CURSO DE HIBERNATE 5

OpenWebinars

HIBERNATE

(1) INTRODUCCION

Persistencia, desfase objecto-relacional, ORM. Productos y estándares

(2) HIBERNATE

Más que un ORM. Comparativa con otros productos. JPA. Maven. Módulos

(3) PRIMER PROYECTO

Hibernate.cfg.xml, EntityManager y persistence.xml

(4) ENTIDADES

Definición del modelo del dominio. Entidades y ciclo de vida. XML y anotaciones. Tipos de datos.

(5) ASOCIACIONES

ManyToOne, OneToMany, OneToOne, ManyToMany

HIBERNATE

(6) ELEMENTOS AVANZADOS

Campos calculados, herencia.

(7) COLECCIONES

Mapeo de colecciones. Tipos (list, set, map). Colecciones ordenadas (sorted vs. ordered).

(8) GENERACION DEL ESQUEMA

Customización del proceso de generación del esquema.

(9) CONTEXTO DE PERSISTENCIA

Almacenamiento, recuperación y borrado de entidades.

(10) TRANSACCIONES

Control de concurrencia. Patrones y antipatrones.

HIBERNATE

(12)
ENVERS

Introducción a la
auditoria de entidades.

(11)
CONSULTAS
HPQL VS JPQL

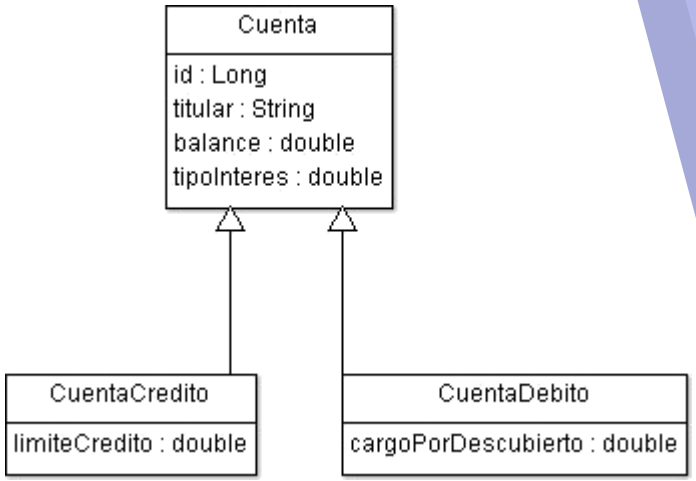
Consultas con
parámetros,
Anotaciones. SQL nativo

1.

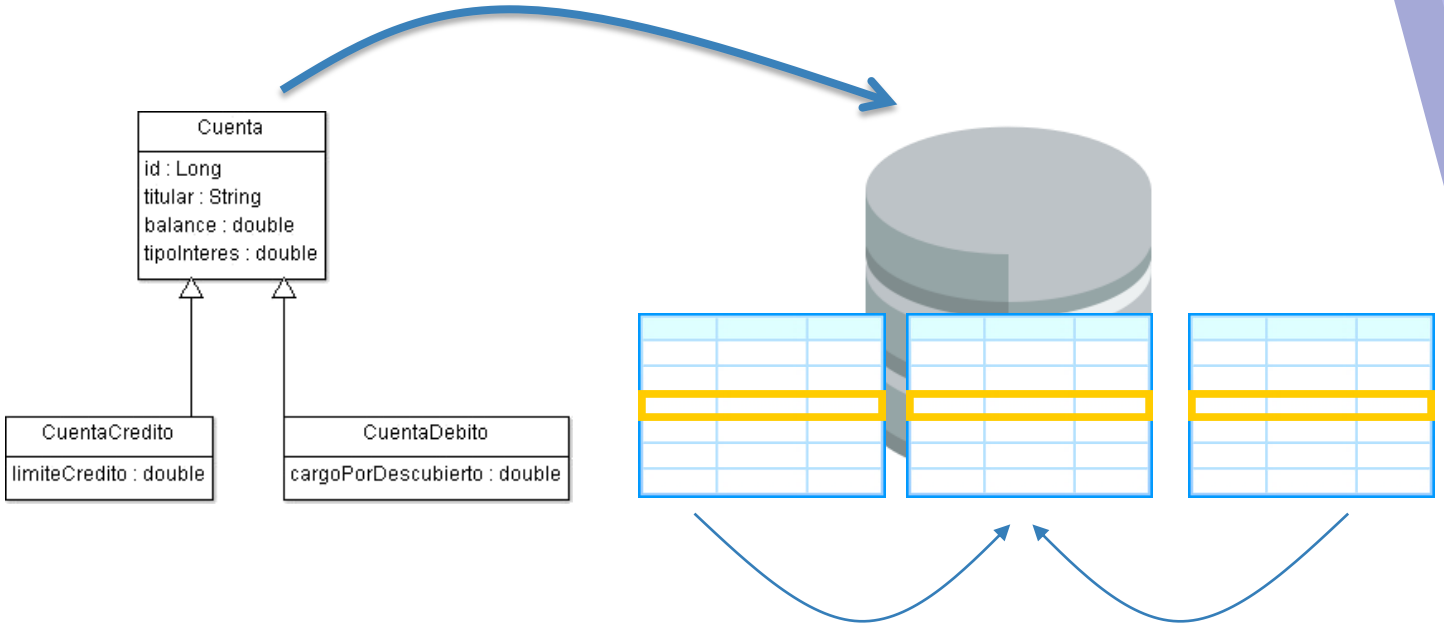
HERENCIA DE ENTIDADES

HERENCIA

Mediante este mecanismo, una serie de clases ***extendidas*** pueden incorporar el comportamiento y la estructura de otra clase, llamada ***base.***



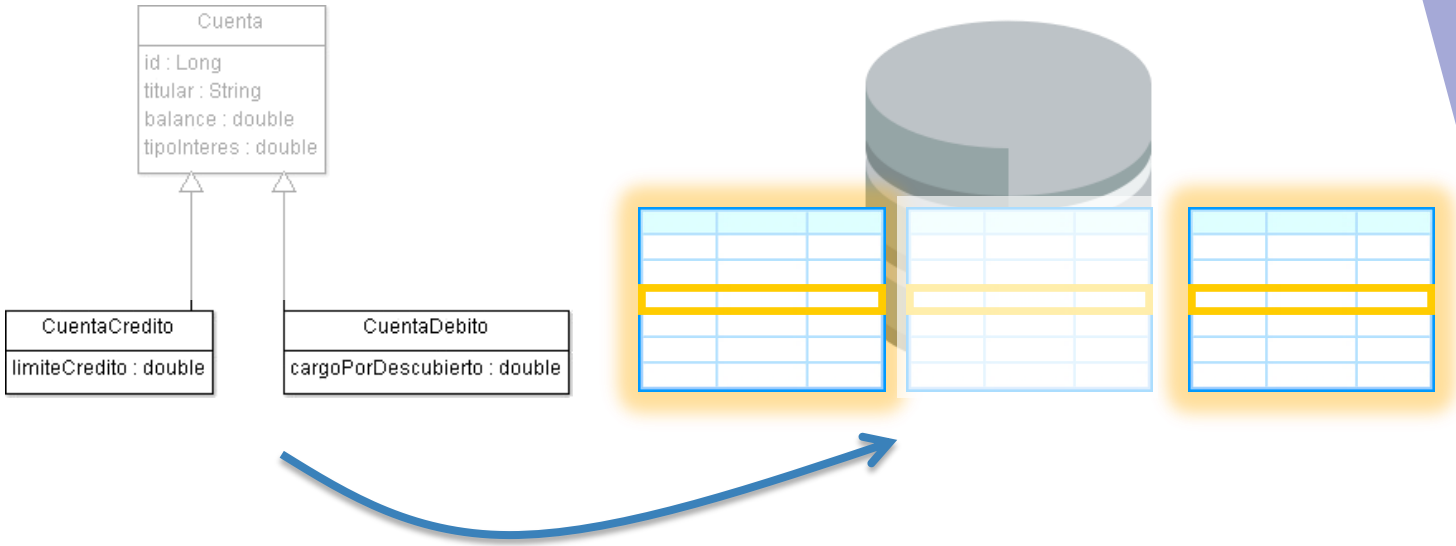
HERENCIA


Cuenta
id : Long
titular : String
balance : double
tipoInteres : double
CuentaCredito
limiteCredito : double
CuentaDebito
cargoPorDescubierto : double

2.

@MappedSuperclass

@MappedSuperclass



@MappedSuperclass

- Indica que hay herencia
- No traslada a tabla la clase ***base***
- Cada tabla asociada a una clase ***extendida*** tiene los atributos de base + propios.
- No permite consultas polimórficas.
- Herencia Java.

```java
@MappedSuperclass
public class Cuenta implements Serializable {
```

```java
@Entity
public class CuentaCredito extends Cuenta implements Serializable {
```

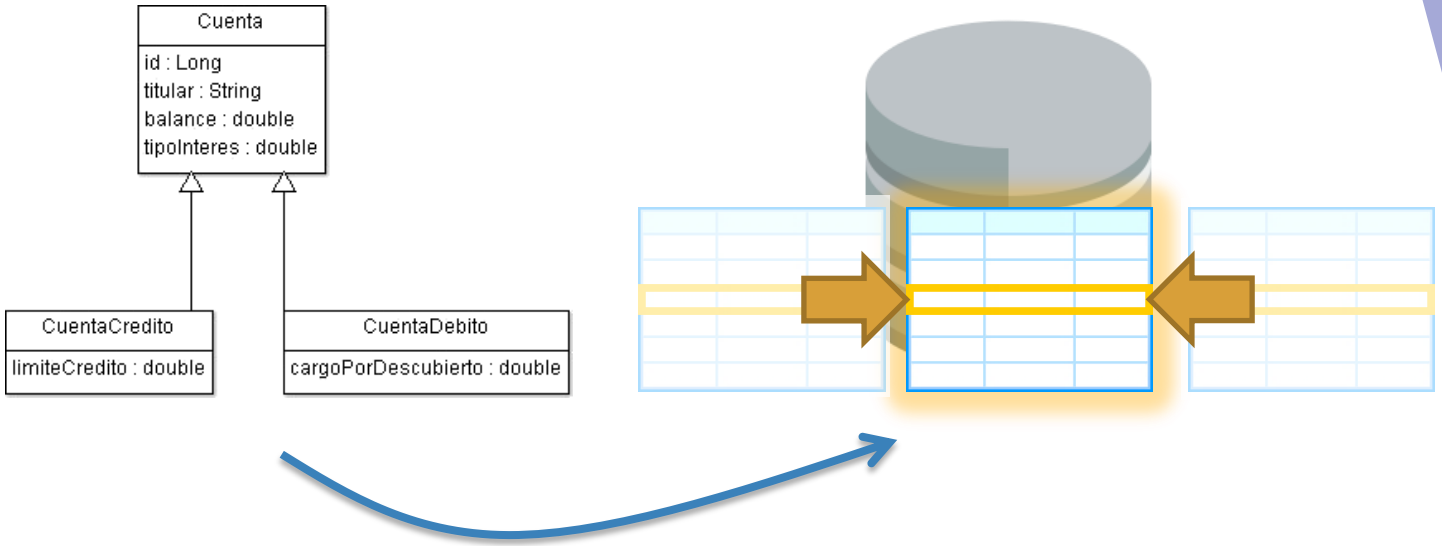
@MappedSuperclass

```
create table CuentaCredito (
       id bigint not null,
        balance double precision not null,
        tipoInteres double precision not null,
        titular varchar(255),
        limiteCredito double precision not null,
        primary key (id)
    ) engine=InnoDB

    create table CuentaDebito (
           id bigint not null,
            balance double precision not null,
            tipoInteres double precision not null,
            titular varchar(255),
            cargoPorDescubierto double precision not null,
            primary key (id)
        ) engine=InnoDB
```

3.

SINGLE TABLE

SINGLE TABLE



@Inheritance(strategy = InheritanceType.SINGLE_TABLE)

- Para toda la jerarquía construye una sola tabla
- Traslada todos los atributos de las entidades extendidas a la tabla de la entidad base.
- Podemos añadir un discriminante con ***@DiscriminatorValue.***
- Permite consultas polimórficas.

```
@Entity
@Inheritance(strategy = InheritanceType.SINGLE_TABLE)
public class Cuenta implements Serializable {
```

```
@Entity
@DiscriminatorValue("CC")
public class CuentaCredito extends Cuenta implements Serializable {
```

@Inheritance(strategy = InheritanceType.SINGLE_TABLE)

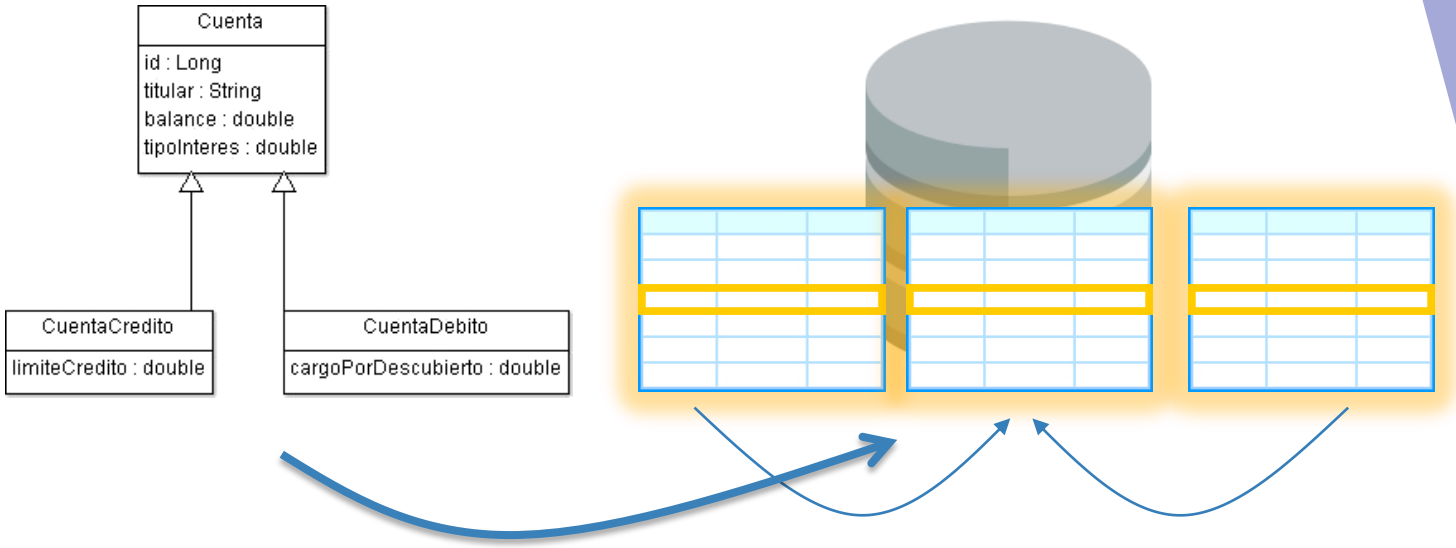
```
create table Cuenta (
       DTYPE varchar(31) not null,
        id bigint not null,
        balance double precision not null,
        tipoInteres double precision not null,
        titular varchar(255),
        cargoPorDescubierto double precision,
        limiteCredito double precision,
        primary key (id)
    ) engine=InnoDB
```

DTYPE	id	balance	tipoInteres	titular	cargoPorDescubierto	limiteCredito
CuentaCredito	1	500	0.1	Luismi	NULL	600
CuentaDebito	2	200	0	Luismi	6.5	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL

4.

JOINED TABLE

JOINED TABLE



@Inheritance(strategy = InheritanceType.JOINED)

- Construye una tabla para cada entidad.
- La entidad base traslada todos sus atributos.
- Las entidades extendidas trasladan los atributos propios y una clave externa.
- Por cada instancia de entidad extendida insertamos 1 fila en dos tablas.

```
@Entity
@Inheritance(strategy = InheritanceType.JOINED)
public class Cuenta implements Serializable {
```

```
@Entity
public class CuentaCredito extends Cuenta implements Serializable {
```

@Inheritance(strategy = InheritanceType.JOINED)

```
create table Cuenta (
   id bigint not null,
    balance double precision not null,
    tipoInteres double precision not null,
    titular varchar(255),
    primary key (id)
) engine=InnoDB


create table CuentaCredito (
   limiteCredito double precision not null,
    id bigint not null,
    primary key (id)
) engine=InnoDB

create table CuentaDebito (
   cargoPorDescubierto double precision not null,
    id bigint not null,
    primary key (id)
) engine=InnoDB
```

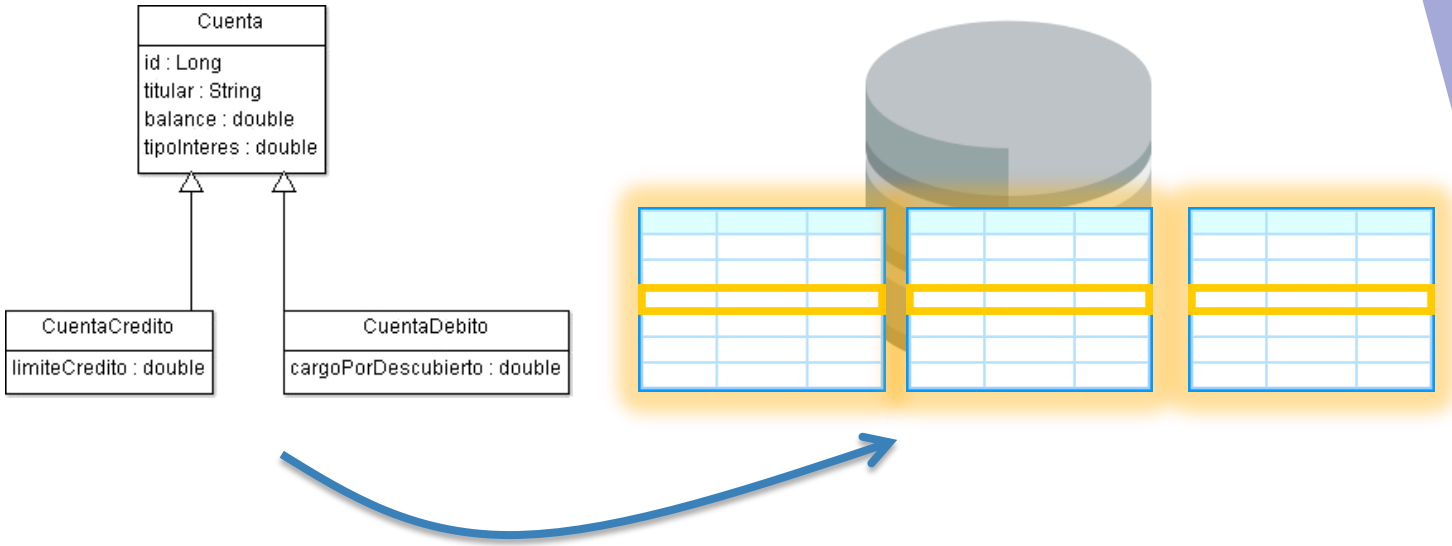
```
alter table CuentaCredito
   add constraint FK6641o76fphgs98cbv18sd7htc
   foreign key (id)
   references Cuenta (id)

alter table CuentaDebito
   add constraint FK2gigt3h95mq590key3xvkhqk0
   foreign key (id)
   references Cuenta (id)
```

USO **EXHAUSTIVO** DE JOIN

5.

TABLE PER CLASS

TABLE PER CLASS



@Inheritance(strategy = InheritanceType.TABLE_PER_CLASS)

- Construye una tabla para cada entidad.
- La entidad base traslada todos sus atributos.
- Las entidades extendidas trasladan los atributos propios y los de la entidad base.
- No hay claves externas.
- Las consultas polimórficas usan UNION.

```
@Entity
@Inheritance(strategy=InheritanceType.TABLE_PER_CLASS)
public class Cuenta implements Serializable {
```

```
@Entity
public class CuentaCredito extends Cuenta implements Serializable {
```

@Inheritance(strategy = InheritanceType.TABLE_PER_CLASS)

```
create table Cuenta (
  id bigint not null,
  balance double precision not null,
  tipoInteres double precision not null,
  titular varchar(255),
  primary key (id)
) engine=InnoDB
```

```
create table CuentaCredito (
  id bigint not null,
  balance double precision not null,
  tipoInteres double precision not null,
  titular varchar(255),
  limiteCredito double precision not null,
  primary key (id)
) engine=InnoDB
```

```
create table CuentaDebito (
  id bigint not null,
  balance double precision not null,
  tipoInteres double precision not null,
  titular varchar(255),
  cargoPorDescubierto double precision not null,
  primary key (id)
) engine=InnoDB
```

6.

CAMPOS GENERADOS O DERIVADOS

@Generated

- Define una propiedad cuyo valor viene de un ***cálculo***.
- Generado por la base de datos (no Java).
- Actualización en inserción o siempre.

```
@Generated(value=GenerationTime.ALWAYS)
@Column(columnDefinition=
    " varchar(512) AS (CONCAT(email,' (', city,')')) "
)
private String generated;
```

@CreationTimestamp

- Asociada a una fecha (varios tipos).
- Fecha y hora de JVM

```
@CreationTimestamp
private Date createdDate;
```

@ColumnTransformer

- Personaliza como se lee o escribe una columna de una entidad.
- Funciones de la base de datos.

```
@Column(name="pswd")
@ColumnTransformer(
        write= " MD5(?) "
)
private String password;
```