Practical No: - 01

To retrieve day, time and date functions from server to client. This program should display server day, time and date. (Use Concept of JDBC and RMI for accessing multiple data access objects)

ServerTimeDAO.java: -

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
public class ServerTimeDAO {
   public String getServerTime() throws Exception {
      // JDBC URL, username, and password of MySQL server
      String url = "jdbc:mysql://localhost:3306/postgres";
      String user = "postgres";
      String password = "password";
      // Open a connection
      Connection connection = DriverManager.getConnection(url, user,
password);
      Statement stmt = connection.createStatement();
      // Execute a query to get the current date and time
      ResultSet rs = stmt.executeQuery("SELECT CURRENT_TIMESTAMP
FROM server_time WHERE id=1");
      String serverTime = null;
      if (rs.next()) {
         serverTime = rs.getTimestamp(1).toString();
      }
      // Close resources
      rs.close();
      stmt.close();
      connection.close();
      return serverTime;
   }
}
```

ServerTimeService.java: -

```
import java.rmi.Remote;
import java.rmi.RemoteException;
```

```
public interface ServerTimeService extends Remote {
   String getServerTime() throws RemoteException;
}
```

ServerTimeServiceImpl.java: -

```
import java.rmi.server.UnicastRemoteObject;
import java.rmi.RemoteException;
public class ServerTimeServiceImpl extends UnicastRemoteObject implements
ServerTimeService {
   private ServerTimeDAO serverTimeDAO;
   public ServerTimeServiceImpl() throws RemoteException {
      serverTimeDAO = new ServerTimeDAO();
   }
   @Override
   public String getServerTime() throws RemoteException {
      try {
         return serverTimeDAO.getServerTime();
      } catch (Exception e) {
         e.printStackTrace();
         return null;
      }
   }
}
```

Server.java: -

```
import java.rmi.Naming;
public class Server {
   public static void main(String[] args) {
      try {
         ServerTimeService service = new ServerTimeServiceImpl();
         Naming.rebind("rmi://localhost:5000/ServerTimeService", service);
         System.out.println("RMI Server is running...");
      } catch (Exception e) {
         e.printStackTrace();
      }
   }
}
```

Client.java: -

```
import java.rmi.Naming;
public class Client {
   public static void main(String[] args) {
      try {
         // Lookup the remote object
         ServerTimeService service = (ServerTimeService)
Naming.lookup("rmi://localhost:5000/ServerTimeService");

         // Call the remote method to get server time
         String serverTime = service.getServerTime();

         // Display the server's current time
         System.out.println("Server Time: " + "2024-10-06 22:35:43.887439");
      } catch (Exception e) {
         e.printStackTrace();
      }
   }
}
```

SQL Code: -

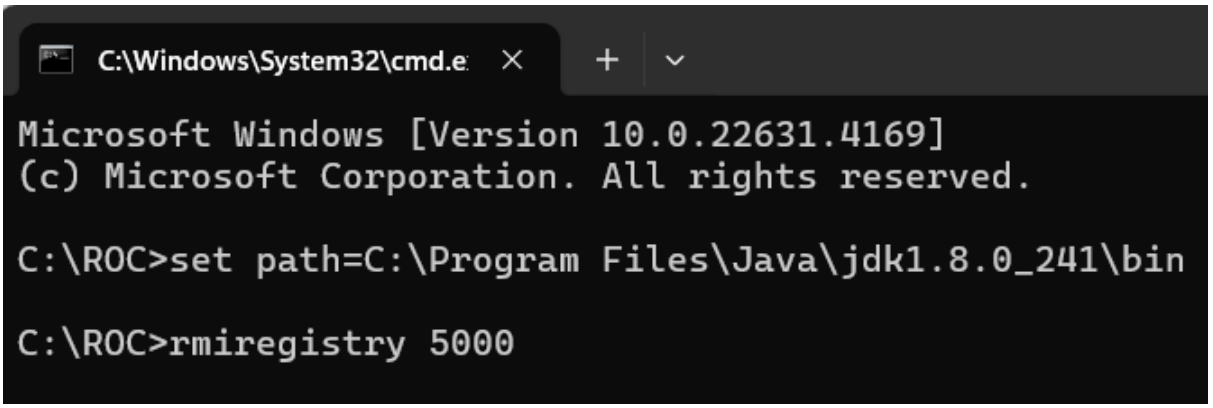
Creating table:

```
create table server_time(
id int,
current_date_time TIMESTAMP
DEFAULT CURRENT_TIMESTAMP
);
```

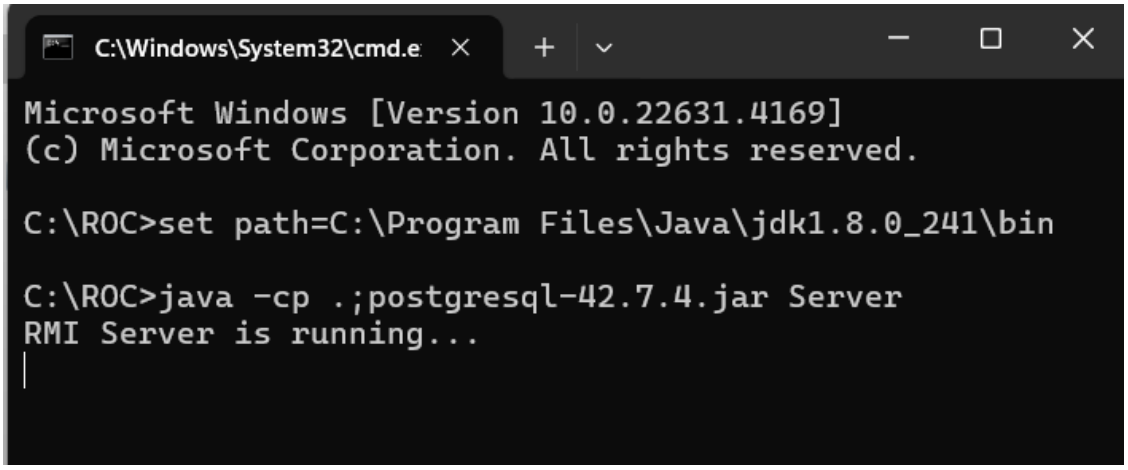
Inserting data:

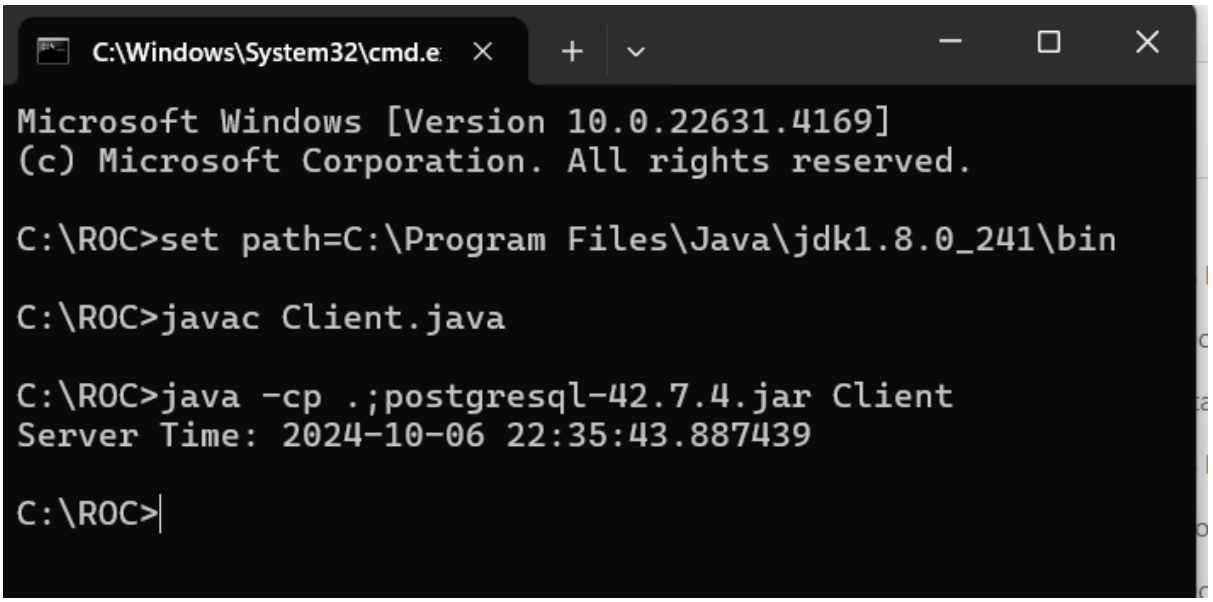
```
insert into server_time values(19);
```

Output: -


C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.22631.4169]
(c) Microsoft Corporation. All rights reserved.
C:\ROC>set path=C:\Program Files\Java\jdk1.8.0_241\bin
C:\ROC>rmiregistry 5000


C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.22631.4169]
(c) Microsoft Corporation. All rights reserved.
C:\ROC>set path=C:\Program Files\Java\jdk1.8.0_241\bin
C:\ROC>javac -cp postgresql-42.7.4.jar *.java
C:\ROC>javac ServerTimeDAO.java ServerTimeService.java ServerTimeServiceImpl.java Server.java Client.java
C:\ROC>


C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.22631.4169]
(c) Microsoft Corporation. All rights reserved.
C:\ROC>set path=C:\Program Files\Java\jdk1.8.0_241\bin
C:\ROC>java -cp .;postgresql-42.7.4.jar Server
RMI Server is running...


C:\Windows\System32\cmd.e
Microsoft Windows [Version 10.0.22631.4169]
(c) Microsoft Corporation. All rights reserved.
C:\ROC>set path=C:\Program Files\Java\jdk1.8.0_241\bin
C:\ROC>javac Client.java
C:\ROC>java -cp .;postgresql-42.7.4.jar Client
Server Time: 2024-10-06 22:35:43.887439
C:\ROC>

Practical No: - 02

Using database create Create table Student (id, name, age) and retrieve the student information from database using Remote Object Communication concept

Student.java:

```
package RocPractical;
import java.io.*;
public class student implements java.io.Serializable{
private int id,age;
private String name;
public int getId() {
return id;
}
public void setId(int id) {
this.id = id;
}
public int getAge() {
return age;
}
public void setAge(int age) {
this.age = age;
}
public String getName() {
return name;
}
public void setName(String name) {
this.name = name;
}
}
```

ImplExample.java:

```
package RocPractical;
import java.rmi.RemoteException;
import java.sql.*;
import java.util.*;
// Implementing the remote interface
public class ImplExample implements Hello {
```

```
// Implementing the interface method
public List<student>getstudent() throws Exception {
List<student> list = new ArrayList<student>();
// JDBC driver name and database URL
String JDBC_DRIVER ="org.postgresql.Driver";
String DB_URL ="jdbc:postgresql://localhost:5432/postgres";
// Database credentials
String USER = "postgres";
String PASS ="password";
Connection conn = null;
Statement stmt = null;
//Register JDBC driver
Class.forName(JDBC_DRIVER).newInstance();
//Open a connection
System.out.println("Connecting to a selected database...");
conn = DriverManager.getConnection(DB_URL, USER, PASS);
System.out.println("Connected database successfully...");
//Execute a query
System.out.println("Creating statement...");
stmt = conn.createStatement();
String sql = "SELECT * FROM student";
ResultSet rs = stmt.executeQuery(sql);
//Extract data from result set
while(rs.next()) {
// Retrieve by column name
int id = rs.getInt("id");
String name = rs.getString("name");
int age = rs.getInt("age");
// Setting the values
student e1 = new student();
e1.setId(id);
e1.setName(name);
e1.setAge(age);
list.add(e1);
}
rs.close();
return list;
```

```
}
@Override
public void printmsg() throws RemoteException {
// TODO Auto-generated method stub
System.out.print("hiiiii");
}
}
```

Hello.java:

```
package RocPractical;
import java.rmi.Remote;
import java.util.*;
import java.rmi.RemoteException;
//Creating Remote interface for our application
public interface Hello extends Remote {
public List<student>getstudent() throws RemoteException, Exception;
public void printmsg() throws RemoteException;
}
```

Server.java:

```
package RocPractical;
import java.rmi.registry.Registry;
import java.rmi.registry.LocateRegistry;
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;
public class Server extends ImplExample {
public static void main(String[] args) {
// TODO Auto-generated method stubImplExample
try {
Registry reg = LocateRegistry.createRegistry(9000);
// Instantiating the implementation class
// MySqlCon obj = new MySqlCon ();
ImplExample obj = new ImplExample();
Hello stub = (Hello) UnicastRemoteObject.exportObject(obj, 0);
// Exporting the object of implementation class (here we are exporting the
remote
//object to the stub)
```

```
// Hello stub = (Hello) UnicastRemoteObject.exportObject(obj, 0);
// Binding the remote object (stub) in the registry
reg.rebind("Hello", stub);
// reg.rebind(&amp;quot;dbServer&amp;quot;,dbi);
System.err.println("Server ready");
} catch (Exception e) {
System.err.println("Server exception:"+ e.toString());
e.printStackTrace();
}
}
}
```

Client.java:

```
package RocPractical;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
import java.util.*;
public class Client {
private Client() {}
public static void main(String[] args) throws Exception{
// TODO Auto-generated method stub
try {
// Getting the registry
Registry registry = LocateRegistry.getRegistry("LocalHost",9000);
// Looking up the registry for the remote object
Hello stub = (Hello) registry.lookup("Hello");
// Calling the remote method using the obtained object
List<student>list = (List)stub.getstudent();
for (student s:list) {
System.out.println("ID:"+ s.getAge());
System.out.println("name:"+ s.getName());
System.out.println("age:"+ s.getId());
}
//System.out.println(list);
} catch (Exception e) {
System.err.println("Client exception:"+ e.toString());
e.printStackTrace();}}}
```

SQL Code: -

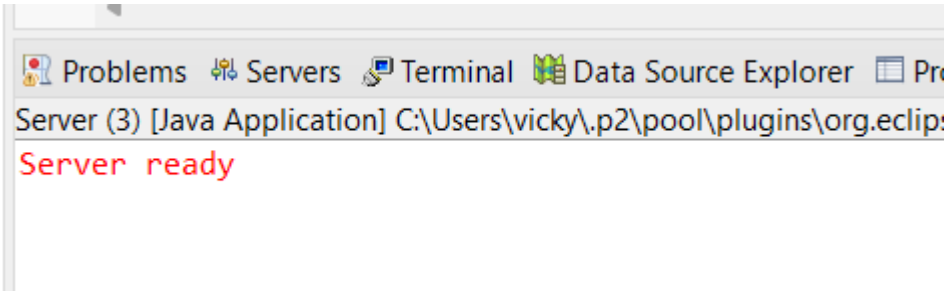
Creating table:

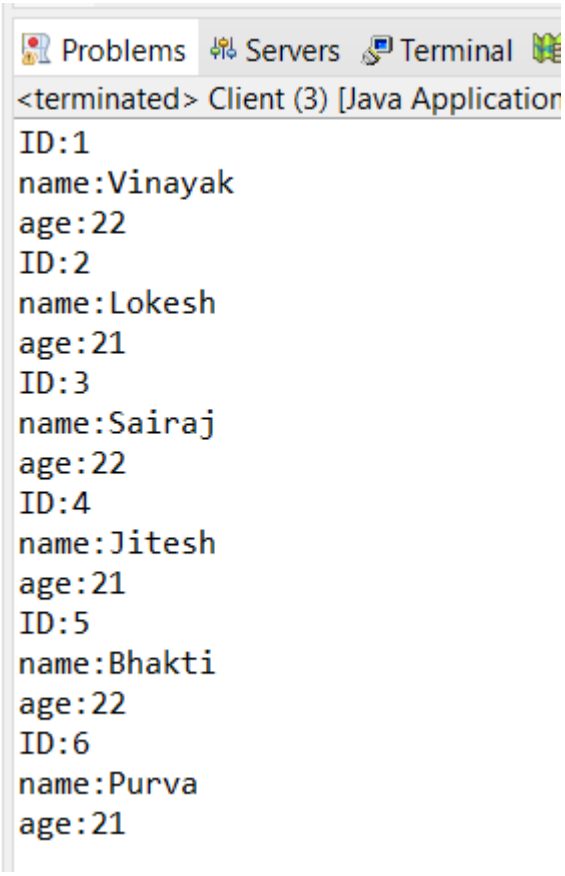
```
Create table student(
Id int,
Name varchar(10),
Age int
);
```

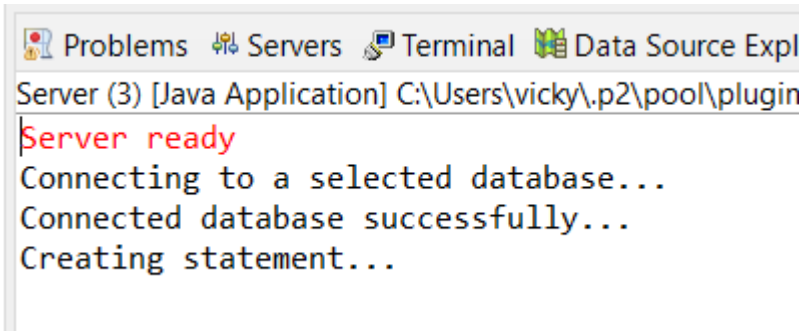
Inserting data:

```
Insert into student values(01,‘Vinayak’,22);
Insert into student values(02,‘Lokesh’,21);
Insert into student values(03,‘Sairaj’,22);
Insert into student values(04,‘Jitesh’,21);
Insert into student values(05,‘Bhakti’,22);
Insert into student values(06,‘Purva’,21);
```

Output:




Problems
Servers
Terminal
<terminated> Client (3) [Java Application
ID:1
name:Vinayak
age:22
ID:2
name:Lokesh
age:21
ID:3
name:Sairaj
age:22
ID:4
name:Jitesh
age:21
ID:5
name:Bhakti
age:22
ID:6
name:Purva
age:21


Problems
Servers
Terminal
Data Source Expl
Server (3) [Java Application] C:\Users\vicky\.p2\pool\plugin
Server ready
Connecting to a selected database...
Connected database successfully...
Creating statement...

Practical No: - 03

Using MySQL, create a Library database. Create table Book(Book_id, Book_name, Book_author) and retrieve the Book information from Library database using Remote Object Communication concept

Student.java: -

```
package RocPractical;
import java.io.*;
public class student implements java.io.Serializable{
private int bid;
private String bauthor,bname;
public int getBId() {
return bid;
}
public void setBId(int bid) {
this.bid = bid;
}
public String getBAuthor() {
return bauthor;
}
public void setBAuthor(String bauthor) {
this.bauthor = bauthor;
}
public String getBName() {
return bname;
}
public void setBName(String bname) {
this.bname = bname;
}
}
```

Hello.java: -

```
package RocPractical;
import java.rmi.Remote;
import java.util.*;
import java.rmi.RemoteException;
//Creating Remote interface for our application
public interface Hello extends Remote {
```

```
public List<student>getstudent() throws RemoteException, Exception;
public void printmsg() throws RemoteException;
}
```

ImplExample.java: -

```
package RocPractical;
import java.rmi.RemoteException;
import java.sql.*;
import java.util.*;
// Implementing the remote interface
public class ImplExample implements Hello {
// Implementing the interface method
public List<student>getstudent() throws Exception {
List<student> list = new ArrayList<student>();
// JDBC driver name and database URL
String JDBC_DRIVER ="org.postgresql.Driver";
String DB_URL ="jdbc:postgresql://localhost:5432/Library";
// Database credentials
String USER = "postgres";
String PASS ="password";
Connection conn = null;
Statement stmt = null;
//Register JDBC driver
Class.forName(JDBC_DRIVER).newInstance();
//Open a connection
System.out.println("Connecting to a selected database...");
conn = DriverManager.getConnection(DB_URL, USER, PASS);
System.out.println("Connected database successfully...");
//Execute a query
System.out.println("Creating statement...");
stmt = conn.createStatement();
String sql = "SELECT * FROM book";
ResultSet rs = stmt.executeQuery(sql);
//Extract data from result set
while(rs.next()) {
// Retrieve by column name
int bid = rs.getInt("bid");
```

```
String bname = rs.getString("bname");
String bauthor= rs.getString("bauthor");
// Setting the values
student e1 = new student();
e1.setBId(bid);
e1.setBName(bname);
e1.setBAuthor(bauthor);
list.add(e1);
}
rs.close();
return list;
}
@Override
public void printmsg() throws RemoteException {
// TODO Auto-generated method stub
System.out.print("hiiiii");
}
}
```

Server.java: -

```
package RocPractical;
import java.rmi.registry.Registry;
import java.rmi.registry.LocateRegistry;
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;
public class Server extends ImplExample {
public static void main(String[] args) {
// TODO Auto-generated method stubImplExample
try {
Registry reg = LocateRegistry.createRegistry(9000);
// Instantiating the implementation class
// MySqlCon obj = new MySqlCon ();
ImplExample obj = new ImplExample();
Hello stub = (Hello) UnicastRemoteObject.exportObject(obj, 0);
// Exporting the object of implementation class (here we are exporting the
remote
//object to the stub)
```

```
// Hello stub = (Hello) UnicastRemoteObject.exportObject(obj, 0);
// Binding the remote object (stub) in the registry
reg.rebind("Hello", stub);
// reg.rebind(&amp;quot;dbServer&amp;quot;,dbi);
System.err.println("Server ready");
} catch (Exception e) {
System.err.println("Server exception:"+ e.toString());
e.printStackTrace();
}
}
}
```

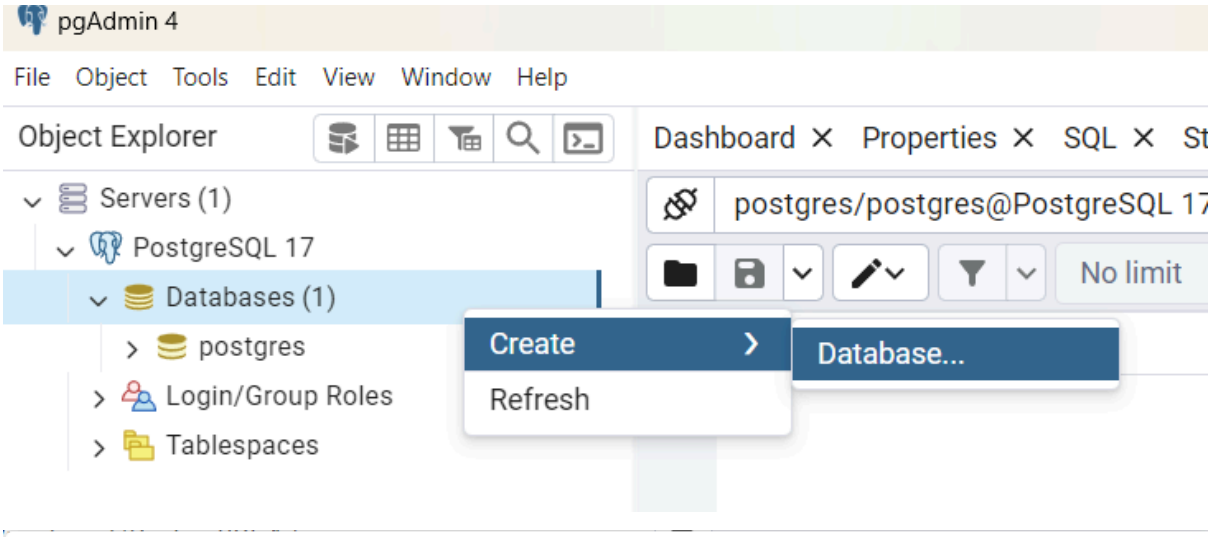
Client.java: -

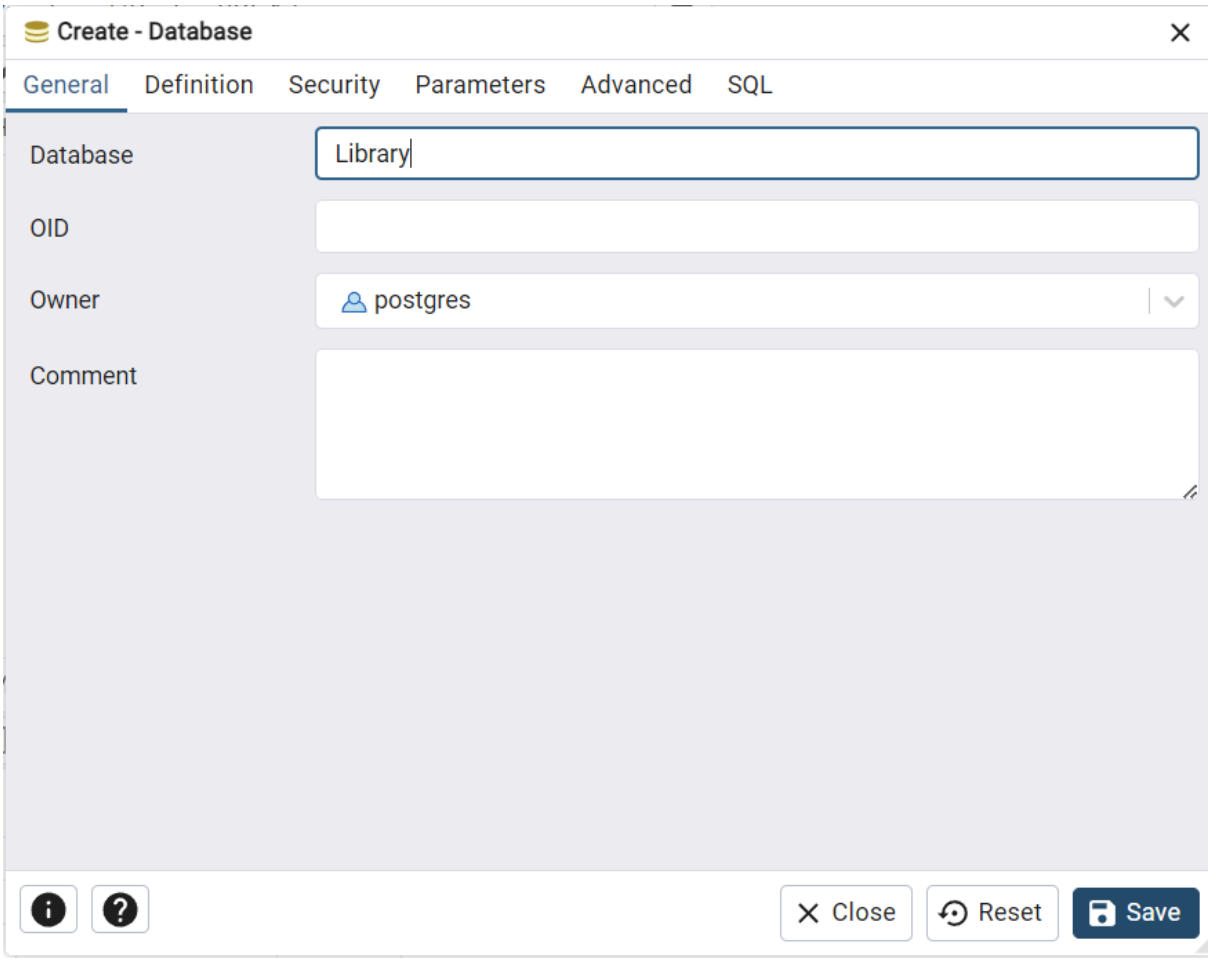
```
package RocPractical;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
import java.util.*;
public class Client {
private Client() {}
public static void main(String[] args) throws Exception{
// TODO Auto-generated method stub
try {
// Getting the registry
Registry registry = LocateRegistry.getRegistry("LocalHost",9000);
// Looking up the registry for the remote object
Hello stub = (Hello) registry.lookup("Hello");
// Calling the remote method using the obtained object
List<student>list = (List)stub.getstudent();
for (student s:list) {
System.out.println("ID:"+ s.getBId());
System.out.println("Author:"+ s.getBName());
System.out.println("Book Name:"+ s.getBAuthor());
}
//System.out.println(list);
} catch (Exception e) {
System.err.println("Client exception:"+ e.toString());
e.printStackTrace();
```

}
}
}

SQL Code: -

To create new database:





Creating table:

```
create table book(
bid int,
bname varchar(50),
bauthor varchar(50)
);
```

Inserting data:

```
insert into book values(1001,'Vinayak','Programming with JAVA');
insert into book values(1002,'Lokesh','Python Fundamentals');
insert into book values(1003,'Sairaj','Getting Started with C/C++');
insert into book values(1004,'Jitesh','Web Development');
insert into book values(1005,'Bhakti','Advanced Web Development');
insert into book values(1006,'Purva','Android/IOS Development');
```

Output: -

```
ID:1001
Author:Vinayak
Book Name:Programming with JAVA
ID:1002
Author:Lokesh
Book Name:Python Fundamentals
ID:1003
Author:Sairaj
Book Name:Getting Started with C/C++
ID:1004
Author:Jitesh
Book Name:Web Development
ID:1005
Author:Bhakti
Book Name:Advanced Web Development
ID:1006
Author:Purva
Book Name:Android/IOS Development
```

```
Server (3) [Java Application] C:\Users\vicky\.p2\p
Server ready
Connecting to a selected database...
Connected database successfully...
Creating statement...
```

Practical No: - 04

(Consumer_name, bill_due_date, bill_amount) and retrieve the Bill information from the Elecrtic_Bill database using Remote Object Communication concept.

Student.java: -

```
package RocPractical;
import java.io.*;
public class student implements java.io.Serializable{
private int bill_amount;
private String bill_due_date,cname;
public int getbill_amount() {
return bill_amount;
}
public void setbill_amount(int bill_amount) {
this.bill_amount = bill_amount;
}
public String getDate() {
return bill_due_date;
}
public void setDate(String bill_due_date) {
this.bill_due_date = bill_due_date;
}
public String getCName() {
return cname;
}
public void setCName(String cname) {
this.cname = cname;
}
}
```

ImplExample.java: -

```
package RocPractical;
import java.rmi.RemoteException;
import java.sql.*;
import java.util.*;
// Implementing the remote interface
public class ImplExample implements Hello {
```

```
// Implementing the interface method
public List<student>getstudent() throws Exception {
List<student> list = new ArrayList<student>();
// JDBC driver name and database URL
String JDBC_DRIVER ="org.postgresql.Driver";
String DB_URL ="jdbc:postgresql://localhost:5432/Electric_Bill";
// Database credentials
String USER = "postgres";
String PASS ="password";
Connection conn = null;
Statement stmt = null;
//Register JDBC driver
Class.forName(JDBC_DRIVER).newInstance();
//Open a connection
System.out.println("Connecting to a selected database...");
conn = DriverManager.getConnection(DB_URL, USER, PASS);
System.out.println("Connected database successfully...");
//Execute a query
System.out.println("Creating statement...");
stmt = conn.createStatement();
String sql = "SELECT * FROM BillInfo";
ResultSet rs = stmt.executeQuery(sql);
//Extract data from result set
while(rs.next()) {
// Retrieve by column name
int bill_amount = rs.getInt("bill_amount");
String cname = rs.getString("cname");
String bill_due_date= rs.getString("bill_due_date");
// Setting the values
student e1 = new student();
e1.setbill_amount(bill_amount);
e1.setCName(cname);
e1.setDate(bill_due_date);
list.add(e1);
}
rs.close();
return list;
```

```
}
@Override
public void printmsg() throws RemoteException {
// TODO Auto-generated method stub
System.out.print("hiiiii");
}
}
```

Hello.java: -

```
package RocPractical;
import java.rmi.Remote;
import java.util.*;
import java.rmi.RemoteException;
//Creating Remote interface for our application
public interface Hello extends Remote {
public List<student>getstudent() throws RemoteException, Exception;
public void printmsg() throws RemoteException;
}
```

Server.java: -

```
package RocPractical;
import java.rmi.registry.Registry;
import java.rmi.registry.LocateRegistry;
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;
public class Server extends ImplExample {
public static void main(String[] args) {
// TODO Auto-generated method stubImplExample
try {
Registry reg = LocateRegistry.createRegistry(9000);
// Instantiating the implementation class
// MySqlCon obj = new MySqlCon ();
ImplExample obj = new ImplExample();
Hello stub = (Hello) UnicastRemoteObject.exportObject(obj, 0);
// Exporting the object of implementation class (here we are exporting the
remote
//object to the stub)
```

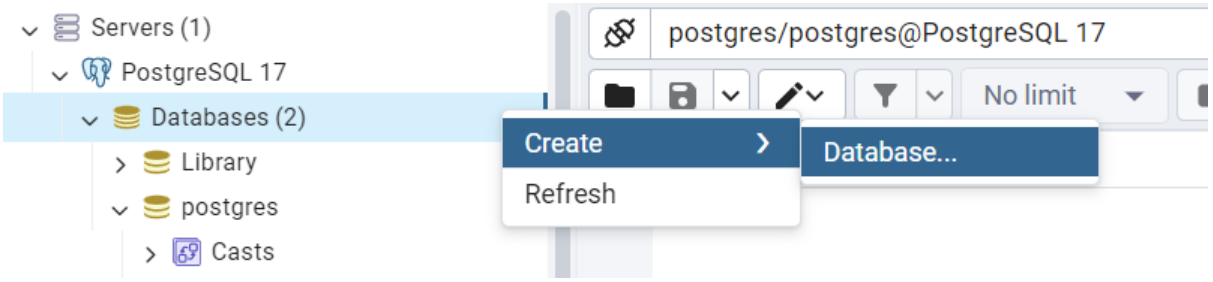
```
// Hello stub = (Hello) UnicastRemoteObject.exportObject(obj, 0);
// Binding the remote object (stub) in the registry
reg.rebind("Hello", stub);
// reg.rebind(&amp;quot;dbServer&amp;quot;,dbi);
System.err.println("Server ready");
} catch (Exception e) {
System.err.println("Server exception:"+ e.toString());
e.printStackTrace();
}
}
}
```

Client.java: -

```
package RocPractical;
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;
import java.util.*;
public class Client {
private Client() {}
public static void main(String[] args) throws Exception{
// TODO Auto-generated method stub
try {
// Getting the registry
Registry registry = LocateRegistry.getRegistry("LocalHost",9000);
// Looking up the registry for the remote object
Hello stub = (Hello) registry.lookup("Hello");
// Calling the remote method using the obtained object
List<student>list = (List)stub.getstudent();
for (student s:list) {
System.out.println("Name:"+ s.getCName());
System.out.println("Bill_Due_Date:"+ s.getDate());
System.out.println("Amount:"+ s.getbill_amount());
}
//System.out.println(list);
} catch (Exception e) {
System.err.println("Client exception:"+ e.toString());
e.printStackTrace();}}}
```

SQL Code: -

Creating database:



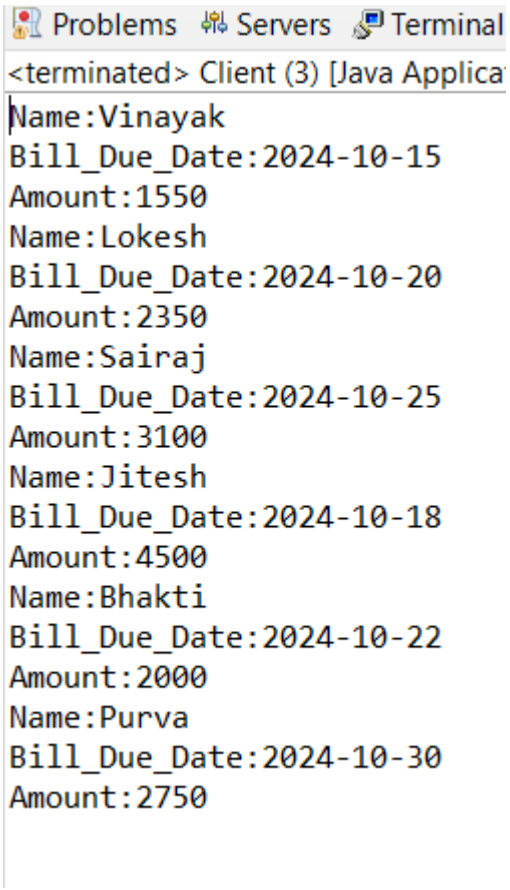
Creating table:

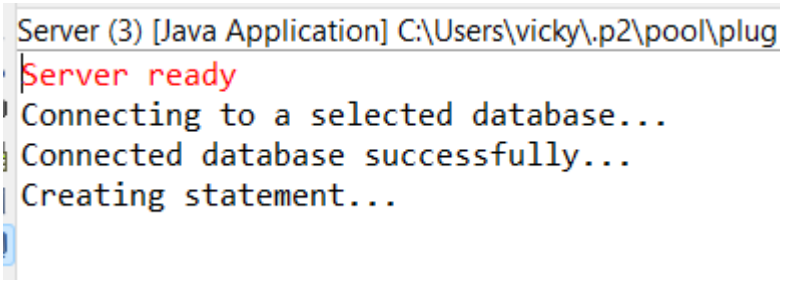
```
create table BillInfo(
cname varchar(10),
bill_due_date date,
bill_amount int
);
```

Inserting data:

```
insert into BillInfo values('Vinayak','2024-10-15',1550);
insert into BillInfo values('Lokesh','2024-10-20',2350);
insert into BillInfo values('Sairaj','2024-10-25',3100);
insert into BillInfo values('Jitesh','2024-10-18',4500);
insert into BillInfo values('Bhakti','2024-10-22',2000);
insert into BillInfo values('Purva','2024-10-30',2750);
```

Output: -


Problems
Servers
Terminal
<terminated> Client (3) [Java Applica
Name:Vinayak
Bill_Due_Date:2024-10-15
Amount:1550
Name:Lokesh
Bill_Due_Date:2024-10-20
Amount:2350
Name:Sairaj
Bill_Due_Date:2024-10-25
Amount:3100
Name:Jitesh
Bill_Due_Date:2024-10-18
Amount:4500
Name:Bhakti
Bill_Due_Date:2024-10-22
Amount:2000
Name:Purva
Bill_Due_Date:2024-10-30
Amount:2750


Server (3) [Java Application] C:\Users\vicky\.p2\pool\plug
Server ready
Connecting to a selected database...
Connected database successfully...
Creating statement...