HIBERNATE FRAMEWORK

- Hibernate is framework, which is used to store data to database.
- It has ORM Tool, Object Relational Model to perform CRUD Operation.
- Due to Drawbacks of JDBC, we used Hibernate Framework
- Drawbacks of JDBC is,
  In JDBC we can’t use OOP’s Concepts
  In JDBC we can’t Create Object
  In JDBC we need write query for database, table.
  In JDBC doesn’t Support ORM Tool
- Due to Drawbacks of JDBC we are using Hibernate Framework.
- In Hibernate we can use OOP’s Concepts
- In Hibernate we can Create Object
- In Hibernate will generate table and Id it’s automatically.
- In Hibernate have in-built methods and dependencies are there.
- It is simple and store data to database.
- We used store data to database we have MySQL Workbench.

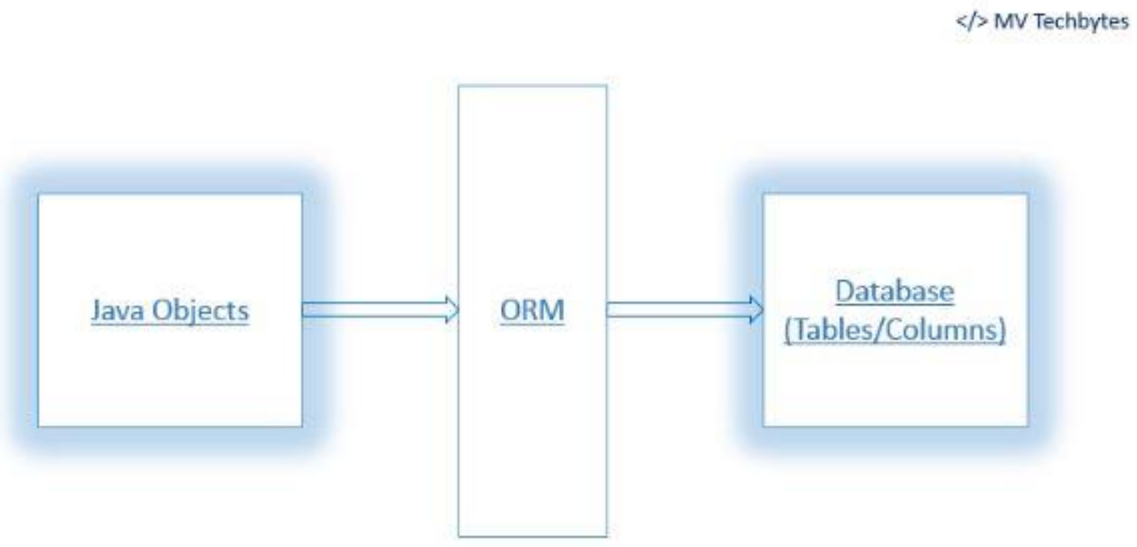
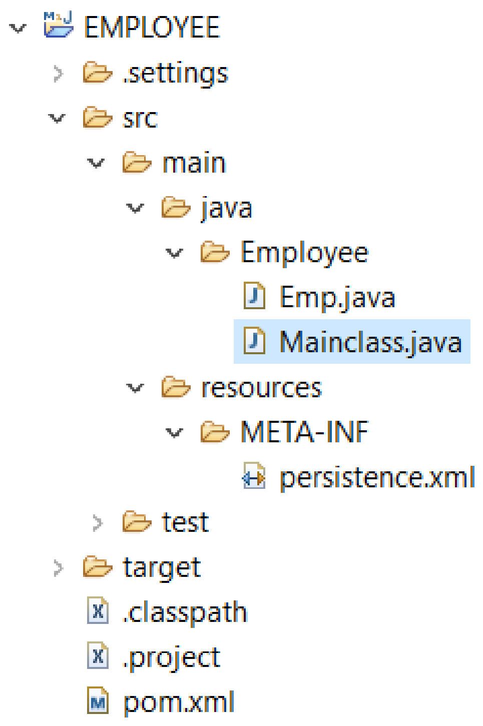


Fig. Hibernate flow diagram.

STEPS TO CREATE HIBERNATE PROJECT:

Step1: Open Eclipse application, create maven project. While creating Project don't forget to click the checkbox for create a sample project for (maven archetype). And Add Dependencies.



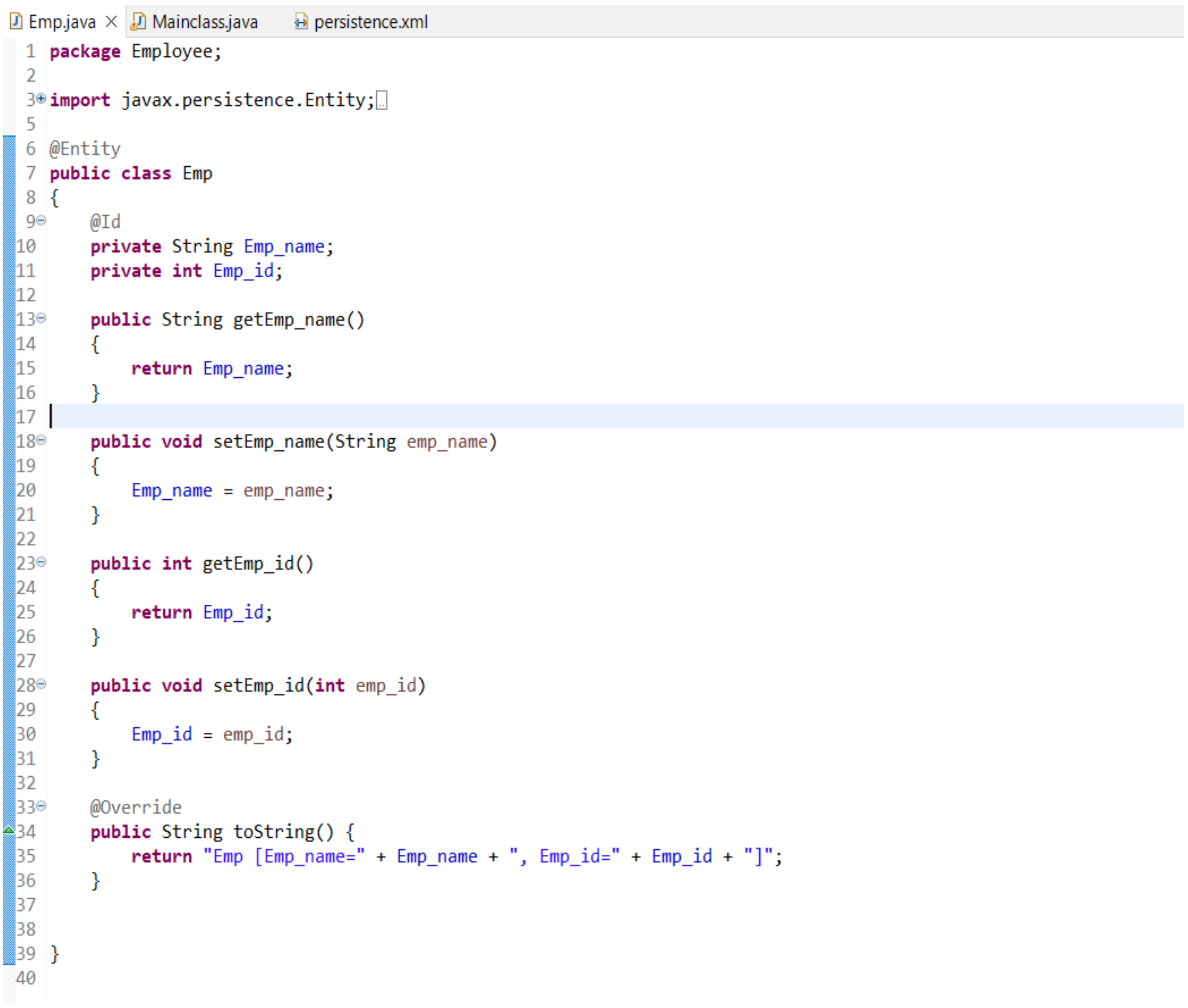
```
EMPLOYEE
  .settings
  src
    main
      java
        Employee
          Emp.java
          Mainclass.java
      resources
        META-INF
          persistence.xml
    test
  target
  .classpath
  .project
  pom.xml
```

Step2: In src/main/resource, create a folder META-INF, inside folder create xml file persistence.xml.

```
<persistence xmlns="http://xmlns.jcp.org/xml/ns/persistence"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence
  http://xmlns.jcp.org/xml/ns/persistence/persistence_2_1.xsd"
      version="2.1">

      <persistence-unit name="dev">
            <provider>org.hibernate.jpa.HibernatePersistenceProvider</provider>
            <shared-cache-mode>ENABLE_SELECTIVE</shared-cache-mode>      <!-- for
caching -->
            <properties>
                  <property name="javax.persistence.jdbc.driver"
                        value="com.mysql.cj.jdbc.Driver" />
                  <property name="javax.persistence.jdbc.url"
                        value="jdbc:mysql://localhost:3306/employee" />
                  <property name="javax.persistence.jdbc.user"
                        value="root" />
                  <property name="javax.persistence.jdbc.password"
                        value="root" />
                  <property name="hibernate.show_sql" value="true" />
                  <property name="hibernate.hbm2ddl.auto" value="update" />
                  <property name="hibernate.dialect"
value="org.hibernate.dialect.MySQL8Dialect"/>


            </properties>
      </persistence-unit>
</persistence>
```

Step3: Open MySQL Work bench application and minimize it.

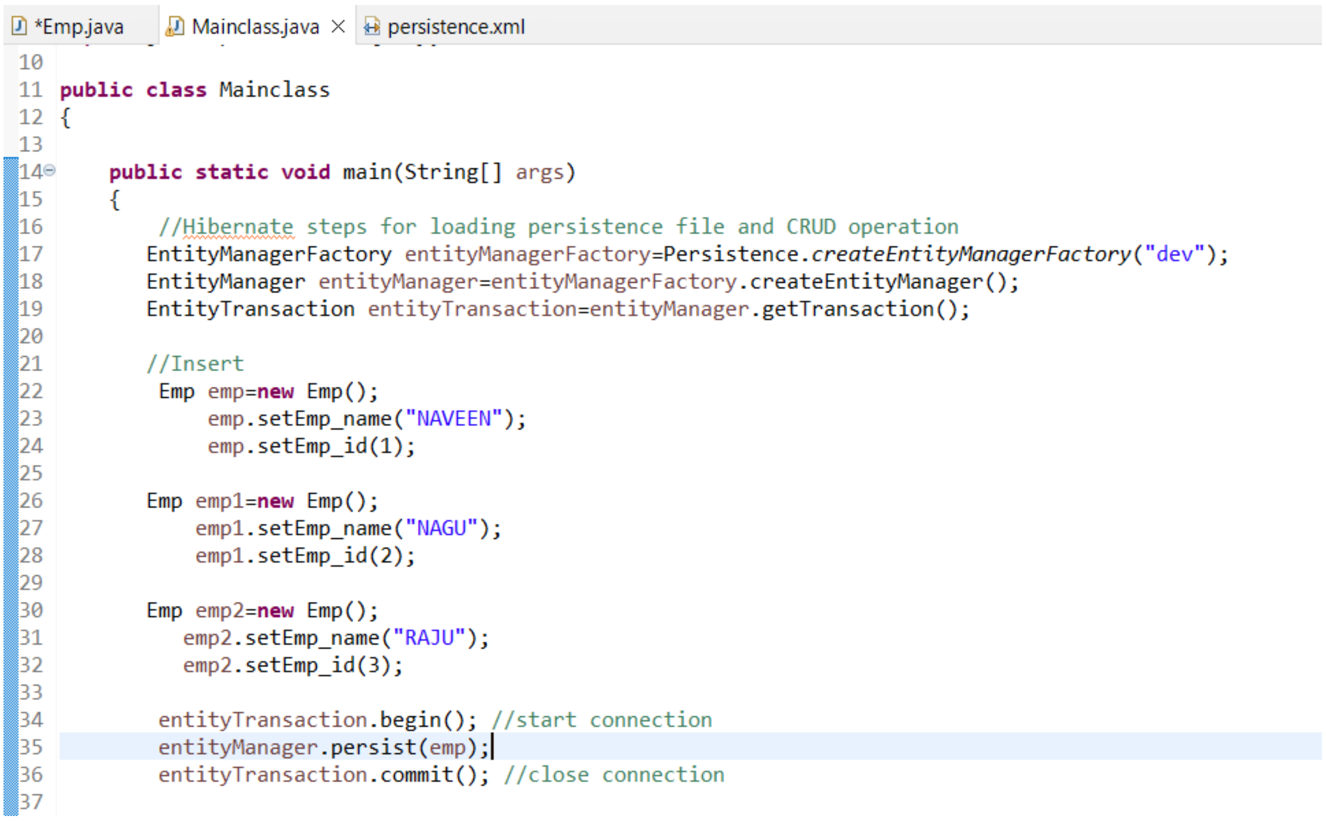
Step4: In eclipse src/main/java create class emp



```java
package Employee;

import javax.persistence.Entity;

@Entity
public class Emp
{
    @Id
    private String Emp_name;
    private int Emp_id;

    public String getEmp_name()
    {
        return Emp_name;
    }

    public void setEmp_name(String emp_name)
    {
        Emp_name = emp_name;
    }

    public int getEmp_id()
    {
        return Emp_id;
    }

    public void setEmp_id(int emp_id)
    {
        Emp_id = emp_id;
    }

    @Override
    public String toString() {
        return "Emp [Emp_name=" + Emp_name + ", Emp_id=" + Emp_id + "]";
    }


}
```

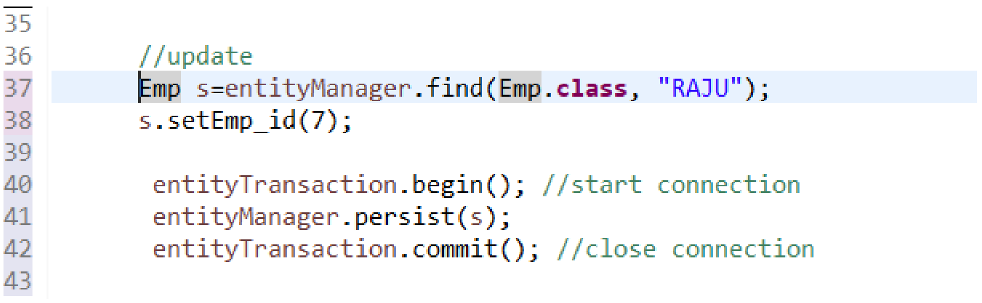
Step5: In eclipse src/main/java create class Mainclass

1) **Insert data:**



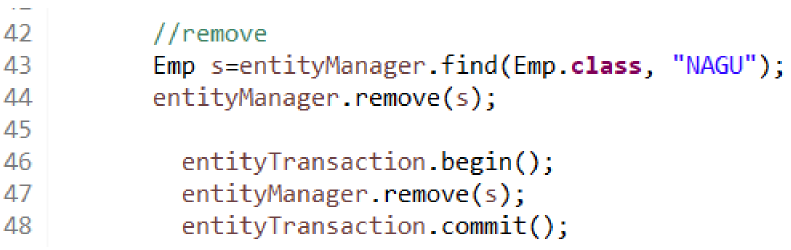
```

public class Mainclass
{

    public static void main(String[] args)
    {
        //Hibernate steps for loading persistence file and CRUD operation
       EntityManagerFactory entityManagerFactory=Persistence.createEntityManagerFactory("dev");
       EntityManager entityManager=entityManagerFactory.createEntityManager();
       EntityTransaction entityTransaction=entityManager.getTransaction();

       //Insert
        Emp emp=new Emp();
            emp.setEmp_name("NAVEEN");
            emp.setEmp_id(1);

       Emp emp1=new Emp();
           emp1.setEmp_name("NAGU");
           emp1.setEmp_id(2);

       Emp emp2=new Emp();
          emp2.setEmp_name("RAJU");
          emp2.setEmp_id(3);

        entityTransaction.begin(); //start connection
        entityManager.persist(emp);
        entityTransaction.commit(); //close connection

```

2) **Update data:**



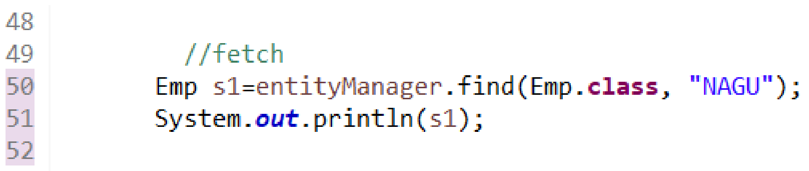
```

      //update
      Emp s=entityManager.find(Emp.class, "RAJU");
      s.setEmp_id(7);

       entityTransaction.begin(); //start connection
       entityManager.persist(s);
       entityTransaction.commit(); //close connection

```

3) **Delete data:**



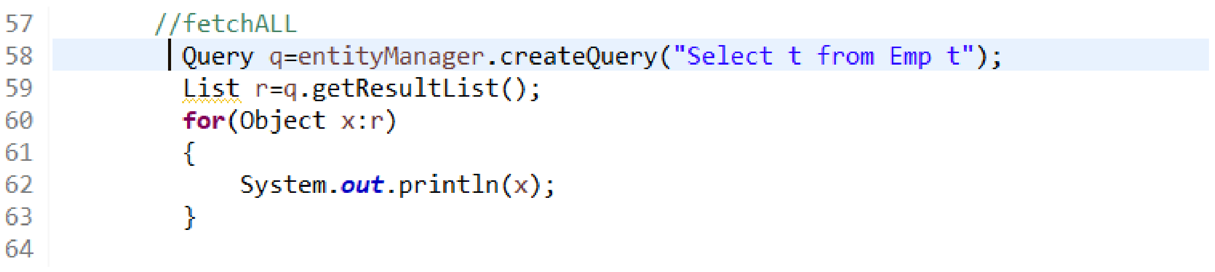
```
       //remove
       Emp s=entityManager.find(Emp.class, "NAGU");
       entityManager.remove(s);

         entityTransaction.begin();
         entityManager.remove(s);
         entityTransaction.commit();
```

4) Fetch Data:



```

    //fetch
  Emp s1=entityManager.find(Emp.class, "NAGU");
  System.out.println(s1);

```

5) Fetch All Data:



```
    //fetchALL
    Query q=entityManager.createQuery("Select t from Emp t");
    List r=q.getResultList();
    for(Object x:r)
    {
        System.out.println(x);
    }

```

MySQL Workbench

Database Schema

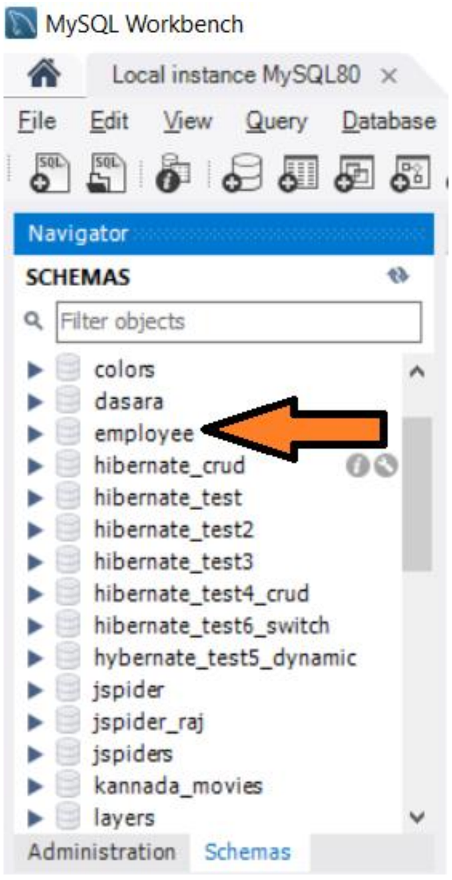
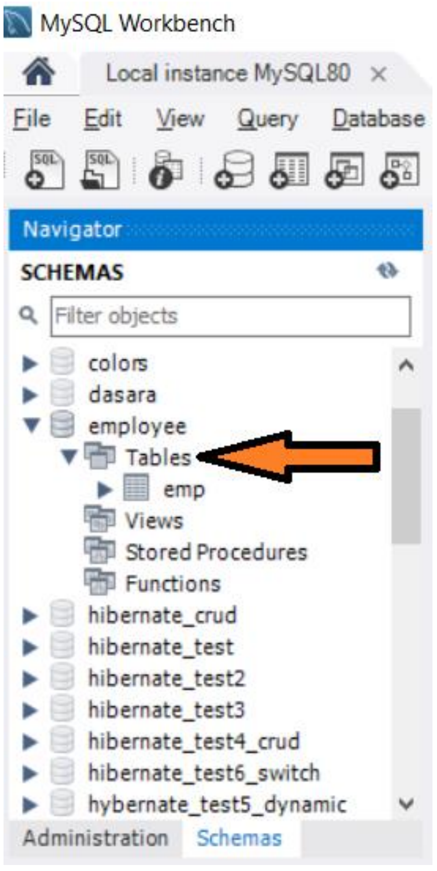
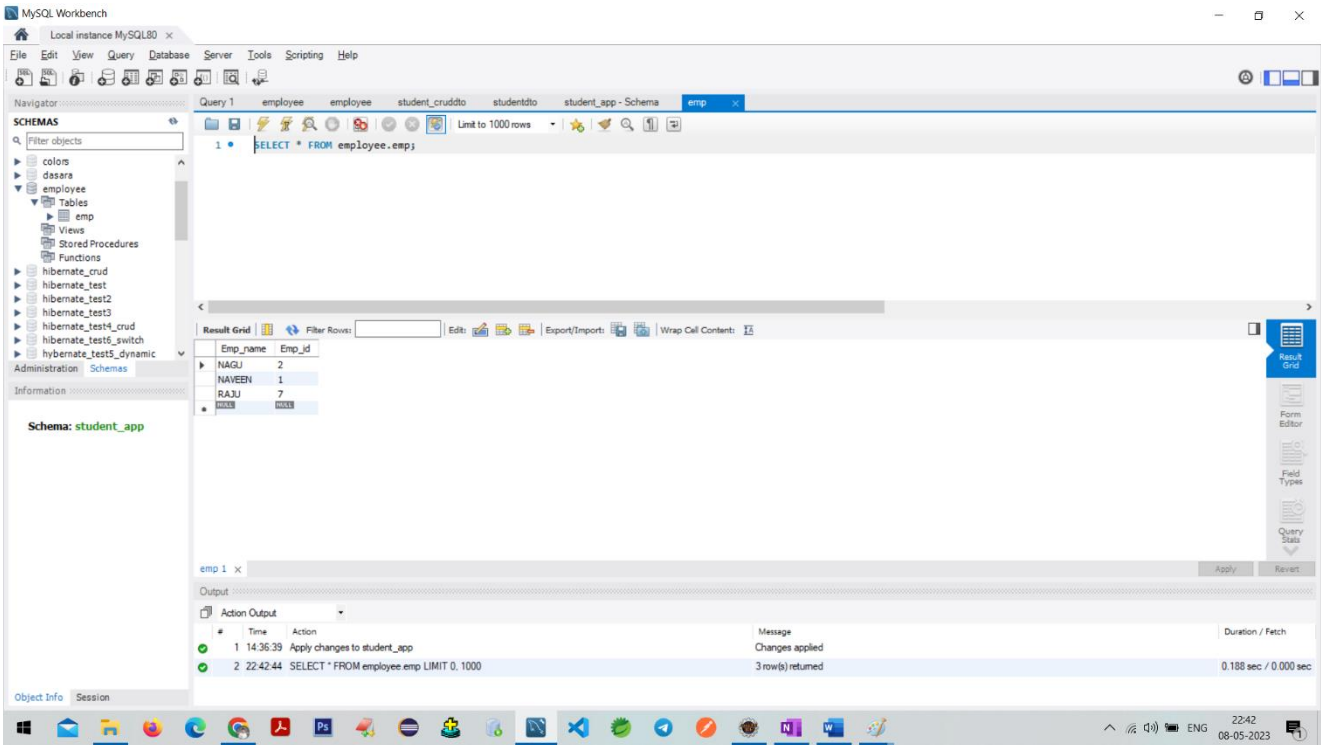


Table created in Database



Stored data and some CRUD operation performed in MySQL Workbench.



```
SELECT * FROM employee.emp;
```

Emp_name	Emp_id
NAGU	2
NAVEEN	1
RAJU	7
NULL	NULL

```
#  Time      Action                                     Message              Duration / Fetch
1  14:36:39  Apply changes to student_app               Changes applied
2  22:42:44  SELECT * FROM employee.emp LIMIT 0, 1000   3 row(s) returned    0.188 sec / 0.000 sec
```