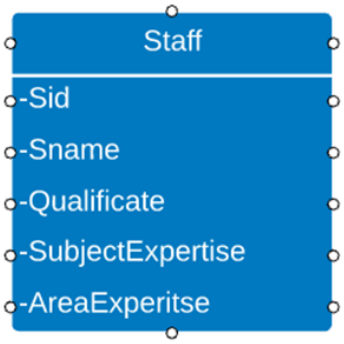
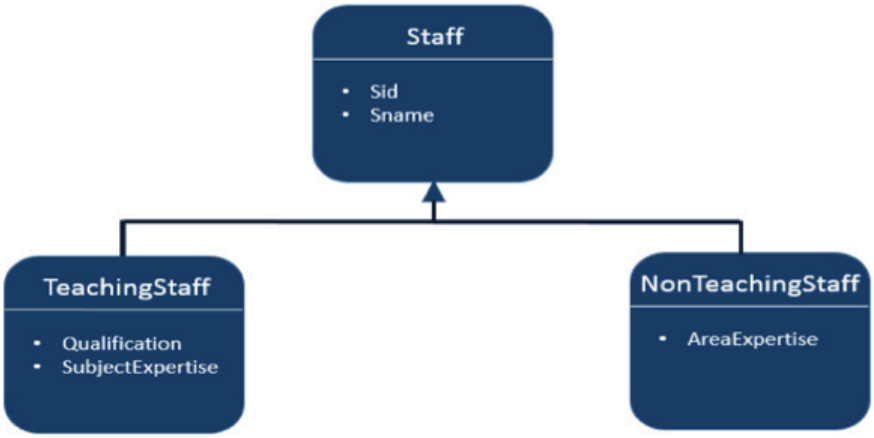
Hands-On Activity JPA-Mapping

Prerequisites

Create a single table name as "Staff" under classicmodels database. Add below column/ fields.

1. sid(Primary ID) 2. sname 3. areaexpertise 4. qualification 5. subjectexpertise	

Inheritance Strategies



- Inheritance is the core concept of object oriented language, therefore we can use inheritance relationships or strategies between entities. JPA supports three types of inheritance strategies such as **SINGLE_TABLE, JOINED_TABLE, and TABLE_PER_CONCRETE_CLASS.**
- Let us consider an example of **Staff, TeachingStaff, NonTeachingStaff classes** and their relationships as shown in above screenshot.

- Thinking about it in an object-oriented way…
  - ➢ Staff is the Parent
  - ➢ TeachingStaff and NonTeachngStaff are children, inheriting the attributes of Staff

Remind the concept of inheritance (is a mechanism of inheriting the properties of superclass by subclass) and therefore sid, sname are the fields which belong to both TeachingStaff and NonTeachingStaff.

Begin Activity

Single Table Example

Create a JPA project by the name of JPA-MAPPING-SINGLE or as you like, and Configure JPA Jars and MariaDB JDBC Jars

Single Table strategy: Single-Table strategy takes all classes fields (both super and subclasses) and maps them down into a single table known as SINGLE_TABLE strategy. Here discriminator value plays a key role in differentiating the values of three entities in one table.

1. **Creating Entities**

a) Create a package named **'com.test.entity'** under the 'src' package. Create a new java class named **Staff.java** under the given package. The Staff entity class is shown as follows:

```
package com.test.entity;
import javax.persistence.DiscriminatorColumn;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Inheritance;
import javax.persistence.InheritanceType;
import javax.persistence.Table;

@Entity
@Table
@Inheritance( strategy = InheritanceType.SINGLE_TABLE )
@DiscriminatorColumn( name = "type" )

public class Staff {
   @Id
   @GeneratedValue( strategy = GenerationType.AUTO )
   private int sid;
   private String sname;

   public Staff( int sid, String sname ) {
      super( );
```

```
        this.sid = sid;
        this.sname = sname;
    }

    public Staff( ) {
        super( );
    }

    public int getSid( ) {
        return sid;
    }

    public void setSid( int sid ) {
        this.sid = sid;
    }

    public String getSname( ) {
        return sname;
    }

    public void setSname( String sname ) {
        this.sname = sname;
    }
}
```

*In the code above **@DescriminatorColumn** specifies the field name (type) and the values of it shows the remaining (Teaching and NonTeachingStaff) fields. **Basically @DiscriminatorColumn will create a new field/column in the staff table along with value.***

b) Create a children class named as "TeachingStaff" under the **com.test.entity** package should extend Staff . The TeachingStaff Entity class is shown as follows.

```
package com.test.entity;

import javax.persistence.DiscriminatorValue;
import javax.persistence.Entity;

@Entity
@DiscriminatorValue( value="TS" )
public class TeachingStaff extends Staff {

    private String qualification;
```

```
   private String subjectexpertise;

   public TeachingStaff( int sid, String sname,

   String qualification,String subjectexpertise ) {
      super( sid, sname );
      this.qualification = qualification;
      this.subjectexpertise = subjectexpertise;
   }

   public TeachingStaff( ) {
      super( );
   }

   public String getQualification( ){
      return qualification;
   }

   public void setQualification( String qualification ){
      this.qualification = qualification;
   }

   public String getSubjectexpertise( ) {
      return subjectexpertise;
   }

   public void setSubjectexpertise( String subjectexpertise ){
      this.subjectexpertise = subjectexpertise;
   }
}
```

c) Create a children class named as "**NonTeachingStaff**" under the **com.test.entity** package should extend Staff . The **NonTeachingStaff** Entity class is shown as follows:

```
package com.test.entity;

import javax.persistence.DiscriminatorValue;
import javax.persistence.Entity;

@Entity
@DiscriminatorValue( value = "NS" )
```

```
public class NonTeachingStaff extends Staff {
   private String areaexpertise;

   public NonTeachingStaff( int sid, String sname, String
areaexpertise ) {
      super( sid, sname );
      this.areaexpertise = areaexpertise;
   }

   public NonTeachingStaff( ) {
      super( );
   }

   public String getAreaexpertise( ) {
      return areaexpertise;
   }

   public void setAreaexpertise( String areaexpertise ){
      this.areaexpertise = areaexpertise;
   }
}
```

2. Persistence.xml

Persistence.xml file contains the configuration information of database and registration information of entity classes. The xml file is shown as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.1" 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">
        <persistence-unit name="JPA-MAPPING-SINGLE">
         <class>com.test.entity.Staff</class>
         <class>com.test.entity.TeachingStaff</class>
          <class>com.test.entity.NonTeachingStaff</class>

        <properties>
       <property name="javax.persistence.jdbc.driver"
value="org.mariadb.jdbc.Driver"/>
                    <property name="javax.persistence.jdbc.url"
value="jdbc:mariadb://localhost/classicmodels"/>
```

```
                    <property name="eclipselink.logging.level" value="FINE"/>
                    <property name="javax.persistence.jdbc.user" value="root"/>
                    <property name="javax.persistence.jdbc.password"
value="password"/>
      <property name="eclipselink.logging.level" value="FINE"/>
      <property name="eclipselink.ddl-generation" value="create-tables"/>
    </properties>
      </persistence-unit>
</persistence>
```

3. Service class

Service classes are the implementation part of the business component. Create a package named **'com.test.service'.**

Create a class named "**SaveClient"** under the given package to store Staff, TeachingStaff, and NonTeachingStaff class fields. The SaveClient class is shown as follows.

```
package com.test.service;

import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;

import com.test.entity.NonTeachingStaff;
import com.test.entity.TeachingStaff;

public class SaveClient {

        public static void main(String[] args) {
                // TODO Auto-generated method stub
                EntityManagerFactory emfactory = Persistence.createEntityManagerFactory(
"JPA-MAPPING-SINGLE" );
            EntityManager entitymanager = emfactory.createEntityManager( );
            entitymanager.getTransaction( ).begin( );

            //Teaching staff entity
            TeachingStaff ts1=new TeachingStaff(1,"Alex","MSc MEd","Maths");
            TeachingStaff ts2=new TeachingStaff(2, "Peterson", "BSc BEd", "English");

            //Non-Teaching Staff entity
            NonTeachingStaff nts1=new NonTeachingStaff(3, "Ramon", "Accounts");
```

```
        NonTeachingStaff nts2=new NonTeachingStaff(4, "Ali", "Office Admin");

        //storing all entities
        entitymanager.persist(ts1);
        entitymanager.persist(ts2);
        entitymanager.persist(nts1);
        entitymanager.persist(nts2);
        entitymanager.getTransaction().commit();

        entitymanager.close();
        emfactory.close();
    }
}
```

Result:

After compilation and execution of the above program you will get notifications in the console panel of Eclipse IDE. Check MariaDB HeidiSQL for output. The output in a tabular format is shown as follows:

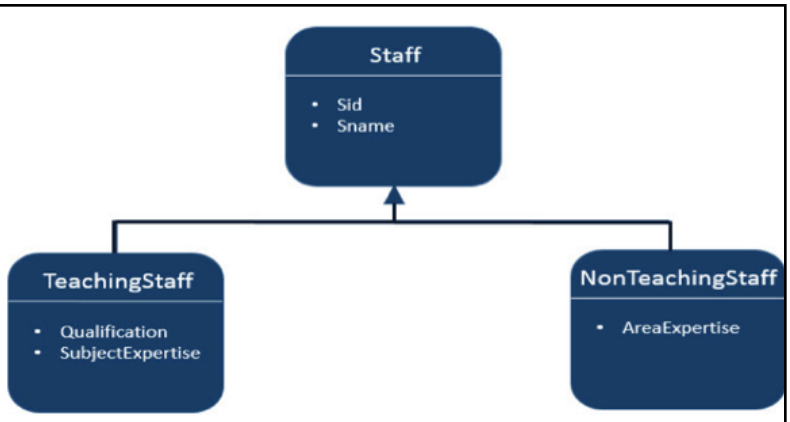
staff (4r × 6c)

sid	sname	areaexpertise	qualification	subjectexpertise	type
1	Alex	(NULL)	MSc MEd	Maths	TS
2	Peterson	(NULL)	BSc BEd	English	TS
3	Ramon	Accounts	(NULL)	(NULL)	NS
4	Ali	Office Admin	(NULL)	(NULL)	NS

Joined Table example

Prerequisites

Create a Three table name as **Staff, TeachingStaff and NonTeachingStaff** under classicmodels database. Add below column/ fields.

Staff Table: sid(Primary ID) sname **Teaching Staff** qualification Subjectexpertise **NonTechingStaff** areaexpertise	

Create a JPA project by the name of JPA-MAPPING-Join or as you like, and Configure JPA Jars and MariaDB JDBC Jars

Joined table strategy is to share the referenced column which contains unique values to join the table and make easy transactions. Let us consider the same example as above.

1. Creating Entities

a) Create a package named **'com.test.entity'** under the 'src' package. Create a new java class named **Staff.java** under the given package. The Staff entity class is shown as follows:

```
omport javax.persistence.DiscriminatorColumn;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Inheritance;
import javax.persistence.InheritanceType;
import javax.persistence.Table;

@Entity
@Table
@Inheritance( strategy = InheritanceType.JOINED )
@DiscriminatorColumn( name = "type" )
```

```
public class Staff {
   @Id
   @GeneratedValue( strategy = GenerationType.AUTO )
   private int sid;
   private String sname;

   public Staff( int sid, String sname ) {
      super( );
      this.sid = sid;
      this.sname = sname;
   }

   public Staff( ) {
      super( );
   }

   public int getSid( ) {
      return sid;
   }

   public void setSid( int sid ) {
      this.sid = sid;
   }

   public String getSname( ) {
      return sname;
   }

   public void setSname( String sname ) {
      this.sname = sname;
   }
}
```

Create a subclass (class) to Staff class named **TeachingStaff.java** under the **'com.test.entity'** package. The **TeachingStaff** Entity class is shown as follows:

```
import javax.persistence.DiscriminatorValue;
import javax.persistence.Entity;
import javax.persistence.PrimaryKeyJoinColumn;

@Entity
@PrimaryKeyJoinColumn(referencedColumnName="sid")
@DiscriminatorValue( value="TS" )
public class TeachingStaff extends Staff {
```

```
  private String qualification;
  private String subjectexpertise;

  public TeachingStaff( int sid, String sname,

  String qualification,String subjectexpertise ) {
    super( sid, sname );
    this.qualification = qualification;
    this.subjectexpertise = subjectexpertise;
  }

  public TeachingStaff( ) {
    super( );
  }

  public String getQualification( ){
    return qualification;
  }

  public void setQualification( String qualification ){
    this.qualification = qualification;
  }

  public String getSubjectexpertise( ) {
    return subjectexpertise;
  }

  public void setSubjectexpertise( String subjectexpertise ){
    this.subjectexpertise = subjectexpertise;
  }
}
```

Create a subclass (class) to Staff class named **NonTeachingStaff.java** under the **'com.test.entity'** package. The **NonTeachingStaff** Entity class is shown as follows:

```
import javax.persistence.DiscriminatorValue;
import javax.persistence.Entity;
import javax.persistence.PrimaryKeyJoinColumn;

@Entity
```

```
@DiscriminatorValue( value = "NS" )

public class NonTeachingStaff extends Staff {
   private String areaexpertise;

   public NonTeachingStaff( int sid, String sname, String areaexpertise ) {
      super( sid, sname );
      this.areaexpertise = areaexpertise;
   }

   public NonTeachingStaff( ) {
      super( );
   }

   public String getAreaexpertise( ) {
      return areaexpertise;
   }

   public void setAreaexpertise( String areaexpertise ){
      this.areaexpertise = areaexpertise;
   }
}
```

Persistence.xml

Persistence.xml file contains the configuration information of database and registration information of entity classes. The xml file is shown as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.1" 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">
          <persistence-unit name="JPA-MAPPING-SINGLE">
           <class>com.test.entity.Staff</class>
           <class>com.test.entity.TeachingStaff</class>
            <class>com.test.entity.NonTeachingStaff</class>


        <properties>
                              <property name="javax.persistence.jdbc.driver"
```

```
value="org.mariadb.jdbc.Driver"/>
                        <property name="javax.persistence.jdbc.url"
value="jdbc:mariadb://localhost/classicmodels"/>
                        <property name="eclipselink.logging.level" value="FINE"/>
                        <property name="javax.persistence.jdbc.user" value="root"/>
                        <property name="javax.persistence.jdbc.password"
value="password"/>
                        <property name="eclipselink.ddl-generation"
value="create-or-extend-tables"/>
                </properties>

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

Service class

Service classes are the implementation part of the business component. Create a package named **'com.test.service'.**

Create a class named "**SaveClient"** under the given package to store Staff, TeachingStaff, and NonTeachingStaff class fields. The SaveClient class is shown as follows.

```
import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;

import com.test.entity.NonTeachingStaff;
import com.test.entity.TeachingStaff;

public class SaveClient {

        public static void main(String[] args) {
                // TODO Auto-generated method stub
                  EntityManagerFactory emfactory =
Persistence.createEntityManagerFactory( "JPA-MAPPING-SINGLE" );
            EntityManager entitymanager = emfactory.createEntityManager( );
            entitymanager.getTransaction( ).begin( );

            //Teaching staff entity
            TeachingStaff ts1=new TeachingStaff(15,"Alex","MSc MEd","Maths");
```

```
            TeachingStaff ts2=new TeachingStaff(16, "Peterson", "BSc BEd",
"English");



            //Non-Teaching Staff entity
            NonTeachingStaff nts1=new NonTeachingStaff(17, "Ramon", "Accounts");
            NonTeachingStaff nts2=new NonTeachingStaff(18, "Ali", "Office Admin");

            //storing all entities
            entitymanager.persist(ts1);
            entitymanager.persist(ts2);
            entitymanager.persist(nts1);
            entitymanager.persist(nts2);


            entitymanager.getTransaction().commit();

            entitymanager.close();
            emfactory.close();
        }

}
```