Name: Md Shajedul Islam StudentID: 614144

Total inc review: 85.5

Midterm 2022-06

CS544 Enterprise Architecture

Theory Section

A. [3 pts] Describe what surrogate keys are (in the context of relational databases):

3 Surrogate keys are the specially declared id attribute that does not belong to the entity's business domain, rather assigned/generated by the database engine. Natural keys lead to brittle schema while surrogates seamlessly maintains associations

B. [3 pts] Describe the difference between the Transient and Detached entity states

3 Transient means not persistent. When we create a new object of an Entity, it is moved to Transient state, but not persisted or managed yet. Detached on the other hand has existence in DB but not managed by EM.

C. [3 pts] Explain how Bi-Directional associations are mapped (what do you need to stop them from being 2 uni-directional associations)

3 Bi-directional associations are mapped by declaring vice-versa One-to-many or Many-To-One annotations on either side. We need to define the owner by declaring mappedBy="owner" on the non-owning class to avoid 2-uni directional ass

D. [3 pts] What does the @MappedSuperClass annotation do?

3 A mappedSuperClass annotation is declared on the Super class if we intend to have it only for code-reuse purpose without having to persist it in the database. Only the child classed will have related tables.

E. [3 pts] What annotations do you need to map a table that has a composite key?

3 We create a new class with the composite attributes and mark it as @Embeddable Then we put a reference to it in the owner class and mark it as @EmbeddedId.

F. [3 pts] Explain what the N+1 problem is in Hibernate

3 N+1 problem occurs when Hibernate executes many small Select queries. To avoid this data can be loaded in one big Select statement using; BatchSize or sub select statements.

G. [3 pts] Describe what Optimistic Concurrency is

0 When two threads tries to access a resource then the thread that comes first aquires a lock on the resource and the other thread waits till the first thread releases the lock. this is called Optimistic concurrency.

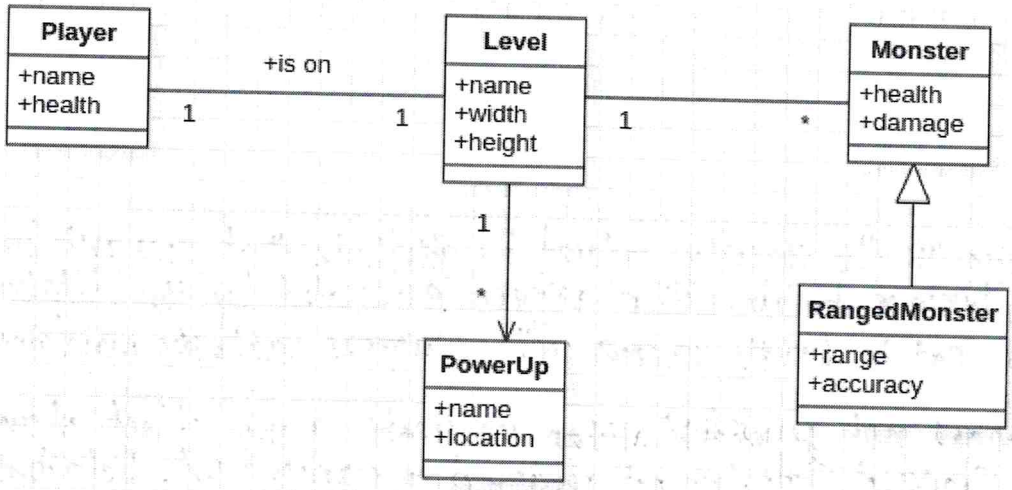
optimistic is no lock but with version

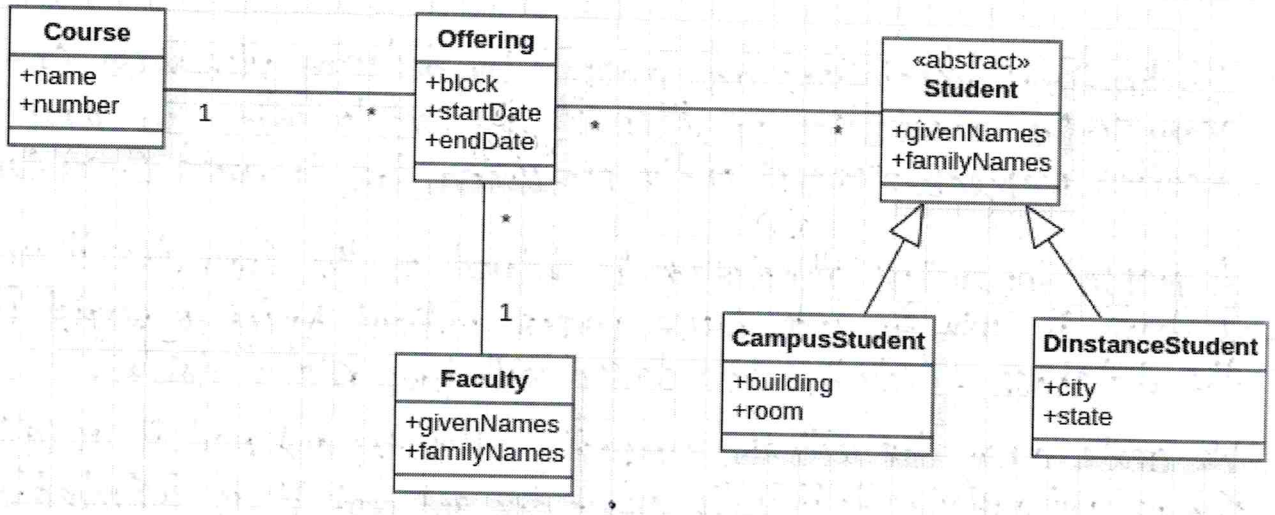
H. [3 pts] Describe what Auto Commit Mode is and how it relates to Hibernate

0 Hibernate does not instantly makes a DB hit or executes a statement when we persist or update or remove an Entity that has an Id with @GeneratedValue But in case of when we explicitly assign an id it does not wait for transactio to commit, instead auto commit at once.

Name: Md Shajedul Islam StudentID: 614144

These are the UML diagrams of the domains for the 2 mapping questions:





You can use these UML diagrams to get an overview of what the code looks like, which is useful when writing queries. Hint: use dates directly in you query string, like: '2022-06-03'

Name: Md Shajedul Islam StudentID: 614144

Exercises:

1. [24 pts] Based on the following classes with annotations write what the tables names, column names, and data types will be (also include if a column is auto_increment).

```java
@Entity
public class Player {
    @Id
    @GeneratedValue
    private Long id;
    private String name;
    private int health;
    @OneToOne(mappedBy = "player")
    private Level level;
}

@Entity
public class Level {
    @Id
    private Long id;
    private String name;
    private int width;
    private int height;
    @OneToOne
    @MapsId
    @JoinColumn(name="id")
    private Player player;
    @OneToMany
    private List<PowerUp> items =
        new ArrayList<>();
    @OneToMany(mappedBy = "level")
    private List<Monster> monsters =
        new ArrayList<>();
}
```

Bi-directional FK in Level

Unidirectional FK in PowerUP

Bi-directional FK in monster

```java
@Entity(name="Item")
public class PowerUp {
    @Id
    @GeneratedValue
    private Long id;
    private String name;
    private String location;
}

@Entity
@Inheritance(strategy = InheritanceType.JOINED)
public class Monster {
    @Id
    @GeneratedValue
    private Long id;
    private int health;
    private int damage;
    @ManyToOne
    private Level level;
}

@Entity
public class RangedMonster extends Monster {
    @Column(name="shootDistance")
    private int range;
    private double accuracy;
}
```

TABLE: Player

Field	Type	Key	Extra
id	bigint(20)	PRI	auto_increment
name	varchar(255)		
health	varchar(255)		

TABLE: Level

Field	Type	Key	Extra
id	bigint(20)	PRI	auto_incr
name	varchar(255)		
width	int(11)		
height	int(11)		
Player_id	bigint(20)	MUL	

TABLE: Item

Field	Type	Key	Extra
id	bigint(20)	PRI	auto_increm
name	varchar(255)		
location	varchar(255)		
Level-id	bigint(20)	MUL	

TABLE: Monster

Field	Type	Key	Extra
Id	bigint(20)	PRI	auto_increment
health	int(11)		
damage	int(11)		
Level-id	bigint(20)	MUL	

TABLE: Range_Monster

Field	Type	Key	Extra
Monster_id	bigint(20)	MUL	
shootDistance	int(11)		
accuracy	bigint		

18

Name: Md Shajedul Islam StudentID: 614144

2. [24 pts] Add annotations to the following classes to map to the tables shown on the next page.

```
@Embeddable
public class Course {

    private String name;

    @Column(nullable=false)
    private int number;
}
```

```
@Entity
public class Offering {
    @Id
    @GeneratedValue
    private Long id;

    @Temporal(TemporalType.DATE)
    private Date startDate;

    @Temporal(TemporalType.DATE)
    private Date endDate;

    @Embedded
    private Course course;

    @ManyToOne
    private Faculty faculty;

    @ManyToMany(mappedBy="offerings")
    private List<Student> students =
        new ArrayList<>();
}
```

```
@Entity
//default discriminator value
public class CampusStudent
        extends Student {

    private String building;

    private String room;
}
```

```
@Entity
public class Faculty {
    @Id
    @GeneratedValue
    private Long id;

    private String givenNames;

    private String familyNames;

    @OneToMany(mappedBy="faculty")
    private List<Offering> offerings =
        new ArrayList<>();
}
```

```
@Entity
@InheritanceStrategy=InheritanceType.SINGLE_TABLE)
public abstract class Student {//Single Table
                               //default
    @Id
    @GeneratedValue
    private Long id;

    private String givenNames;

    private String familyNames;

    @ManyToMany
    private List<Offering> courses =
        new ArrayList<>();
}
```

```
@Entity
// default discriminator value
public class DinstanceStudent
        extends Student {

    private String city;

    private String state;
}
```

22

Name: Md Shajedul Islam StudentID: 6141144

```
describe Offering;
+------------+--------------+------+-----+---------+----------------+
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| id         | bigint(20)   | NO   | PRI | NULL    | auto_increment |
| name       | varchar(255) | YES  |     | NULL    |                |
| number     | int(11)      | NO   |     | NULL    |                |
| endDate    | date         | YES  |     | NULL    |                |
| startDate  | date         | YES  |     | NULL    |                |
| faculty_id | bigint(20)   | YES  | MUL | NULL    |                |
+------------+--------------+------+-----+---------+----------------+

describe Faculty;
+-------------+--------------+------+-----+---------+----------------+
| Field       | Type         | Null | Key | Default | Extra          |
+-------------+--------------+------+-----+---------+----------------+
| id          | bigint(20)   | NO   | PRI | NULL    | auto_increment |
| familyNames | varchar(255) | YES  |     | NULL    |                |
| givenNames  | varchar(255) | YES  |     | NULL    |                |
+-------------+--------------+------+-----+---------+----------------+

describe Student;
+-------------+--------------+------+-----+---------+----------------+
| Field       | Type         | Null | Key | Default | Extra          |
+-------------+--------------+------+-----+---------+----------------+
| DTYPE       | varchar(31)  | NO   |     | NULL    |                |
| id          | bigint(20)   | NO   | PRI | NULL    | auto_increment |
| familyNames | varchar(255) | YES  |     | NULL    |                |
| givenNames  | varchar(255) | YES  |     | NULL    |                |
| building    | varchar(255) | YES  |     | NULL    |                |
| room        | varchar(255) | YES  |     | NULL    |                |
| city        | varchar(255) | YES  |     | NULL    |                |
| state       | varchar(255) | YES  |     | NULL    |                |
+-------------+--------------+------+-----+---------+----------------+

describe Student_Offering;
+---------------+------------+------+-----+---------+-------+
| Field         | Type       | Null | Key | Default | Extra |
+---------------+------------+------+-----+---------+-------+
| students_id   | bigint(20) | NO   | PRI | NULL    |       |
| courses_id    | bigint(20) | NO   | MUL | NULL    |       |
| courses_ORDER | int(11)    | NO   | PRI | NULL    |       |
+---------------+------------+------+-----+---------+-------+
```

Name: Md Shajedul Islam StudentID: 614144

3. [12 pts] Based on the game domain write queries to retrieve:
   a. All players whose health is greater than 50 and are on the level named "Beach"

3.5

```
SELECT DISTINCT P FROM Player P JOIN p.level l JOIN l.monsters m
WHERE l.name="Beach" AND m.health > 50
```

   b. All PowerUp items on the level named "Mountains"

4

```
SELECT l.items FROM Level l WHERE l.name="Mountains"
```

   c. All levels that have a RangedMonster with health greater than 100

4

```
SELECT DISTINCT l FROM Level l JOIN l.monsters m
WHERE m.health > 100 AND type(m) = RangedMonster
```

4. [12 pts] Based on the university domain write queries to retrieve:
   a. All Students with the familyNames "Smith"

4

```
SELECT s FROM Student s WHERE s.familyNames = "Smith"
```

   b. All Faculty teaching the course "Enterprise Applications"

4

```
SELECT DISTINCT f FROM Faculty f
  JOIN f.offerings o WHERE o.course.name = "Enterprise
                                           Applications"
```

   c. All Offerings with a startDate after 2022-01-01 that has CampusStudents with the givenNames "John"

4

```
SELECT DISTINCT o FROM Offering o
    JOIN o.students s
      WHERE o.startDate > '2022-01-01'
        AND s.givenNames = "John"
          AND type(s) = CampusStudent
```