Hibernate + JPA + JDBC : Diagram की Complete Summary

1️⃣ Application Start पर क्या होता है

- Application start होते ही

  `EntityManagerFactory (EMF)`

  create होता है।
- EMF:
  - **thread-safe** होता है
  - सभी `@Entity`, `@Table`, `@Column` annotations पढ़ता है
  - Table, column, relation, ID आदि की **metadata जानकारी** रखता है
- EMF:
  - JDBC connection pool को **contain नहीं करता**
  - सिर्फ **DataSource / connection pool का reference** रखता है

👉 इस stage पर:

- कोई JDBC connection नहीं बनता
- कोई SQL execute नहीं होती

2️⃣ Runtime पर Thread और EntityManager

- हर request / transaction के लिए:
  - एक नया **EntityManager (EM)** बनता है
- EM:
  - **thread-safe नहीं होता**
  - सिर्फ उसी thread से जुड़ा रहता है
- EM के अंदर:
  - एक **Persistence Context (PC)** बनता है

3️⃣ Persistence Context (PC) का काम

- PC Hibernate का **first-level cache** है
- PC:
  - Entity objects को memory में रखता है
  - उनकी state (managed / dirty) track करता है
- जब आप लिखते हैं:

```
Product p = new Product();
em.persist(p);
```

तो:

- p object PC के अंदर चला जाता है
- DB अभी hit नहीं होता

4 ActionQueue का रोल

- `em.persist()`, `update`, `delete` पर:
  - Hibernate तुरंत JDBC call नहीं करता
- बल्कि:
  - **ActionQueue** में DB operations जमा करता है (INSERT / UPDATE / DELETE actions)
- ActionQueue:
  - एक internal Java collection है
  - DB को भेजने से पहले operations को **buffer** करता है

5 Transaction Commit पर असली काम

जब आप लिखते हैं:

```
tx.commit();
```

Hibernate internally:

1. **flush()** करता है
2. ActionQueue से actions उठाता है
3. EMF की metadata से:
   - SQL **runtime पर generate करता है**
4. Hibernate के अंदर पहले से लिखे:
   - **JDBC code** को call करता है
5. JDBC driver (MySQL) के ज़रिए:
   - Connection लेता है
   - PreparedStatement बनाता है
   - SQL execute करता है
6. काम पूरा होते ही:
   - Connection release हो जाता है

6 JDBC और Bytecode को लेकर साफ़ सच्चाई

- JDBC code:
  - Hibernate JAR के अंदर **पहले से लिखा हुआ** होता है

- Runtime पर generate नहीं होता
- Bytecode manipulation से नहीं बनता

- Bytecode manipulation:
  - सिर्फ entities (lazy loading, proxy, dirty checking) के लिए होती है
  - JDBC के लिए नहीं

7️⃣ पूरे Diagram का Final Flow (एक लाइन में)

```
EMF (metadata + datasource ref)
   ↓
EntityManager (per thread)
   ↓
Persistence Context (entities)
   ↓
ActionQueue (pending DB actions)
   ↓
flush / commit
   ↓
Hibernate internal JDBC code
   ↓
JDBC Driver
   ↓
Database
```

Final Statement

Hibernate में JDBC code पहले से लिखा हुआ होता है।
Application start पर Hibernate JPA annotations से metadata बनाता है।
Runtime पर EntityManager और Persistence Context entities को manage करते हैं।
Database operations पहले ActionQueue में जमा होते हैं।
Transaction commit पर Hibernate metadata से SQL generate करता है
और pre-written JDBC code के ज़रिए DB में execute करता है।