Case Study Title: Citizen and Passport Management System

Business Context:

A national government agency maintains records of citizens and the passports issued to them. The rule of the system is:

- **Each citizen can hold exactly one passport**
- **Each passport must be assigned to only one citizen**

This kind of relationship is a textbook example of a One-to-One association, where one record in the Citizen table corresponds to one record in the Passport table, and vice versa.

Solution:

HibernetAssignment/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>com.example</groupId>
  <artifactId>CitizenPassportHibernate</artifactId>
  <version>1.0-SNAPSHOT</version>
  <dependencies>
    <dependency>
      <groupId>org.hibernate.orm</groupId>
      <artifactId>hibernate-core</artifactId>
      <version>6.1.7.Final</version>
    </dependency>
    <dependency>
      <groupId>jakarta.persistence</groupId>
```

```
    <artifactId>jakarta.persistence-api</artifactId>
    <version>3.1.0</version>
  </dependency>
  <dependency>
  <groupId>com.mysql</groupId>
  <artifactId>mysql-connector-j</artifactId>
  <version>8.0.33</version>
</dependency>
 </dependencies>
</project>
```

Citizen.java

```
package com.example.entity;
import jakarta.persistence.*;
@Entity
public class Citizen {
  @Id
  @GeneratedValue(strategy = GenerationType.IDENTITY)
  private int id;
  private String name;
  @OneToOne(cascade = CascadeType.ALL)
  @JoinColumn(name = "passport_id", referencedColumnName = "id")
  private Passport passport;
  public Citizen() {}
  public Citizen(String name, Passport passport) {
    this.name = name;
    this.passport = passport;
  }
```

```
    // Getters and Setters
    public int getId() { return id; }
    public void setId(int id) { this.id = id; }
    public String getName() { return name; }
    public void setName(String name) { this.name = name; }
    public Passport getPassport() { return passport; }
    public void setPassport(Passport passport) { this.passport = passport; }
}
```

Passport.java

```
package com.example.entity;
import jakarta.persistence.*;
@Entity
public class Passport {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;
    @Column(name = "passport_number", nullable = false, unique = true)
    private String passportNumber;
    public Passport() {}
    public Passport(String passportNumber) {
        this.passportNumber = passportNumber;
    }
    // Getters and Setters
    public int getId() { return id; }
    public void setId(int id) { this.id = id; }
```

```
    public String getPassportNumber() { return passportNumber; }
    public void setPassportNumber(String passportNumber) { this.passportNumber =
passportNumber; }
}
```

hibernate.cfg.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<hibernate-configuration>
    <session-factory>
        <property
        name="hibernate.connection.driver_class">com.mysql.cj.jdbc.Driver</property>
        <property
        name="hibernate.connection.url">jdbc:mysql://localhost:3306/citizen_passpo
        rt_db</property>
        <property name="hibernate.connection.username">root</property>
        <property name="hibernate.connection.password">root</property>
        <property
         name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>
        <property name="hibernate.hbm2ddl.auto">update</property>
        <property name="hibernate.show_sql">true</property>
        <property name="hibernate.format_sql">true</property>
        <mapping class="com.example.entity.Citizen"/>
        <mapping class="com.example.entity.Passport"/>
    </session-factory>
</hibernate-configuration>
```

HibernateUtil.java

```
package com.example.util;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;
public class HibernateUtil {
   private static final SessionFactory sessionFactory;
   static {
      try {
         sessionFactory = new Configuration()
               .configure("hibernate.cfg.xml")
               .buildSessionFactory();
       } catch (Throwable ex) {
         System.err.println("Initial SessionFactory creation failed." + ex);
         throw new ExceptionInInitializerError(ex);
      }
   }
   public static SessionFactory getSessionFactory() {
      return sessionFactory;
   }
}
```

App.java

```
package com.example.app;
import com.example.entity.Citizen;
import com.example.entity.Passport;
import com.example.util.HibernateUtil;
import org.hibernate.Session;
import org.hibernate.Transaction;
```

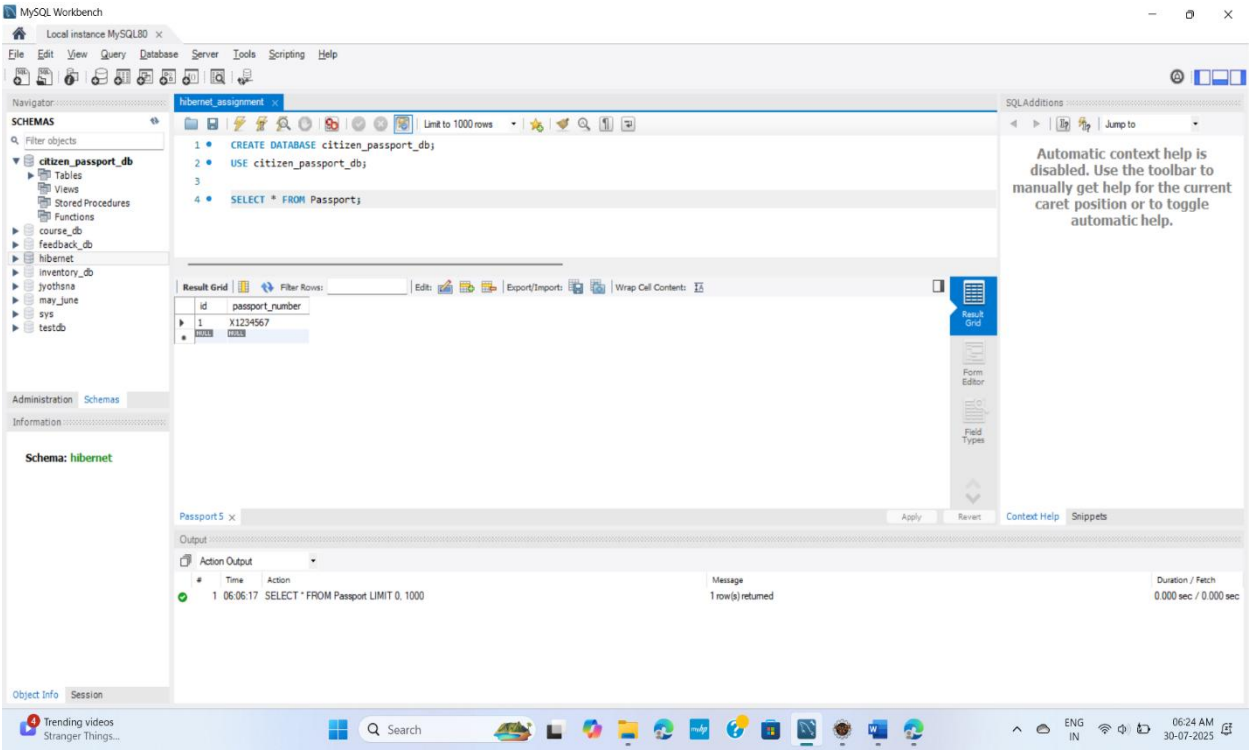
```
public class App {
    public static void main(String[] args) {
        Passport passport = new Passport("X1234567");
        Citizen citizen = new Citizen("John Doe", passport);
        Session session = HibernateUtil.getSessionFactory().openSession();
        Transaction tx = session.beginTransaction();
        session.persist(citizen);  // Cascade saves both Citizen and Passport
        tx.commit();
        session.close();
        System.out.println("Citizen and Passport saved successfully.");
    }
}
```

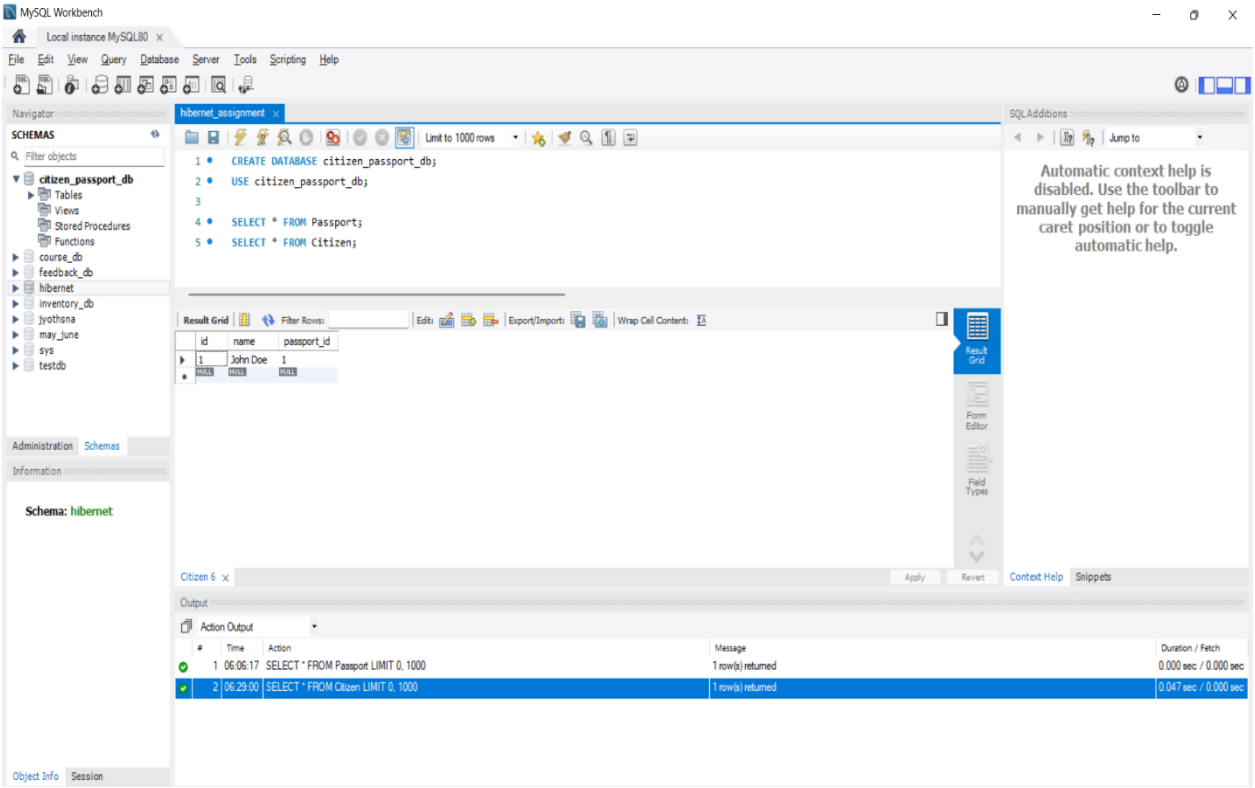
Output:

```
Hibernate:insert into Passport (passportNumber) values (?)
Hibernate:insert into Citizen (name,passport_id) values (?,?)
Citizen and Passport saved successfully.
```

SQL Query:

```
CREATE DATABASE citizen_passport_db;
USE citizen_passport_db;
SELECT * FROM Passport;
SELECT * FROM Citizen;
```


MySQL Workbench
Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
citizen_passport_db
Tables
Views
Stored Procedures
Functions
course_db
feedback_db
hibernet
inventory_db
jyothsna
may_june
sys
testdb
Administration Schemas
Information
Schema: hibernet
Object Info Session
hibernet_assignment
Limit to 1000 rows
1 • CREATE DATABASE citizen_passport_db;
2 • USE citizen_passport_db;
3
4 • SELECT * FROM Passport;
Result Grid
Filter Rows:
Edit:
Export/Import:
Wrap Cell Content:
id passport_number
1 X1234567
NULL NULL
Result Grid
Form Editor
Field Types
Passport 5
Apply
Revert
SQLAdditions
Jump to
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Context Help
Snippets
Output
Action Output
Time Action Message Duration / Fetch
1 06:06:17 SELECT * FROM Passport LIMIT 0, 1000 1 row(s) returned 0.000 sec / 0.000 sec
Trending videos
Stranger Things...
Search
ENG
IN
06:24 AM
30-07-2025


MySQL Workbench
Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
citizen_passport_db
Tables
Views
Stored Procedures
Functions
course_db
feedback_db
hibernet
inventory_db
jyothsna
may_june
sys
testdb
Administration Schemas
Information
Schema: hibernet
Object Info Session
hibernet_assignment
Limit to 1000 rows
1 • CREATE DATABASE citizen_passport_db;
2 • USE citizen_passport_db;
3
4 • SELECT * FROM Passport;
5 • SELECT * FROM Citizen;
Result Grid
Filter Rows:
Edit:
Export/Import:
Wrap Cell Content:
id name passport_id
1 John Doe 1
NULL NULL NULL
Result Grid
Form Editor
Field Types
Citizen 6
Apply
Revert
SQLAdditions
Jump to
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Context Help
Snippets
Output
Action Output
Time Action Message Duration / Fetch
1 06:06:17 SELECT * FROM Passport LIMIT 0, 1000 1 row(s) returned 0.000 sec / 0.000 sec
2 06:29:00 SELECT * FROM Citizen LIMIT 0, 1000 1 row(s) returned 0.047 sec / 0.000 sec