ADVANCED JAVA

AMITY INSTITUTE OF INFORMATION TECHNOLOGY

LAB-3

Name : Priya Kumari
Course : Advanced java
Program/Semester : BCA – 6 'B'
Enrollment Number : A45304821056

Submitted to :-

Dr. Naveen Kumar Singh

Department:- Amity Institude Of Information Technology
Session:- 2021-24

HIBERNATE (THROUGH A SEPARATE XML FILE)

Problem Statement :

Hibernate program to demonstrate crud operations with the help of a separate xml file.

Introduction :

The objective of this proj ect is to develop a Hibernate program that performs CRUD (Create, Read, Update, Delete) operations on a table named "politics". The program will utilize two XML files for configuration and mapping purposes. The first XML file, hibernate.cfg.xml, will contain properties related to the database connection, including driver class name, URL username, password, dialect, and a mapping tag specifying the location of the second XMLT11e, leaders.hbm.xml. The second XML file, leaders.hbm.xml, will define the mapping between Java objects and the "politics" table, linking each attribute of the leader class to the corresponding columns in the table.

Problem Description:

The program will consist of the following functionalities:

1. Insert Record :

Users will have the option to insert a new record into the "politics" table. Upon selecting this option, the program will prompt the user to enter details about the political leader, including the leader's name, party affiliation, title, and a notable achievement. The entered information will then be added as a new record in the database.

2. Retrieve a Particular Record :

Users can retrieve a specific political leader by providing the leader's ID. If a leader with the provided ID exists in the "politics" table, the program will display the corresponding information, including the leader's name, party affiliation, title, and notable achievement. If the leader is not found, the program will display a message indicating that the leader with the entered ID is not found.

3. Retrieve All Records :

Users can retrieve all records from the "politics" table. The program will display information about all political leaders stored in the database, including their names, party affiliations, titles, and notable achievements.

4. Update Record :

Users will be able to update information about a particular political leader. Upon selecting this option, the program will prompt the user to enter the ID of the leader whose information they want to update. If the provided ID corresponds to a leader in the database, the program will allow the user to modify the leader's party affiliation. If the leader is not found, the program will display a message indicating that the leader with the entered ID is not found.

5. Delete Record :
Users can delete a record of a political leader from the "politics" table by providing the leader's ID. If the provided ID matches a leader in the database, the program will delete the corresponding record. If the leader is not found, the program will display a message indicating that the leader with the entered ID is not found.

6. Exit Program :
Users will have the option to exit the program. Upon selecting this option, the program will close the Hibernate session and factory, allowing users to exit the program gracefully.

Implementation Approach:

The program will be implemented using Hibernate, a popular object-relational mapping (ORM) framework for Java. Hibernate will handle the mapping between Java objects and the database tables, utilizing a separate mapping file to define the mapping configuration. The program will provide a userfriendly menu interface for interacting with the database, ensuring ease of use and clarity for users. Each CRUD operation will be implemented as a separate method or class, following a modular and object-oriented approach.

Expected Input and Output:

1. Insert Record :
User inputs : Leader's name, party affiliation, title, notable achievement.

2. Retrieve a Particular Record :
User inputs : Leader's ID.
Output : Information about the specified leader if found, or a message indicating that the leader with the entered ID is not found.

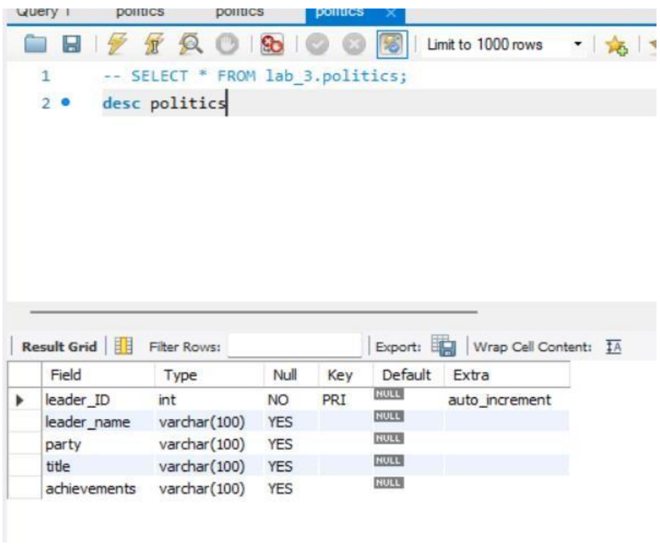
3. Retrieve All Records :
Output : Information about all political leaders stored in the database.

4. Update Record :
User inputs : Leader's ID and new party name.
Output : Confirmation message upon successful update, or a message indicating that the leader with the entered ID is not found.

5. Delete Record :
User inputs : Leader's ID.
Output : Confirmation message upon successful deletion, or a message indicating that the leader with the entered ID is not found.

6. Exit Program :
Output : Termination of the program.

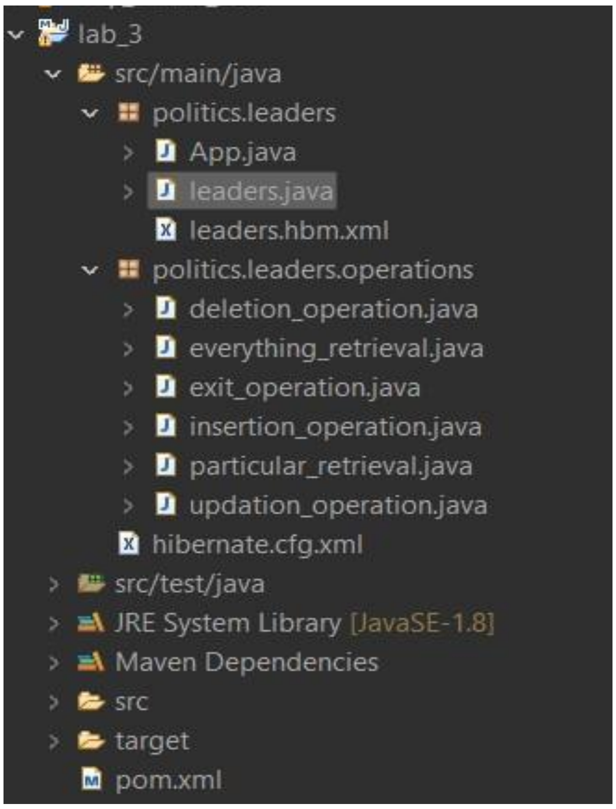
Describe table:


Limit to 1000 rows

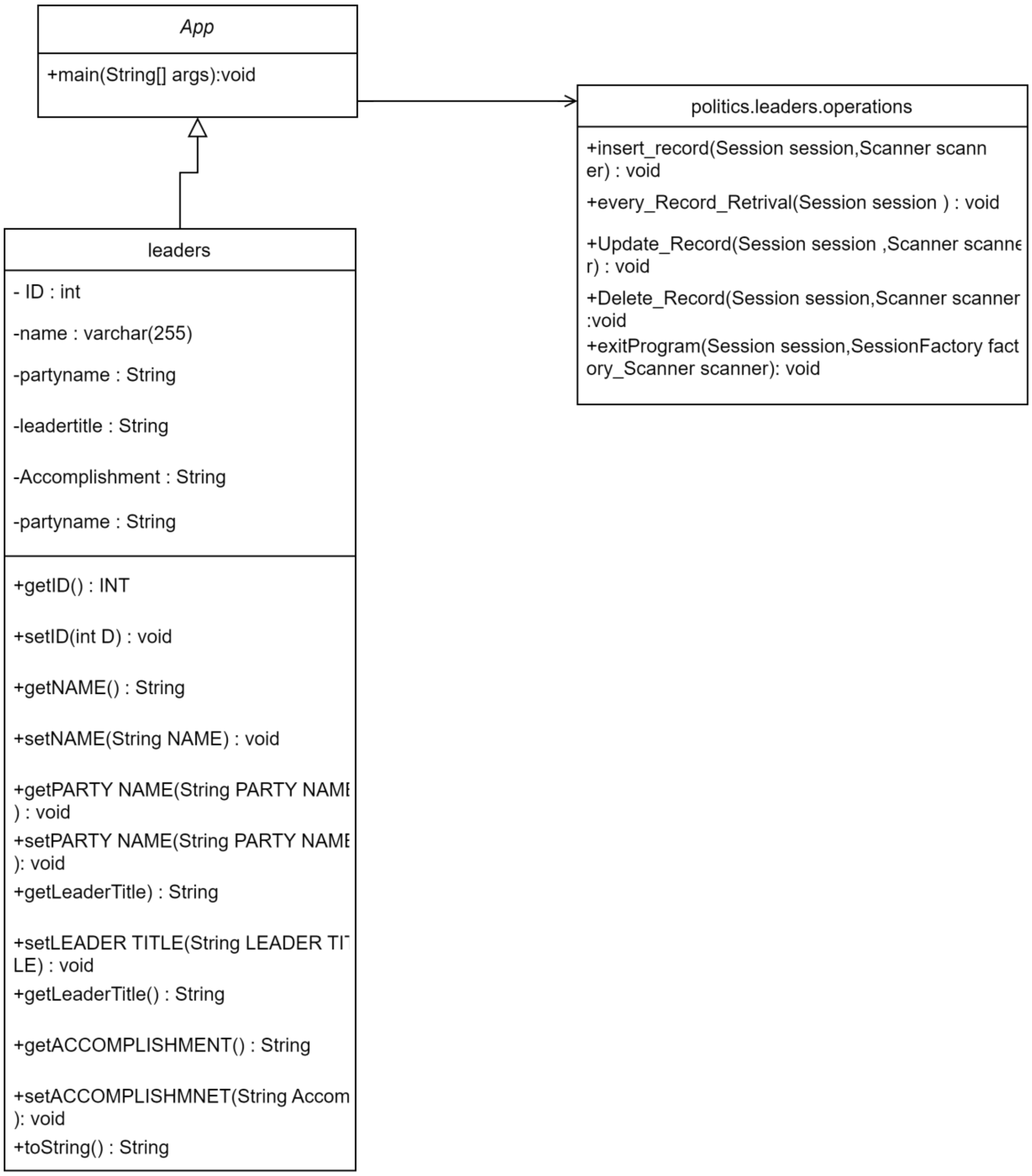
```
-- SELECT * FROM lab_3.politics;
desc politics
```

Field	Type	Null	Key	Default	Extra
leader_ID	int	NO	PRI	NULL	auto_increment
leader_name	varchar(100)	YES		NULL	
party	varchar(100)	YES		NULL	
title	varchar(100)	YES		NULL	
achievements	varchar(100)	YES		NULL	

Folder Structure


```
lab_3
  src/main/java
    politics.leaders
      App.java
      leaders.java
      leaders.hbm.xml
    politics.leaders.operations
      deletion_operation.java
      everything_retrieval.java
      exit_operation.java
      insertion_operation.java
      particular_retrieval.java
      updation_operation.java
    hibernate.cfg.xml
  src/test/java
  JRE System Library [JavaSE-1.8]
  Maven Dependencies
  src
  target
  pom.xml
```

Class Diagram



App
+main(String[] args):void
politics.leaders.operations
+insert_record(Session session,Scanner scann er) : void
+every_Record_Retrival(Session session) : void
+Update_Record(Session session ,Scanner scanne r) : void
+Delete_Record(Session session,Scanner scanner :void
+exitProgram(Session session,SessionFactory fact ory_Scanner scanner): void
leaders
- ID : int
-name : varchar(255)
-partyname : String
-leadertitle : String
-Accomplishment : String
-partyname : String
+getID() : INT
+setID(int D) : void
+getNAME() : String
+setNAME(String NAME) : void
+getPARTY NAME(String PARTY NAME) : void
+setPARTY NAME(String PARTY NAME): void
+getLeaderTitle) : String
+setLEADER TITLE(String LEADER TI LE) : void
+getLeaderTitle() : String
+getACCOMPLISHMENT() : String
+setACCOMPLISHMNET(String Accom): void
+toString() : String

CODE

```
package politics.leaders;

import org.hibernate.Session;

import org.hibernate.SessionFactory; import

org.hibernate.cfg.Configuration; import

java.util.Scanner; import

politics.leaders.operations.*;

 public class App

{     public static void main( String[] args )

    {

      Configuration config = new Configuration();          config.configure();

         SessionFactory factory = config.buildSessionFactory();

         Session session = factory.openSession();

         Scanner scanner = new Scanner(System.in);

         //for insertion          insertion_operation I = new insertion_operation();

         //for retrieval of any particular record          particular_retrieval P = new

particular_retrieval();

         //for retrieval of each and every record          everything_retrieval E = new

everything_retrieval();

         //Updating any record from the table          updation_operation U = new

updation_operation();

         //Deleting any record from the table          deletion_operation D = new

deletion_operation();

         //For exiting the program          exit_operation Exit = new

exit_operation();          try          {          while(true)

         {

                    System.out.println("\nChoose the operation you want to perform:");
                System.out.println("1. Insert a record into the table");

                System.out.println("2. Retrieve a particular record from the table");

                System.out.println("3. Retrieve all the records from the table");
```

```
              System.out.println("4. Update a record in the table");
              System.out.println("5. Delete a record from the table");
              System.out.println("6. Exit");
              System.out.print("\nPlease enter your choice: ");               int choice =
scanner.nextInt();
                         switch(choice)
              {                case 1 : I.insert_Record(session, scanner);
break;
              case 2 : P.single_Record_Retrieval(session, scanner);
break;                    case 3 : E.every_Record_Retrieval(session);
break;                case 4 : U.Update_Record(session, scanner);
break;                case 5 : D.Delete_Record(session, scanner);
break;
              case 6 : Exit.exitProgram(session, factory, scanner);                    break;
             default : System.out.println("\nInvalid choice. Please try again.\n");
              }
    }
    }
    finally        {
session.close();            factory.close();
scanner.close();
    }
  }
}
```

Leaders.java

```
  package politics.leaders;
  public class leaders {
```

CODE

```
private int ID; private String NAME;

private String PARTY_NAME; private

String LEADER_TITLE; private String

ACCOMPLISHMENT;

 public leaders() {

super();

 // TODO Auto-generated constructor stub

}   public leaders(int iD, String nAME, String pARTY_NAME, String lEADER_TITLE,

String aCCOMPLISHMENT) {  super();  ID = iD;

 NAME = nAME;

 PARTY_NAME = pARTY_NAME;
 LEADER_TITLE = lEADER_TITLE;

 ACCOMPLISHMENT = aCCOMPLISHMENT;

}  public int getID() {

return ID;

}  public void setID(int iD) {

 ID = iD;

}  public String getNAME() {

return NAME;

}  public void setNAME(String nAME) {

 NAME = nAME;

}  public String getPARTY_NAME() {

return PARTY_NAME;

}  public void setPARTY_NAME(String pARTY_NAME) {

 PARTY_NAME = pARTY_NAME;

}  public String getLEADER_TITLE() {

return LEADER_TITLE;

}
 public void setLEADER_TITLE(String lEADER_TITLE) {
```

CODE

```
 LEADER_TITLE = lEADER_TITLE;
}  public String getACCOMPLISHMENT() {
return ACCOMPLISHMENT;
}  public void setACCOMPLISHMENT(String aCCOMPLISHMENT) {
 ACCOMPLISHMENT = aCCOMPLISHMENT;
}
 @Override public String toString() {  return "leaders [ID=" + ID + ", NAME=" + NAME + ",
PARTY_NAME="
+ PARTY_NAME + ", LEADER_TITLE=" + LEADER_TITLE
                              + ", ACCOMPLISHMENT=" + ACCOMPLISHMENT + "]";
}


}
```

insertion operation.java

```
  package politics . leaders . operations;
   import java. util . Scanner;
   import org . hibernate . Session;
   import org . hibernate . Transaction;
   import politics . leaders . leaders;
public class insertion operation {

public void insert Record Session session, Scanner scanner)

new leaders ( ) ; session . begin Transaction ( );

System . out . print In ( " \nPerforming INSERT
```

CODE

```
                    System. out. print In ("\nPlease enter the name of

                        String name scanner . next ( ) ;
                    System. out . print In ("Please enter the name of the
party, the leader is associated with-");

                        String party scanner . next ( ) •
                    System. out. print In ("Please enter the title of the
leader in the party-");

                        String title scanner. next ( ) •

                    System. out. print In ("Please enter any one major
achievement of the leader_"\ .

String achieve scanner . next ( ) ; System. out . print In ( ) ; setNAME (name) ;
```

particular_retrival.java

```
package politics.leaders.operations;


import java.util.Scanner;

import org.hibernate.Session;

import org.hibernate.Transaction;

 import politics.leaders.leaders;

 public class particular_retrieval {

 public void single_Record_Retrieval(Session session, Scanner scanner)

        {

                Transaction transaction = session.beginTransaction();           leaders

Leader = new leaders();

                System.out.println("\nRetrieving a particular leader from the table based on the
id of the leader......");

        System.out.println("\nEnter the id of the leader who's information you want to

retrieve - ");                int retrieved_id = scanner.nextInt();

                    System.out.println();

                Leader = session.get(leaders.class, retrieved_id);          if (Leader !=

null)
```

```
                    {
                            System.out.println("Leader - " + Leader);
                    }
          else              {


     System.out.println("Leader with ID " + retrieved_id + " is not available.\n");
          }              transaction.commit();
          }
}
```

package

Everything_retrival.java

```
politics.leaders.operations;
 import java.util.List; import
org.hibernate.Session; import
org.hibernate.Transaction;
import
org.hibernate.query.Query;
import
politics.leaders.leaders;
 public class everything_retrieval {
  public void every_Record_Retrieval(Session session)
          {
                              Transaction transaction = session.beginTransaction();
  System.out.println("\nRetrieving everything from the table......\n");
               Query<leaders> query = session.createQuery("FROM leaders", leaders.class);
               List<leaders> leaders_list = query.list();                  for (leaders
leader : leaders_list)
                    {
```

```
                                System.out.println("Leader - " + leader);
                        }
                            transaction.commit();
            }
}
```

updation operation . java

```
package politics.leaders.operations;
 import java.util.Scanner; import
org.hibernate.Session; import
org.hibernate.Transaction; import
politics.leaders.leaders;
 public class updation_operation {
        public void Update_Record(Session
session, Scanner scanner)

            {
                    Transaction transaction = session.beginTransaction();                    leaders Leader
= new leaders();
                    System.out.println("\nUpdating the party of the leader......");
                    System.out.println("\nEnter the id of the leader who's party you want to update -
");
                            int id = scanner.nextInt();
                    Leader = session.get(leaders.class, id);                    if(Leader != null)
                        {
  System.out.println("\nEnter the name of the new party for the leader -");
                            String new_party = scanner.next();
```

CODE

```
                System.out.println();              Leader.setPARTY_NAME(new_party);
        session.saveOrUpdate(Leader);
                System.out.println("Party - " + Leader + " updated successfully.\n");
                    }
                    else
                    {
    System.out.println("\nLeader with ID " + id + " is not found.\n");
                    }
                        transaction.commit();
            }
}
```

deletion operation . java

```
package politics.leaders.operations;
 import java.util.Scanner; import
org.hibernate.Session; import
org.hibernate.Transaction; import
politics.leaders.leaders;
 public class deletion_operation {              public void Delete_Record(Session session,
Scanner scanner)
        {
                Transaction transaction = session.beginTransaction();                    leaders Leader
= new leaders();
  System.out.println("\nDeleting some record from the table......");
  System.out.println("\nEnter the id of the leader you want to delete -");
                        int id = scanner.nextInt();
                       System.out.println();
```

CODE

```
                Leader = session.get(leaders.class, id);                    if

(Leader != null)

                  {

                        session.delete(Leader);

                System.out.println("Record - " + Leader + " deleted successfully.\n");

                  }

                   else

                  {

  System.out.println("\nLeader with ID " + id + " is not found.\n");

                  }

                       transaction.commit();


            }
```

```
 <?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
 "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

 <hibernate-configuration>
 <session-factory>

       <property name="connection.driver_class">com.mysql.cj.jdbc.Driver</proper
ty>
       <property name="connection.url">jdbc:mysql://localhost:3306/lab_3</proper ty>
       <property name="connection.username">root</property>
       <property name="connection.password">Mysql@2024</property>
       <property name="dialect">org.hibernate.dialect.MySQLDialect</property>
       <property name="current_session_context_class">thread</property>
       <property name="cache.provider_class">org.hibernate.cache.internal.NoCach
eProvider</property>
       <property name="show_sql">true</property>
       <property name="hbm2ddl.auto">update</property>
     <mapping resource="politics/leaders/leaders.hbm.xml"/>
```

```
            </session-factory>

         </hibernate-configuration>
```

pom.Xml

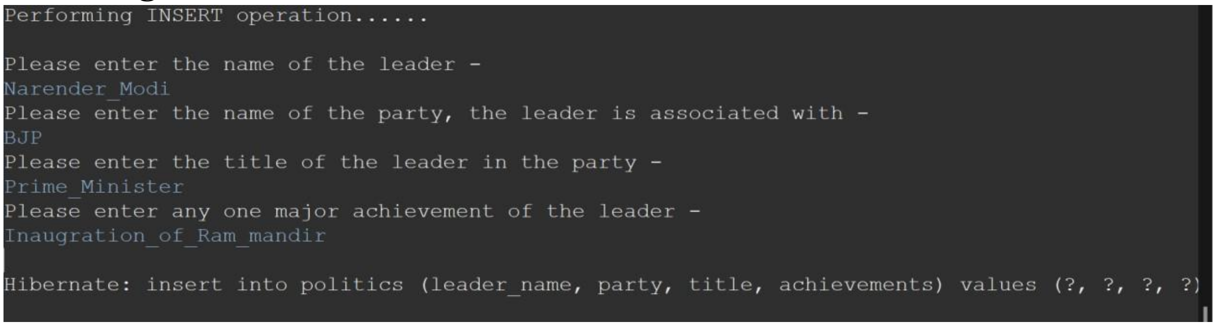
```
<project xmlns="http://maven.apache.org/POM/4.0.0"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
             <modelVersion>4.0.0</modelVersion>

             <groupId>politics</groupId>
               <artifactId>leaders</artifactId>
            <version>0.0.1-SNAPSHOT</version>
            <packaging>jar</packaging>

          <name>leaders</name>
              <url>http://maven.apache.org</url>

          <properties>
                     <project.build.sourceEncoding>UTF-
8</project.build.sourceEncoding>
           </properties>

          <dependencies>
                 <dependency>
                        <groupId>junit</groupId>
                          <artifactId>junit</artifactId>
                        <version>3.8.1</version>
                       <scope>test</scope>
                 </dependency>
              <!-- https://mvnrepository.com/artifact/mysql/mysqlconnector-java -->
                 <dependency>
                       <groupId>mysql</groupId>
                           <artifactId>mysql-connector-java</artifactId>
                        <version>8.0.33</version>
                 </dependency>
                 <!--
```

CODE

```
https://mvnrepository.com/artifact/org.hibernate/hibernate-core -->
                    <dependency>
                                  <groupId>org.hibernate</groupId>
                                      <artifactId>hibernate-core</artifactId>
                                  <version>5.4.5.Final</version>
                    </dependency>
            </dependencies>
</project>
```

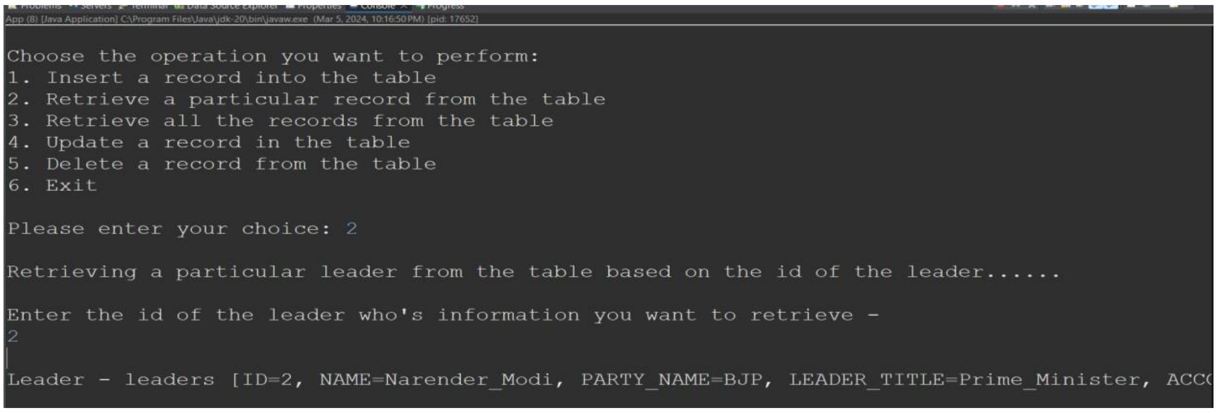
INPUT/OUTPUT

Inserting into database



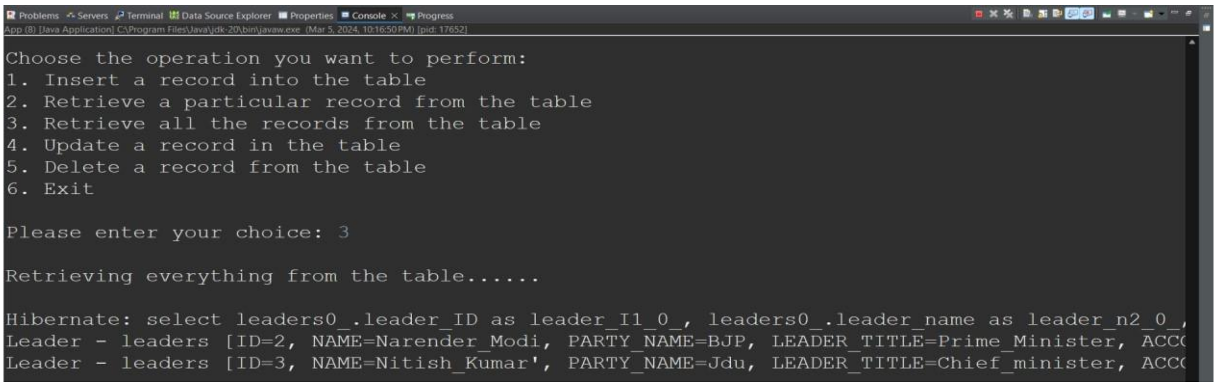
```
Performing INSERT operation......

Please enter the name of the leader -
Narender_Modi
Please enter the name of the party, the leader is associated with -
BJP
Please enter the title of the leader in the party -
Prime_Minister
Please enter any one major achievement of the leader -
Inaugration_of_Ram_mandir

Hibernate: insert into politics (leader_name, party, title, achievements) values (?, ?, ?, ?)
```

Retrieving a particular record from the table



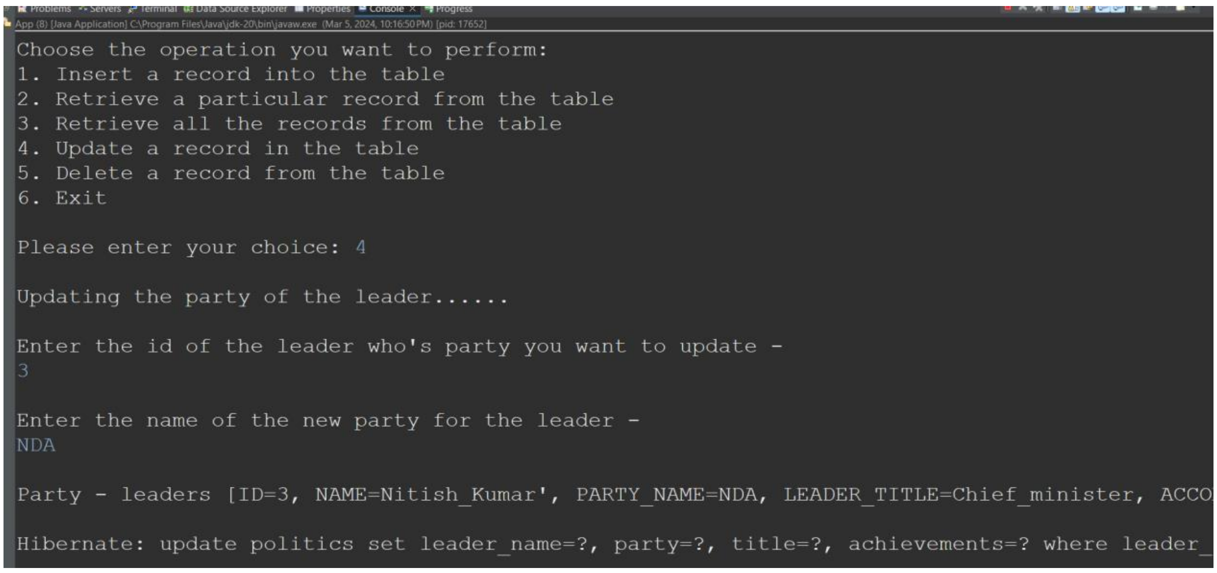
```
App (8) [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (Mar 5, 2024, 10:16:50 PM) [pid: 17652]

Choose the operation you want to perform:
1. Insert a record into the table
2. Retrieve a particular record from the table
3. Retrieve all the records from the table
4. Update a record in the table
5. Delete a record from the table
6. Exit

Please enter your choice: 2

Retrieving a particular leader from the table based on the id of the leader......

Enter the id of the leader who's information you want to retrieve -
2

Leader - leaders [ID=2, NAME=Narender_Modi, PARTY_NAME=BJP, LEADER_TITLE=Prime_Minister, ACC
```

Retrieving a all the record from the table



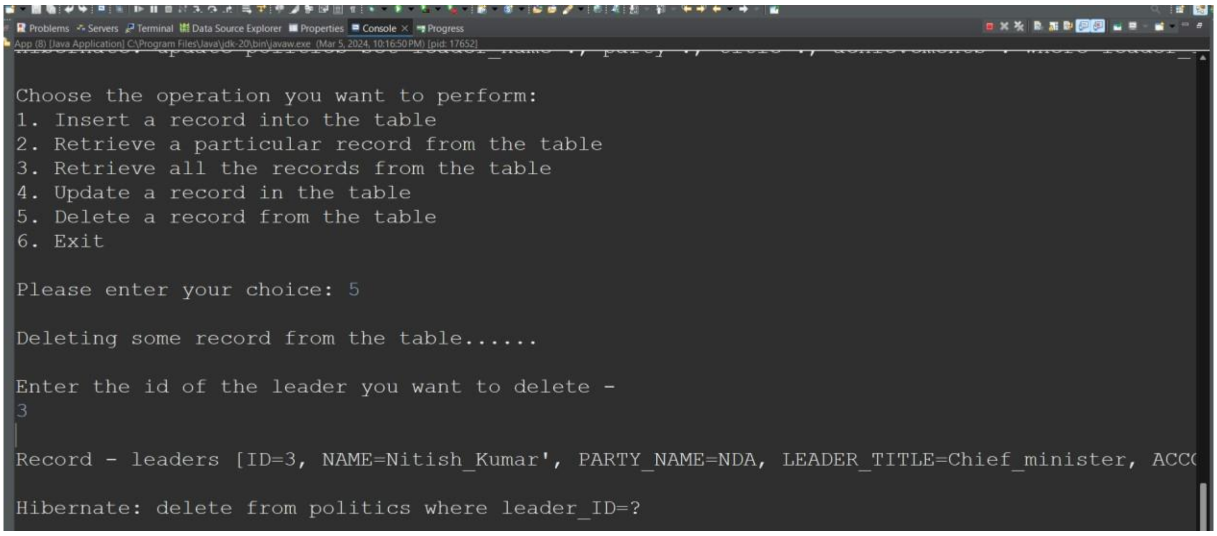
```
App (8) [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (Mar 5, 2024, 10:16:50 PM) [pid: 17652]
Choose the operation you want to perform:
1. Insert a record into the table
2. Retrieve a particular record from the table
3. Retrieve all the records from the table
4. Update a record in the table
5. Delete a record from the table
6. Exit

Please enter your choice: 3

Retrieving everything from the table......

Hibernate: select leaders0_.leader_ID as leader_I1_0_, leaders0_.leader_name as leader_n2_0_,
Leader - leaders [ID=2, NAME=Narender_Modi, PARTY_NAME=BJP, LEADER_TITLE=Prime_Minister, ACC
Leader - leaders [ID=3, NAME=Nitish_Kumar', PARTY_NAME=Jdu, LEADER_TITLE=Chief_minister, ACC
```

Updating any record from the table



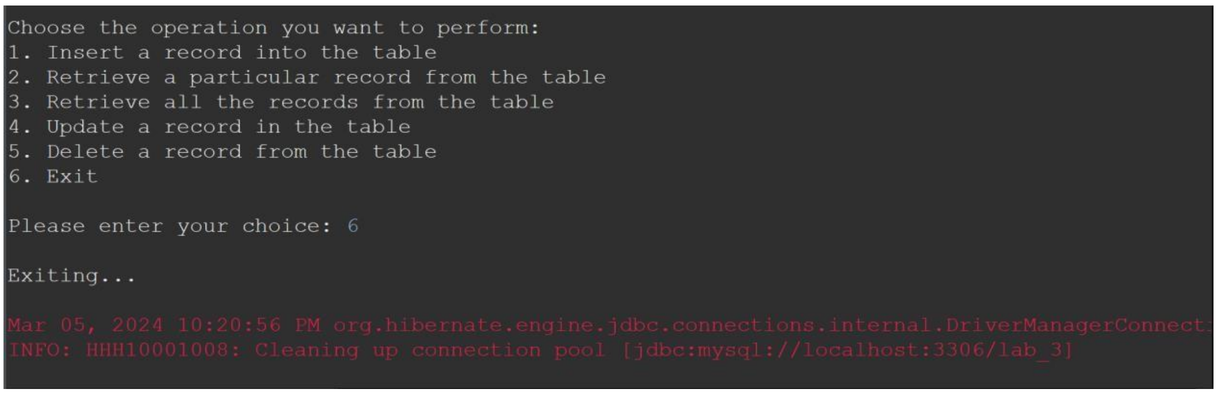
```
App (8) [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (Mar 5, 2024, 10:16:50 PM) [pid: 17652]
Choose the operation you want to perform:
1. Insert a record into the table
2. Retrieve a particular record from the table
3. Retrieve all the records from the table
4. Update a record in the table
5. Delete a record from the table
6. Exit

Please enter your choice: 4

Updating the party of the leader......

Enter the id of the leader who's party you want to update -
3

Enter the name of the new party for the leader -
NDA

Party - leaders [ID=3, NAME=Nitish_Kumar', PARTY_NAME=NDA, LEADER_TITLE=Chief_minister, ACCO

Hibernate: update politics set leader_name=?, party=?, title=?, achievements=? where leader_
```

Deleting any record from the table



```
App (8) [Java Application] C:\Program Files\Java\jdk-20\bin\javaw.exe (Mar 5, 2024, 10:16:50 PM) [pid: 17652]

Choose the operation you want to perform:
1. Insert a record into the table
2. Retrieve a particular record from the table
3. Retrieve all the records from the table
4. Update a record in the table
5. Delete a record from the table
6. Exit

Please enter your choice: 5

Deleting some record from the table......

Enter the id of the leader you want to delete -
3

Record - leaders [ID=3, NAME=Nitish_Kumar', PARTY_NAME=NDA, LEADER_TITLE=Chief_minister, ACCO

Hibernate: delete from politics where leader_ID=?
```

Exiting the program



```
Choose the operation you want to perform:
1. Insert a record into the table
2. Retrieve a particular record from the table
3. Retrieve all the records from the table
4. Update a record in the table
5. Delete a record from the table
6. Exit

Please enter your choice: 6

Exiting...

Mar 05, 2024 10:20:56 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnect
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost:3306/lab_3]
```