

# JPA Mapping II

**“Code with Passion!”**



# Agenda

- Entity Inheritance relationship
- Single table strategy
- Joined strategy
- Which one to use

# Entity Inheritance

- Entities can have inheritance relationship
  - > They are POJO's
- Three inheritance mapping strategies (mapping entity inheritance to database tables)
  - > Single table
  - > Joined subclass
  - > Table per class (rarely used)
- Use annotation `@Inheritance(..)`

# Single Table Strategy

# Single Table Strategy

- All the classes in a hierarchy are mapped to a single table
- Annotation to the parent Entity
  - > `@Inheritance(strategy=InheritanceType.SINGLE_TABLE)`
- Root table has a discriminator column whose value identifies the specific subclass to which the instance represented by row belongs
  - > `@DiscriminatorColumn(columnDefinition="MYDTYPE")`

# Single Table Strategy Example

**// Parent Entity**

**@Entity**

**@Inheritance(strategy=InheritanceType.SINGLE\_TABLE)**

**@DiscriminatorColumn(columnDefinition="MYDTYPE")**

**public class Person implements Serializable {...}**

**// Child Entity**

**@Entity**

**public class Student extends Person {...}**

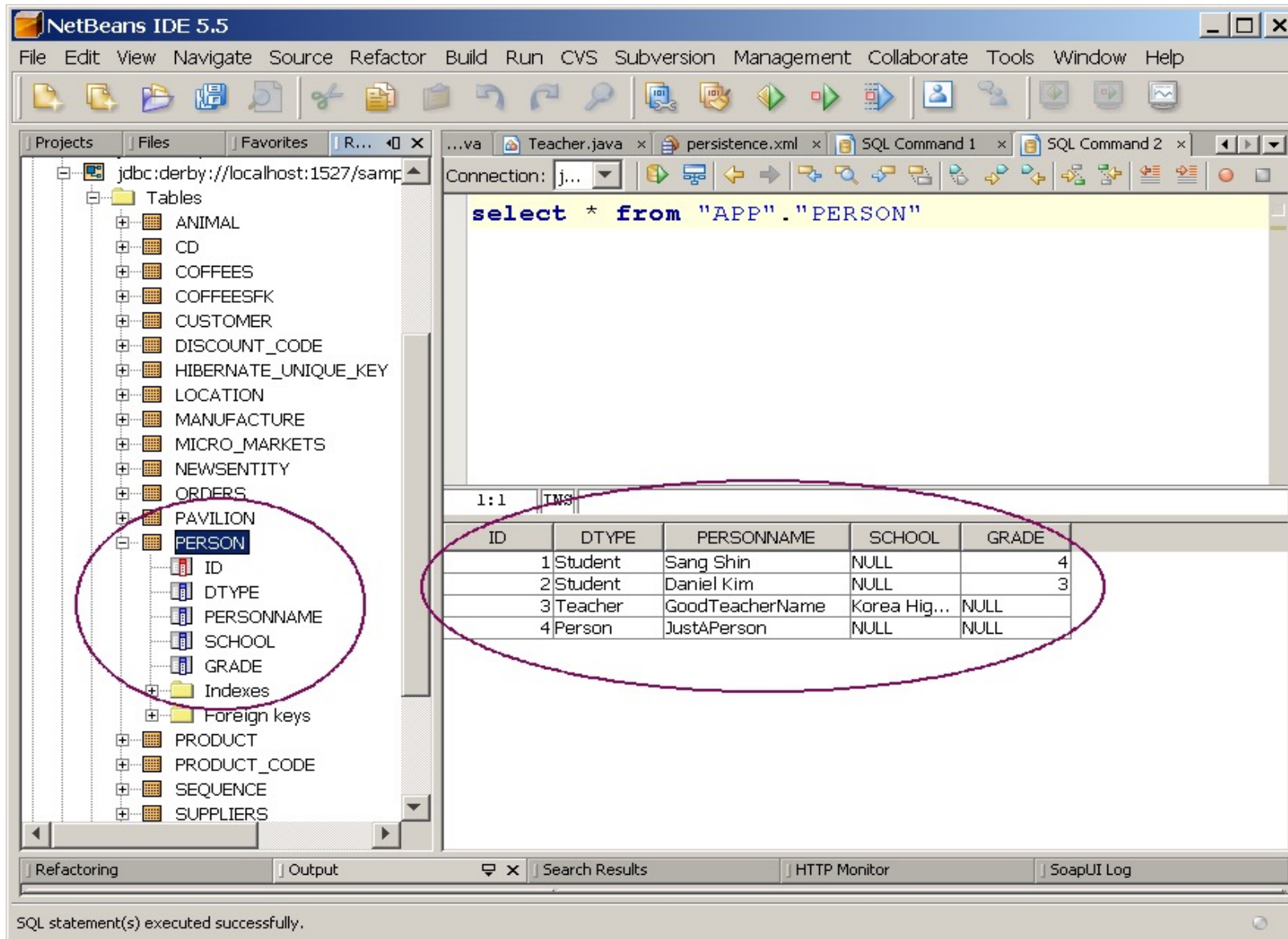
**// Child Entity**

**@Entity**

**public class Teacher extends Person {...}**



# Single Table Strategy Example



NetBeans IDE 5.5

File Edit View Navigate Source Refactor Build Run CVS Subversion Management Collaborate Tools Window Help

Projects Files Favorites R... x

jdbc:derby://localhost:1527/samp

Tables

- ANIMAL
- CD
- COFFEES
- COFFEESFK
- CUSTOMER
- DISCOUNT\_CODE
- HIBERNATE\_UNIQUE\_KEY
- LOCATION
- MANUFACTURE
- MICRO\_MARKETS
- NEWSENTITY
- ORDERS
- PAVILION
- PERSON**
- Indexes
- Foreign keys
- PRODUCT
- PRODUCT\_CODE
- SEQUENCE
- SUPPLIERS

Connection: j...

```
select * from "APP"."PERSON"
```

1:1 TWS

ID	DTYPE	PERSONNAME	SCHOOL	GRADE
1	Student	Sang Shin	NULL	4
2	Student	Daniel Kim	NULL	3
3	Teacher	GoodTeacherName	Korea Hig...	NULL
4	Person	JustAPerson	NULL	NULL

Refactoring Output Search Results HTTP Monitor SoapUI Log

SQL statement(s) executed successfully.

# Lab:

## Exercise 1: Single Table Inheritance 4324\_jpa\_mapping2.zip





# Joined Strategy

# Joined Strategy

- One table for each class in the hierarchy
  - > A parent class is represented by a single common table
  - > Each child class is represented by a separate table that contains fields specific to the child class
  - > Foreign key relationship exists between parent common table and subclass tables
- Annotation to the parent Entity
  - > `@Inheritance(strategy=InheritanceType.JOINED)`

# Joined Table Strategy Example

```
// Parent Entity
@Entity
@Inheritance(strategy=InheritanceType.JOINED)
@DiscriminatorColumn(columnDefinition="MYDTYPE")
public class Person implements Serializable {...}
```

```
// Child Entity
@Entity
public class Student extends Person {...}
```

```
// Child Entity
@Entity
public class Teacher extends Person {...}
```

**Which One to Use?**

# So Which One Should You Use?

- SINGLE\_TABLE or JOIN\_TABLE?

# SINGLE\_TABLE

- Advantages
  - > Offers best performance even for in the deep hierarchy since single select may suffice
- Disadvantages
  - > Changes to members of the hierarchy require column to be altered, added or removed from the table



# JOIN\_TABLE

- Advantages
  - > Does not require complex changes to the schema when a single parent class is modified
  - > Works well with shallow hierarchy
- Disadvantages
  - > Can result in poor performance – as hierarchy grows, the number of joins required to construct a leaf class also grows

**Code with Passion!**

