

# JPA JPQL Basic Operations

JPQL allows us to create both static as well as dynamic queries. Now, we will perform some basic JPQL operations using both type of queries on the below table.

S_ID	S_NAME	S_AGE
101	Gaurav	24
102	Rahul	22
103	Chris	20
104	Ronit	26
105	Roy	21

## JPQL Dynamic Query Example

In this example, we will fetch single column from database by using **createQuery()** method .

### StudentEntity.java

```
package com.javatpoint.jpa;
import javax.persistence.*;

@Entity
@Table(name="student")
public class StudentEntity {

    @Id
    private int s_id;
    private String s_name;
    private int s_age;

    public StudentEntity(int s_id, String s_name, int s_age) {
        super();
        this.s_id = s_id;
        this.s_name = s_name;
        this.s_age = s_age;
    }
}
```



```
}

public StudentEntity() {
    super();
}

public int getS_id() {
    return s_id;
}

public void setS_id(int s_id) {
    this.s_id = s_id;
}

public String getS_name() {
    return s_name;
}

public void setS_name(String s_name) {
    this.s_name = s_name;
}

public int getS_age() {
    return s_age;
}

public void setS_age(int s_age) {
    this.s_age = s_age;
}
}
```

### Persistence.xml

```
<persistence>
<persistence-unit name="Student_details">

    <class>com.javatpoint.jpa.StudentEntity</class>
```



```
<properties>
  <property name="javax.persistence.jdbc.driver" value="com.mysql.jdbc.Driver"/>
  <property name="javax.persistence.jdbc.url" value="jdbc:mysql://localhost:3306/studentdata"/>
  <property name="javax.persistence.jdbc.user" value="root"/>
  <property name="javax.persistence.jdbc.password" value=""/>
  <property name="eclipselink.logging.level" value="SEVERE"/>
  <property name="eclipselink.ddl-generation" value="create-or-extend-tables"/>
</properties>

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

## FetchColumn.java

```
package com.javatpoint.jpa.jpql;
import javax.persistence.*;
import java.util.*;

public class FetchColumn {

    public static void main( String args[]) {

        EntityManagerFactory emf = Persistence.createEntityManagerFactory( "Student_details" );
        EntityManager em = emf.createEntityManager();
        em.getTransaction().begin( );

        Query query = em.createQuery("Select s.s_name from StudentEntity s");
        @SuppressWarnings("unchecked")
        List<String> list =query.getResultList();
        System.out.println("Student Name :");
        for(String s:list) {

            System.out.println(s);

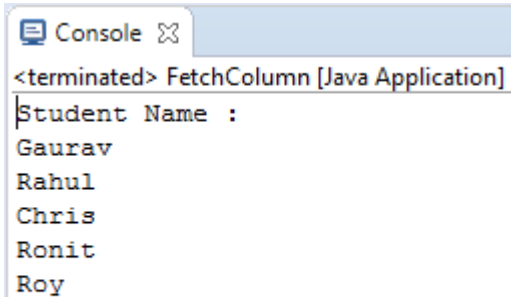
        }


        em.close();
        emf.close();
    }
}
```



```
}  
  
}
```

### Output:



```
Console   
<terminated> FetchColumn [Java Application]  
Student Name :  
Gaurav  
Rahul  
Chris  
Ronit  
Roy
```

## JPQL Static Query Example

In this example, we will fetch single column from database by using **createNamedQuery()** method .

### StudentEntity.java

```
package com.javatpoint.jpa;  
  
import javax.persistence.*;  
  
@Entity  
@Table(name="student")  
@NamedQuery(name = "find name" , query = "Select s from StudentEntity s")  
public class StudentEntity {  
  
    @Id  
    private int s_id;  
    private String s_name;  
    private int s_age;  
  
    public StudentEntity(int s_id, String s_name, int s_age) {  
        super();  
        this.s_id = s_id;  
        this.s_name = s_name;  
    }  
}
```



```
        this.s_age = s_age;
    }

    public StudentEntity() {
        super();
    }

    public int getS_id() {
        return s_id;
    }

    public void setS_id(int s_id) {
        this.s_id = s_id;
    }

    public String getS_name() {
        return s_name;
    }

    public void setS_name(String s_name) {
        this.s_name = s_name;
    }

    public int getS_age() {
        return s_age;
    }

    public void setS_age(int s_age) {
        this.s_age = s_age;
    }
}
```

## Persistence.xml

```
<persistence>
<persistence-unit name="Student_details">

    <class>com.javatpoint.jpa.StudentEntity</class>
```



```
<properties>
  <property name="javax.persistence.jdbc.driver" value="com.mysql.jdbc.Driver"/>
  <property name="javax.persistence.jdbc.url" value="jdbc:mysql://localhost:3306/studentdata"/>
  <property name="javax.persistence.jdbc.user" value="root"/>
  <property name="javax.persistence.jdbc.password" value=""/>
  <property name="eclipselink.logging.level" value="SEVERE"/>
  <property name="eclipselink.ddl-generation" value="create-or-extend-tables"/>
</properties>

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

### FetchColumn.java

```
package com.javatpoint.jpa.jpql;
import javax.persistence.*;

import com.javatpoint.jpa.StudentEntity;

import java.util.*;

public class FetchColumn {

    public static void main( String args[]) {

        EntityManagerFactory emf = Persistence.createEntityManagerFactory( "Student_details" );
        EntityManager em = emf.createEntityManager();
        em.getTransaction().begin( );

        Query query = em.createNamedQuery("find name");
        @SuppressWarnings("unchecked")
        List<StudentEntity> list =query.getResultList();
        System.out.println("Student Name :");
        for(StudentEntity s:list) {

            System.out.println(s.getS_name());

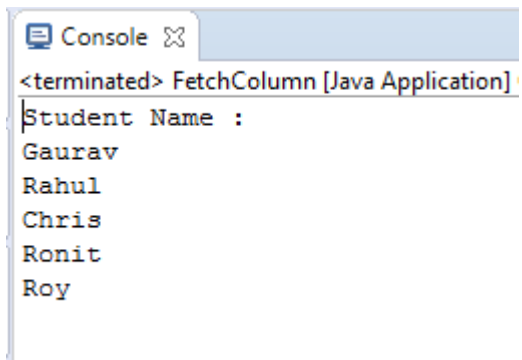
        }

    }

}
```



```
        em.close();  
        emf.close();  
    }  
  
}
```



The screenshot shows a console window titled "Console" with a close button. The output text is as follows:

```
<terminated> FetchColumn [Java Application]  
Student Name :  
Gaurav  
Rahul  
Chris  
Ronit  
Roy
```

[< prev](#)[next >](#)

## Please Share



## Learn Latest Tutorials