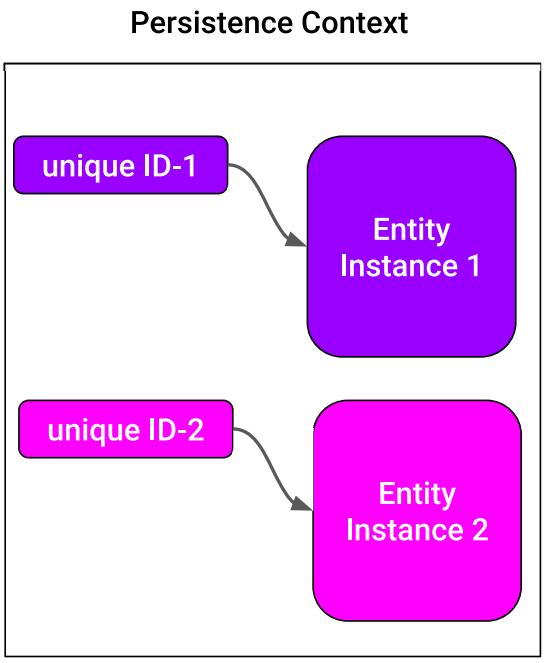
Entity Instance State and State Transition

Dr. Harshad Prajapati
15 Oct 2023

We use Existing Project jpademo

Persistence Context

- A persistence context is a sort of cache.
- The persistence context uniquely knows all entity instances:
  - Read from the database, OR
  - New entity made persistent within the context.
- The context holds a unique mapping from the identifier of the entity instance to the instance itself.



States of an Entity Instance within Context

- An entity instance may be in one of three states with respect to a given persistence context.
  - transient:
    - Never persistent and not associated with the persistence context.
  - persistent:
    - Currently associated with the persistence context.
  - detached:
    - Previously persistent in another session, but not currently associated with this persistence context.
- At any given moment, an instance may be associated with at most one persistence context.

State Transition for a New Entity Instance GitHub Repo: jpademo/state-transition-new-instance

hibernate.properties File

hibernate.properties

```
hibernate.connection.url=jdbc:mysql://localhost/test
hibernate.connection.driver_class=com.mysql.jdbc.Driver
hibernate.dialect=org.hibernate.dialect.MySQLDialect
hibernate.connection.username=root
hibernate.connection.password=rootpassword
hibernate.show_sql=true
hibernate.format_sql=true
hibernate.hbm2ddl.auto=create
```

Entity Class

- Student Entity Class
  - Fields
    - id
    - studentName
  - Methods:
    - Getter/setter

Student.java

```java
package org.example.entities;
import jakarta.persistence.*;

6 usages
@Entity
public class Student {
    @Id @GeneratedValue
    private long id;

    2 usages
    @Column(name = "student_name")
    private String studentName;

    no usages
    public long getId() {
        return id;
    }

    no usages
    public void setId(long id) { this.id = id; }

    no usages
    public String getStudentName() { return studentName; }

    1 usage
    public void setStudentName(String studentName) {
        this.studentName = studentName;
    }
}
```


Main Class

We perform persist() twice, but only one instance gets stored in the database.

Main.java

```java
package org.example;

import org.example.entities.Student;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class Main {
    public static void main(String[] args) {
        Configuration config = new Configuration();
        config.addAnnotatedClass(org.example.entities.Student.class);
        SessionFactory sessionFactory = config.buildSessionFactory();
        Session session = sessionFactory.openSession();

        try {
            session.beginTransaction();

            Student student1 = new Student();
            student1.setStudentName("Tom Cruise");

            session.persist(student1);
            session.persist(student1);

            session.getTransaction().commit();
        } finally{
            session.close();
            sessionFactory.close();
        }
    }
}
```

Run main() method

- From the database, we **remove test schema** and **recreate test schema** so that any existing **tables get removed**.
- Then run the main() method.

```
INFO: HHH000489: No JTA platform available (set 'hibernate.transaction.jta.platform' to enable JTA platform integrat
Hibernate:
    drop table if exists Student
Oct 15, 2023 9:47:47 AM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl ge
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnviron
Hibernate:
    drop table if exists Student_SEQ
Hibernate:
    create table Student (
        id bigint not null,
        student_name varchar(255),
        primary key (id)
    ) engine=InnoDB
Hibernate:
    create table Student_SEQ (
        next_val bigint
    ) engine=InnoDB
```

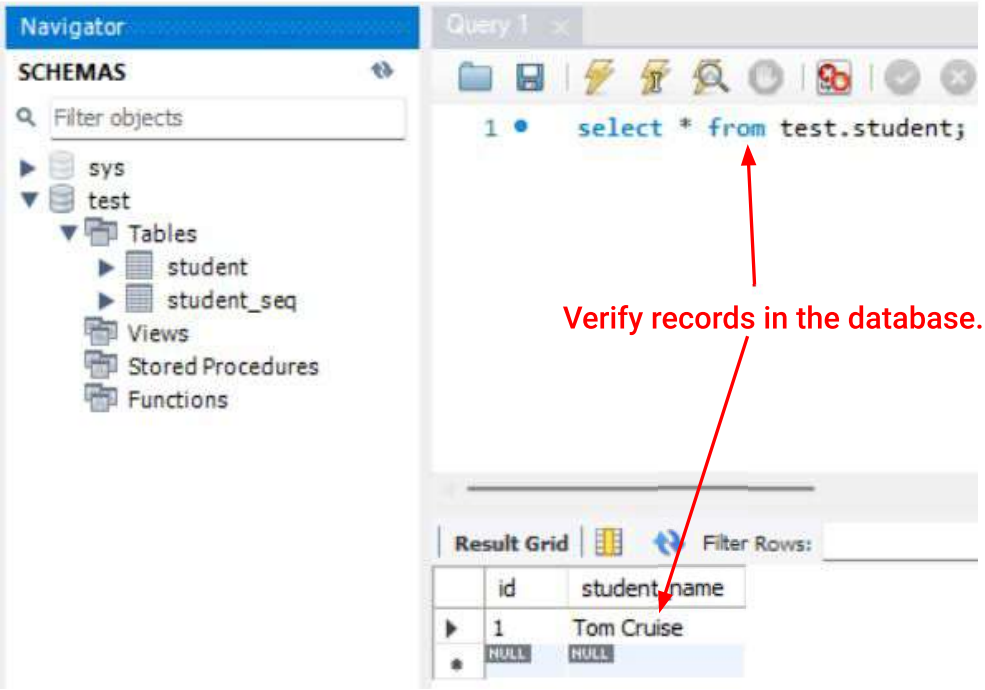

Run main() method

```
Oct 15, 2023 9:47:47 AM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsola
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentIr
Hibernate:
    insert into Student_SEQ values ( 1 )
Hibernate:
    select
        next_val as id_val
    from
        Student_SEQ for update
Hibernate:
    update
        Student_SEQ
    set
        next_val= ?
    where
        next_val=?
Hibernate:
    insert
    into
        Student
        (student_name, id)
    values
        (?, ?)
Oct 15, 2023 9:47:47 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PoolState stop
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost/test]
```

```
21    session.persist(student1);
22    session.persist(student1);
```

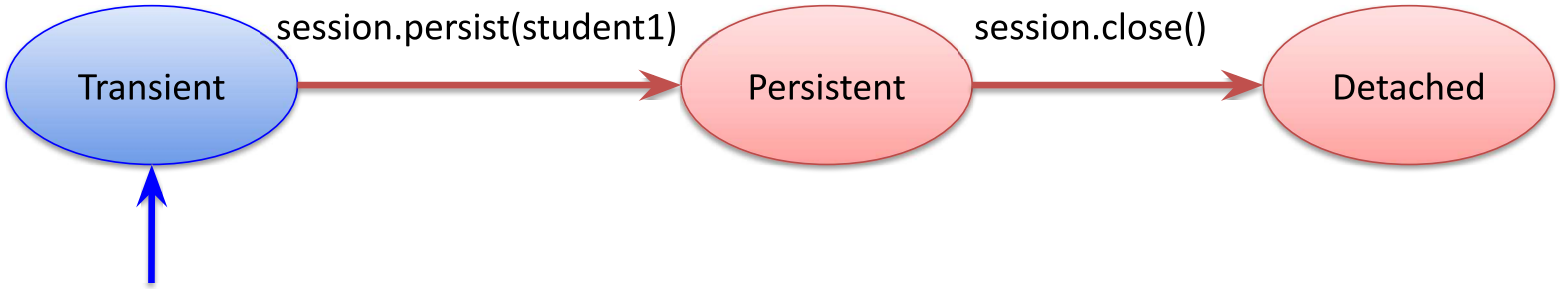
Only one SQL INSERT query is fired by Hibernate.

MySQL Workbench



State Transition for a New Entity Instance

- We **create** student instance:

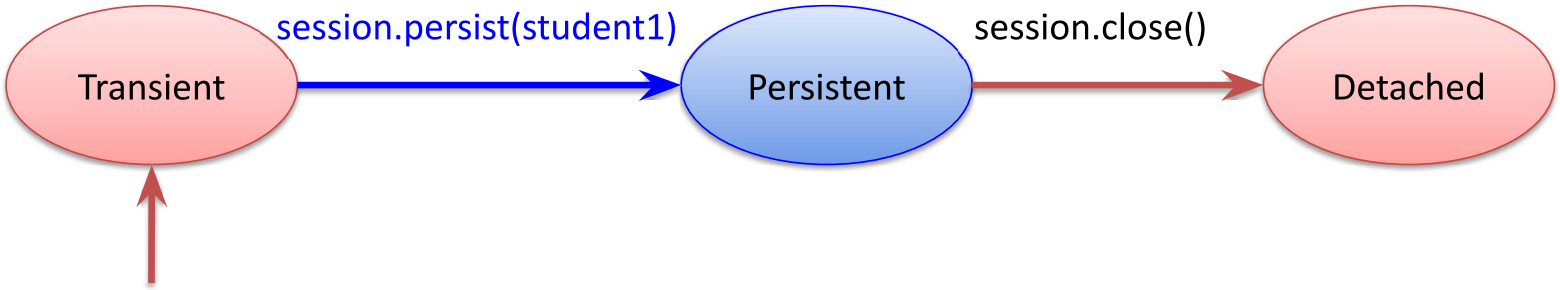



```
Student student1 = new Student();
student1.setStudentName("Tom Cruise");
```

- From creating a student entity instance, using new, until persisting it using session.persist(), the entity instance remains in transient state.

State Transition When a New Instance is persisted

- We **persist** student instance:

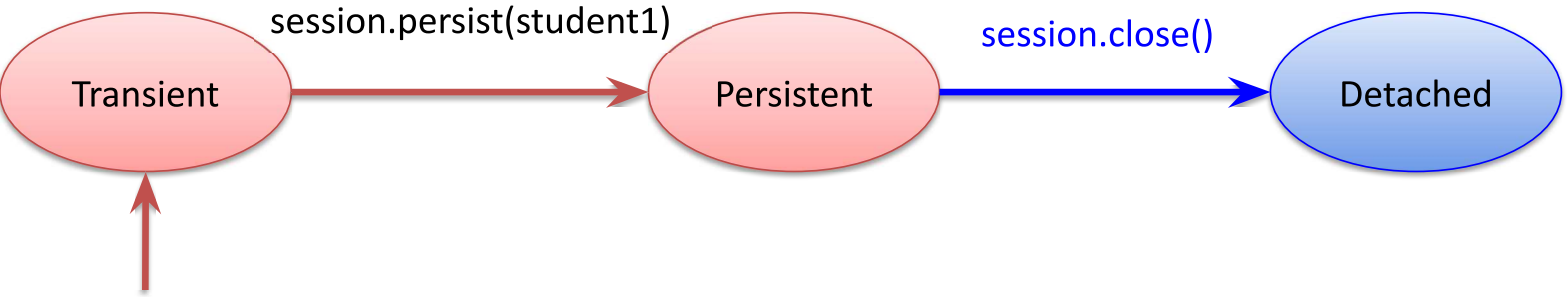



```
Student student1 = new Student();
student1.setStudentName("Tom Cruise");
```

- After persisting the instance until we close the session, the instance's state remains persistent state.

State Transition when the Session is Closed

- We **close** session:




```
Student student1 = new Student();
student1.setStudentName("Tom Cruise");
```

- After session is closed, the instance goes into detached state. That is, no effect on db record for any change on entity instance after session.close().

Session State and Database State

- Transient:
  - Instance has no corresponding record in the database.
- Persistent:
  - Instance has corresponding record in the database and the instance is associated with hibernate session.
  - Any modifications done to the instance, will be reflected into database, when the session is closed.
- Detached:
  - Instance has corresponding record in the database, but the instance is not associated with any hibernate session.

Update Instance while in Persistent State GitHub Repo: jpademo/state-transition-update

Update Instance while in Persistent State

```
15        try {
16            session.beginTransaction();
17
18            Student student1 = new Student();
19            student1.setStudentName("Tom Cruise");
20
21            session.persist(student1);
22
23            student1.setStudentName("Tomkumar Cruise");
24
25            session.getTransaction().commit();
26        } finally{
27            session.close();
28            sessionFactory.close();
29        }
```

We persist student instance.

We update student instance. Then, we do not persist the instance again.

hibernate.properties File

- We keep hibernate.properties file as it is, i.e., hbm2ddl.auto value as create.

hibernate.properties

```
1  hibernate.connection.url=jdbc:mysql://localhost/test
2  hibernate.connection.driver_class=com.mysql.jdbc.Driver
3  hibernate.dialect=org.hibernate.dialect.MySQLDialect
4  hibernate.connection.username=root
5  hibernate.connection.password=rootpassword
6  hibernate.show_sql=true
7  hibernate.format_sql=true
8  hibernate.hbm2ddl.auto=create
```

Running the main Method

```
INFO: HHH000489: No JTA platform available (set 'hibernate.transaction.jta.platform' to enable JTA platform integration)
Hibernate:
    drop table if exists Student
Oct 15, 2023 9:15:30 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIso
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironment
Hibernate:
    drop table if exists Student_SEQ
Hibernate:
    create table Student (
        id bigint not null,
        student_name varchar(255),
        primary key (id)
    ) engine=InnoDB
Oct 15, 2023 9:15:30 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIso
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironment
```


Running the main Method

```
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEn
Hibernate:
    create table Student_SEQ (
        next_val bigint
    ) engine=InnoDB
Hibernate:
    insert into Student_SEQ values ( 1 )
Hibernate:
    select
        next_val as id_val
    from
        Student_SEQ for update
Hibernate:
    update
        Student_SEQ
    set
        next_val= ?
    where
        next_val=?
```

Running the main Method

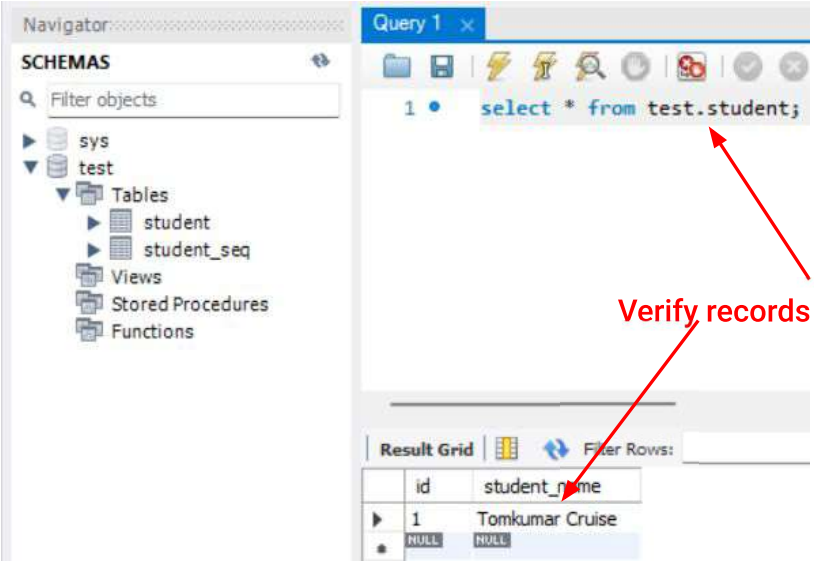
```
Hibernate:
    insert
    into
        Student
        (student_name, id)
    values
        (?, ?)
Hibernate:
    update
        Student
    set
        student_name=?
    where
        id=?
Oct 15, 2023 9:15:31 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConn
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost/test]

Process finished with exit code 0
```

Resulted in INSERT query

Resulted in UPDATE query

```
try {
    session.beginTransaction();

    Student student1 = new Student();
    student1.setStudentName("Tom Cruise");

    session.persist(student1);

    student1.setStudentName("Tomkumar Cruise");

    session.getTransaction().commit();
} finally{
    session.close();
    sessionFactory.close();
}
```


MySQL Workbench

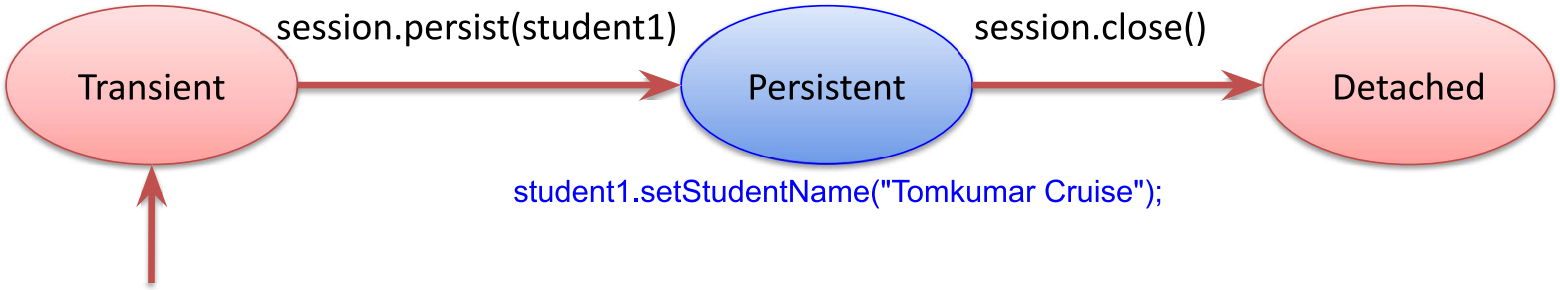


Verify records in the database.

We find that if an instance is in persistent state, and if we change values of fields, they get reflected into database automatically, without again calling session.persist(...)

Update Entity Instance while in Persistent State

- We **update** student instance (in persistent state):



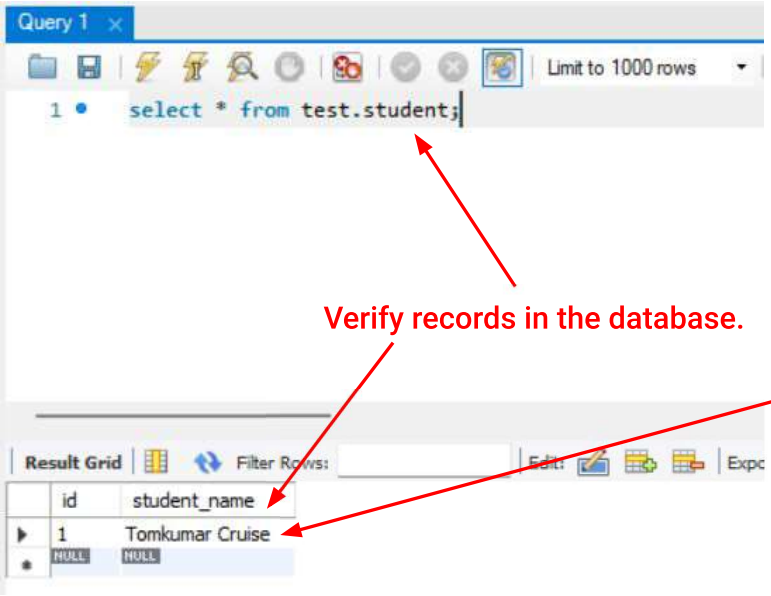

```
Student student1 = new Student();
student1.setStudentName("Tom Cruise");
```

- If an instance is in persistent state, and if we do any modification in the instance, the modifications will be reflected into the database by hibernate automatically, without need of writing session.persist()

State Transition for a Retrieved Instance GitHub Repo: jpademo/state-transition-get-instance

MySQL Workbench

- We check existing records in database:



Verify records in the database.

- This is an existing record.
- We will retrieve it

Comment hbm2ddl.auto in hibernate.properties file

hibernate.properties

```
hibernate.connection.url=jdbc:mysql://localhost/test
hibernate.connection.driver_class=com.mysql.jdbc.Driver
hibernate.dialect=org.hibernate.dialect.MySQLDialect
hibernate.connection.username=root
hibernate.connection.password=rootpassword
hibernate.show_sql=true
hibernate.format_sql=true
#hibernate.hbm2ddl.auto=create
```

- We want to retrieve existing record. So, we comment hbm2ddl.auto property.
- Hibernate will not do anything with table schema

Add toString in Student Entity Class

Student.java

```
1   package org.example.entities;
2   import jakarta.persistence.*;
3
    6 usages
4   @Entity
5   public class Student {
6       @Id @GeneratedValue
7       private long id;
8
        3 usages
9       @Column(name = "student_name")
10      private String studentName;
11
        no usages
12      public long getId() { return id; }
15
        no usages
16      public void setId(long id) { this.id = id; }
```


Add toString in Student Entity Class

```
19
        no usages
20      public String getStudentName() { return studentName; }
23
        no usages
24      public void setStudentName(String studentName) { this.studentName = studentName; }
27
28      @Override
29      public String toString() {
30          return "Student{" +
31                  "id=" + id +
32                  ", studentName='" + studentName + '\'' +
33                  '}';
34      }
35  }
```

Add toString() method using Alt+INSERT, then select override toString()

Main Class: Retrieve Existing Student with Id

Main.java

```java
package org.example;

import org.example.entities.Student;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class Main {
    public static void main(String[] args) {
        Configuration config = new Configuration();
        config.addAnnotatedClass(org.example.entities.Student.class);
        SessionFactory sessionFactory = config.buildSessionFactory();
        Session session = sessionFactory.openSession();
```


Main Class: Retrieve Existing Student with Id

Retrieve existing student with ID value 1

```java

        try {
            session.beginTransaction();

            Student student1 = session.get(Student.class, o: 1);
            System.out.println("Retrieved Student = " + student1);
            session.getTransaction().commit();
        } finally{
            session.close();
            sessionFactory.close();
        }
    }
}
```

Run main() Method

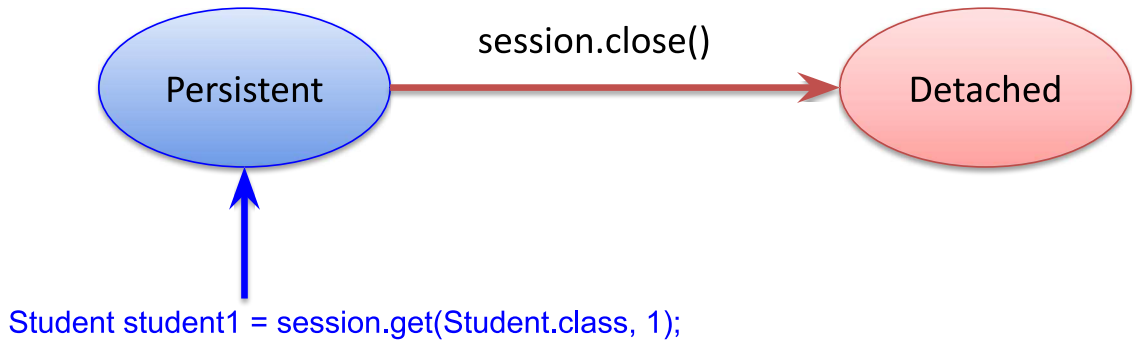
```
INFO: HHH000489: No JTA platform available (set 'hibernate.transaction.jta.p
Hibernate:
    select
        s1_0.id,
        s1_0.student_name
    from
        Student s1_0
    where
        s1_0.id=?
Retrieved Student = Student{id=1, studentName='Tomkumar Cruise'}
Oct 19, 2023 7:40:34 AM org.hibernate.engine.jdbc.connections.internal.Drive
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost/test]

Process finished with exit code 0
```

SELECT query is fired by Hibernate

Retrieved student record from the database

State transition for a retrieved object

- We retrieve existing student record using id value.
  - The Student instance directly goes into persistent state.



Retrieving Non Existent Student with ID

Pass ID value that is not existing

```
try {
    session.beginTransaction();

//          Student student1 = session.get(Student.class, 1);
    Student student1 = session.get(Student.class,  o: 2);

    System.out.println("Retrieving student " + student1);
    session.getTransaction().commit();
} finally{
    session.close();
    sessionFactory.close();
}
```


Run main() Method

No instance is retrieved.

```
INFO: HHH000489: No JTA platform available (set 'hibernate.transaction.jta.p
Hibernate:
    select
        s1_0.id,
        s1_0.student_name
    from
        Student s1_0
    where
        s1_0.id=?
Retrieved Student = null
Oct 19, 2023 7:56:00 AM org.hibernate.engine.jdbc.connections.internal.Drive
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost/test]

Process finished with exit code 0
```

Retrieving Entity Instance using find() Method

Retrieve Student instance using find() method

```
try {
    session.beginTransaction();

//      Student student1 = session.get(Student.class, 1);
//      Student student1 = session.get(Student.class, 2);
    Student student1 = session.find(Student.class, o: 1);

    System.out.println("Retrieving student " + student1);
    session.getTransaction().commit();
} finally{
    session.close();
    sessionFactory.close();
}
```


Run main() Method

Retrieved Student instance using find() method

```
INFO: HHH000489: No JTA platform available (set 'hibernate.transaction.jta.p
Hibernate:
    select
        s1_0.id,
        s1_0.student_name
    from
        Student s1_0
    where
        s1_0.id=?
Retrieved Student = Student{id=1, studentName='Tomkumar Cruise'}
Oct 19, 2023 8:02:23 AM org.hibernate.engine.jdbc.connections.internal.Drive
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost/test]

Process finished with exit code 0
```

Modification to Retrieved Instance
GitHub Repo:
jpademo/state-transition-get-then-update

Modify Retrieved Instance

Main.java

```
package org.example;

import org.example.entities.Student;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class Main {
    public static void main(String[] args) {
        Configuration config = new Configuration();
        config.addAnnotatedClass(org.example.entities.Student.class);
        SessionFactory sessionFactory = config.buildSessionFactory();
        Session session = sessionFactory.openSession();

```

Modify Retrieved Instance

Get entity instance

Update entity instance

commit

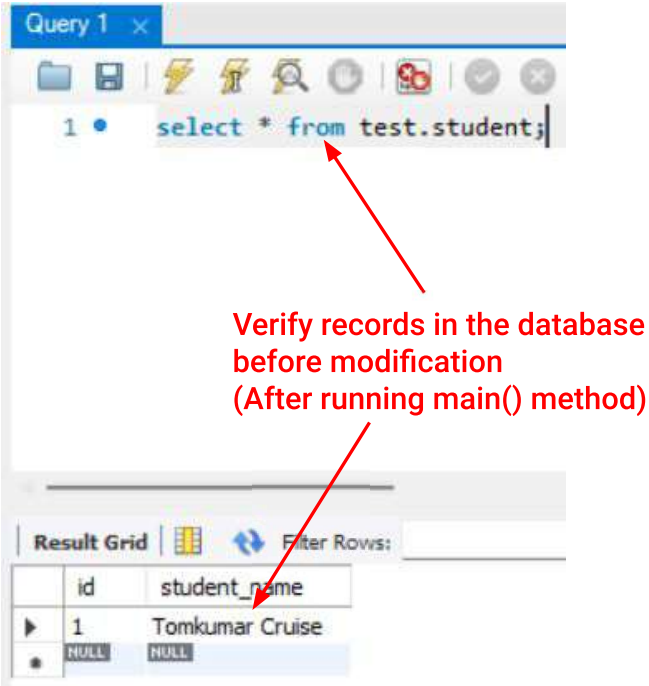
```
15          try {
16              session.beginTransaction();
17
18              Student student1 = session.find(Student.class, o: 1);
19              System.out.println("Retrieved Student = " + student1);
20              student1.setStudentName("Tom Cruise");
21              System.out.println("Retrieved Student = " + student1);
22              session.getTransaction().commit();
23              student1.setStudentName("Tomkumar Cruise");
24          } finally{
25              session.close();
26              sessionFactory.close();
27          }
28      }
29  }
```

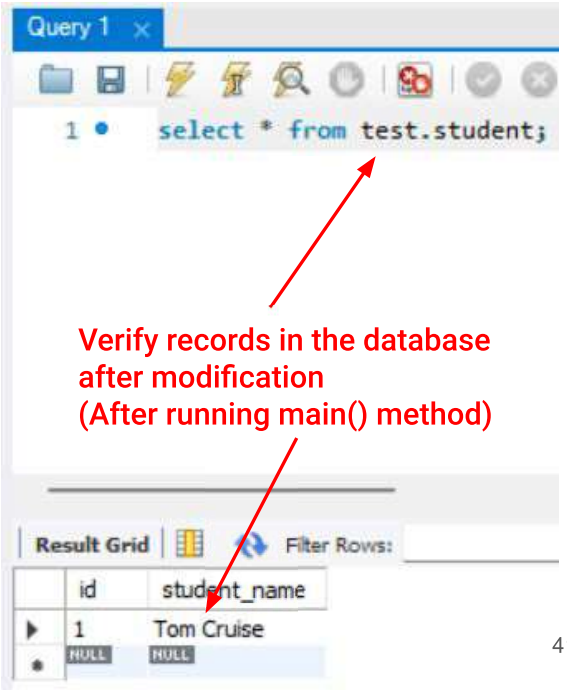
Update after commit has no effect on db.

Run main() Method

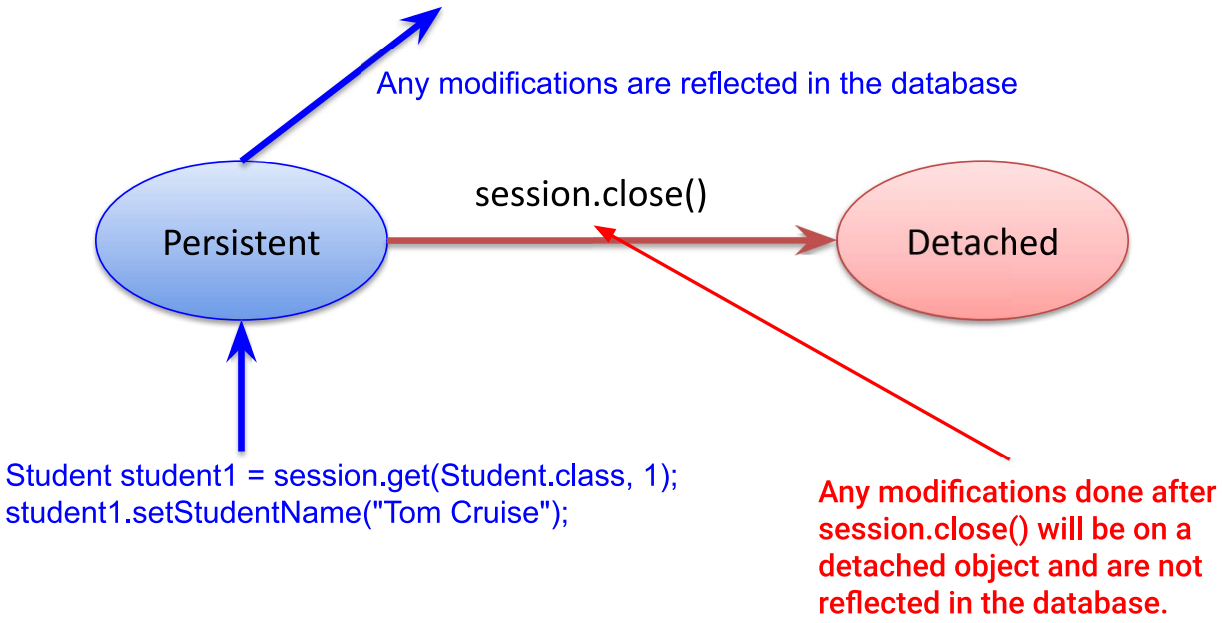
```
INFO: HHH000489: No JTA platform available (set 'hibernate.transaction.jta.pl
Hibernate:
    select
        s1_0.id,
        s1_0.student_name
    from
        Student s1_0
    where
        s1_0.id=?
Retrieved Student = Student{id=1, studentName='Tomkumar Cruise'}
Retrieved Student = Student{id=1, studentName='Tom Cruise'}
Hibernate:
    update
        Student
    set
        student_name=?
    where
        id=?
Oct 19, 2023 9:29:46 AM org.hibernate.engine.jdbc.connections.internal.Drivel
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost/test]
```

MySQL Workbench





State Transition for Modification in Instance



Changes After Commit

- There will **not** be **any effect** on database record if **changes** are **done** in **transient instance**.

```
Student student1 = session.find(Student.class, o: 1);
System.out.println("Retrieved Student = " + student1);
student1.setStudentName("Tom Cruise");
System.out.println("Retrieved Student = " + student1);
session.getTransaction().commit();
student1.setStudentName("Tomkumar Cruise");
```

Change after commit

No change in database

Result Grid | Filter Rows:

id	student_name
1	Tom Cruise
NULL	NULL

Modifications Should be done within Transaction

```
15        try {
16            session.beginTransaction();
17
18            Student student1 = session.find(Student.class, o: 1);
19            System.out.println("Retrieved Student = " + student1);
20            student1.setStudentName("Tom Cruise");
21            System.out.println("Retrieved Student = " + student1);
22    //        session.getTransaction().commit();
23            student1.setStudentName("Tomkumar Cruise");
24        } finally{
25            session.close();
26            sessionFactory.close();
27        }
```

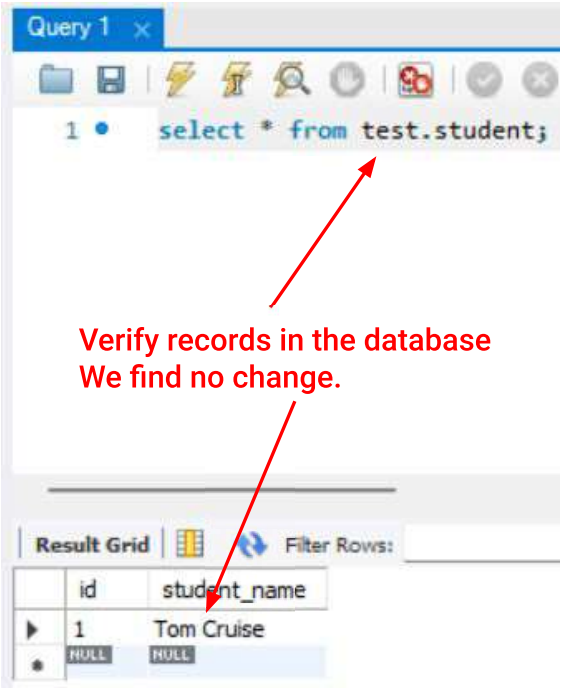
We comment this commit()

Run main() Method

```
INFO: HHH000489: No JTA platform available (set 'hibernate.transaction.jta.pl
Hibernate:
    select
        s1_0.id,
        s1_0.student_name
    from
        Student s1_0
    where
        s1_0.id=?
Retrieved Student = Student{id=1, studentName='Tom Cruise'}
Retrieved Student = Student{id=1, studentName='Tom Cruise'}
Oct 19, 2023 9:38:16 AM org.hibernate.engine.jdbc.connections.internal.Driver
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost/test]
```

Hibernate does not fire UPDATE query

MySQL Workbench



Verify records in the database
We find no change.

Deleting a Persistent Instance
GitHub Repo:
jpademo/state-transition-delete-instance

Deleting a Persistent Instance

Main.java

```
package org.example;

import org.example.entities.Student;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class Main {
    public static void main(String[] args) {
        Configuration config = new Configuration();
        config.addAnnotatedClass(org.example.entities.Student.class);
        SessionFactory sessionFactory = config.buildSessionFactory();
        Session session = sessionFactory.openSession();
        Student student1=null;

```

Declare reference outside try-catch-finally block

Deleting a Persistent Instance

```
16          try {
17              session.beginTransaction();
18
19              student1 = session.find(Student.class, o: 1);
20              System.out.println("Retrieved Student = " + student1);
21              session.remove(student1);
22
23              session.getTransaction().commit();
24          } finally{
25              System.out.println("Retrieved Student = " + student1);
26              session.close();
27              sessionFactory.close();
28          }
29      }
30  }
```

Delete persistent instance

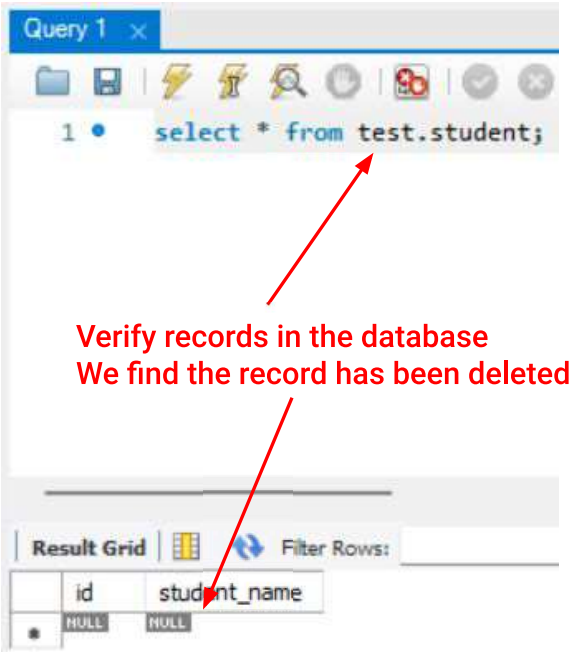
Print instance after delete operation

Run main() Method

```
INFO: HHH000489: No JTA platform available (set 'hibernate.transaction.jta.p
Hibernate:
    select
        s1_0.id,
        s1_0.student_name
    from
        Student s1_0
    where
        s1_0.id=?
Retrieved Student = Student{id=1, studentName='Tom Cruise'}
Hibernate:
    delete
    from
        Student
    where
        id=?
Retrieved Student = Student{id=1, studentName='Tom Cruise'}
Oct 19, 2023 9:57:16 AM org.hibernate.engine.jdbc.connections.internal.Drive
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost/test]
```

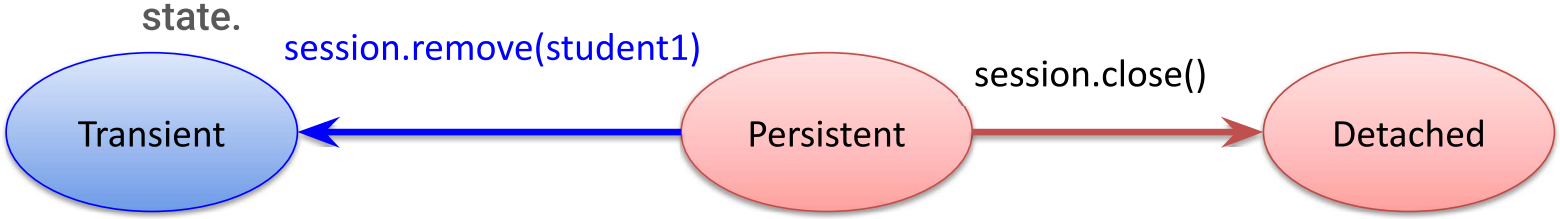
Hibernate fires DELETE query

Delete instance holds values.



State Transition for a Deleted Object

- After remove(), the instance goes into Transient state and not in Detached state.



Recall:

- Transient object has no presence in database.
- Detached object has presence in database.

Convert Instance from Detached State to Persistent State
GitHub Repo: jpademo/state-transition-detach-to-persistent

Convert Instance from Detached State to Persistent State

- After session is closed, the instance goes to detached state; therefore, any modification will not be reflected to the database.
- When to create a new session?
  - If there is a long gap between data (record/instance) is retrieved and data (record/instance) is updated.
  - For example, data update using a webpage.
- Solution:
  - Open a new session.
  - Update the values in persistent instance.

First We Store a New Student

- As database table is empty, we first store a new Student record.

Main.java

```java
package org.example;

import org.example.entities.Student;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class Main {
    public static void main(String[] args) {
        Configuration config = new Configuration();
        config.addAnnotatedClass(org.example.entities.Student.class);
        SessionFactory sessionFactory = config.buildSessionFactory();
        Session session = sessionFactory.openSession();
        Student student1=null;
```


First We Store a New Student

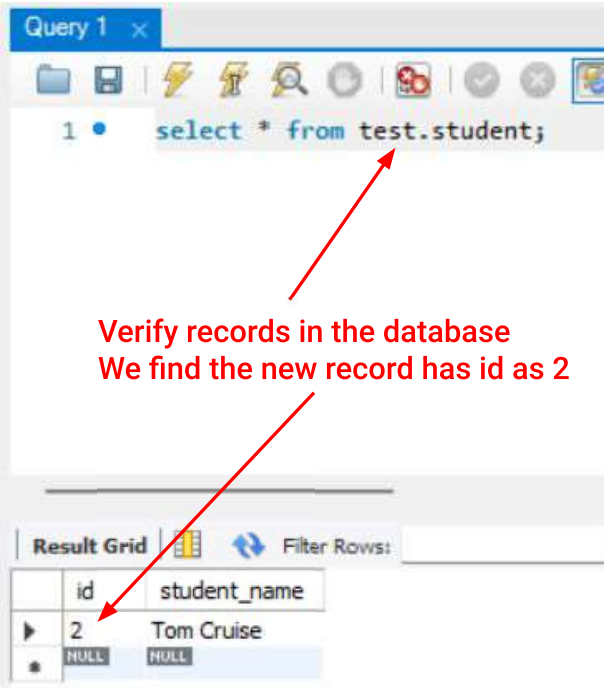
```java
        try {
            session.beginTransaction();
            student1 = new Student();
            student1.setStudentName("Tom Cruise");
            session.persist(student1);
            session.getTransaction().commit();
        } finally{
            System.out.println("Retrieved Student = " + student1);
            session.close();
            sessionFactory.close();
        }
    }
}
```

Run main() Method

```
INFO: HHH000489: No JTA platform available (set 'hibernate.transaction.jta.p
Hibernate:
    select
        next_val as id_val
    from
        Student_SEQ for update
Hibernate:
    update
        Student_SEQ
    set
        next_val= ?
    where
        next_val=?
Hibernate:
    insert
    into
        Student
        (student_name, id)
    values
        (?, ?)
Retrieved Student = Student{id=2, studentName='Tom Cruise'}
Oct 19, 2023 6:59:03 PM org.hibernate.engine.jdbc.connections.internal.Drive
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost/test]

Process finished with exit code 0
```

New record has id value as 2

MySQL Workbench



Verify records in the database
We find the new record has id as 2

Convert Instance from Detached State to Persistent State
GitHub Repo:
jpademo/state-transition-detach-to-persistent-2

Convert Instance from Detached State to Persistent State

Main.java

```java
package org.example;

import org.example.entities.Student;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class Main {
    public static void main(String[] args) {
        Configuration config = new Configuration();
        config.addAnnotatedClass(org.example.entities.Student.class);
        SessionFactory sessionFactory = config.buildSessionFactory();
        Session session1 = sessionFactory.openSession();
        Student student1=null;
        Session session2=null;

```

Convert Instance from Detached State to Persistent State

```
17          try {
18              session1.beginTransaction();
19              student1 = session1.get(Student.class, o: 2);
20              session1.getTransaction().commit();
21              session1.close();
22              System.out.println("Retrieved Student = " + student1);
23
24              student1.setStudentName("Tomkumar Cruise");
25              session2 = sessionFactory.openSession();
26              session2.beginTransaction();
27              session2.merge(student1);
28              session2.getTransaction().commit();
29
30          } finally{
31              session2.close();
32              sessionFactory.close();
33          }
34      }
35  }
```

One session

student1 is now detached.

Another session

student1 will be in persistent state

Run main() Method

```
INFO: HHH000489: No JTA platform available (set 'hibernate.transaction.jta.platform' to e
Hibernate:
    select
        s1_0.id,
        s1_0.student_name
    from
        Student s1_0
    where
        s1_0.id=?
Retrieved Student = Student{id=2, studentName='Tom Cruise'}
```

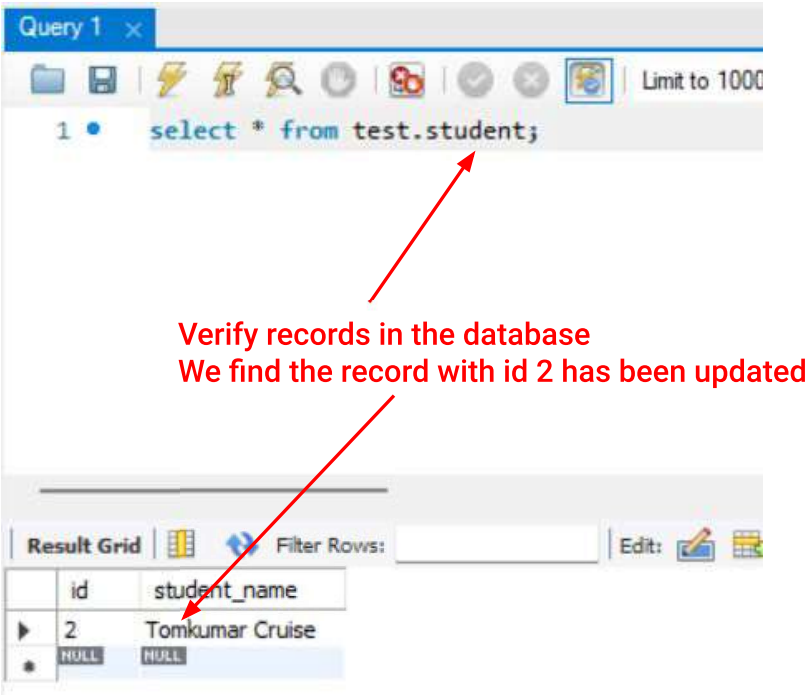
Run main() Method

```
Hibernate:
    select
        s1_0.id,
        s1_0.student_name
    from
        Student s1_0
    where
        s1_0.id=?
Hibernate:
    update
        Student
    set
        student_name=?
    where
        id=?
Oct 19, 2023 7:09:18 PM org.hibernate.engine.jdbc.connections.internal.Drive
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost/test]

Process finished with exit code 0
```

Student record is retrieved in one session.

Student record is updated in another session.

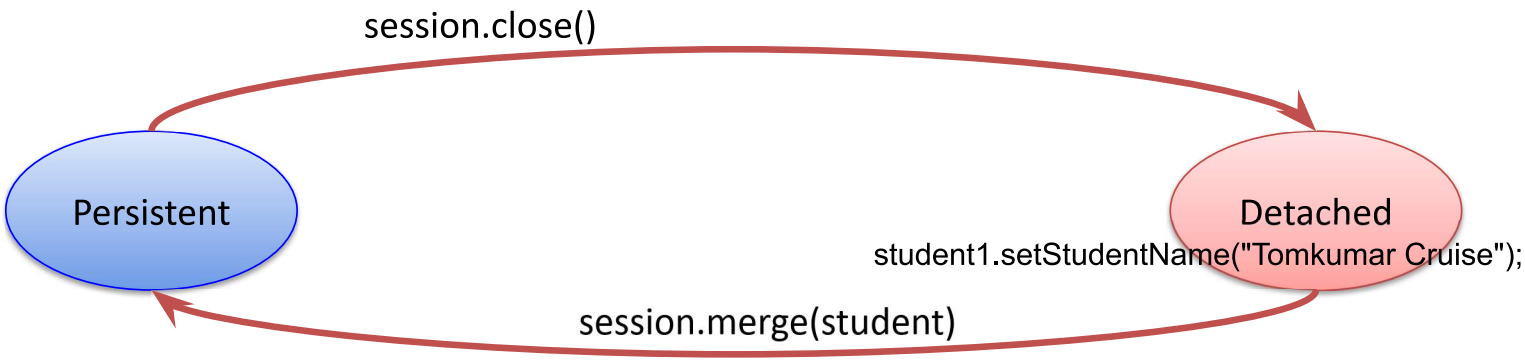
MySQL Workbench



Verify records in the database
We find the record with id 2 has been updated

State Transition for Instance from Detached to Persistent

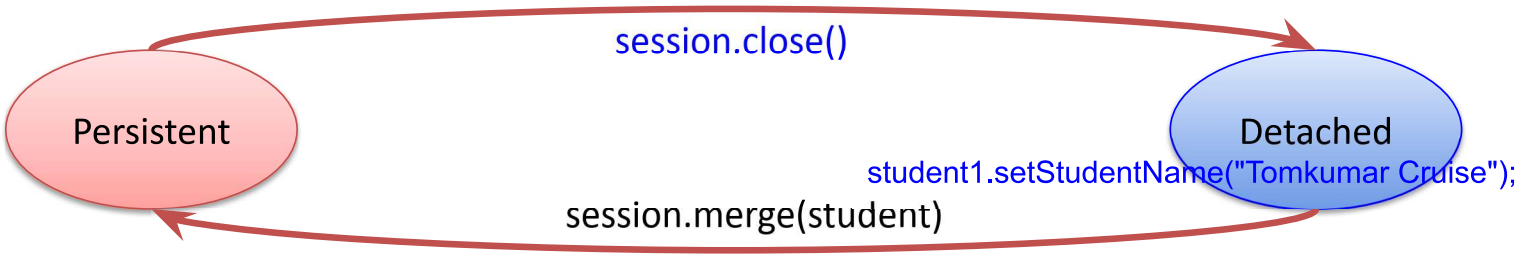
- We first load an existing record in Entity Instance (**Persistence** state).




```
student1 = session1.get(Student.class, 2);
session1.getTransaction().commit();
```


State Transition for Instance from Detached to Persistent

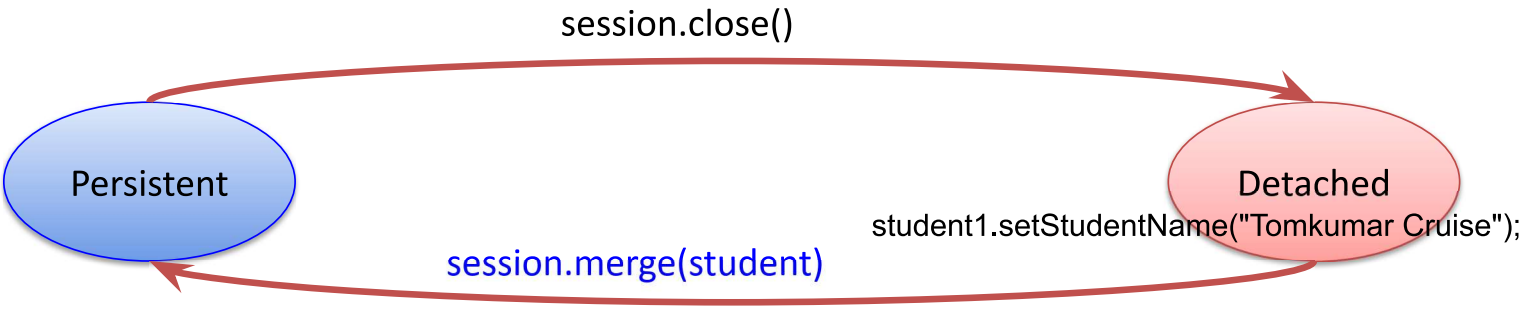
- We then **close** the **session** to make the Instance Detached.




```
student1 = session1.get(Student.class, 2);
session1.getTransaction().commit();
```

State Transition for Instance from Detached to Persistent

- Finally, we make a Detached Instance Persistent using a new session.




```
student1 = session1.get(Student.class, 2);
session1.getTransaction().commit();
```


References

- Java Persistence with Hibernate, Second Edition, 2016, Christian Baver, Gavin King, Gary Gregory, Publisher: Manning.
- Official Documentation https://docs.jboss.org/hibernate/orm/6.3/introduction/html_single/Hibernate_Introduction.html

State Transition Diagram

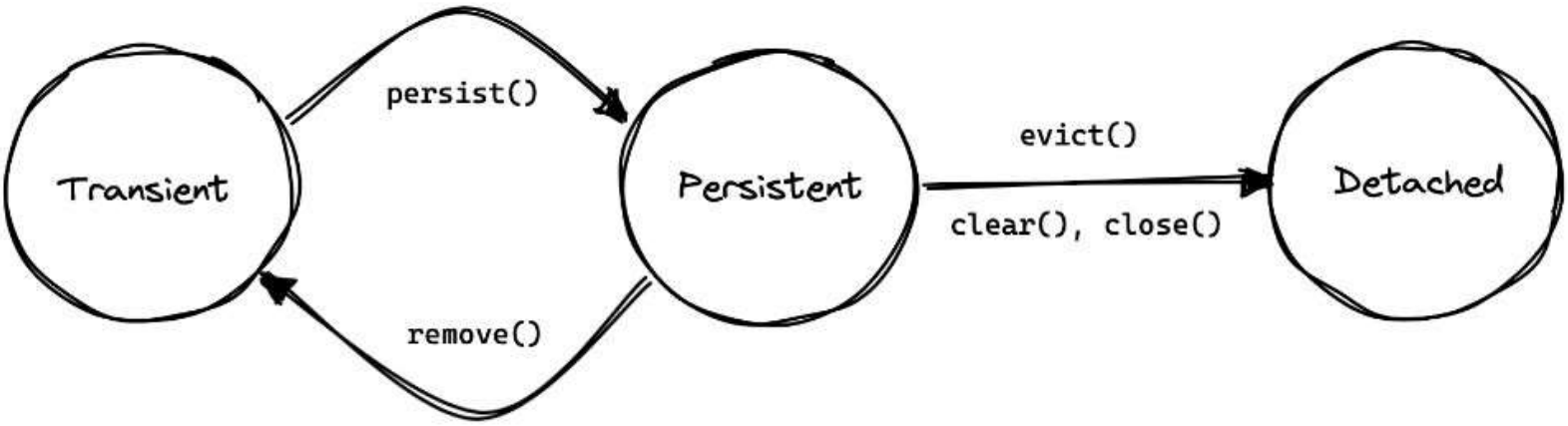


Image Source: https://docs.jboss.org/hibernate/orm/6.3/introduction/html_single/Hibernate_Introduction.html