**Inheritance Mapping/Strategy in Spring JPA**

There are three inheritance mapping strategies defined in the hibernate:

* Table Per Hierarchy. Also known as Single Table
* Table Per Concrete class. Also known as Per Class
* Table Per Subclass.

**Table Per Hierarchy – Single Table**

@Entity @Table(name = "employee")

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

@DiscriminatorColumn(name="type",discriminatorType=DiscriminatorType.***STRING***)

@Data

**public** **class** Employee {

@Id @GeneratedValue

**private** **long** id;

@Column

**private** String name;

}

@Entity

@DiscriminatorValue("contract")

**public** **class** ContractEmployee **extends** Employee {

}

@Entity

@DiscriminatorValue("permanent")

**public** **class** PermanetEmployee **extends** Employee {

}

**Database Table Design**

Table

Description automatically generated

**Table Per Concrete class – Table Per Class**

@Entity @Table(name = "vehicle")

@Inheritance(strategy = InheritanceType.***TABLE\_PER\_CLASS***)

@Data

**public** **class** Vehicle {

@Id @GeneratedValue

**private** **long** id;

@Column

**private** String name;

}

@Entity

@Table(name="fourwheeler")

@Data

**public** **class** FourWheelerVehicle **extends** Vehicle {

}

@Entity

@Table(name="twowheeler")

@Data

**public** **class** TwoWheelerVehicle **extends** Vehicle {

}

**Database Table Design**

**SELECT** \* **FROM** vehicle;

**SELECT** \* **FROM** twowheeler;

**SELECT** \* **FROM** fourwheeler;

Graphical user interface

Description automatically generated Text

Description automatically generated with medium confidence Graphical user interface

Description automatically generated with low confidence

**Table Per Subclass**

@Entity @Table(name = "car")

@Inheritance(strategy = InheritanceType.***JOINED***)

@Entity @Table(name = "sedan")

@PrimaryKeyJoinColumn(name="id")

@Data

**public** **class** SedanCar **extends** Car {

}

@Data

**public** **class** Car {

@Id @GeneratedValue

**private** **long** id;

@Column

**private** String name;

@Entity @Table(name = "hatchback")

@PrimaryKeyJoinColumn(name="id")

@Data

**public** **class** HatchBackCar **extends** Car {

}

}

**Database Table Design**

**SELECT** \* **FROM** car;

**SELECT** \* **FROM** sedan;

**SELECT** \* **FROM** hatchback;

Table

Description automatically generated A picture containing graphical user interface

Description automatically generated Graphical user interface, application

Description automatically generated

**MappedSuperClass**

@Entity @Table(name = "debit") @Data

**public** **class** DebitCard **extends** BaseCard {

@Column

**private** String type;

}

@Entity @Table(name = "credit") @Data

**public** **class** CreditCard **extends** BaseCard {

@Column

**private** String type;

}

@MappedSuperclass @Data

**public** **class** BaseCard {

@Id @GeneratedValue

**private** **long** id;

@Column

**private** String name;

}

**BaseCard is not an Entity.**

**Database Table Design**

**SELECT** \* **FROM** credit;

**SELECT** \* **FROM** debit;

Table

Description automatically generated Table

Description automatically generated

Inheritance Mapping in Spring JPA Hibernate

Relational databases don't have a straightforward way to map class hierarchies onto database tables.

To address this, the JPA specification provides several strategies:

* *MappedSuperclass* – the parent classes, can't be entities
* Single Table – the entities from different classes with a common ancestor are placed in a single table
* Joined Table – each class has its table and querying a subclass entity requires joining the tables
* Table-Per-Class – all the properties of a class, are in its table, so no join is required

Each strategy results in a different database structure.

**Entity inheritance means that we can use polymorphic queries for retrieving all the sub-class entities when querying for a super-class.**

There are three types of inheritance mapping in hibernate  
**1. Table per concrete class with unions**  
**2. Table per class hierarchy(**Single Table Strategy**)**  
**3. Table per subclass**

**Table Per Hierarchy**

@Data

@Entity

@Table(name = "Employee")

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

@DiscriminatorColumn(name = "type", discriminatorType = DiscriminatorType.***STRING***)

@DiscriminatorValue(value = "E")

**public** **class** Employee {

@Id

@GeneratedValue

**private** **int** id;

**private** String name;

}

@Data

@Entity

@DiscriminatorValue(value = "CE")

**public** **class** ContractEmployee **extends** Employee {

@Column(name = "hourlyRate")

**private** **double** hourlyRate;

}

@Data

@Entity

@DiscriminatorValue(value = "PE")

**public** **class** PermanentEmployee **extends** Employee {

@Column(name = "salary")

**private** **double** salary;

}

@Repository

**public** **interface** EmployeeRepository **extends** JpaRepository<Employee, Long> {

}

@Service

**public** **class** EmployeeService {

@Autowired

**private** EmployeeRepository repo;

**public** **void** saveInfo(Employee emp) {

repo.save(emp);

}

}

**public** **class** TestTablePerHierarchy {

**public** **static** **void** storeInfo(ApplicationContext applicationContext) {

EmployeeService service = applicationContext.getBean(EmployeeService.**class**);

Employee employee = **new** Employee();

employee.setName("John");

service.saveInfo(employee);

PermanentEmployee permanentEmployee = **new** PermanentEmployee();

permanentEmployee.setName("Jacob");

permanentEmployee.setSalary(30000);

service.saveInfo(permanentEmployee);

ContractEmployee contractEmployee = **new** ContractEmployee();

contractEmployee.setName("Raj");

contractEmployee.setHourlyRate(2000);

service.saveInfo(contractEmployee);

}

}

**Database Design**

Table

Description automatically generated

# **Table per concrete class**

@Data

@Entity

@Table(name = "Employee2")

@Inheritance(strategy = InheritanceType.***TABLE\_PER\_CLASS***)

**public** **class** Employee {

@Id

@GeneratedValue

**private** **int** id;

**private** String name;

}

@Data

@Entity

@Table(name = "ContractEmployee")

**public** **class** ContractEmployee **extends** Employee {

@Column(name = "hourlyRate")

**private** **double** hourlyRate;

}

@Data

@Entity

@Table(name = "PermanentEmployee")

**public** **class** PermanentEmployee **extends** Employee {

@Column(name = "salary")

**private** **double** salary;

}

@Repository

**public** **interface** EmployeeRepository **extends** JpaRepository<Employee, Long> {

}

@Service

**public** **class** EmployeeService {

@Autowired

**private** EmployeeRepository repo;

**public** **void** saveInfo(Employee emp) {

repo.save(emp);

}

}

**Database Design**

Graphical user interface, table

Description automatically generated Graphical user interface, text, application

Description automatically generated Graphical user interface, text, table

Description automatically generated

**public** **class** TestTablePerConcreteClass {

**public** **static** **void** storeInfo(ApplicationContext applicationContext) {

EmployeeService service = applicationContext.getBean(EmployeeService.**class**);

Employee employee=**new** Employee();

employee.setName("John");

service.saveInfo(employee);

PermanentEmployee permanentEmployee=**new** PermanentEmployee();

permanentEmployee.setName("Jacob");

permanentEmployee.setSalary(30000);

service.saveInfo(permanentEmployee);

ContractEmployee contractEmployee=**new** ContractEmployee();

contractEmployee.setName("Raj");

contractEmployee.setHourlyRate(2000);

service.saveInfo(contractEmployee);

}

}

# **Table per subclass**

@Data

@Entity

@Table(name = "Employee2")

@Inheritance(strategy=InheritanceType.***JOINED***)

**public** **class** Employee {

@Id

@GeneratedValue

**private** **int** id;

**private** String name;

}

@Data

@Entity

@Table(name="ContractEmployee")

@PrimaryKeyJoinColumn(name="Id")

**public** **class** ContractEmployee **extends** Employee {

@Column(name = "hourlyRate")

**private** **double** hourlyRate;

}

@Data

@Entity

@Table(name="PermanentEmployee")

@PrimaryKeyJoinColumn(name="Id")

**public** **class** PermanentEmployee **extends** Employee {

@Column(name = "salary")

**private** **double** salary;

}

@Repository

**public** **interface** EmployeeRepository **extends** JpaRepository<Employee, Long> {

}

@Service

**public** **class** EmployeeService {

@Autowired

**private** EmployeeRepository repo;

**public** **void** saveInfo(Employee emp) {

repo.save(emp);

}

}

**Database Design**

Table

Description automatically generated Graphical user interface, application

Description automatically generated Graphical user interface, application, table

Description automatically generated

**public** **class** TestTablePerConcreteClass {

**public** **static** **void** storeInfo(ApplicationContext applicationContext) {

EmployeeService service = applicationContext.getBean(EmployeeService.**class**);

Employee employee=**new** Employee();

employee.setName("John");

service.saveInfo(employee);

PermanentEmployee permanentEmployee=**new** PermanentEmployee();

permanentEmployee.setName("Jacob");

permanentEmployee.setSalary(30000);

service.saveInfo(permanentEmployee);

ContractEmployee contractEmployee=**new** ContractEmployee();

contractEmployee.setName("Raj");

contractEmployee.setHourlyRate(2000);

service.saveInfo(contractEmployee);

}

}

**Complete Spring Boot Class is given below**

@SpringBootApplication

**public** **class** StandaloneJPAInheritanceMappingApp {

**public** **static** **void** main(String[] args) {

ApplicationContext applicationContext = SpringApplication.*run*(StandaloneJPAInheritanceMappingApp.**class**, args);

// TestTablePerHierarchy.storeInfo(applicationContext);

// TestTablePerConcreteClass.storeInfo(applicationContext);

TestTablePerConcreteClass.*storeInfo*(applicationContext);

}

}