Inheritance with ORM

- Three strategies:
  - Single table strategy (`SINGLE_TABLE`): combine all instance variables in the inheritance hierarchy into the topmost class in the hierarchy. "Flatten upwards".
  - Joined table strategy (`JOINED`): Each table representing a class in the inheritance hierarchy only contains its own instance variables. Tables are joined by ids.
  - A separate table for each class (`TABLE_PER_CLASS`): Each class is represented by a separate table, all instance variables in the classes upper in the inheritance hierarchy appears in all child classes. "Flatten downwards".

Single table strategy

```
@Entity
@Inheritance(strategy = InheritanceType.SINGLE_TABLE)
public class Team {
    private Long id;
    private String name;
    private League league;
    private Set<Player> players = new HashSet<Player>();

...
    @Id
    @GeneratedValue
    public Long getId() {
...
```

All classes inheriting from Team are "folded" into Team table

```
@Entity
public class SoccerTeam extends Team {
    private String formation = new String();

    public SoccerTeam() {}
...
```

DTYPE identifies the Java class for the database row (here for example 'SoccerTeam')

```
+-----------+--------------+------+-----+---------+----------------+
| Field     | Type         | Null | Key | Default | Extra          |
+-----------+--------------+------+-----+---------+----------------+
| DTYPE     | varchar(31)  | NO   |     | NULL    |                |
| id        | bigint(20)   | NO   | PRI | NULL    | auto_increment |
| name      | varchar(255) | YES  |     | NULL    |                |
| coach     | varchar(255) | YES  |     | NULL    |                |
| formation | varchar(255) | YES  |     | NULL    |                |
| league_id | bigint(20)   | YES  | MUL | NULL    |                |
+-----------+--------------+------+-----+---------+----------------+
```

Joined tables strategy

```
@Entity
@Inheritance(strategy = InheritanceType.JOINED)
public class Team {
     private Long id;
     private String name;
     private League league;
     private Set<Player> players = new HashSet<Player>();

...
```

Inheritance hierarchy is created by joining father and child classes by id

```
@Entity
public class SoccerTeam extends Team {
     private String formation = new String();

     public SoccerTeam() {}
...
```

Note: no change in child class

```
+----------------+
| Tables_in_hib2 |
+----------------+
| BasketballTeam |
| League         |
| League_Team    |
| Player         |
| SoccerTeam     |
| Team           |
| Team_Player    |
+----------------+
```

```
+-----------+--------------+------+-----+---------+----------------+
| Field     | Type         | Null | Key | Default | Extra          |
+-----------+--------------+------+-----+---------+----------------+
| id        | bigint(20)   | NO   | PRI | NULL    | auto_increment |
| name      | varchar(255) | YES  |     | NULL    |                |
| league_id | bigint(20)   | YES  | MUL | NULL    |                |
+-----------+--------------+------+-----+---------+----------------+
```

Team table - only team instance variables

```
+-----------+--------------+------+-----+---------+-------+
| Field     | Type         | Null | Key | Default | Extra |
+-----------+--------------+------+-----+---------+-------+
| formation | varchar(255) | YES  |     | NULL    |       |
| id        | bigint(20)   | NO   | PRI | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
```

SoccerTeam table - id for joining

Table per class strategy

```
@Entity
@Inheritance(strategy = InheritanceType.TABLE_PER_CLASS)
public class Team {
     private Long id;
     private String name;
     private League league;
     private Set<Player> players = new HashSet<Player>();

...
     @Id
     @GeneratedValue(strategy = GenerationType.TABLE)
     public Long getId() {
...
```

Inheritance hierarchy is created by creating a separate table for each class in inheritance hierarchy

id is shared by tables in the inheritance hierarchy

```
@Entity
public class SoccerTeam extends Team {
       private String formation = new String();

       public SoccerTeam() {}
...
```

Child classes unchanged

```
mysql> select * from soccerteam;
+----+-----------+-----------+-----------+
| id | name      | league_id | formation |
+----+-----------+-----------+-----------+
|  4 | Barcelona |         1 | 4-3-2-1   |
+----+-----------+-----------+-----------+

mysql> select * from basketballteam;
+----+------------+-----------+---------------+
| id | name       | league_id | coach         |
+----+------------+-----------+---------------+
|  1 | L.A Lakers |         1 | Mike Dunleavy |
|  2 | Celtics    |         1 | Chris Brown   |
|  3 | Bulls      |         1 | Phil Jackson  |
+----+------------+-----------+---------------+
```

Notice shared id