Hibernate Concepts Notes

1. Hibernate

- Hibernate is an ORM (Object Relational Mapping) framework in Java.

- Maps Java classes (objects) to relational database tables.

- Eliminates boilerplate JDBC code and provides features like caching, lazy loading, transactions, etc.

2. ORM (Object Relational Mapping)

- Technique to map Java objects → database tables and object fields → table columns.

- Benefits:

- No manual SQL for basic CRUD.

- Database-independent code.

- Supports inheritance, associations, caching, and transactions.

3. Session

- A lightweight, non-thread-safe object used to interact with the database.

- Represents a unit of work between application and DB.

- Provides CRUD operations, query execution, transaction handling.

- Closed after use.

- Not thread safe → Each thread should have its own Session.

4. SessionFactory

- A heavyweight, thread-safe object.

- Built once per database and used to open Session objects.

- Stores metadata & connection pool.

- Recommended to keep only one SessionFactory per DB in an app.

5. Transaction

- Ensures ACID properties.

- In Hibernate:

Transaction tx = session.beginTransaction();

// operations

tx.commit();

- If error occurs → tx.rollback().

6. hbm2ddl (hibernate.hbm2ddl.auto)

Controls schema generation/update:

- validate → Validate schema against entities (no changes).
- update → Update schema (risky in prod).
- create → Drops and recreates schema each time.
- create-drop → Drops schema after SessionFactory close.
- none → No action.

■■ In production → Avoid update/create-drop, can lead to data loss.

7. @Embeddable

- Used for value types (composite fields, not entities).
- Example:

```
@Embeddable
class Address {
String city;
String state;
}

@Entity
class Employee {
@Embedded
Address address;
}
```

8. Associations

- OneToOne → One entity linked to one entity.
- OneToMany → One entity linked to many.
- ManyToOne → Many entities linked to one.
- ManyToMany → Many entities linked to many.

9. mappedBy

- Used in bidirectional relationships to tell Hibernate which side owns the relationship.

10. CascadeType

Defines what happens to related entities when parent is updated/deleted:

- ALL → Applies all operations.
- PERSIST → Save parent also saves children.
- REMOVE → Delete parent also deletes children.
- MERGE → Updates propagate.
- REFRESH, DETACH.

11. Eager vs Lazy Loading

- Eager → Loads related entities immediately with parent (JOIN).
- Lazy (default for collections) → Loads related entities only when accessed (proxy used).

12. Get vs Load

- get():
- Immediate fetch from DB.
- Returns null if entity not found.
- load():
- Returns a proxy (lazy loading).
- If entity not found → ObjectNotFoundException.
- In Hibernate 6, behavior changed.

13. Hibernate 6 - load() Changes

- Hibernate 6 made load() void-returning for entity-by-id.
- Old:

```
Flight f = session.load(Flight.class, 1); // proxy
```

- New (Hibernate 6):

```
Flight f = new Flight();
session.load(f, 1); // populates into given entity instance
```

- get() is now recommended for fetching by id.
- load() mainly for natural id / proxy initialization.

14. First-level Cache

- Session-level cache (default).
- Same session → Repeated fetch of same entity → returns from cache, no new SQL query.
- Cleared when session closes.

15. Second-level Cache

- SessionFactory-level cache (optional).
- Shared across sessions.
- Providers:
- EhCache (older, widely used).
- Caffeine (modern, high-performance).
- Needs config in hibernate.cfg.xml or application.properties.