




Trainer: Nilesh Ghule

Wake up from Hibernate, Spring up!!!



Agenda

- Auto-generated primary key ✓
- Composite primary key ✓
-  Hibernate relations ✓
 - @OneToMany ✓
 - @ManyToOne ✓
 - @ManyToMany ✓
 - @OneToOne ✓
 - HQL Joins ✓



Auto-generate Primary keys - JPA compliant

- @GeneratedValue annotation is used to auto-generate primary key. ^{→ JPA}
- This annotation is used with @Id column.
- There are different strategies for generating ids.

◉ AUTO: Depends on database dialect.

- MySQL 8: id will be taken from "next_val" column of "gen" table.
- IDENTITY: RDBMS AUTO_INCREMENT / IDENTITY

- MySQL 8: id will be AUTO INCREMENT

◉ TABLE: Dedicated table for PK generation → for multiple entities

- MySQL 8: @TableGenerator(name="gen", initialValue=1000, pkColumnName="book_ids", valueColumnName="id", table="id_gen", allocationSize=1)

• SEQUENCE: RDBMS sequence using @SequenceGenerator

- MySQL 8: Emulated with table.

generator
→ name =
→ strategy =
1 row 1 col (next_val)
↑
hibernate_sequence (default)

→ insert into table(,,-) values(-,-,-)

id_gen	
en	next
book	1
customer	10
order	14

↳ oracle
pg-sql



Auto-generate Primary keys - Hibernate Specific

→ org.hibernate

- @GenericGenerator is used to specify config of hibernate's id generation strategies.
- Hibernate allows few more strategies.
 - native: depends on db/dialect. → like AUTO
 - guid: 128-bit unique id → globally unique id (nic no + time)
 - hilo: id gen using high low algo.
 - identity: auto incr or identity col. → like identity x
 - sequence: db seq. → sequence like
 - increment: select max(id)+1 from table.
 - foreign: id corresponding to foreign key @PrimaryKeyJoinColumn.

1 ✓
2 ✓
3
4
:
:
100 -
101 ✓



Composite Primary key

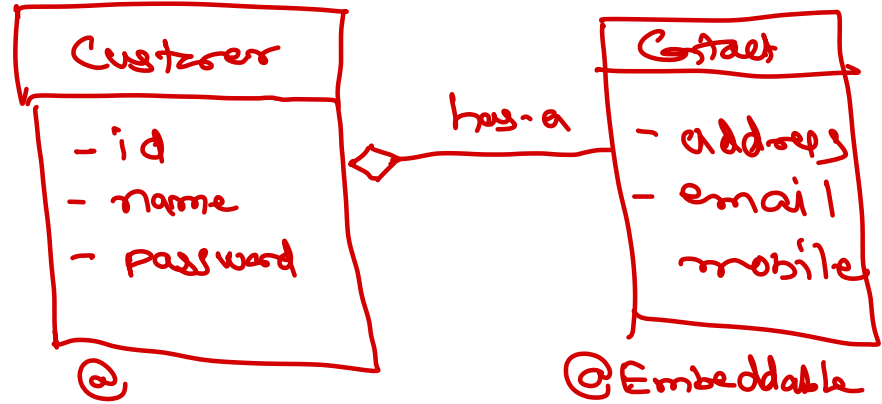
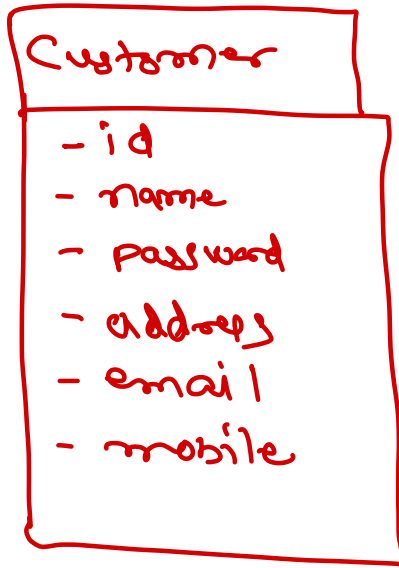
- Hibernate @Id is always serializable. field
- In case of composite PK, create a new class for PK and mark as @Embeddable.
- Create object of that class into entity class & mark it as @EmbeddedId.
- @Embeddable class must implement equals() and hashCode().

→ int / string -- primitive types - Serializable.
session.set (Entity.class, id);
↑
only one in an entity.
Serializable

Students

Std	roll	name	mark
1	1	A	1
1	2	B	1
1	3	C	1
2	1	X	1
2	2	Y	1

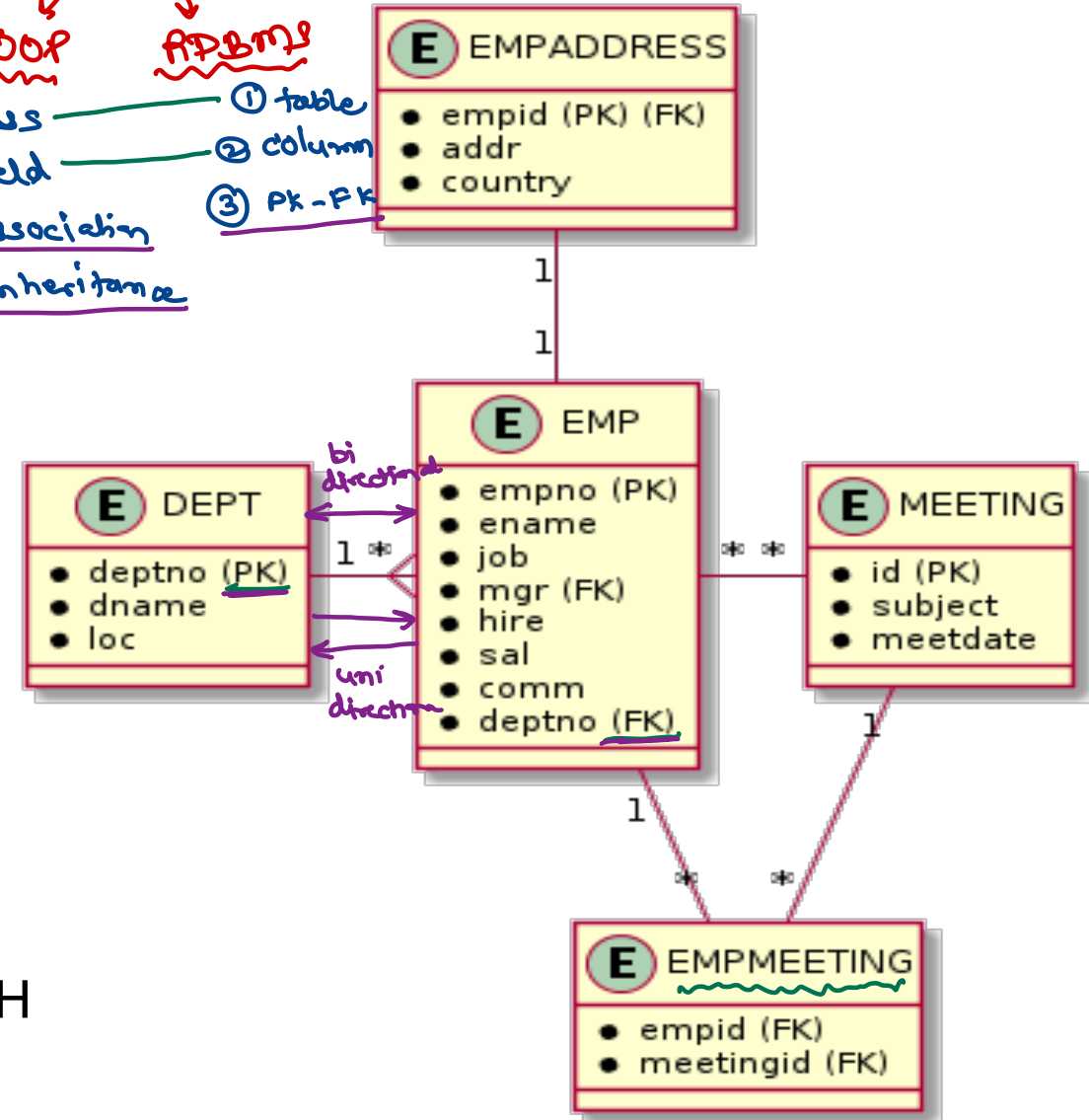
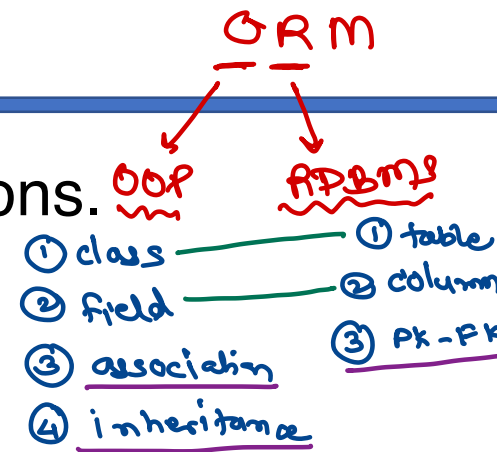




Homework

Hibernate Relations (associations)

- Hibernate represents RDBMS table relations.
 - OneToOne ✓
 - OneToMany ✓
 - ManyToOne ✓
 - ManyToMany ✓
- OneToMany & ManyToOne represent parent-child relation between tables.
- Primary key of parent table is mapped to foreign key of child table.
- FetchType
 - Lazy or Eager
- CascadeType
 - PERSIST, MERGE, DETACH, REMOVE, REFRESH



@OneToMany (uni-directional)

- A Dept has Many Emp.
 - mappedBy – foreign key field in Emp table (that reference to primary key of Dept table).
 - FetchType → SELECT
 - LAZY: Fetch Dept only (simple SELECT query) and fetch Emp only when empList is accessed (simple SELECT query with WHERE clause on deptno)
 - EAGER: Fetch Dept & Emp data in single query (OUTER JOIN query)
 - CascadeType →
 - PERSIST: insert Emp in list while inserting Dept (persist())
 - REMOVE: delete Emp in dept while deleting Dept (remove())
 - DETACH: remove Emp in dept from session while removing Dept from session (detach())
 - REFRESH: re-select Emp in dept while re-selecting Dept (refresh())
 - ✓ MERGE: add Emp in dept ^{into} ~~into~~ session while adding Dept into session (merge())

```
@Entity @Table(name="dept")
class Dept {
    @Id
    @Column private int deptno;
    @Column private String dname;
    @Column private String loc;
    @OneToMany(mappedBy="deptno")
    private List<Emp> empList;
    // ...
}

@Entity @Table(name="emp")
class Emp {
    @Id
    @Column private int empno;
    @Column private String ename;
    @Column private double sal;
    @Column private int deptno;
    // ...
}
```

Handwritten annotations:

- A purple arrow labeled "pk" points to the `@Id` annotation on the `deptno` field in the `Dept` class.
- A purple arrow labeled "FK" points to the `deptno` field in the `Emp` class.
- A purple arrow labeled "FK field" points to the `mappedBy="deptno"` attribute in the `@OneToMany` annotation.





Thank you!

Nilesh Ghule <nilesh@sunbeaminfo.com>

