luv2code

Mapping - Enums

Enum

A special Java data type for a

set of predefined constants

Examples of Java Enums

Core Java

Examples of Java Enums

Core Java

- Define logging levels

```
public enum Level {
   SEVERE, WARNING, INFO
}
```

Examples of Java Enums

Core Java

- Define logging levels

```
public enum Level {
   SEVERE, WARNING, INFO
}
```

- Define payment types

```
public enum Payment {
   DEBIT, CREDIT, PAYPAL, BITCOIN
}
```

Examples of Java Enums

Core Java

- Define logging levels

```
public enum Level {
   SEVERE, WARNING, INFO
}
```

- Define payment types

```
public enum Payment {
   DEBIT, CREDIT, PAYPAL, BITCOIN
}
```

Java Enums Tutorial

www.luv2code.com/java-enums-tutorial

Use Case for Enums

Use Case for Enums

- A student will have a **status**

Use Case for Enums

- A student will have a **status**
  - Status can have constant value of **`ACTIVE`** or **`INACTIVE`**

Use Case for Enums

- A student will have a **status**
  - Status can have constant value of **`ACTIVE`** or **`INACTIVE`**

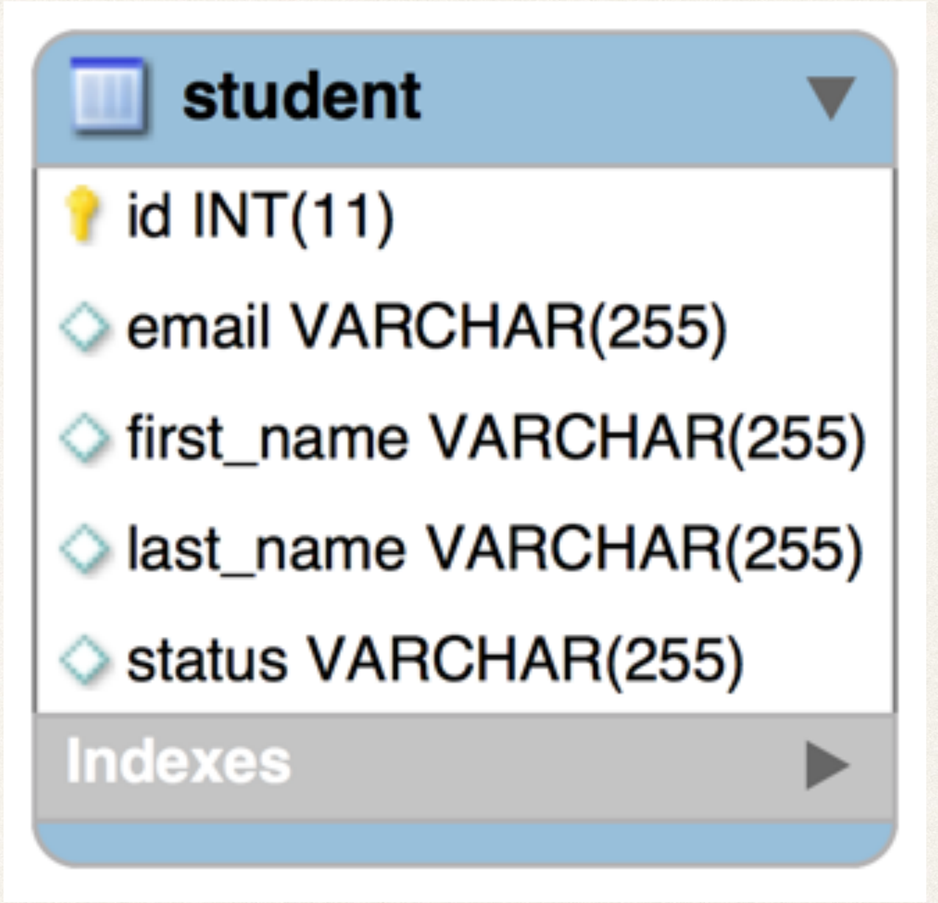
- The status field will be in the student database table

Use Case for Enums

- A student will have a **status**
  - Status can have constant value of **`ACTIVE`** or **`INACTIVE`**
- The status field will be in the student database table
- In Java code, model the status as an Enum

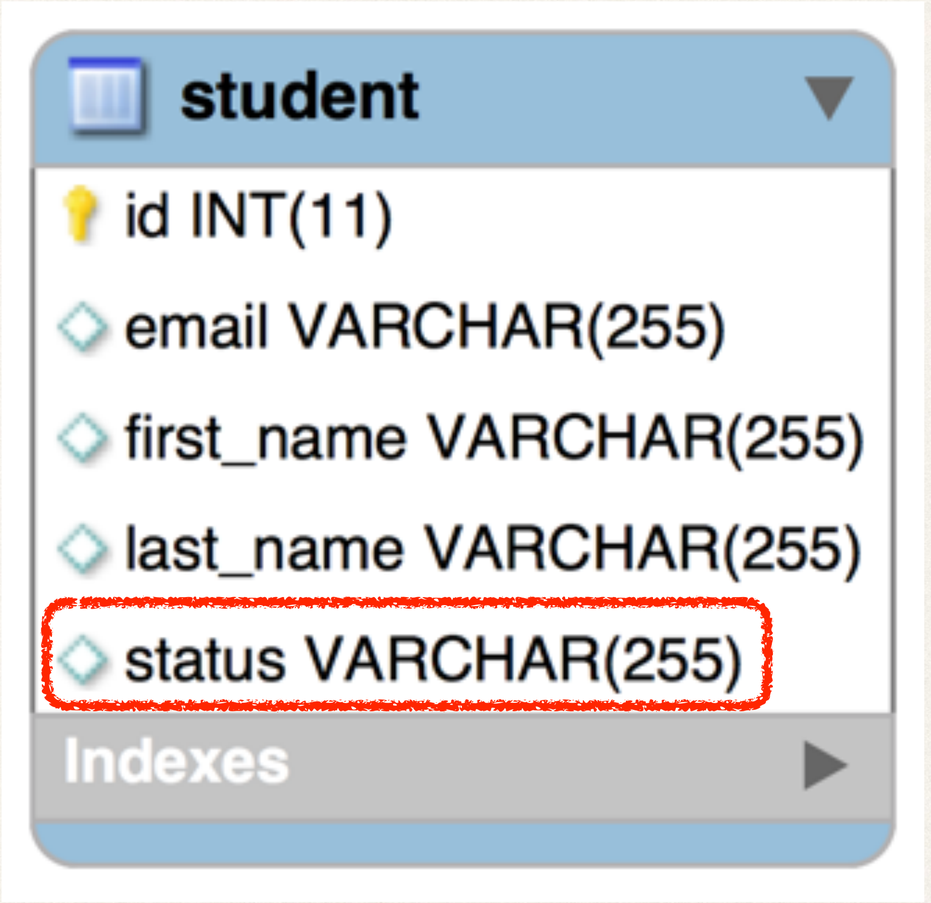
```
public enum Status {
  ACTIVE, INACTIVE
}
```

Database Diagram



Database Diagram

Status

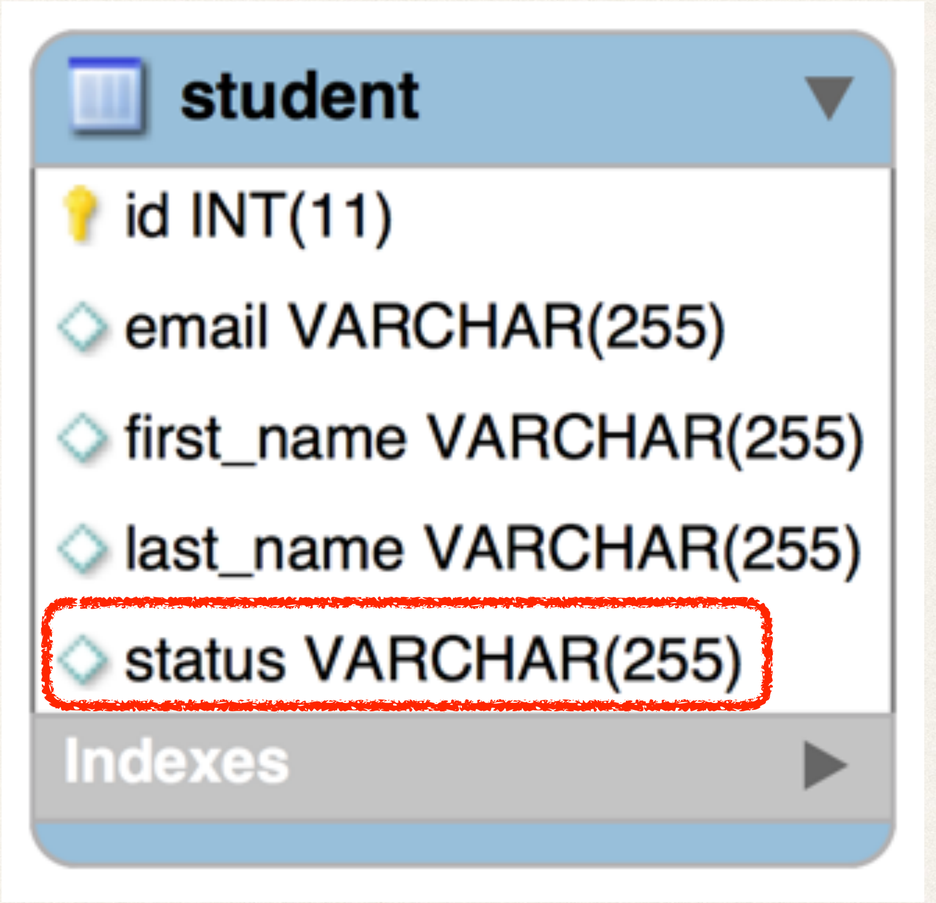


Database Diagram

Status

Reuse Status in other parts of our app

For: Student, Customer, Group etc…



Development Process

Step-By-Step

Development Process

Step-By-Step

1. Define the `Status` enum type

Development Process

Step-By-Step

1. Define the **`Status`** enum type

2. Reference the **`Status`** enum type in **`Student`** class

Development Process

Step-By-Step

1. Define the **`Status`** enum type

2. Reference the **`Status`** enum type in **`Student`** class

3. Develop the main application

Annotation for Enums

Annotation for Enums

Annotation	Description

Annotation for Enums

Annotation	Description
`@Enumerated`	**Used to reference an Enum type** **For example: `Student` can use this to refer to `Status`**

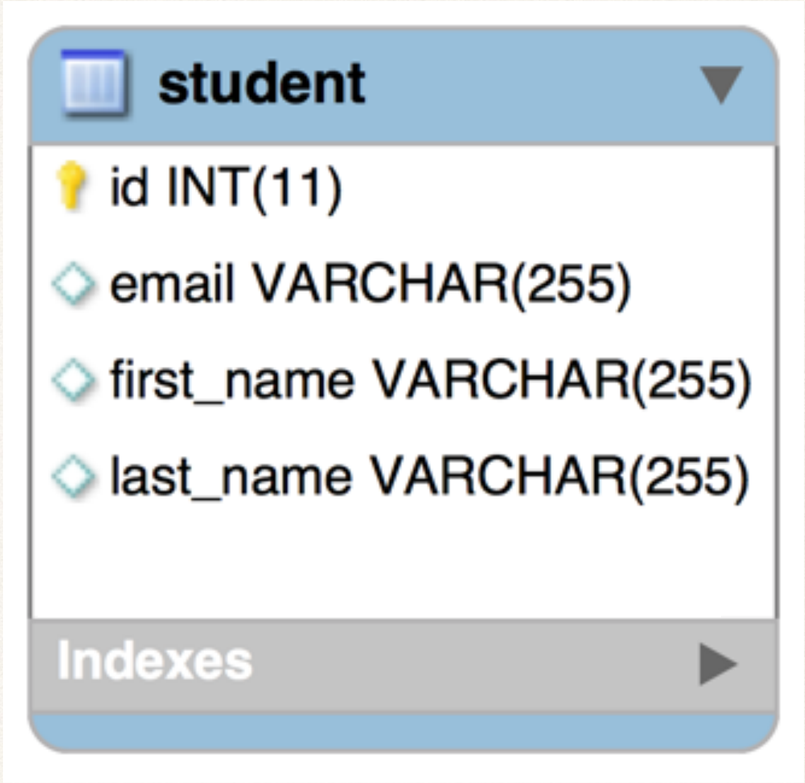
Step 1: Define the Status Enum Type

Step 1: Define the Status Enum Type

```
public enum Status {
  ACTIVE, INACTIVE
}
```

Step 2: Reference the Status Enum Type

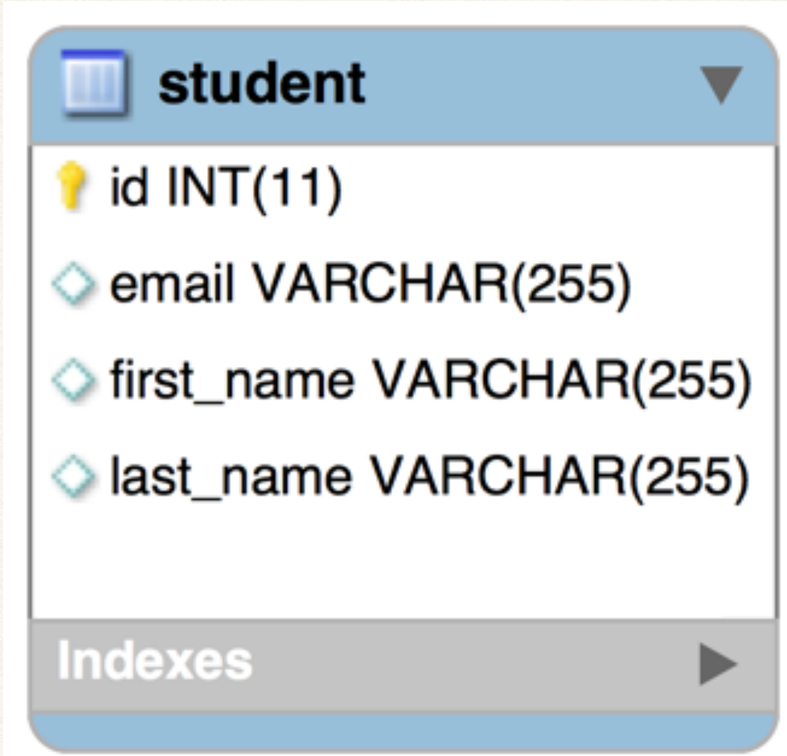
```
@Entity
@Table(name="student")
public class Student {

   @Id
   @GeneratedValue(strategy=GenerationType.IDENTITY)
   private int id;

   @Column(name="first_name")
   private String firstName;
   …




}
```



Step 2: Reference the Status Enum Type

```
@Entity
@Table(name="student")
public class Student {

   @Id
   @GeneratedValue(strategy=Ge
   private int id;

   @Column(name="first_name")
   private String firstNa
   …

   @Enumerated(EnumType.STRING)
   @Column(name="status")
   private Status status;

   …
}
```

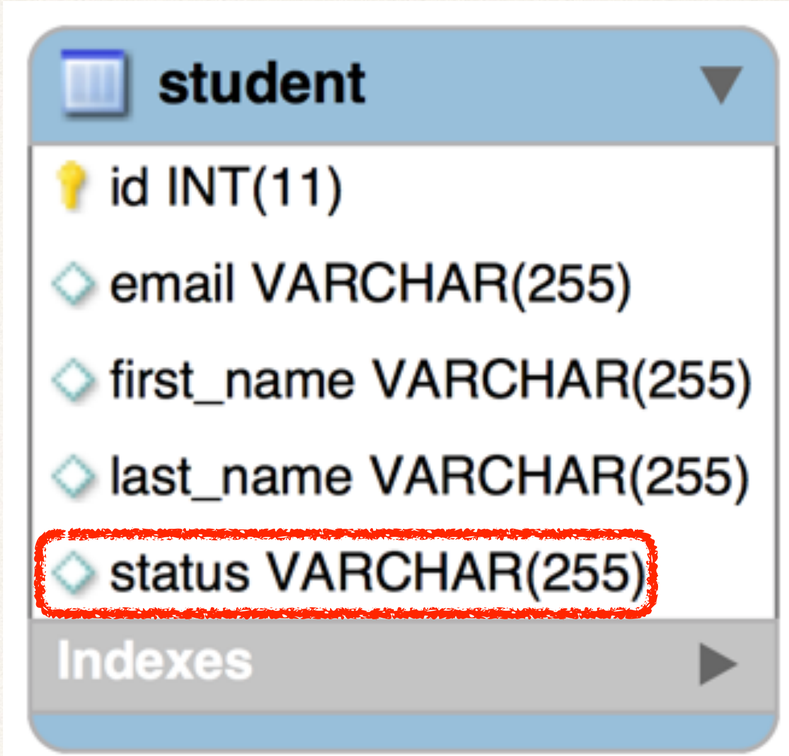
Reference the enum type



Step 2: Reference the Status Enum Type

```
@Entity
@Table(name="student")
public class Student {

   @Id
   @GeneratedValue(strategy=Ge
   private int id;

   @Column(name="first_name")
   private String firstNa
   …

   @Enumerated(EnumType.STRING)
   @Column(name="status")
   private Status status;

   …
}
```

Reference the enum type



Step 3: Develop the main application

Step 3: Develop the main application

```
// create the objects
Student tempStudent1 =  new Student("John", "Doe", "john@luv2code.com", Status.ACTIVE);
Student tempStudent2 =  new Student("Mary", "Public", "mary@luv2code.com", Status.INACTIVE);
```

Step 3: Develop the main application

Pass in the status

```
// create the objects
Student tempStudent1 =  new Student("John", "Doe", "john@luv2code.com", Status.ACTIVE);
Student tempStudent2 =  new Student("Mary", "Public", "mary@luv2code.com", Status.INACTIVE);
```

Step 3: Develop the main application

Pass in the status

```
// create the objects
Student tempStudent1 =  new Student("John", "Doe", "john@luv2code.com", Status.ACTIVE);
Student tempStudent2 =  new Student("Mary", "Public", "mary@luv2code.com", Status.INACTIVE);

// start a transaction
session.beginTransaction();
```

Step 3: Develop the main application

Pass in the status

```
// create the objects
Student tempStudent1 =  new Student("John", "Doe", "john@luv2code.com", Status.ACTIVE);
Student tempStudent2 =  new Student("Mary", "Public", "mary@luv2code.com", Status.INACTIVE);

// start a transaction
session.beginTransaction();

// save the objects
System.out.println("Saving the student...");
session.save(tempStudent1);
session.save(tempStudent2);
```

Step 3: Develop the main application

Pass in the status

```
// create the objects
Student tempStudent1 =  new Student("John", "Doe", "john@luv2code.com", Status.ACTIVE);
Student tempStudent2 =  new Student("Mary", "Public", "mary@luv2code.com", Status.INACTIVE);

// start a transaction
session.beginTransaction();

// save the objects
System.out.println("Saving the student...");
session.save(tempStudent1);
session.save(tempStudent2);

// commit the transaction
session.getTransaction().commit();
```

Run the App

Run the App

Console

```
Hibernate: insert into student (email, first_name, last_name, status) values (?, ?, ?, ?)
Hibernate: insert into student (email, first_name, last_name, status) values (?, ?, ?, ?)
```

Run the App

Console

```
Hibernate: insert into student (email, first_name, last_name, status) values (?, ?, ?, ?)
Hibernate: insert into student (email, first_name, last_name, status) values (?, ?, ?, ?)
```

Table: student

id	email	first_name	last_name	status
1	john@luv2code.com	John	Doe	ACTIVE
2	mary@luv2code.com	Mary	Public	INACTIVE

Run the App

Console

```
Hibernate: insert into student (email, first_name, last_name, status) values (?, ?, ?, ?)
Hibernate: insert into student (email, first_name, last_name, status) values (?, ?, ?, ?)
```

Table: student

id	email	first_name	last_name	status
1	john@luv2code.com	John	Doe	ACTIVE
2	mary@luv2code.com	Mary	Public	INACTIVE

Run the App

Console

```
Hibernate: insert into student (email, first_name, last_name, status) values (?, ?, ?, ?)
Hibernate: insert into student (email, first_name, last_name, status) values (?, ?, ?, ?)
```

Table: student

id	email	first_name	last_name	status
1	john@luv2code.com	John	Doe	ACTIVE
2	mary@luv2code.com	Mary	Public	INACTIVE

```
@Entity
@Table(name="student")
public class Student {

    @Enumerated(EnumType.STRING)
    @Column(name="status")
    private Status status;
```

```
public enum Status {
    ACTIVE, INACTIVE;
}
```