Object Relational Mapping

Recitation 6
February 20th, 2020
Outline

• Introduction to Object Relational Mapping
• ORM case study (1): JPA
• ORM case study (2): Django ORM
Outline

• Introduction to Object Relational Mapping
• ORM case study (1): JPA
• ORM case study (2): Django ORM
Building a web application...

Create Student

ID: 
Name: 
GPA: 

```java
class Student {
    int sid;
    String name;
    double gpa;
}
Student s = new Student (…)
```

<table>
<thead>
<tr>
<th>Sid</th>
<th>Name</th>
<th>gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>Adam</td>
<td>3.6</td>
</tr>
<tr>
<td>23</td>
<td>Joy</td>
<td>3.2</td>
</tr>
<tr>
<td>24</td>
<td>AG</td>
<td>4.0</td>
</tr>
</tbody>
</table>
Building a web application...

WebSocket, JSON, Restful WS, JSP + HTML
Building a web application...

- WebSockets
- JSON
- Restful WS
- JSP + HTML

Servlets
Building a web application...

- WebSockets
- JSON
- Restful WS
- JSP + HTML

Servlets

Database
Building a web application…

Presentation

Logic / Business

Data Layer

What happens here?
Building a web application...

What happens here?

```java
class Student {
    int sid;
    String name;
    double gpa;
}
Student s = new Student (...)
```

<table>
<thead>
<tr>
<th>Sid</th>
<th>Name</th>
<th>gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>Adam</td>
<td>3.6</td>
</tr>
<tr>
<td>23</td>
<td>Joy</td>
<td>3.2</td>
</tr>
</tbody>
</table>
Building a web application...

What happens here?

**JDBC Example:**

```java
class Student {
    int sid;
    String name;
    double gpa;
}

Student s = new Student (...)
```

<table>
<thead>
<tr>
<th>Sid</th>
<th>Name</th>
<th>gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>Adam</td>
<td>3.6</td>
</tr>
<tr>
<td>23</td>
<td>Joy</td>
<td>3.2</td>
</tr>
</tbody>
</table>

JDBC API
Building a web application...

Static SQL code

Student is an object, we need to break it down to an SQL code...

What if I decide to change my database from SQL to Oracle?

JDBC Example:

```java
Class.forName("org.postgresql.Driver");
connection = DriverManager.getConnection(jdbcURL, dbUser, dbPassword);
statement = connection.prepareStatement("INSERT INTO STUDENT (sid, name, gpa) VALUES (" + 22 + ", " + "Adam", 3.6 + ");");
ResultSet rs = statement.executeQuery();
```
Building a web application...
Building a web application...

- WebSockets
- JSON
- Restful WS
- JSP + HTML

- Servlets
- ORM
- JPA
- SQL

- JDBC
- Database
“Think like an object”

Relation / Table ⇔ Class
Record / Row / Tuple ⇔ Object
Attribute / Column ⇔ Member / Field
Hierarchy (Is-A) ⇔ Inheritance
Relationship ⇔ Composition / Aggregation

Sailors (sid: integer, sname: string, rating: integer, age: real)

class Sailors {
    int sid;
    String sname;
    int rating;
    double age;
}

<table>
<thead>
<tr>
<th>Sid</th>
<th>Sname</th>
<th>Rating</th>
<th>Age</th>
</tr>
</thead>
</table>

"Think like an object"

Relation / Table ⇔ Class
Record / Row / Tuple ⇔ Object
Attribute / Column ⇔ Member / Field
Hierarchy (Is-A) ⇔ Inheritance
Relationship ⇔ Composition / Aggregation

Sailors \( (\text{sid}: \text{integer}, \) 
\text{sname}: \text{string}, \text{rating}: \text{integer}, \text{age}: \text{real}) \)

<table>
<thead>
<tr>
<th>Sid</th>
<th>Sname</th>
<th>Rating</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Zein..</td>
<td>9</td>
<td>26</td>
</tr>
</tbody>
</table>

class Sailors {
    int sid;
    String sname;
    int rating;
    double age;
}

Sailors S1 = new Sailor()
S1.sid = 123;
S1.sname = Zeinab;
S1.rating = 9;
S1.age = 26
“Think like an object”

Relation / Table ⇔ Class
Record / Row / Tuple ⇔ Object
Attribute / Column ⇔ Member / Field
Hierarchy (Is-A) ⇔ Inheritance
Relationship ⇔ Composition / Aggregation

Sailors (sid: integer, 
sname: string, rating: 
integer, age: real)

class Sailors {
    int sid;
    String sname;
    int rating;
    double age;
}
Sailors S2 = new Sailor()
S2.sid = 124;
S2.sname = Omar;
S2.rating = 10;
S2.age = 32
“Think like an object”

Relation / Table ⇔ Class
Record / Row / Tuple ⇔ Object
Attribute / Column ⇔ Member / Field
Hierarchy (Is-A) ⇔ Inheritance
Relationship ⇔ Composition / Aggregation

Sailors (sid: integer, sname: string, rating: integer, age: real)

<table>
<thead>
<tr>
<th>Sid</th>
<th>Sname</th>
<th>Rating</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Zein..</td>
<td>9</td>
<td>26</td>
</tr>
<tr>
<td>124</td>
<td>Oma..</td>
<td>10</td>
<td>32</td>
</tr>
</tbody>
</table>

class Sailors {
    int sid;
    String sname;
    int rating;
    double age;
}

Sailors S2 = new Sailor();
S2.sid = 124;
S2.sname = Omar;
S2.rating = 10;
S2.age = 32
Outline

• Introduction to Object Relational Mapping
• ORM case study (1): JPA (Entity & Inheritance)
• ORM case study (2): Django ORM
JPA Walkthrough
class Customer
{
    String username;
    String password;
    String full_name;
    String cc_number;
    boolean logged_in;
    Set<Order> orders;
}

@Entity

- uid
- pass
- name
- ccn

Customer
@Entity

class Customer {
    @Id
    String username;
    String password;
    String full_name;
    String cc_number;
    @Transient
    boolean logged_in;
    ...
    Set<Order> orders;
}
@Entity
@Table (name = "Customers", schema = "V2")
class Customer {
    @Id
    @Column (name = "cid")
    String username;

    @Column (nullable = false)
    String password;

    @Column (name = "name", nullable = false)
    String full_name;

    @Column (name = "ccn")
    String cc_number;

    @Transient
    boolean logged_in;

    @...
    Set<Order> orders;
}
Inheritance
Inheritance

```java
@Entity
... 
public class User {
    @Id
    @Column (name = "uid")
    String username;

    @Column (nullable = false)
    String password;

    @Column (nullable = false)
    String full_name;

    @Transient
    boolean logged_in;
}
```

Diagram:
- Customer
  - User
    - Salesman
    - ccn
    - name
    - salary
    - uid
    - pass
Inheritance

```java
@Entity
class Customer extends User {
    String cc_number;
}

@Entity
class Salesman extends User {
    int salary;
}
```
Inheritance

@Entity
@MappedSuperClass
class User {
    @Id
    @Column (name = "uid")
    String username;

    @Column (nullable = false)
    String password;

    @Column (nullable = false)
    String full_name;

    @Transient
    boolean logged_in;
}

Fields get embedded with subclass entities

<table>
<thead>
<tr>
<th>Customer</th>
<th>Salesman</th>
</tr>
</thead>
<tbody>
<tr>
<td>uid</td>
<td>uuid</td>
</tr>
<tr>
<td>pass</td>
<td>pass</td>
</tr>
<tr>
<td>name</td>
<td>name</td>
</tr>
<tr>
<td>ccn</td>
<td>salary</td>
</tr>
<tr>
<td>myahmad</td>
<td></td>
</tr>
<tr>
<td>1234</td>
<td>p@$%</td>
</tr>
<tr>
<td>Yousuf</td>
<td>Tamim</td>
</tr>
<tr>
<td>123456</td>
<td></td>
</tr>
</tbody>
</table>

Carnegie Mellon University Qatar
@Entity
@Inheritance(strategy = InheritanceType.TABLE_PER_CLASS)
@Inheritance(strategy = InheritanceType.SINGLE)
@Inheritance(strategy = InheritanceType.JOIN)
abstract class User
{
    @Id
    @Column (name = "uid")
    String username;

    @Column (nullable = false)
    String password;

    @Column (nullable = false)
    String full_name;

    @Transient
    boolean logged_in;
}
Outline

• Introduction to Object Relational Mapping
• ORM case study (1): JPA
• ORM case study (2): Django ORM:

  you will be using Django for your projects ...
Model View Controller (MVC)
Model View Controller (MVC)
Model View Controller (MVC)
Model View Template (MVT)
Model View Template (MVT)
Next Recitation...

- Mapping relationships
- Practice with Django