

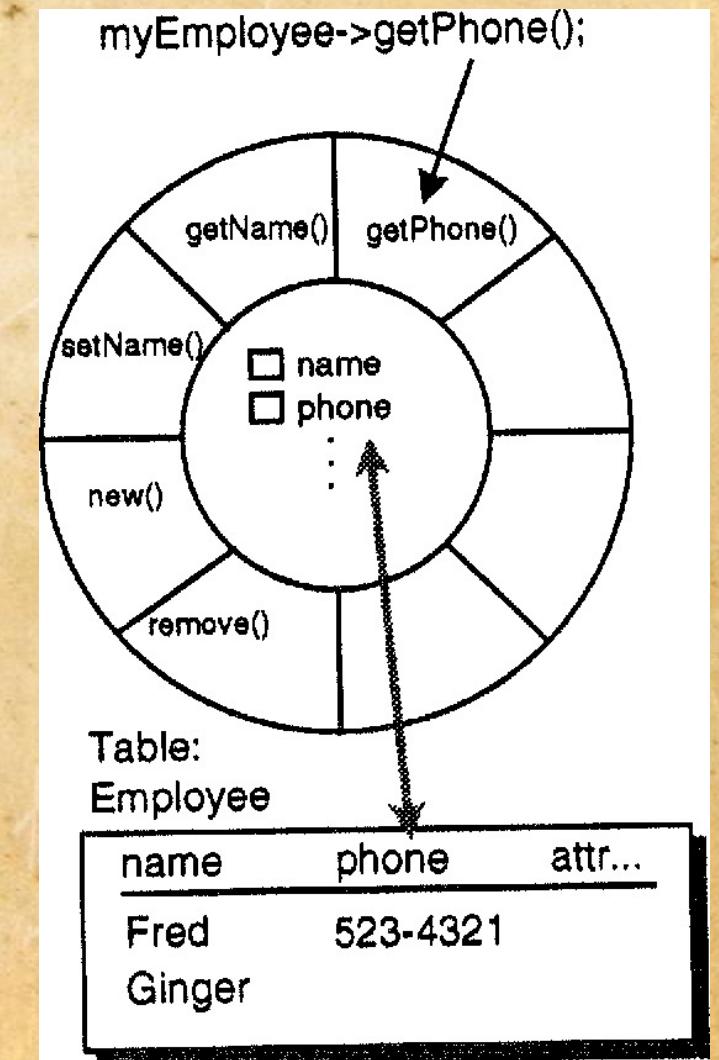
Programació



UT14.1 Introducció a
l'API de persistència

Introducció

La persistència d'objectes té com a objectiu preservar l'estat dels objectes (el valor dels seus membres, incloent-hi les referències a altres objectes) de manera que la informació persistisca més enllà del temps en el qual l'objecte es troba en memòria (RAM).



Introducció

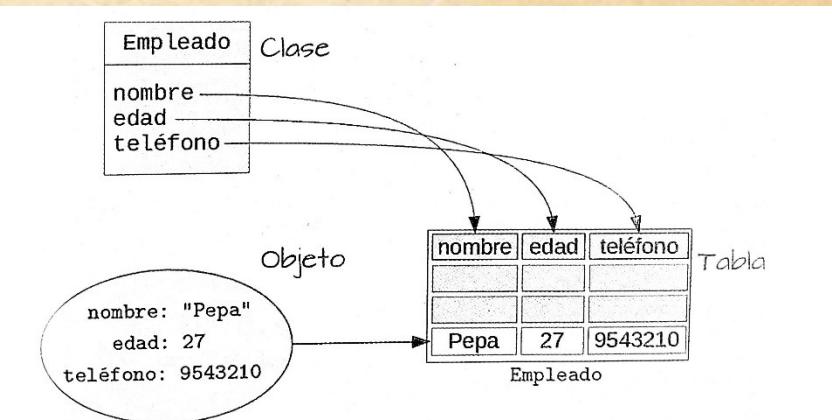
Entre les alternatives per a aconseguir este propòsit tenim:

- **La serialització d'objectes.** Consistix a convertir un objecte en una seqüència de bytes usualment per a guardar-la en un arxiu o transmetre-la via xarxa.
- **Emmagatzemar l'estat en una BD relacional/SQL.** Per a este propòsit es recomana usar alguna ferramenta ORM.
- **Emmagatzemar l'estat en una BD no-relacional/no-SQL.**
Per exemple, les BD orientades a documents en format JSON com MongoDB tenen un esquema d'emmagatzemament molt més compatible amb els objectes que el d'una BD relacional.

Mapatge Objecte-Relacional (ORM)

La tècnica del mapatge objecte-relacional consisteix a convertir (o traduir) les classes i objectes en taules i registres. D'esta manera emmagatzemarem objectes en un SGBD relacional.

Habitualment es converteix cada classe en una taula, però es possible que la mateixa classe es puga convertir en més d'una taula, o que diverses classes amb cert grau d'associació es convertisquen en una única taula.



Java Persistence API

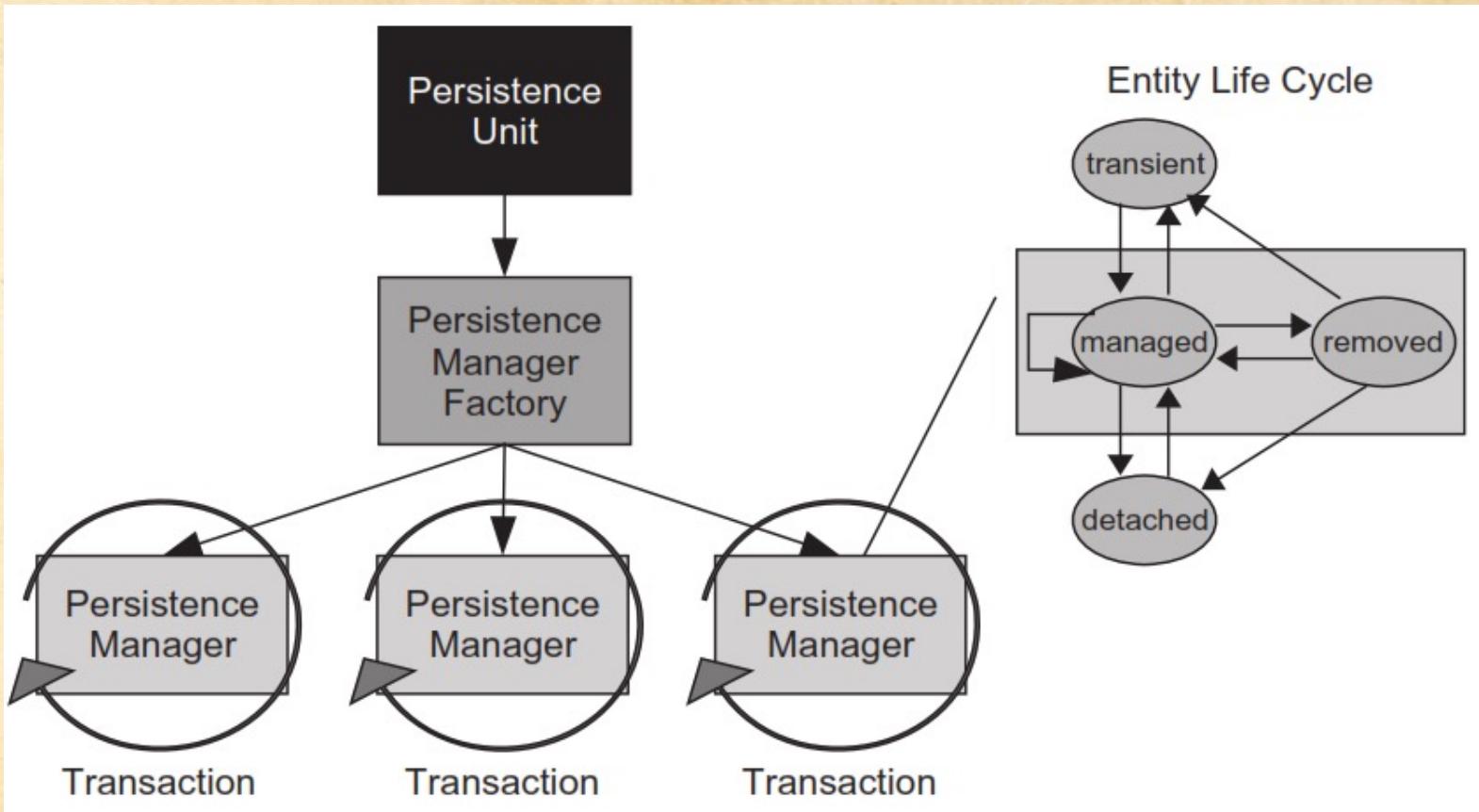
JPA és una **especificació** tècnica que descriu les interfícies de programació, els components de configuració i el llenguatge de consulta empleat per a **persistir l'estat dels objectes** en bases de dades relacionals (ORM).

Hi ha diverses implementacions d'esta especificació com Hibernate, OpenJPA i EclipseLink entre d'altres.



Components JPA

Estos són els component més importants de JPA



Components JPA

La Persistence Unit (Unitat de Persistència) organitza i gestiona metades relacionades amb la connexió a la BD, el tipus de gestió de transacció o les que descriuen la correspondència ("mapejat") entre classes/atributs i taules/columnes; la **unitat de persistència** es definix en l'arxiu **persistence.xml**. Les metades del mapejat es poden expressar en dos formats: xml i anotacions Java.

En temps d'execució, el Persistence Unit és l'entrada per a crear un **Persistence Manager Factory** (Fàbrica de Gestor de Persistència) que per la seu banda s'usa per a crear instàncies **Persistence Manager** (Gestor de Persistència).

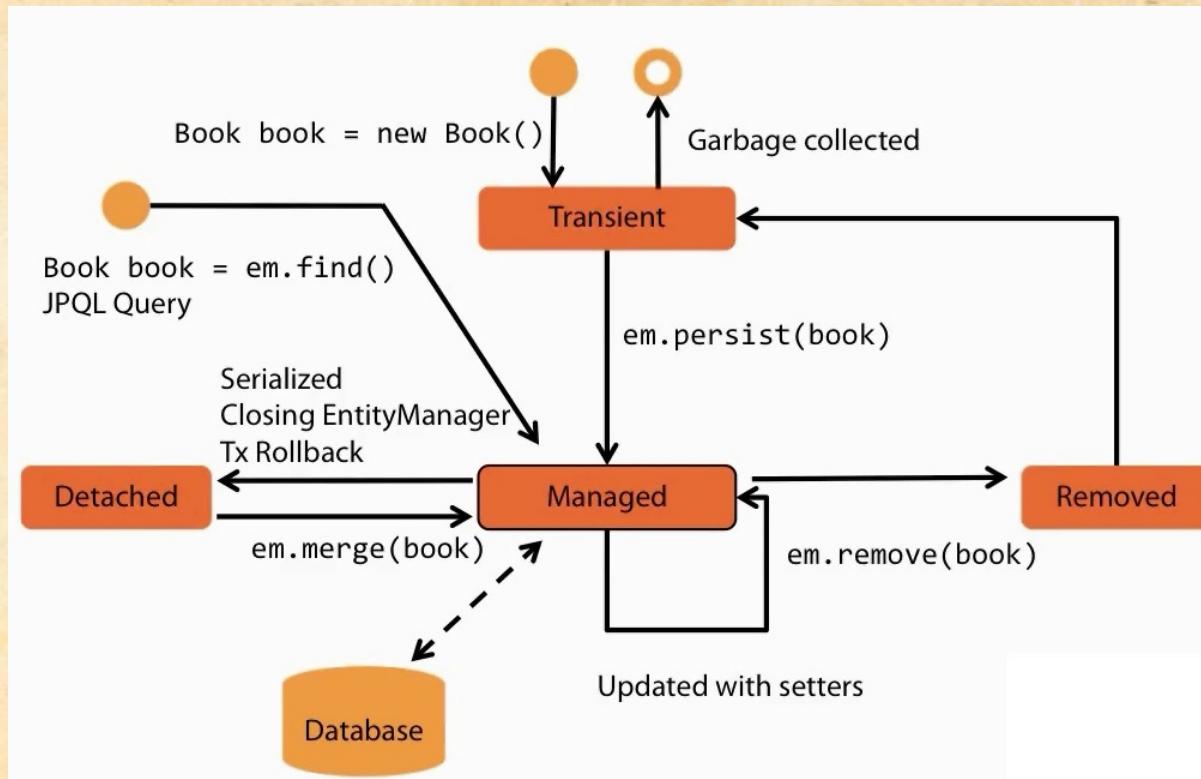
Components JPA

Instàncies de la classe Persistence Mànager són responsables **d'executar operacions CRUD** que mouen les entitats entre la màquina virtual de Java i la base de dades relacional en un procés conegut com el **cicle de vida de l'entitat**. Les operacions realitzades per un Persistence Mànager es realitzen en el context d'una **transacció**.

A més d'oferir una interfície que permet realitzar operacions CRUD, un Entity Mànager gestiona les entitats monitoritzant els canvis realitzats a l'estat dels objectes. El conjunt d'entitats gestionades per un Entity Mànager es coneix com a **context de persistència**.

Components JPA

Finalment, per la seua relació amb un EntityManager una entitat pot estar en algun dels estats següents:



Unitat de persistència

La unitat de persistència és l'element fonamental per a configurar el mapatge objecte-relacional ja que vincula un seguit d'entitats amb una base de dades.

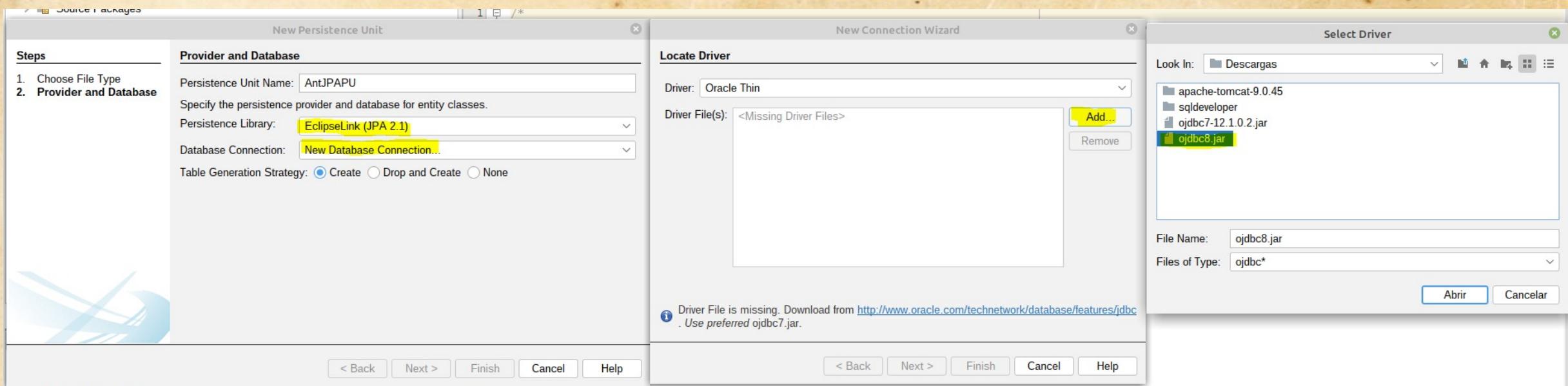
En un mateix projecte poden existir diverses unitats de persistència, que d'identifiquen amb un nom únic.

Per a crear una unitat de persistència:

- File > New File
- En la categoria seleccionem “Persistence” i en el tipus d'arxiu “Persistence Unit”
 - Pots observar que es crea “persistence.xml” a META-INF

Unitat de persistència

És possible utilitzar una connexió existent a la base de dades, però si no existeix cal crear-la mitjançant “New Database Connection”



Unitat de persistència

És possible utilitzar una connexió existent a la base de dades, però si no existeix cal crear-la mitjançant “New Database Connection”

The image displays three windows from an IDE:

- New Connection Wizard - Customize Connection:** Shows connection details for "ORCLCDB": Driver Name (Oracle Thin (Service ID (SID))), Host (localhost), Port (1521), Service ID (SID) (ORCLCDB), User Name (javauser), Password (redacted), and JDBC URL (jdbc:oracle:thin:@localhost:1521:ORCLCDB). A "Connection Succeeded" message is present.
- New Connection Wizard - Choose Database Schema:** A step in the wizard where the user selects the schema (JAVAUSER) to be displayed.
- New Persistence Unit:** A configuration dialog for creating a persistence unit:
 - Steps:** Step 2: Provider and Database.
 - Provider and Database:** Persistence Unit Name (AntJPAPU), Persistence Library (EclipseLink (JPA 2.1)), Database Connection (jdbc:oracle:thin:@localhost:1521:ORCLCDB [javauser on JAV...]), and Table Generation Strategy (Create selected).

Unitat de persistència

Si intentem compilar l'aplicació ens donarà un error:

The screenshot shows a Java code editor with the following code:

```
/*
 * public static void main(String[] args) {
 *     EntityManagerFactory emf = Persistence.createEntityManagerFactory("AntJPAPU");
 * }
 */
antjpa.AntJPA > main > emf >
```

Below the code editor is a terminal window showing the following error message:

```
Exception in thread "main" javax.persistence.PersistenceException: Exception [EclipseLink-4003]
[EL Severe]: ejb: 2022-07-05 11:35:39.473--ServerSession(1765250898)--Exception [EclipseLink-4003]
Exception Description: Configuration error. Class [oracle.jdbc.OracleDriver] not found.
Exception Description: Configuration error. Class [oracle.jdbc.OracleDriver] not found.
```

Ens falta incloure el driver OJDBC:

- Botó dret sobre el directori “Libraries”
- Add JAR/Folder
- Seleccionem el nostre fitxer ojdbc8.jar



Unitat de persistència

Podem comprovar que la connexió funciona correctament:

```
public static void main(String[] args) {
    EntityManagerFactory emf = Persistence.createEntityManagerFactory("AntJPAPU");
    EntityManager em = emf.createEntityManager();
    System.out.println("Nom de la categoria 1: " + em.createNativeQuery("SELECT nom FROM categories WHERE id='1'").getSingleResult());
}

out x
Run (MavenJPA) × AntJPA (run) ×
run:
Nom de la categoria 1: Teclats
BUILD SUCCESSFUL (total time: 7 seconds)
```