      |
| ap555  | ac      |
| ap8830 | metro   |
| ap891  | nonac   |
+--------+---------+
5 rows in set (0.00 sec)
```

mysql> select *from ticket555;

```
+--------+------------+--------+------+----------+----------+-----------+

| tic_no | joudate    | source | dest | deptime  | reatime  | busnumber |
+--------+------------+--------+------+----------+----------+-----------+
|  1111 | 2010-08-05 | srpt   | hyd  | 09:00:05 | 19:15:00 | ap555     |
|  2222 | 2010-08-05 | srpt   | hyd  | 10:00:05 | 20:15:00 | ap501     |
|  3333 | 2010-08-15 | hyd    | srpt | 09:00:05 | 20:15:00 | ap444     |
|  4444 | 2010-08-18 | hyd    | srpt | 09:30:05 | 20:15:00 | ap891     |
|  5555 | 2010-08-08 | hyd    | vij  | 09:10:05 | 22:15:00 | ap8830    |
+--------+------------+--------+------+----------+----------+-----------+
5 rows in set (0.00 sec)set (0.00 sec)
```

mysql> select *from passenger;

```
+-------+-----------+---------+------+------+--------+
| pnrno | ticnumber | pname   | age  | sex  | ppno   |
+-------+-----------+---------+------+------+--------+
| 1001 |      1111 | subbu   |  31 | m    | pp555  |
| 1002 |      2222 | achaith |  22 | m    | pp8830 |
| 1003 |      3333 | padma   |  25 | f    | pp333  |
| 1004 |      4444 | ravi    |  23 | m    | pp444  |
| 1005 |      5555 | nirma   |  42 | f    | pp666  |
+-------+-----------+---------+------+------+--------+
5 rows in set (0.03 sec)
```

mysql> select *from reserve;

```
+----------+----------+------------------+-----------+--------+
| pnrnumber | noofseats | address          | phno      | status |
+----------+----------+------------------+-----------+--------+
|     1001 |       10 | hno:5-4,srpt,nlg  | 9492506282 | yes    |
|     1001 |        5 | hno:5-4,srpt,nlg  | 9492506282 | yes    |
|     1002 |        5 | hno:15-4,lbngr,hyd | 9491653714 | yes    |
|     1003 |        6 | hno:151-4,dsnr,hyd | 9704613151 | yes    |
|     1004 |        8 | hno:11-4,dsnr,hyd  | 9704613111 | yes    |
|     1005 |        8 | hno:41-4,dsnr,hyd  | 9989503111 | no     |
|     1005 |        5 | hno:41-4,dsnr,hyd  | 9989503111 | yes    |
+----------+----------+------------------+-----------+--------+
7 rows in set (0.02 sec)
```

mysql> select *from cancel;

```
+----------+----------+------------------+-----------+--------+
| pnrnumber | noofseats | address          | phno      | status |
+----------+----------+------------------+-----------+--------+
|     1001 |        5 | hno:5-4,srpt,nlg  | 9492506282 | yes    |
|     1001 |        2 | hno:5-4,srpt,nlg  | 9492506282 | yes    |
|     1002 |        2 | hno:15-4,lbngr,hyd | 9491653714 | no     |
|     1003 |        2 | hno:151-4,dsnr,hyd | 9704613151 | yes    |
|     1004 |        5 | hno:11-4,dsnr,hyd  | 9704613111 | yes    |
|     1005 |        4 | hno:41-4,dsnr,hyd  | 9989503111 | yes    |
+----------+----------+------------------+-----------+--------+
6 rows in set (0.00 sec)
```

Use of update command:
mysql> update passenger set ppno='pp888' where pnrno=1001;
Query OK, 1 row affected (0.03 sec)
Rows matched: 1 changed: 1 warnings: 0

Use of DELETE command:

mysql>delete from cancel where status='no';

Query OK, 1 row affected (0.03 sec)

mysql> select *from cancel;

```
+----------+----------+------------------+-----------+--------+
| pnrnumber | noofseats | address          | phno      | status |
+----------+----------+------------------+-----------+--------+
|     1001 |        5 | hno:5-4,srpt,nlg  | 9492506282 | yes    |
|     1001 |        2 | hno:5-4,srpt,nlg  | 9492506282 | yes    |
|     1003 |        2 | hno:151-4,dsnr,hyd | 9704613151 | yes    |
|     1004 |        5 | hno:11-4,dsnr,hyd  | 9704613111 | yes    |
|     1005 |        4 | hno:41-4,dsnr,hyd  | 9989503111 | yes    |
+----------+----------+------------------+-----------+--------+
6 rows in set (0.00 sec)
```

Experiment- 6

Querying: In this week you are going to practice the queries (along with sub queries) using ANY, ALL, IN, EXISTS, NOT EXISTS, UNION, INTERSECT, Constraints etc.

Practice the following queries:

1. Display unique PNR_no of all passengers.

```
mysql> select distinct pnrno from passenger;
+-------+
| pnrno |
+-------+
| 1001 |
| 1002 |
| 1003 |
| 1004 |
| 1005 |
+-------+
5 rows in set (0.01 sec)
```

2. Display all the names of male passengers.

```
mysql> select pname from passenger where sex='m';
+---------+
| pname   |
+---------+
| subbu   |
| achaith |
| ravi    |
+---------+
3 rows in set (0.00 sec)
```

3. Display ticket numbers and names of all the passengers.

```
mysql> select tic_no,pname from ticket555 t,passenger p where t.tic_no=p.ticnumber;
+--------+---------+
| tic_no | pname   |
+--------+---------+
|  1111 | subbu    |
|  2222 | achaith |gh
|  3333 | padma   |
|  4444 | ravi    |
|  5555 | nirma   |
+--------+---------+
5 rows in set (0.00 sec)
```

4. Find the ticket numbers of passengers whose name starts with ‘R’ and ends with ‘H’.

mysql> **select tic_no from ticket555 where tic_no=any (select ticnumber from passenger where pname like 'r%h');**

```
+--------+
|tic_no |
+--------+
|  1111 |
|  2222 |
+--------+
2 rows in set (0.02 sec)
```

5. Find the name of passengers whose age is between 30 and 45.

mysql> **select pname from passenger where age between 30 and 45;**

```
+-------+
|pname |
+-------+
|subbu  |
|nirma |
+-------+
2 rows in set (0.00 sec)
```

6. Display all the passengers names beginning with ‘A’.

mysql> **select all pname from passenger where pname like 'a%';**

```
+---------+
|pname   |
+---------+
|subbu    |
|achaith |
+---------+
2 rows in set (0.00 sec)
```

7. Display the sorted list of passengers names.

mysql> **select pname from passenger order by pname;**

```
+---------+
|pname   |
+---------+
|subbu    |
|achaith |
|nirma   |
|padma   |
|ravi    |
+---------+
5 rows in set (0.02 sec)
```

Experiment- 7

You are going to practice the queries using Aggregate functions (COUNT, SUM, AVG, MAX and MIN), GROUP BY, HAVING AND Creation of Views.

1. Write a query to display the information present in the Passenger and Cancellation tables.

mysql> **select pnrno from passenger union select pnrnumber from cancel;**

```
+-------+
| pnrno |
+-------+
| 1001 |
| 1002 |
| 1003 |
| 1004 |
| 1005 |
+-------+
5 rows in set (0.00 sec)
```

2. Display the number of days in a week on which 9w01 bus is available.

mysql> **select count(joudate) from ticket555 where busnumber in(select busno from bus555 where busno='9w01') and joudate between '2010-08-14' and '2010-08-20';**

```
+----------------+
| count(joudate) |
+----------------+
|              2 |
+----------------+
1 row in set (0.00 sec)
```

3. Find the ticket numbers booked for each PNR_no using Group By clause.

mysql> **select sum(noofseats) as reserved_seats,pnrnumber from reserve where stat us='yes' group by pnrnumber;**

```
+----------------+-----------+
| reserved_seats | pnrnumber |
+----------------+-----------+
|             15 |      1001 |
|              5 |      1002 |
|              6 |      1003 |
|              8 |      1004 |
|              5 |      1005 |
+----------------+-----------+
5 rows in set (0.33 sec)
```

4. Find the distinct PNR numbers that are present.

mysql> **select distinct pnrnumber from reserve;**

```
+-----------+
| pnrnumber |
+-----------+
|      1001 |
|      1002 |
|      1003 |
|      1004 |
|      1005 |
+-----------+
5 rows in set (0.00 sec)
```

5. Find the number of tickets booked by passenger where the number of seats is greater than1
Mysql>select pnrnumber,sum(noofseats) from reservation group by pnrnumber having sum(nooseats) > 1

mysql> select busno,bustype,sum(noofseats) as booked_seats from bus555,reserve,t icket555,passenger where busno=busnumber and tic_no=ticnumber and pnrno=pnrnumbe r and status='yes'group by tic_no having count(*)>=1;

```
+--------+---------+--------------+
| busno  | bustype | booked_seats |
+--------+---------+--------------+
| ap555  | ac      |           15 |
| ap501  | ac      |            5 |
| ap444  | nonac   |            6 |

| ap891  | nonac   |            8 |
| ap8830 | metro   |            5 |
+--------+---------+--------------+
5 rows in set (0.00 sec)
```

6.find the total number of cancelled seats

Mysql>**select sum(noofseats) as cancelled-seats from cancel where status='yes';**

Cancled-seats

18

Creation of views:

7.Create a view to display the details of passenger who cancelled their tickets

Create view cancel1 as

Select pname,pnrno

From passenger,cancel

Where pnrno=pnrnumber and status='yes';

Experiment -8

create a table for the following schema.
Student(snum:integer,sname:string,major:string,level:string,age:integer)
class(name:string,meets at:time,room:string,fid:integer)
enrolled(snum:integer,cname:string) faculty(fid:integer,fname:string,deptid:integer)

create table Student (snum int(5),sname varchar(10),major varchar(10),level varchar(10),age int(10));

create table Class (name varchar(10),meetsat TIME,room varchar(10),fid int(10));

creta table Enrolled (snum int(10),cname varchar(10));

create table Faculty(fid int(10),fname varchar(10),deptid int(10));

insert into student values(111,'vamsi','dbms','sr',25);
insert into class values('cse','23:00:00','r103',222);
insert into enrolled values(111,'cse');
insert into faculty(222,'uma',12345);

select * from student;
select * from class;
select * from enrolled;
select * from faculty;

Experiment-9.
QUERYING

1.find the names of all juniors(level=JR) who are enrolled in aclass taught by I.teacher

1.SELECT DISTINCT S.Sname
FROM Student S, Class C, Enrolled E, Faculty F
WHERE S.snum = E.snum AND E.cname = C.name AND C.fid = F.fid AND
F.fname = 'I.Teach' AND S.level = 'JR';

2.find the age of all the oldest student who is either a history major or is enrolled in a course taught by I.teacher.

. SELECT MAX(S.age)
FROM Student S
WHERE (S.major = 'History')
OR S.num IN (SELECT E.snum
FROM Class C, Enrolled E, Faculty F
WHERE E.cname = C.name AND C.fid = F.fid
AND F.fname = 'I.Teach');

3.**find the names of all classes that either meet in room R128 or more students enrolled**
SELECT C.name
FROM Class C
WHERE C.room = 'R128'
OR C.name IN (SELECT E.cname
FROM Enrolled E
GROUP BY E.cname
HAVING COUNT (*) >= 5);
4.**Find the names of all the students who are enrolled in two classes that meet at the same time**

SELECT DISTINCT S.sname
FROM Student S
WHERE S.snum IN (SELECT E1.snum
FROM Enrolled E1, Enrolled E2, Class C1, Class C2
WHERE E1.snum = E2.snum AND E1.cname <> E2.cname
AND E1.cname = C1.name
AND E2.cname = C2.name AND C1.time = C2.time);

5.**find the names of all the faculty members who teach in evry room in whioch some class is taught**
SELECT DISTINCT F.fname
FROM Faculty F
WHERE NOT EXISTS ((SELECT *
42 Chapter 5
FROM Class C)
EXCEPT
(SELECTC1.room
FROM Class C1
WHERE C1.fid = F.fid));

6.**find the names of faculty members for whom the combined enrollement of courses that they teach in less than 5.**

```
SELECT DISTINCT F.fname
FROM Faculty F
WHERE 5 > (SELECT E.snum
FROM Class C, Enrolled E
WHERE C.name = E.cname
AND C.fid = F.fid);
```

7.**print the level and average age of students for the level,for each level.**

```
SELECT S.level, AVG(S.age)
FROM Student S
GROUP BY S.level;
```

8. **print the level and average age of students for the level,for all level except JR.9**

```
SELECT S.level, AVG(S.age)
FROM Student S
WHERE S.level <> 'JR'
GROUP BY S.level;
```

9. **print the level and average age of students for that level, whose average age is greater that 20**

```
SELECT S.level,AVG(S.age)
FROM Student S
WHERE AVG(S.age)>20;
```

10. **find the names of all the students who are enrolled in the maximum no. of classes**

```
SELECT DISTINCT S.sname
FROM Student S
WHERE S.snum IN (SELECT E.snum
FROM Enrolled E
GROUP BY E.snum
HAVING COUNT (*) >= ALL (SELECT COUNT (*)
FROM Enrolled E2
GROUP BY E2.snum ));
```

11. **find the names of all the students who are not enrolled in the class.**

```
SELECT DISTINCT S.sname
FROM Student S
WHERE S.snum NOT IN (SELECT E.snum
FROM Enrolled E );
```

12.**count the no. of junior level students.**

```
SELECT COUNT(*)
FROM Student where level='JR';
```

13.**display all the students whose names starts with the letter"p"**

```
SELECT * FROM Student
WHERE sname like 'p%';
```

14. **display all the teachers whose names starts with the letter"a" or "I" in their names.**

```
SELECT * FROM faculty
WHERE fname like '%a%' or Fname like '%I%';
```

Experiment-10

PROCEDURES

In this session you are going to learn Creation of stored procedures, execution of procedure and modification of procedures. Practice the procedures using above database.

A stored procedure is a procedure (like a subprogram in a regular computing language) that is

Stored (in the database). Correctly speaking, MySQL supports "routines" and there are two kinds of routines: stored procedures which you call, or functions whose return values you use in other SQL statements the same way that you use pre-installed MySQL functions like pi(). I'll use the word "stored procedures" more frequently than "routines" because it's what we've used in the past, and what people expect us to use.

```
mysql> create procedure p2(p_age int)
   -> begin
   -> select pname,ticnumber,sex from passenger where age>p_age;
   -> end//
Query OK, 0 rows affected (0.00
sec)

mysql>          call
p2(30)//
+-------+-----------+-----
-+
| pname | ticnumber | sex
|
+-------+-----------+-----
-+
| subbu |      1111 | m    |
| nirma |      5555 | f    |
+-------+-----------+-----
-+
2 rows in set (0.00
sec)

mysql>          call
p2(24)//
+-------+-----------+-----
-+
| pname | ticnumber | sex
|
+-------+-----------+-----
-+
| subbu |      1111 | m    |

| padma |      3333 | f    |
| nirma |      5555 | f    |
+-------+-----------+-----
-+
3 rows in set (0.00
sec)
Query OK, 0 rows affected (0.00
sec)
```

mysql> create procedure p3()
-> begin
-> Select source,dest from ticket555 where hour(timediff(reatime,deptime))>10;
-> End//
Query OK, 0 rows affected (0.02 sec)

mysql> call p3()//
+--------+------+
| Source | dest |
+--------+------+