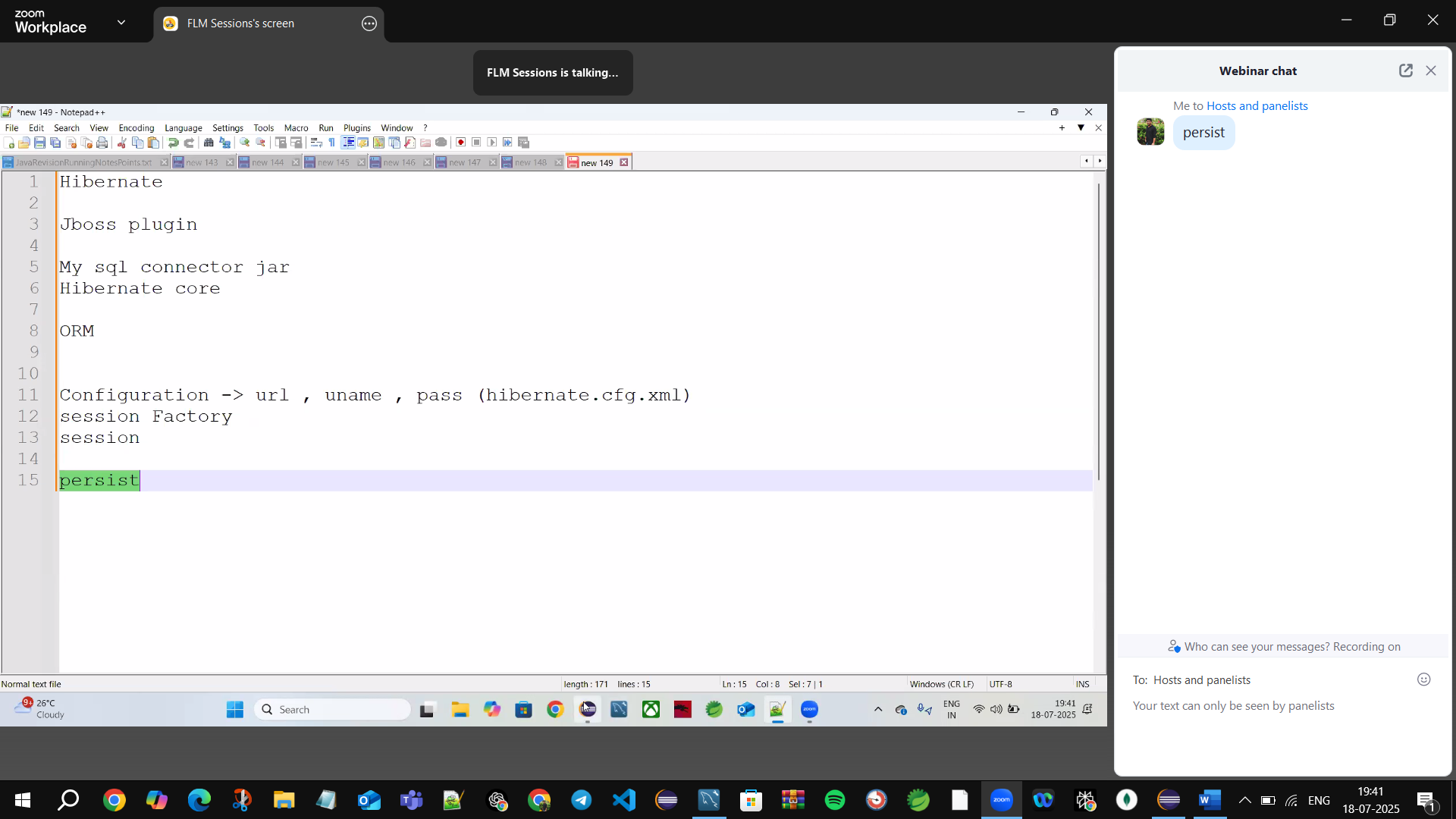
**2.DQL\_DML\_States**



**ORM**🡪**Object relational Mapping**

🡪**Maps** Java objects with SQL Tables

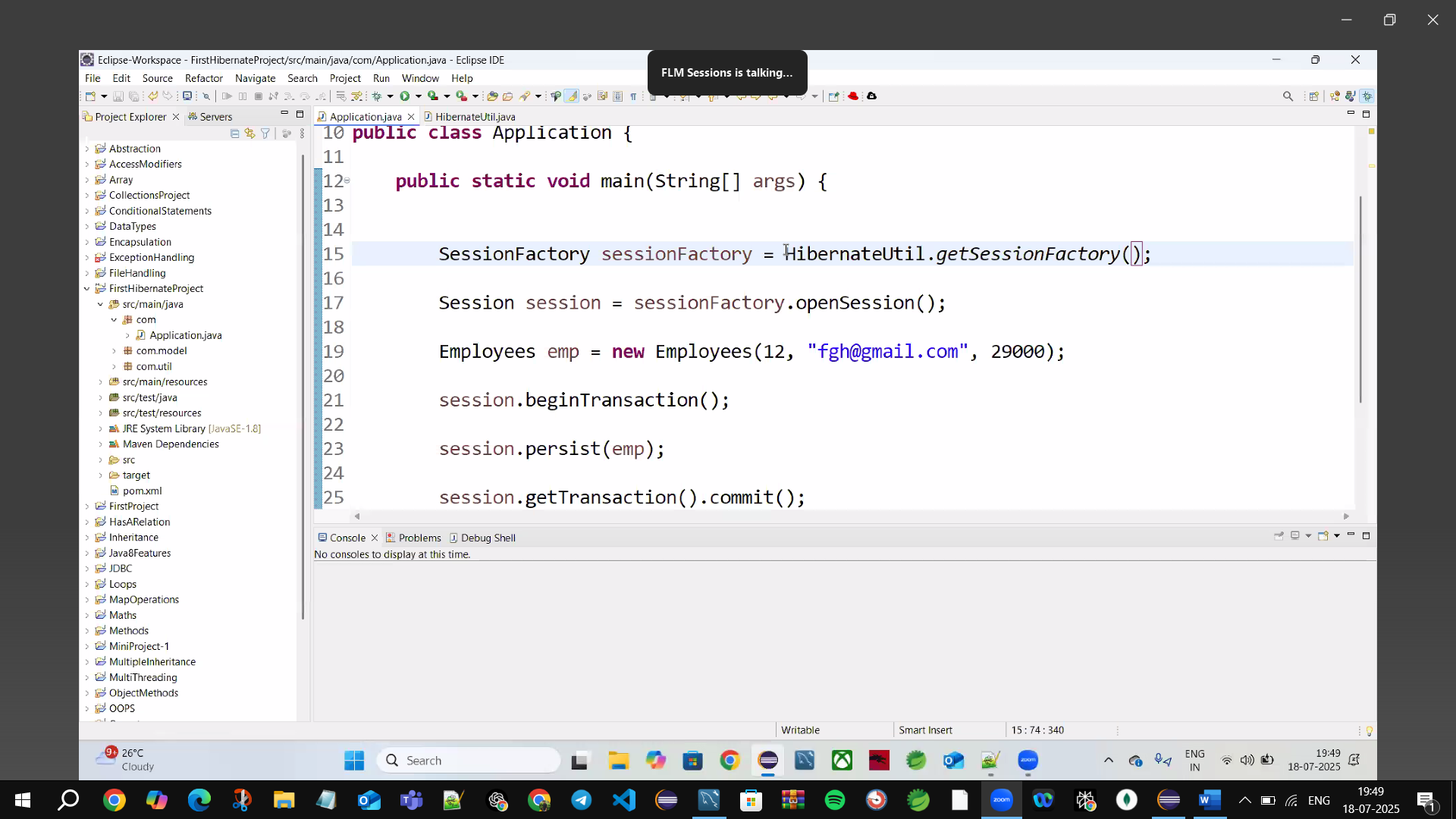
🡪**Session** is used to interact with the databases

Who gives the session?  
🡪Session Factory

Who gives the session Factory?

🡪Configuration🡪needs (URL, username, pass(hibernate.cfg.xml)

**Persist🡪**insert the data

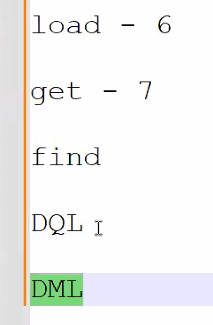


🡪Hibernate by default only support with Primary key (P.K)

**In XML, comments are written using:**

**<!-- Comment goes here -->**





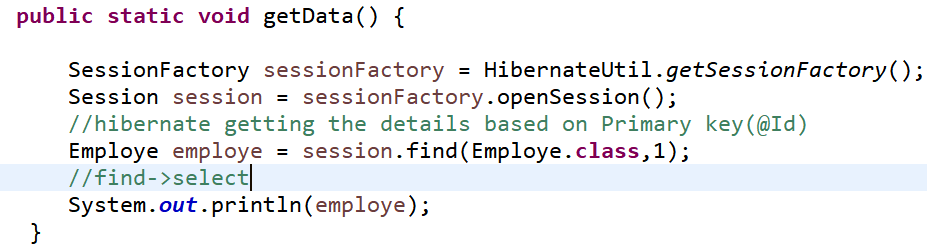
**DQL🡪No changes in a TABLE**

**DML🡪Changes in a Table must do (commit()) just do find()**

load()🡪 works in version 6 but deprecated and replaced with

get()🡪works in version 7 but deprecated and replaced with

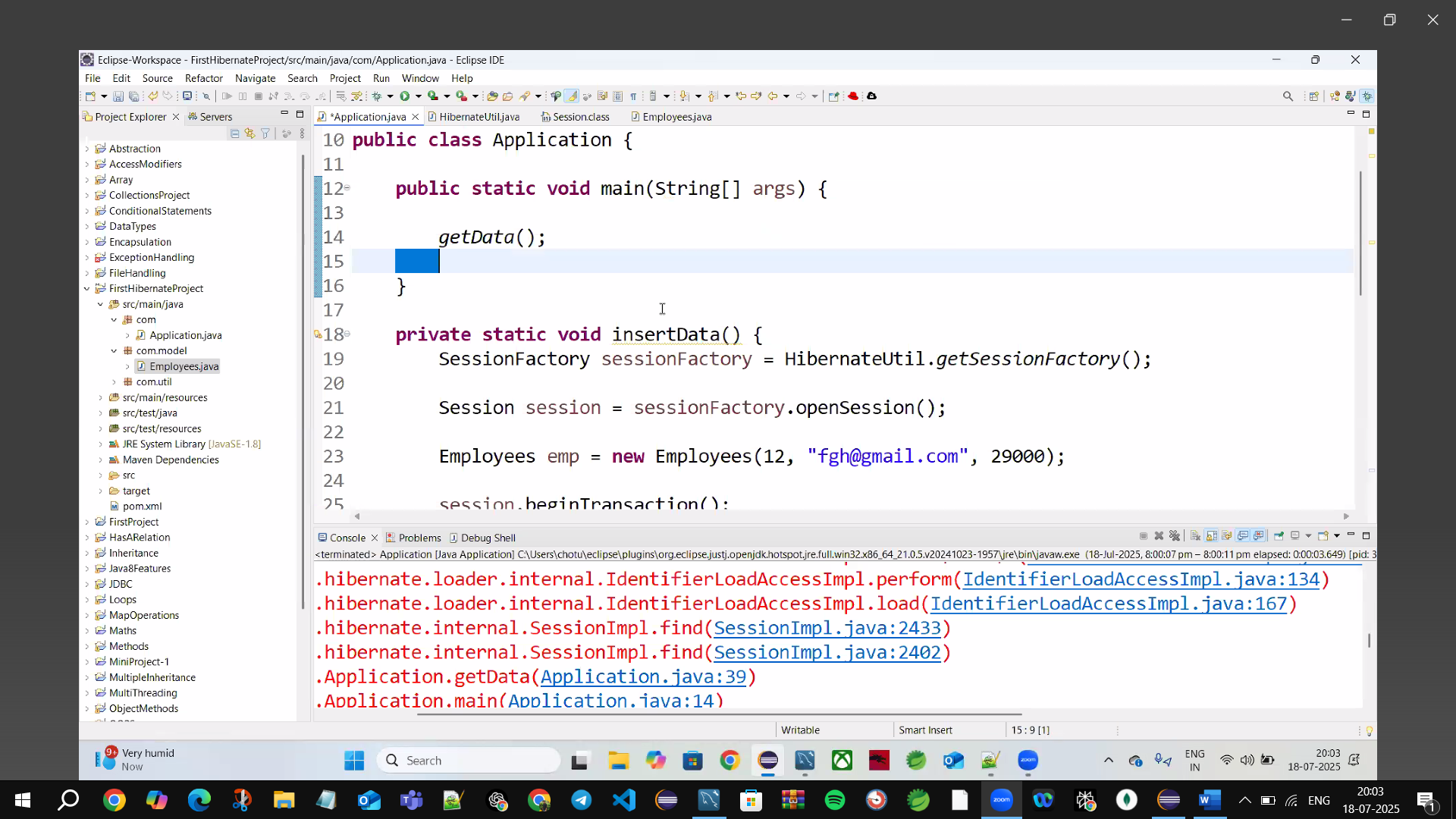
find()🡪selects the data (current)

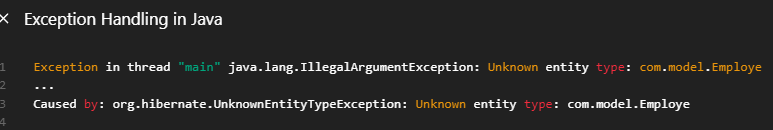




🡪Red colour data= all are Logs

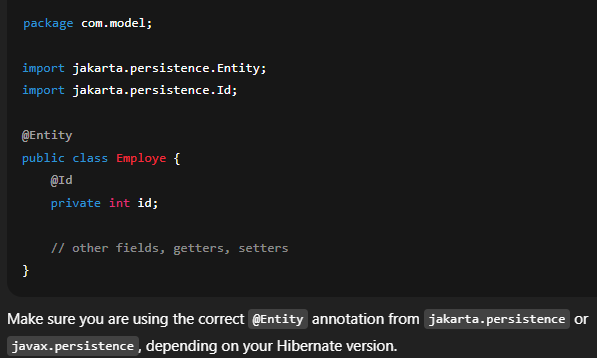
🡪get the employe details based on Primary key(@Id)🡪get details of 1 employe



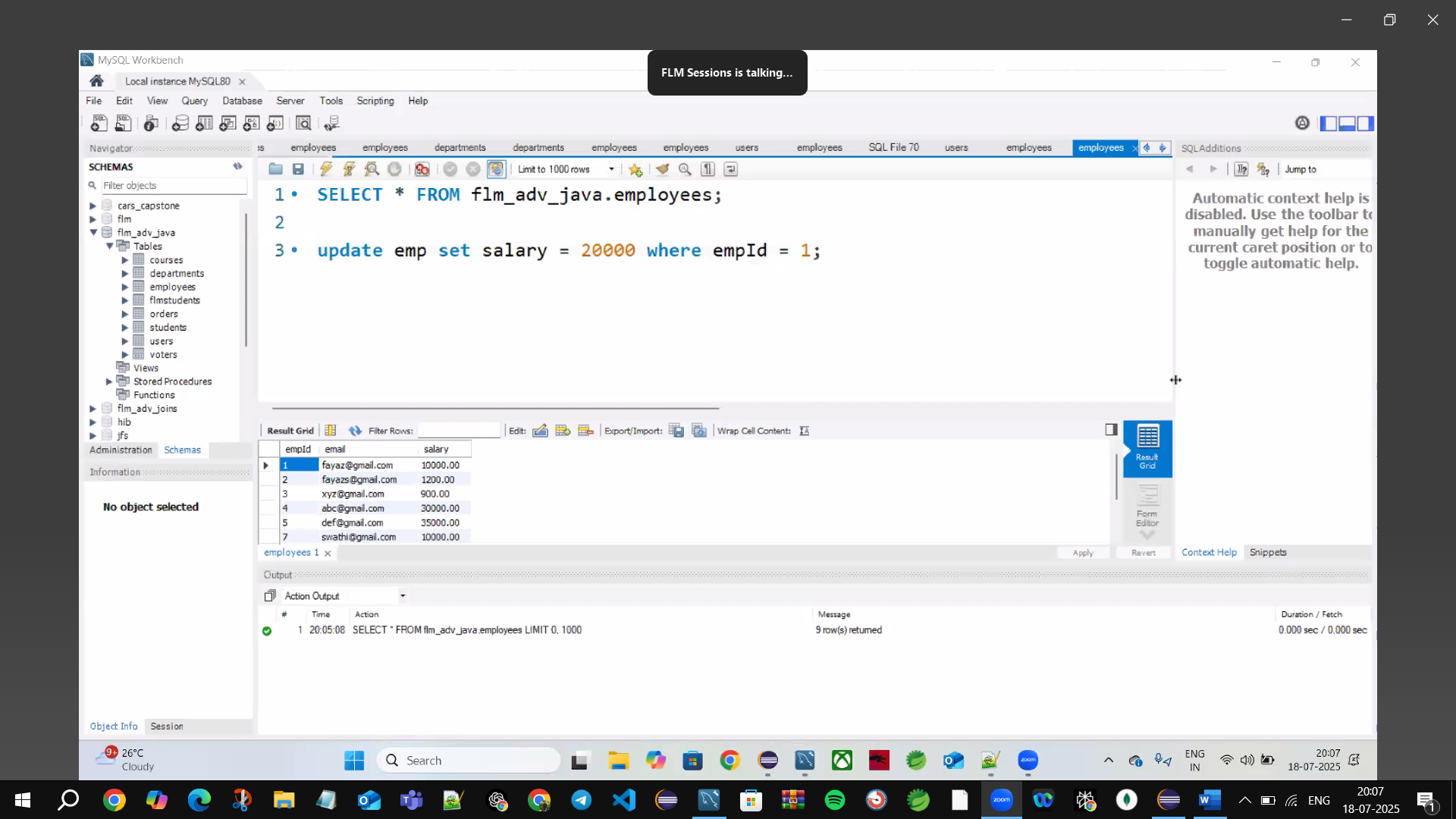


**✅ Final Checklist:**

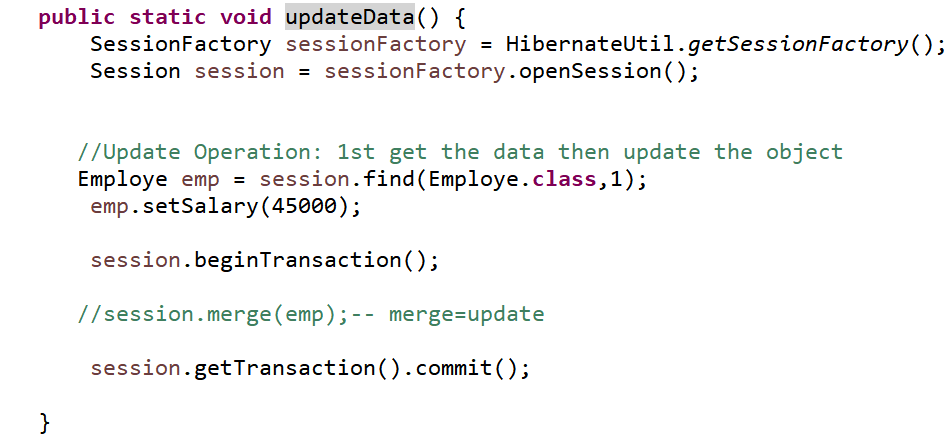
* @Entity annotation present
* Class name is correct (Employe vs Employee)
* @Id field is defined
* Entity is registered in hibernate.cfg.xml or scanned by JPA/Spring

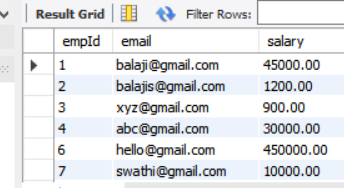


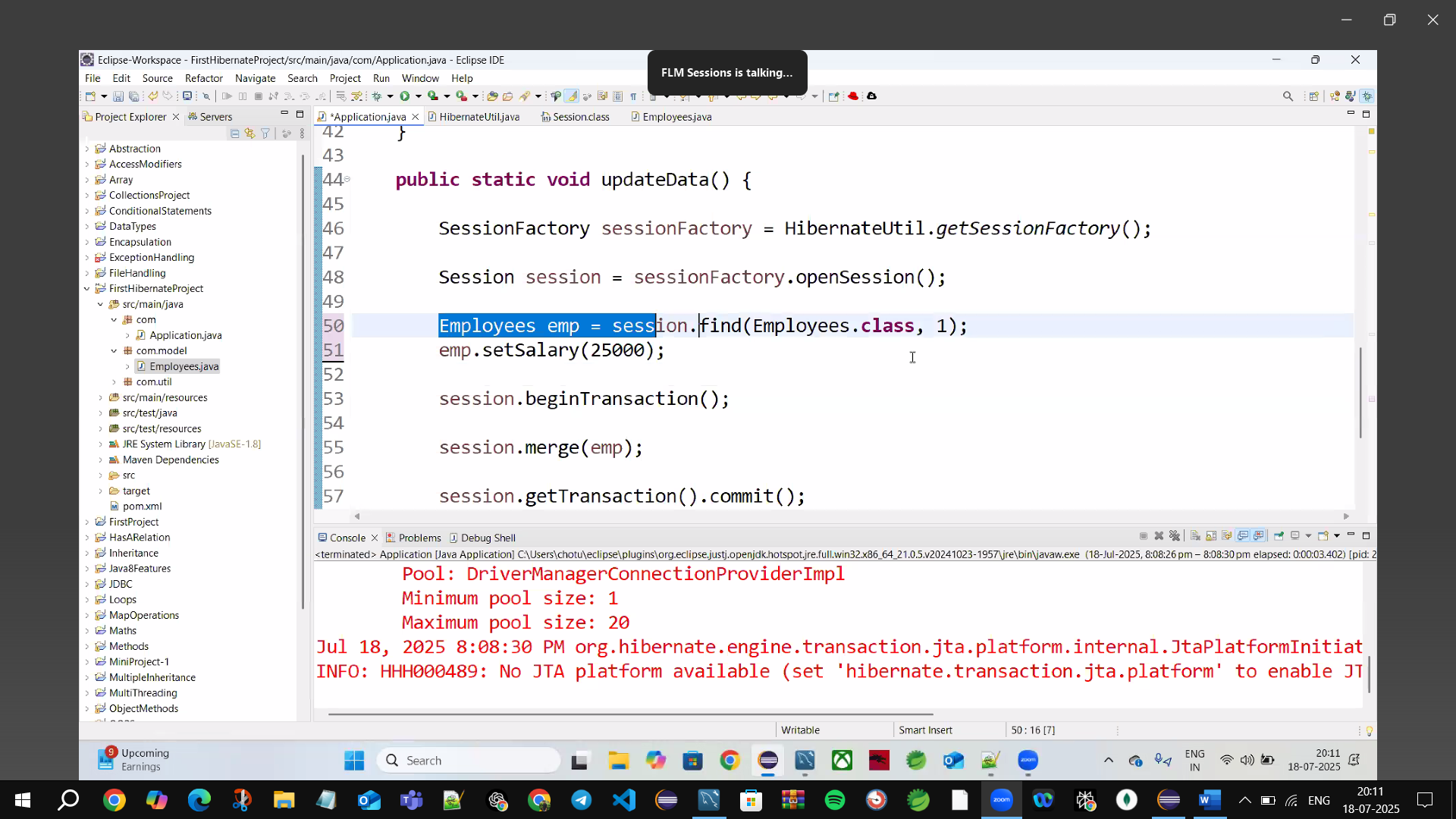
In hibernate Java creates an empty object it will use setters used to set the data Hibernate don’t directly use parameterized constructor

Update salary =20000, where id=1

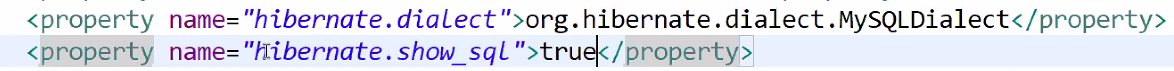
//Update Operation: 1st get the data then update the object







We can show the sql query what its writing inside using **hibernate.show\_sql**

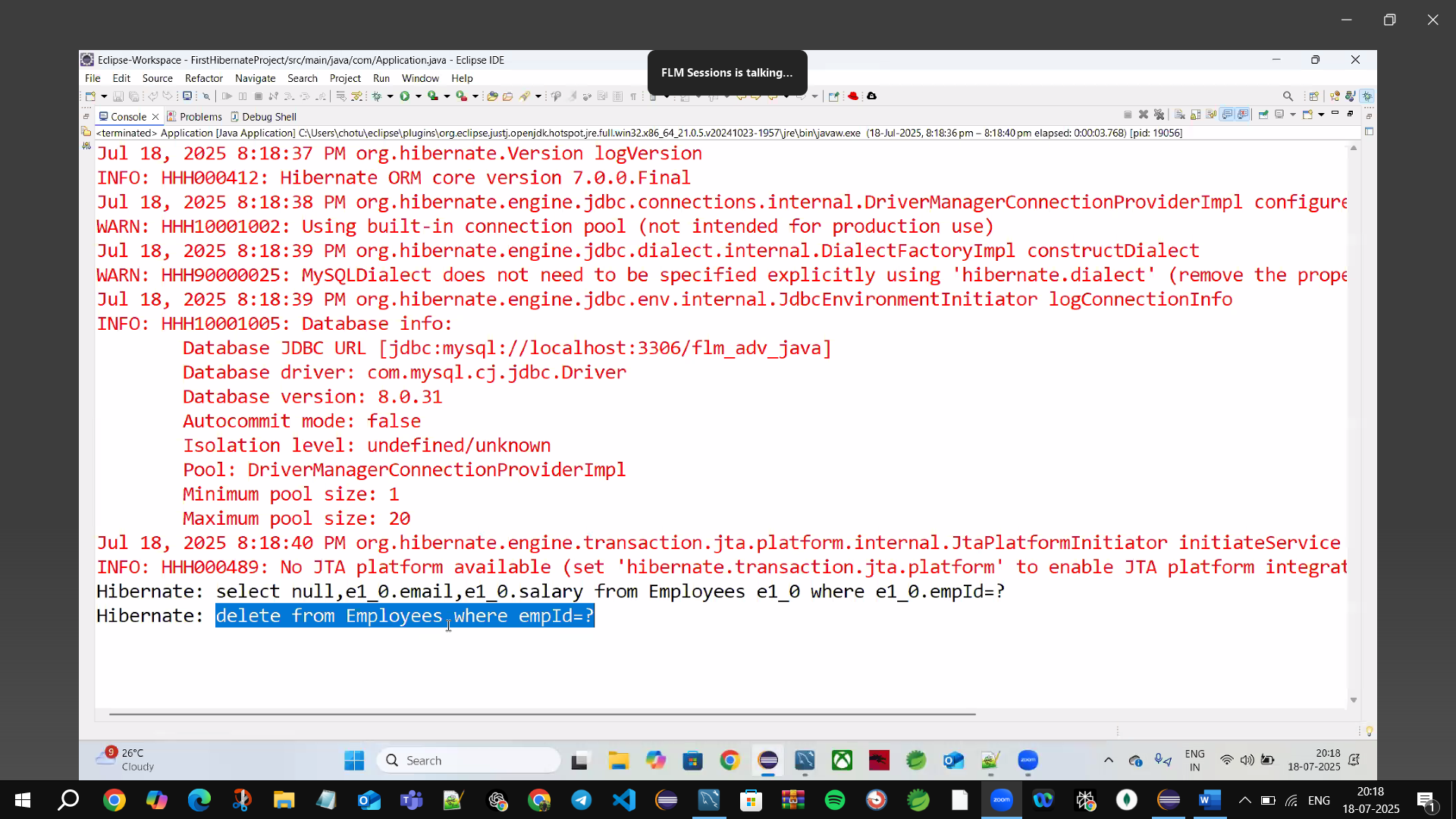


**Delete the employe where employe id = 15**

****

****

**🡪deleteData() 1st select the employe & where empId=? Then deletes**



DML🡪Operations

**persist🡪insert**

**find🡪select**

**merge🡪update**

**remove🡪delete**

**FAMOUS Interview Q’s**

**1.Dirty Checking🡪if you just change the value we needn’t merge back, it will automatically be updated in the Data base (happens by default in hibernate) bcz of persistent**

🡪Whenever we load the db needn’t be update it will automatically be updated

🡪Once we load the db from db if we update it will automatically be updated

🡪Bcz of **persistent** object

**2.States in Hibernate**

🡪3 states

**Transient🡪no relation b/w object & session**

**Creating employe object🡪 Employe emp = new Employe()**

**Eg:🡪 no relation b/w emp object & hibernate session**

**Persistent🡪relation b/w object & session** 🡪**dirty checking only done in Persistent**

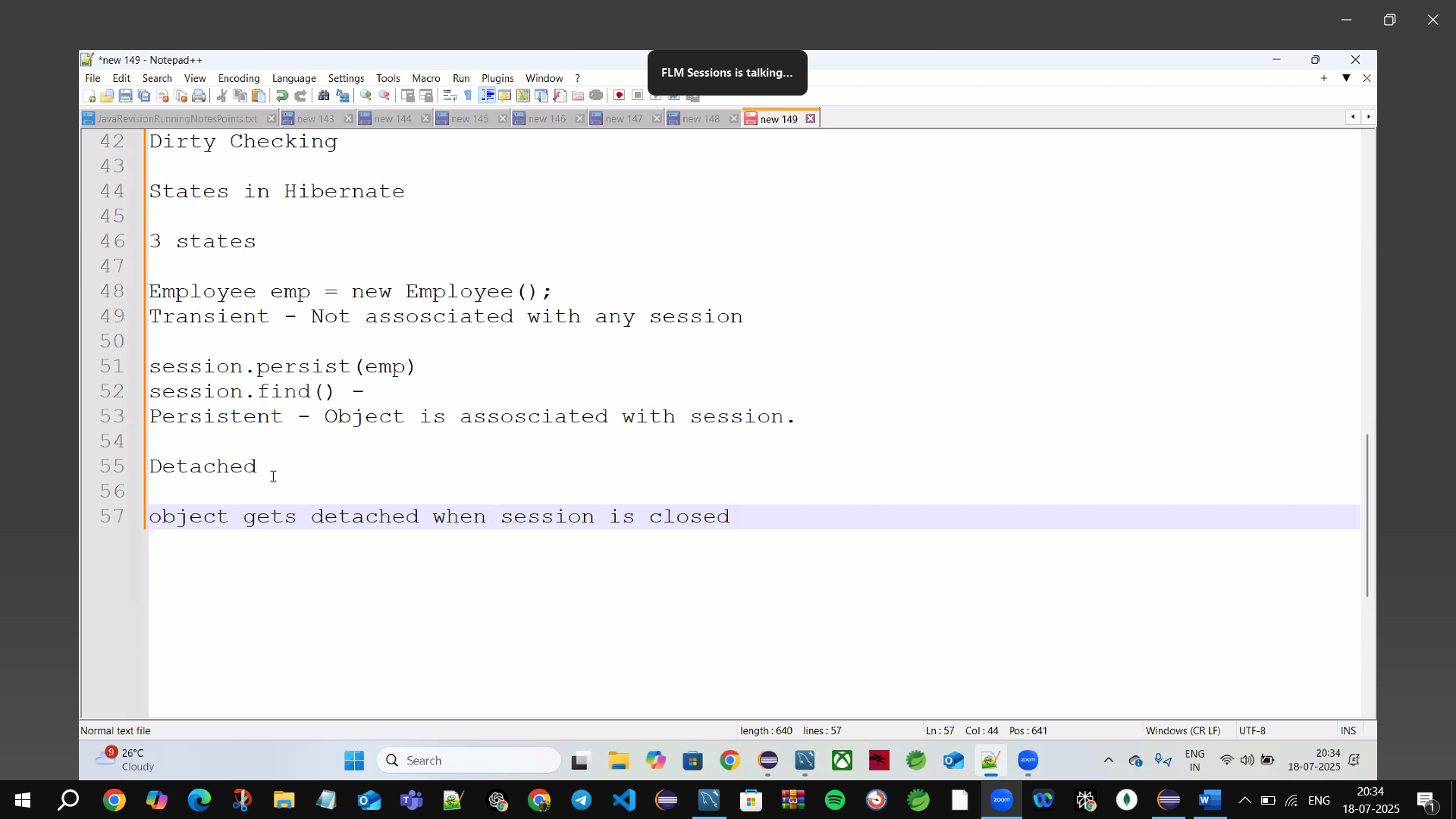
**Eg: session.persist(emp)🡪object moved into persistent state**

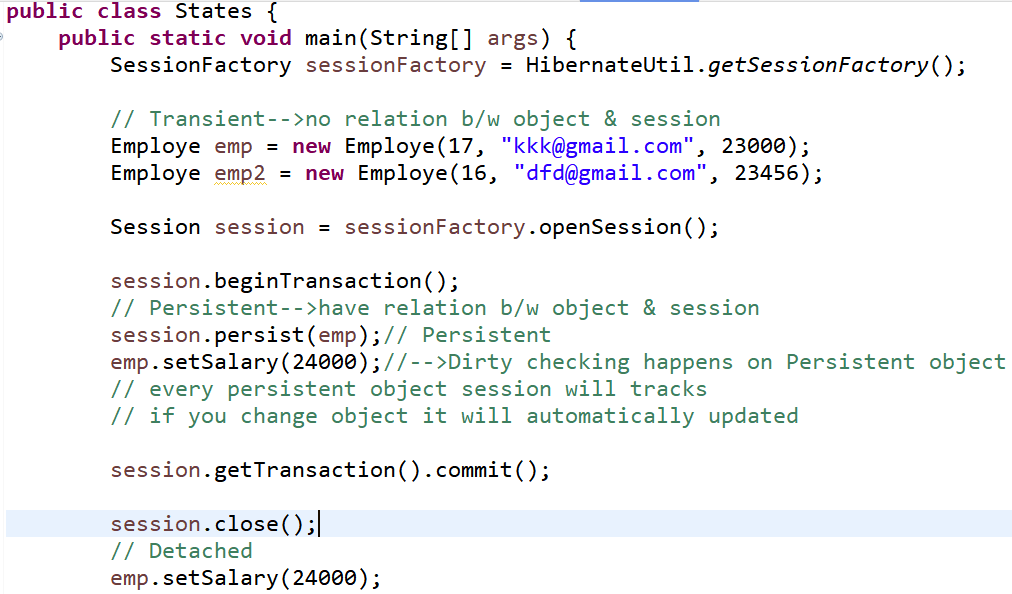
**session.find()🡪also moved into persistent state**

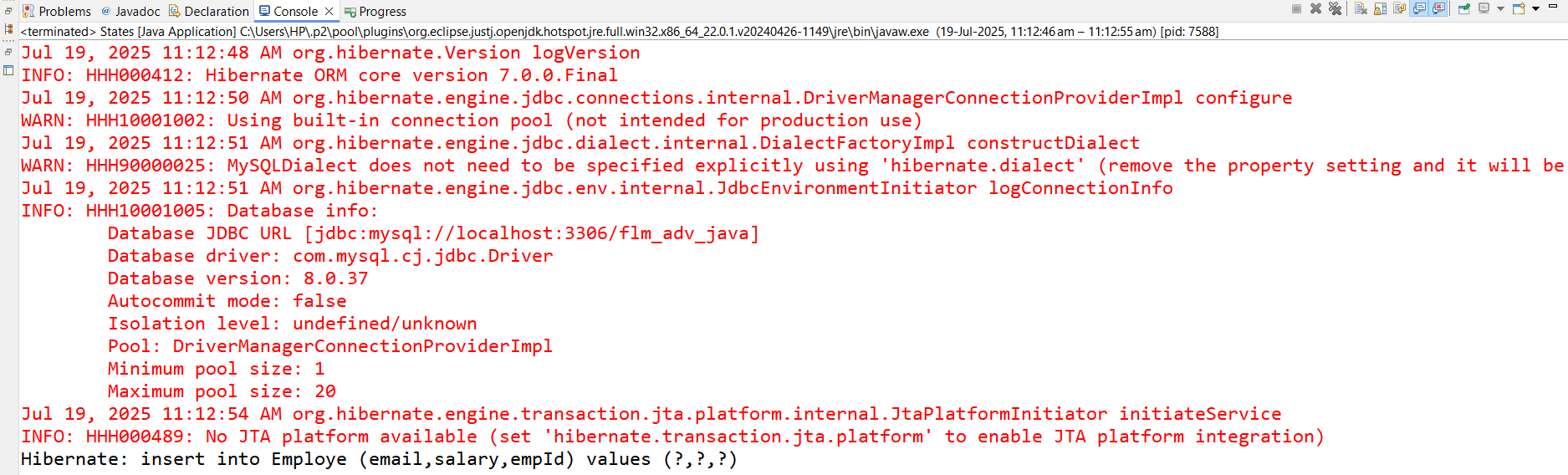
**🡪all the objects which are in Persistent state are prone to Dirty checking**

**Detached🡪object is detached when session is closed**

**🡪unlinked, removed**



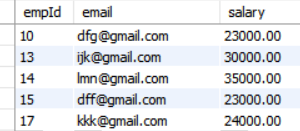




**Transient🡪no relation b/w object & session**

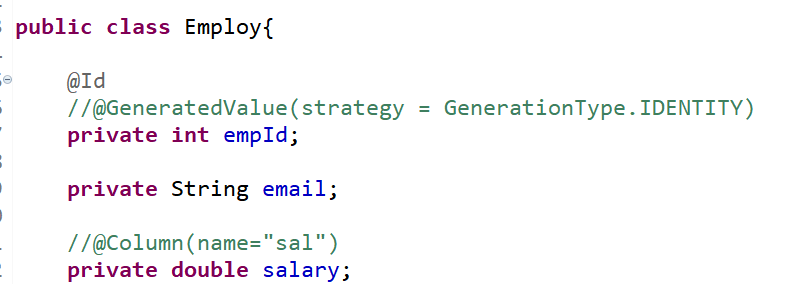
**Persistent🡪relation b/w object & session**

**Detached🡪object is removed when session is closed**

****

**17🡪kkk@gmail.com (inserted 23000 but Dirty checking updates automatically)🡪24000**

****

****

**🡪If class name not match with TABLE name get sqlGrammer exception**

****

**🡪to fix this use @ Table(name=”employe”)**

**@Entity🡪maps java Employe class to SQL Employe Table**

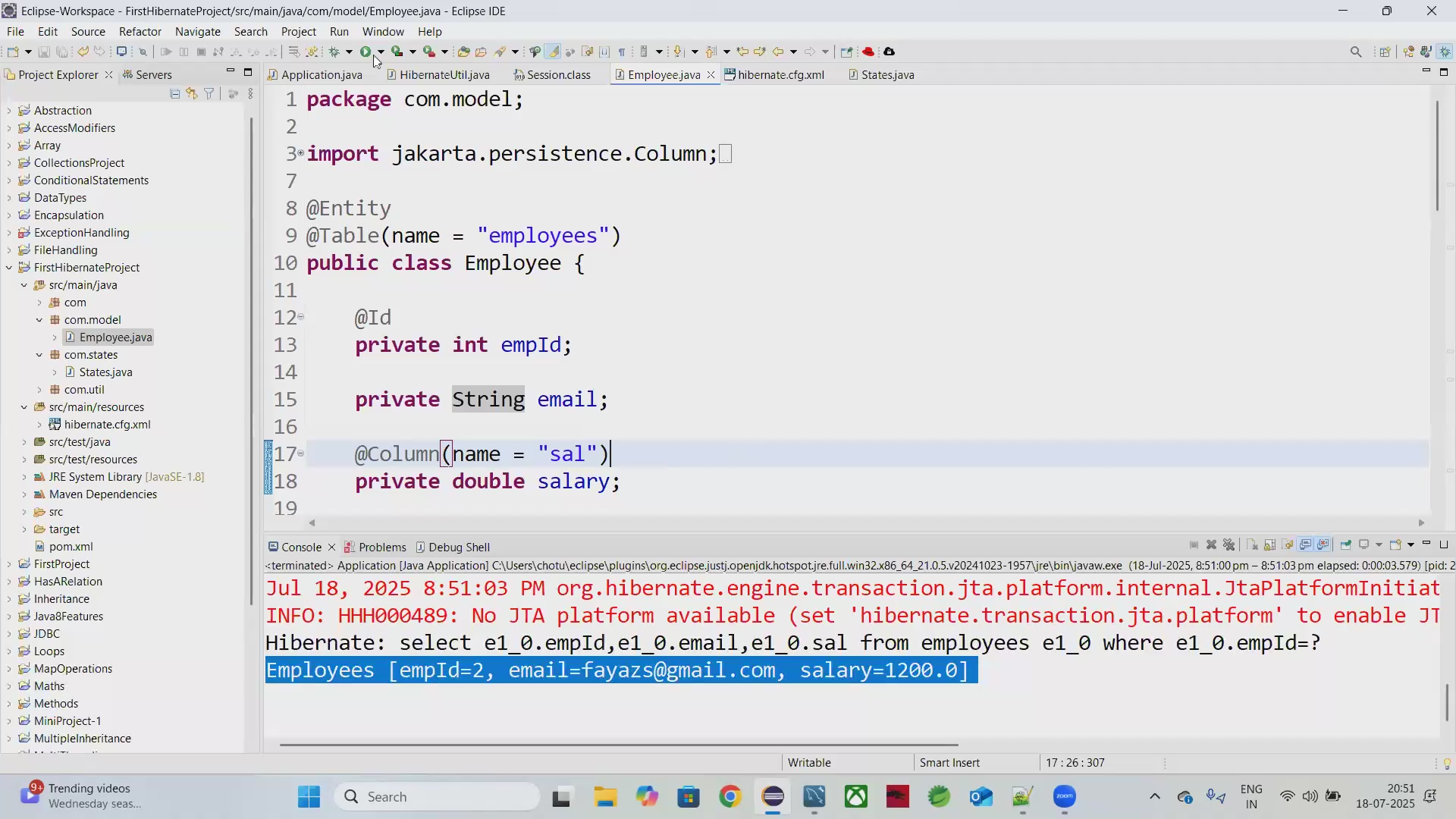
**@Id🡪primary key of the Employe table**

**Get data() from the Employe table**

**🡪if column name not match with TABLE name get sql Grammer exeception**

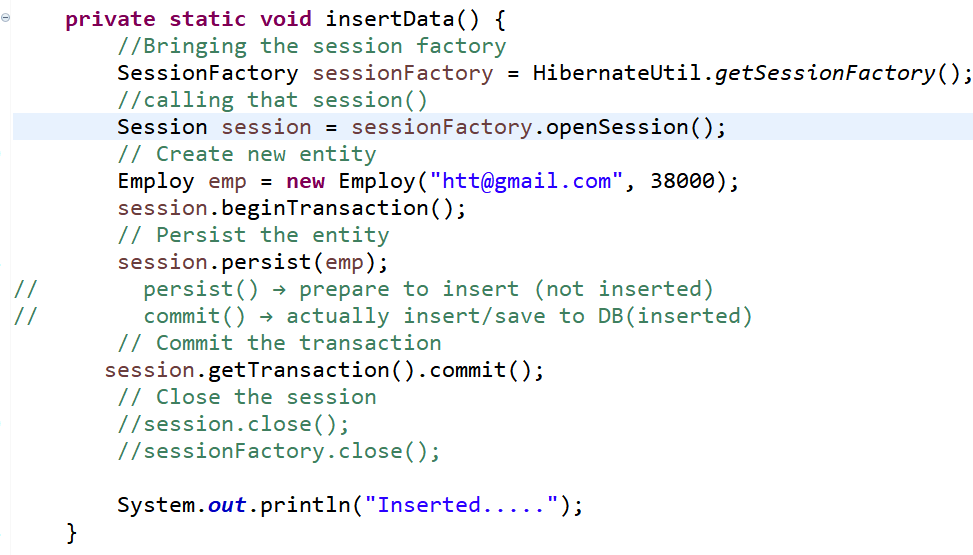
****

**🡪to fix this use @Column(name=”salary”)**



**🡪Tabel supports AutoIncrement**

**🡪Inserted the data without giving Id (Id will automatically generated)**

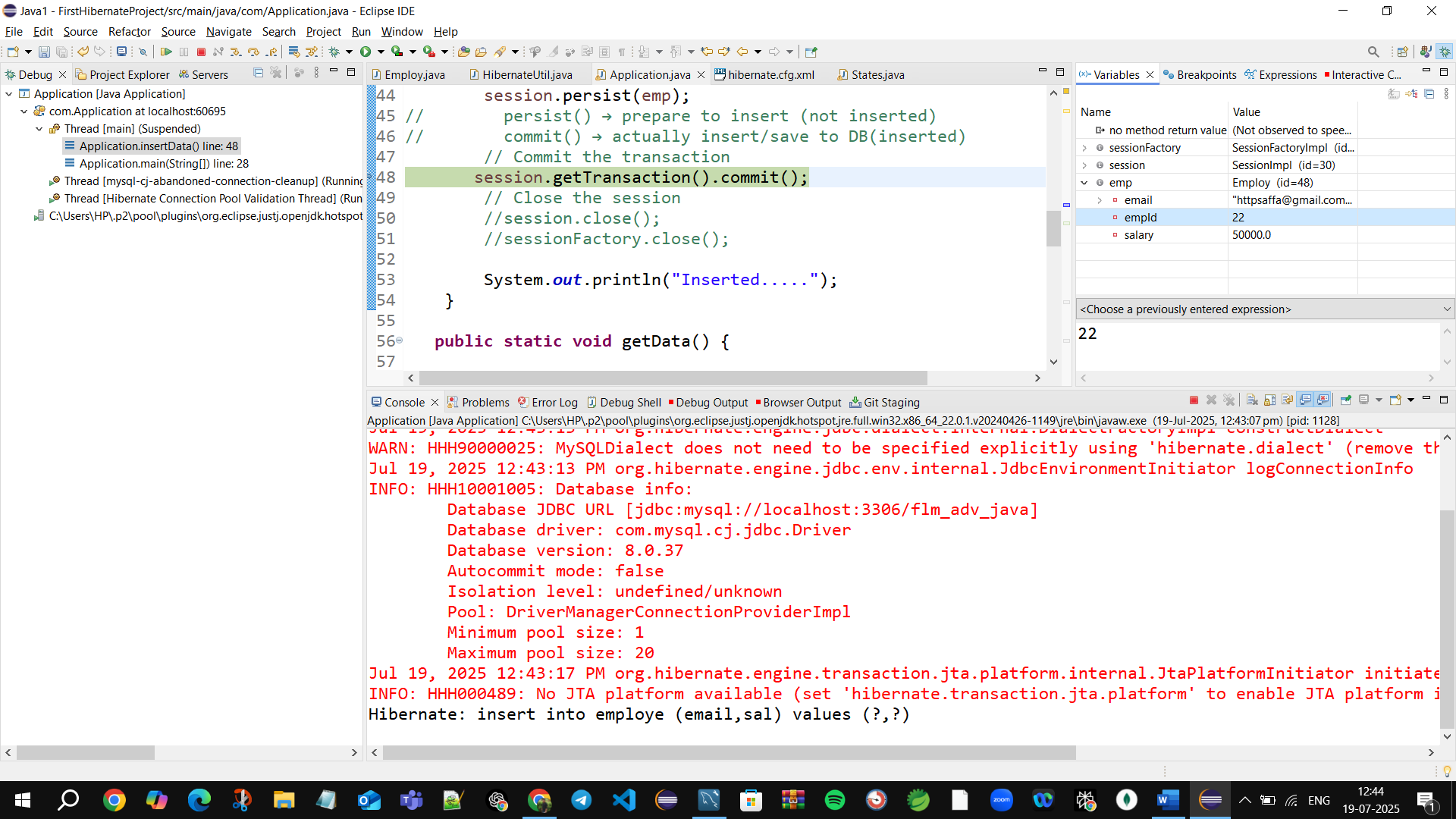


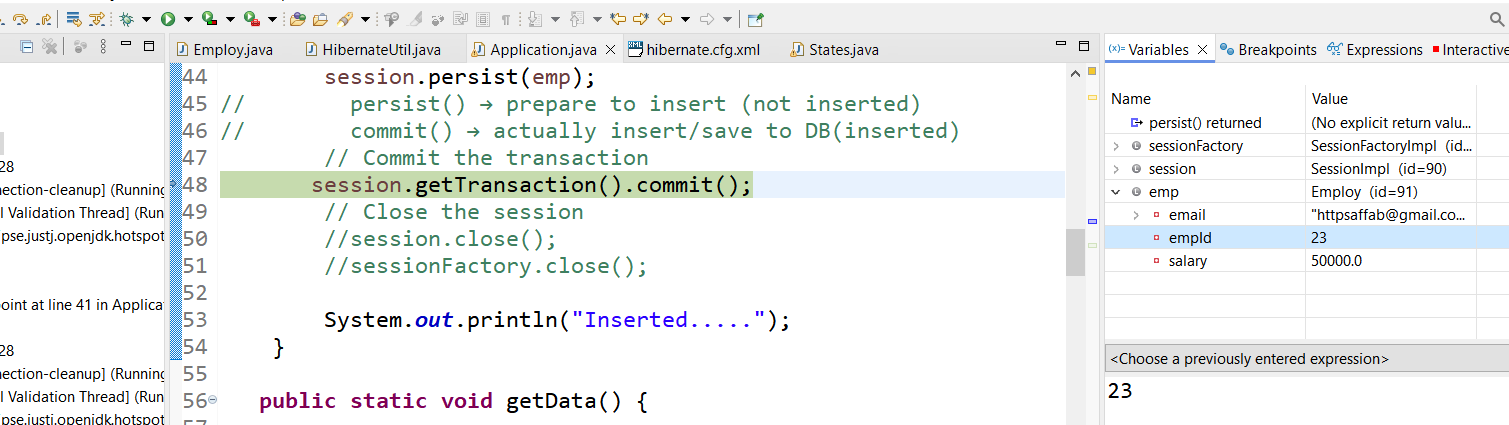
🡪18 is inserted id automatically

🡪id is given by SQL

**🡪to bind autoincrement of SQL with Java we use🡪 @GeneratedValue**

**🡪SQL is generated Autoincrement values & binding those values with Java with help of @GeneratedValue**

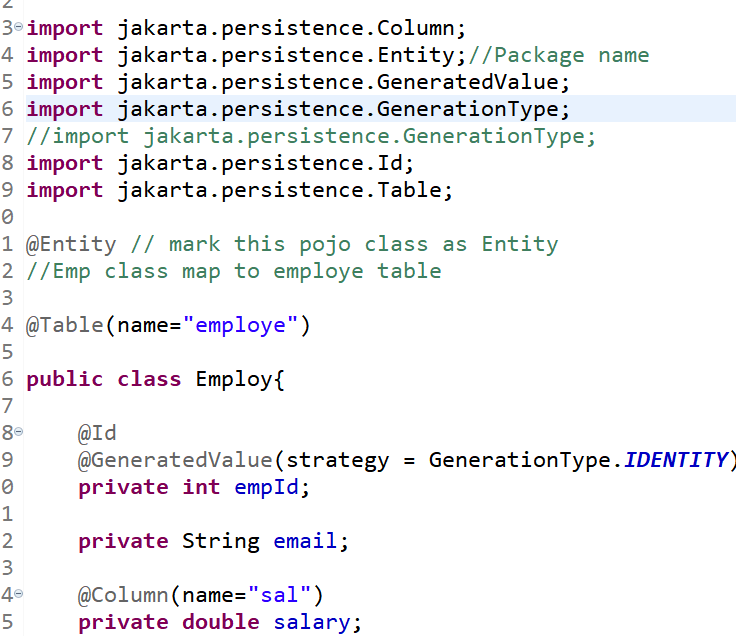


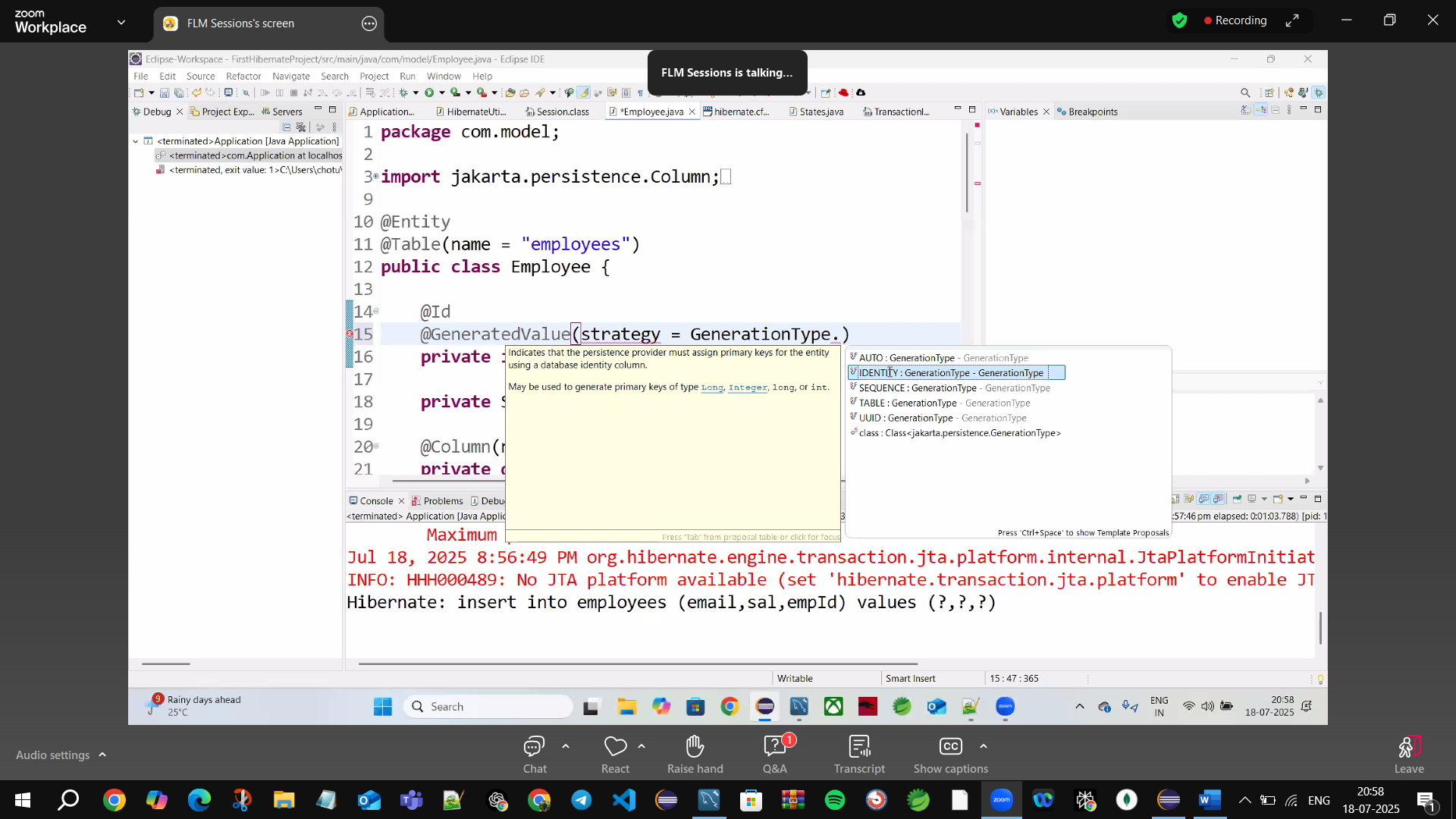
****

**Press f6** ****to next step

**@GeneratedValue not generated new value it only binds SQL generated values to the java**

**🡪must be autoincrement features in SQL**

****



@GenerateValue🡪binding SQL autoincrement to Java

🡪bring back of SQL autoincrement to java

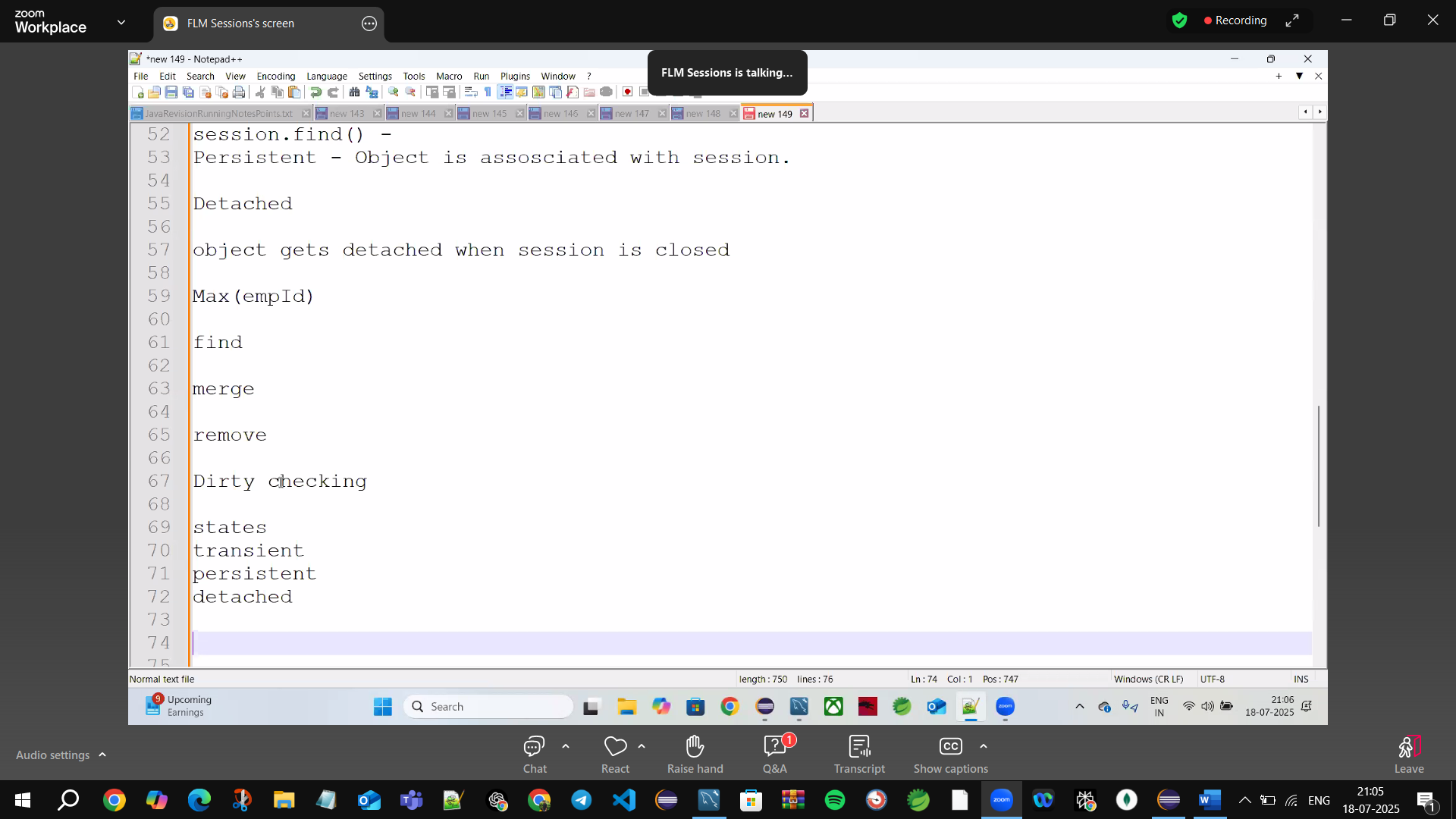
🡪not generating the value

🡪Autoincrement values 🡪can’t roll back

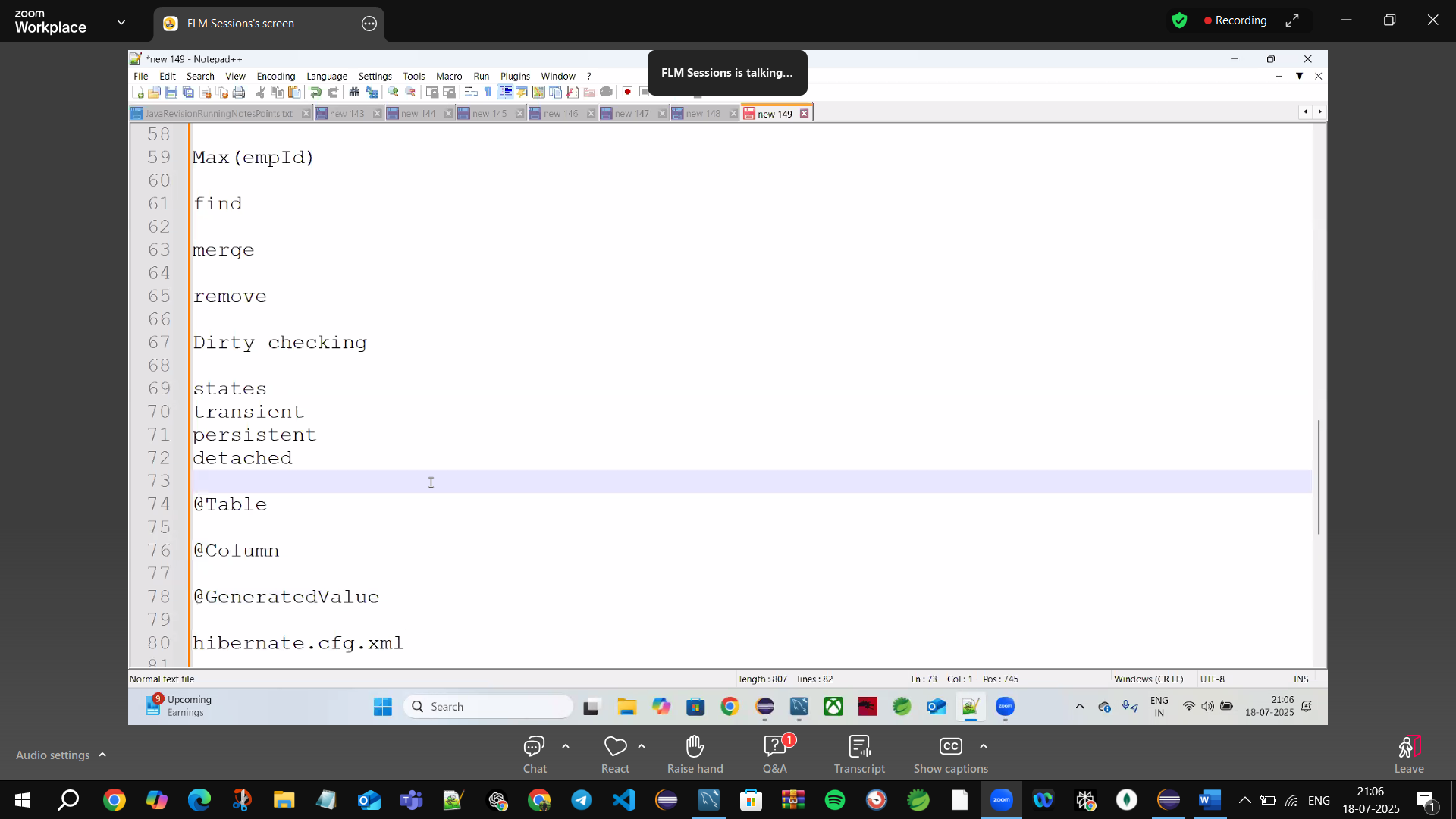
By default it search for 🡪hibernate.cfg.xml file

**🡪if you change hibernate.cfg2.xml give cfg.configure(“hivernate.cfg2.xml”)**

****



**🡪cfg()🡪automatically searches for hibernate.cfg.xml file**



@Table🡪different table

@Column🡪different columns

Org.hibernate.orm

Org.hibernate 🡪moved to 🡪Org.hibernate.orm