

UF13.- ACCÉS A BASES DE DADES

- Teoria -

PROGRAMACIÓ
CFGs DAW

Joan V. Cassany
Guillermo Garrido Portes

jv.cassanycoscolla@edu.gva.es
g.garridoportes@edu.gva.es

2022/2023 1

1. INTRODUCCIÓ

BASE DE DADES



GENERALITAT
VALENCIANA



Una **base de dades** és una col·lecció de dades classificades i estructurades que són guardades en fitxers, però referenciades com si d'un únic fitxer es tractara.

Les dades d'una **base de dades relacional** s'emmagatzemen en **taules** lògicament relacionades entre si utilitzant **camps clau** comuns.

Cada taula disposa les dades en files i columnes. A les **files** se les denomina **tuplas** o registres i a les **columnes** s'anomenen **camps**.

Per a crear i manipular bases de dades relacionals, existeixen en el mercat diversos **sistemes de gestió de bases de dades (SGBD)**; per exemple, Access, SQL Server, Oracle i DB2. Altres SGBD de lliure distribució són MySQL/MariaDB i PostgreSQL.

2. LLENGUATGE SQL



GENERALITAT
VALENCIANA



Els usuaris d'un sistema administrador de bases de dades poden realitzar sobre una determinada base operacions com inserir, recuperar, modificar i eliminar dades, així com afegir noves taules o eliminar-les. Aquesta operacions s'expressen generalment en un llenguatge denominat **SQL (Structured Query Language)**.

Existeixen tres tipus de comandos en SQL: **DCL**

- Els **DDL** (Data Definition Lenguaje), que permeten crear (CREATE), modificar (ALTER) i esborrar (DROP) noves bases de dades, taules, camps i vistes.
- Els **DML** (Data Manipulation Lenguaje), que permeten introduir (INSERT) informació en la BD, esborrar-la (DELETE) i modificar-la (UPDATE).
- Els **DQL** (Data Query Lenguaje), que permeten generar consultes per a ordenar, filtrar i extraure informació de la base de dades (SELECT).
- Els **DCL** (Data Control Lenguaje), que permeten atorgar permisos (GRANT) o revocar-los (REVOKE).

3. JDBC



GENERALITAT
VALENCIANA



Abans de començar necessitem un sistema de gestió de base de dades instal·lat en el nostre ordinador en el qual poder tindre bases de dades a les quals connectar-nos des dels nostres programes escrits en llenguatge Java.

Utilitzar **XAMPP** que incorpora un sistema gestor de bases de dades MySQL/MariaDB, un servidor web Apache i l'eina phpMyAdmin per a treballar amb bases de dades.

<https://www.apachefriends.org>

Nota: Instal·lació en Linux

Canvia els permisos al instal·lador:

Executa el instal·lador:

Se instal·larà en:

Iniciar Xampp:

Finalitzar XAMPP:

Reiniciar XAMPP:

Accés a l'eina gràfica:

```
chmod 755 xampp-linux-*-installer.run
```

```
sudo ./xampp-linux-*-installer.run
```

```
/opt/lampp
```

```
sudo /opt/lampp/lampp start
```

```
sudo /opt/lampp/lampp stop
```

```
sudo /opt/lampp/lampp restart
```

```
cd /opt/lampp
```

```
sudo ./manager-linux-x64.run
```

3. JDBC



GENERALITAT
VALENCIANA



Java pot connectar-se amb diferents SGBD, però ha d'existir sempre un mediador entre l'aplicació i el sistema de base de dades i en Java aqueixa funció la realitza la API estàndard de JAVA denominada **JDBC (Java Data Base Connection)**.

JDBC farà transparent para la codificació Java la BBDD a que ens connectem.

L'**API JDBC** fa possible la realització de les següents tasques:

- Establir una connexió amb una base de dades a través d'un **driver**.
- Enviar sentències **SQL**.
- Manipular dades.
- Processar els resultats de l'execució de les sentències.

4. ACCES BASES DE DADES DES DE NETBEANS

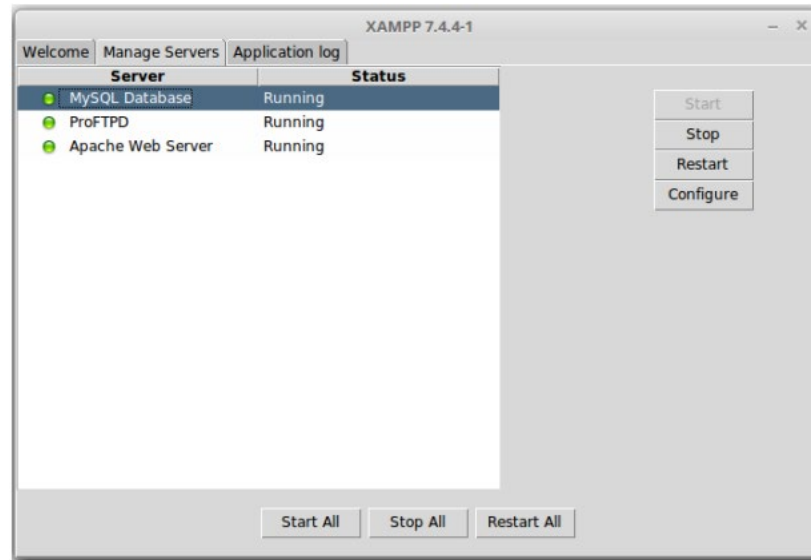
INICIAR XAMPP



GENERALITAT
VALENCIANA

ceedcv
CENTRE ESPECÍFIC
D'EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

Sempre que vulguem treballar amb el servidor i les bases de dades del mateix haurem d'iniciar l'eina XAMPP descarregada anteriorment i iniciar els serveis Apache i MySQL/MariaDB:

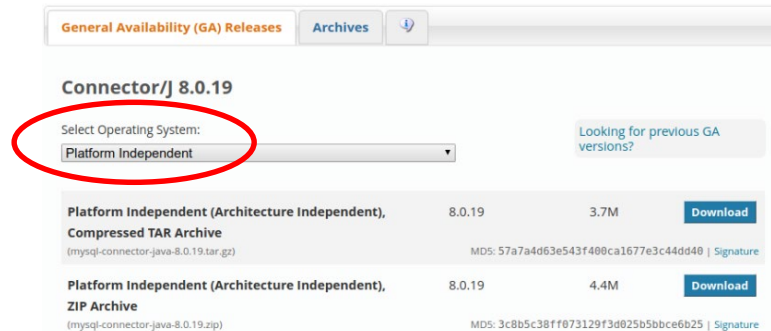


4. ACCES BASES DE DADES DES DE NETBEANS

INSTAL·LACIÓ DEL JDBC DRIVER

Per a poder connectar-nos a una base de dades MySQL/MariaDB des de l'explorador de NetBeans necessitem instal·lar el driver mysql-connector de Java, que el descarregarem de:

<http://dev.mysql.com/downloads/connector/j/>



No fa falta registrar-se per a descarregar-lo, només s'ha de prémer en “No thanks, just start my download.” i descomprimir el ZIP i cercar l'arxiu:

mysql-connector-java-8.0.19.jar

4. ACCES BASES DE DADES DES DE NETBEANS

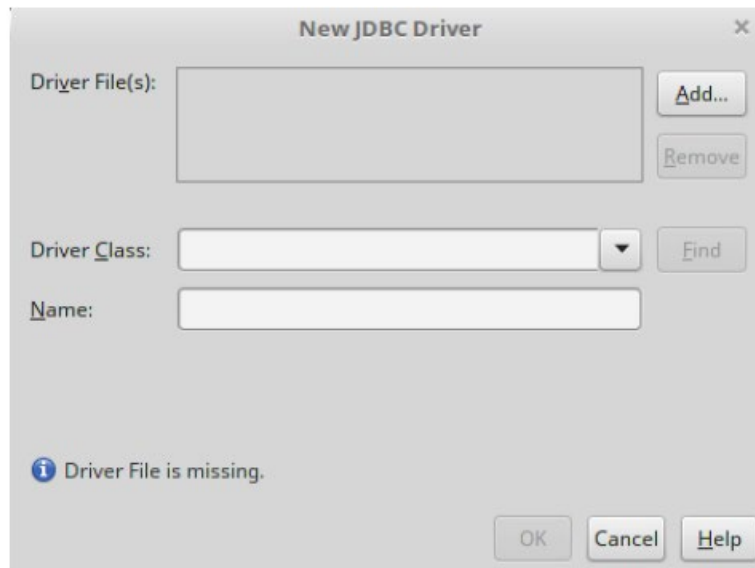
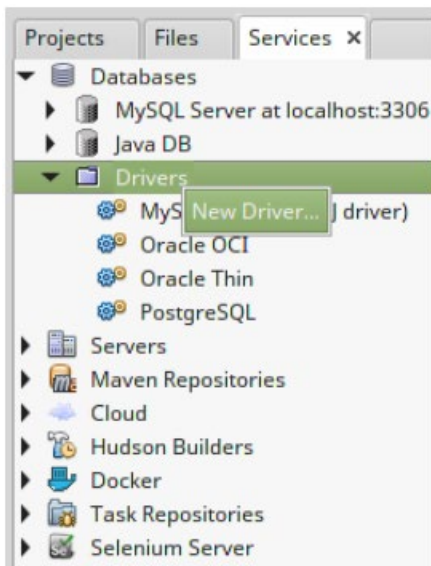
INSTAL·LACIÓ DEL JDBC DRIVER



GENERALITAT
VALENCIANA

ceedcv
CENTRE ESPECÍFIC
D'EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

Ara obri **NetBeans**, veu al panell **Services** → **Databases**, fes clic dret en l'opció **Drivers** i selecciona l'opció **New Driver** del menú contextual. Es mostrarà el diàleg *New JDBC Driver*.



4. ACCES BASES DE DADES DES DE NETBEANS

INSTAL·LACIÓ DEL JDBC DRIVER

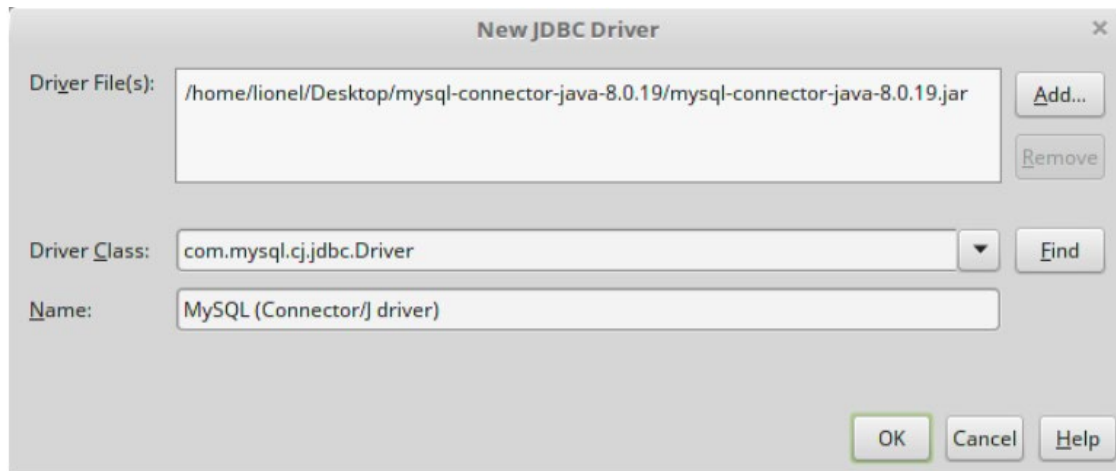


GENERALITAT
VALENCIANA

ceedcv
CENTRE ESPECÍFIC
D'EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

Fes clic en [Add...] i selecciona el fitxer **mysql-connector-java-8.0.19.jar**.

Una vegada seleccionat el fitxer ens ha d'emplenar la classe principal del Driver que en el nostre cas ha de ser **com.mysql.cj.jdbc.Driver** i el nom del nostre driver, per exemple MySQL (Connector/J driver).



4. ACCES BASES DE DADES DES DE NETBEANS

CONNEXIÓ A LA BASE DE DADES

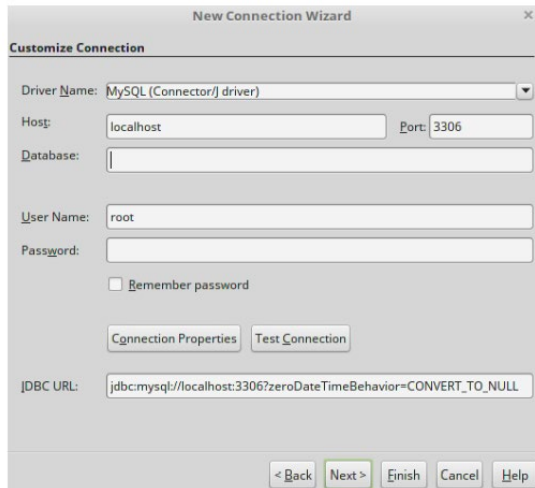
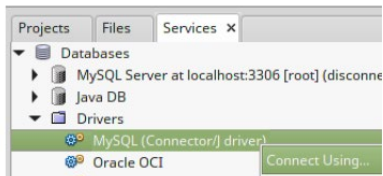


GENERALITAT
VALENCIANA

ceedcv
CENTRE ESPECÍFIC
D'EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

NOTA: Si estàs utilitzant l'última versió de NetBeans és possible que en **Services** → **Databases** → **Drivers** ja t'aparega l'opció '**MySQL (Connector/J driver)**'.

Malgrat això pot ser que necessites instal·lar el driver. Dona-li al driver amb clic dret → '**Connect using**' i comprova si en la finestra t'apareix el missatge 'driver file is missing'. En tal cas, dona-li a 'add driver' i selecciona l'arxiu jar esmentat anteriorment.



Database: El nom de la base de dades a la qual volem accedir. Podem deixar aquest camp buit per a poder accedir a totes les que existisquen en el servidor.

Fes clic a **Finalitzar**

4. ACCES BASES DE DADES DES DE NETBEANS

CONNEXIÓ A LA BASE DE DADES

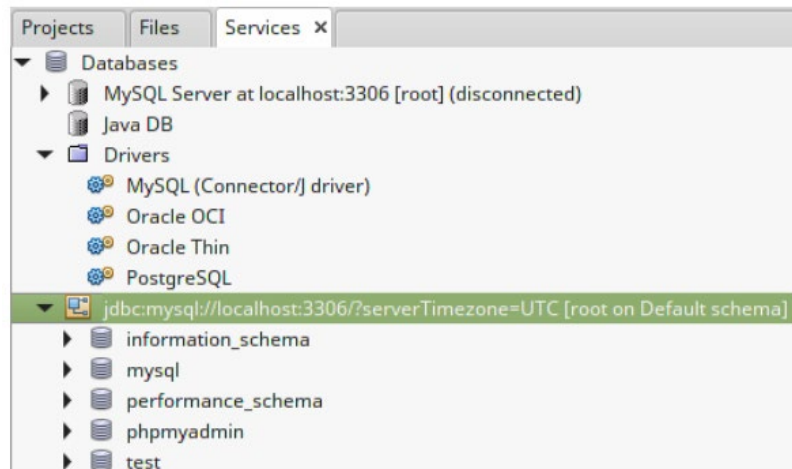


GENERALITAT
VALENCIANA



Si tot va bé, en Services → Databases apareixerà una nova connexió anomenada **`jdbc:mysql://localhost:3306... [root on Default schema]`**

Si falla a causa d'un error 'server time zone' pots solucionar-lo afegint a l'URL la variable `serverTimezone` amb el valor UTC: `jdbc:mysql://localhost:3306/?serverTimezone=UTC`



Si despleguem la connexió creada veurem que podem accedir a totes les bases de dades, taules, etc. emmagatzemades en el servidor al qual estem connectats. D'aquesta senzilla manera podrem consultar o modificar la base de dades i el seu contingut.

4. ACCES BASES DE DADES DES DE NETBEANS


CONNEXIÓ A LA BASE DE DADES

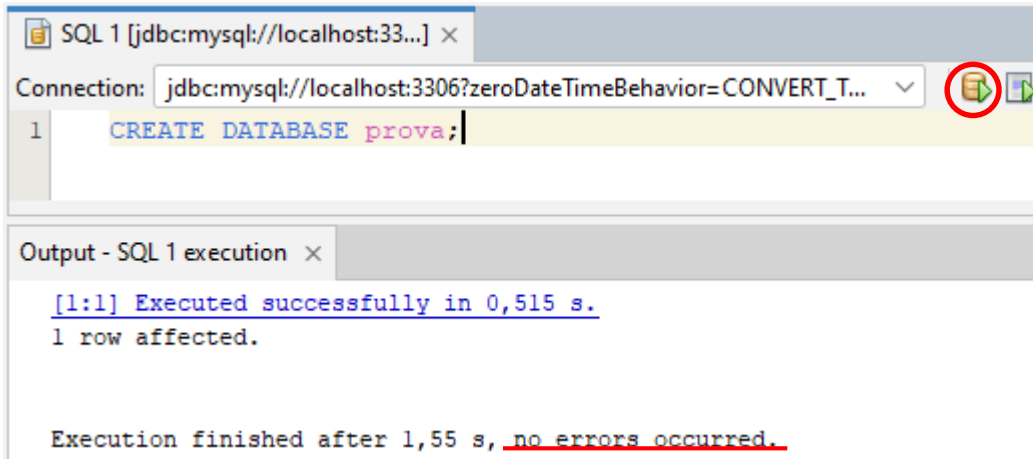
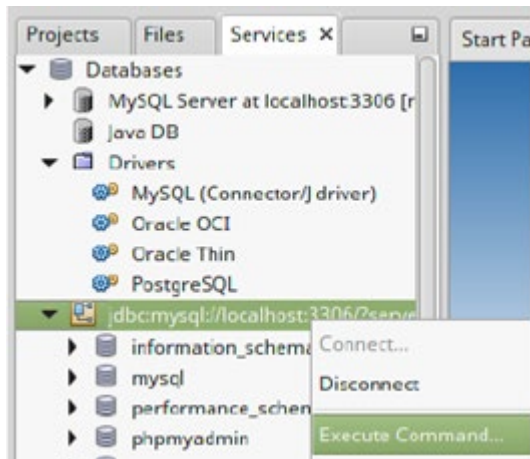


GENERALITAT
VALENCIANA

ceedcv
CENTRE ESPECÍFIC
D'EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

Per a executar un comando sobre la base de dades cal fer clic dret sobre la connexió i seleccionar l'opció 'Execute Command...'.

S'obrirà un panell en l'editor central on podrem introduir sentències SQL i després fer clic en la icona 'Run SQL': 



4. ACCES BASES DE DADES DES DE NETBEANS

CONNEXIÓ A LA BASE DE DADES



GENERALITAT
VALENCIANA



CENTRE ESPECÍFIC
D'EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

La connexió es pot fer de forma genèrica, sense connectar-nos a una BD concreta, o especificant-ne una.

Si no l'he especificat caldrà indicar en cada sentència a quina ens estem connectant:

```
USE tendaonline;  
SELECT * FROM clients;
```

4. ACCES BASES DE DADES DES DE NETBEANS

CONNEXIÓ A LA BASE DE DADES



GENERALITAT
VALENCIANA


ceedcv
CENTRE ESPECÍFIC
D'EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

Si volem connectar-nos a una en concret farem clic dret en

‘MySQL (Connector/J driver)’ → ‘Connect Using...’

Introduir les dades en el diàleg ‘New Connection Wizard’ i en el camp ‘Database’ escriure la base de dades desitjada.

Veurem com la URL de connexió canvia a:

➤  `jdbc:mysql://localhost:3306/prova?zeroDateTimeBehavior=CONVERT_TO_NULL [root on Default schema]`

4. ACCES BASES DE DADES DES DE NETBEANS

EXEMPLE01: CREAR UNA TAULA



GENERALITAT
VALENCIANA

ceedcv
CENTRE ESPECÍFIC
D'EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

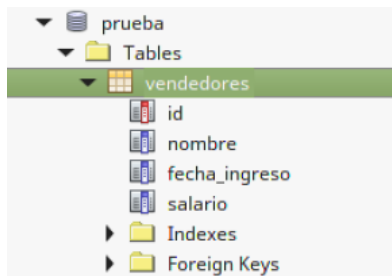
Mitjançant sentències SQL

Seleccionem la connexió

jdbc:mysql://localhost:3306/prova

Fem clic dret sobre ella → **Execute Command**

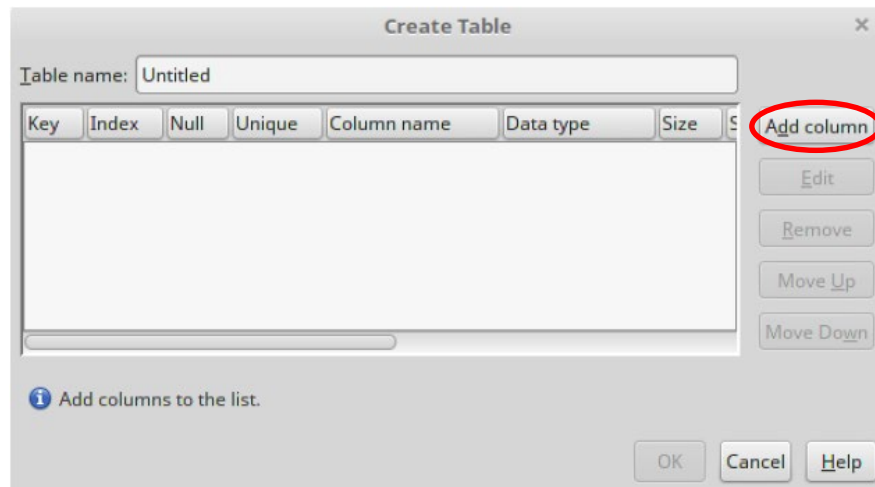
```
CREATE TABLE venedors (  
    id int NOT NULL auto_increment,  
    nom varchar(50) NOT NULL default "",  
    data_ingres date NOT NULL default '0000-00-00',  
    salari float NOT NULL default '0',  
    PRIMARY KEY (id) );
```



Nota: Si no apareix fem clic dret → **Refresh**

Mitjançant l'assistent de creació de taules

Sobre la carpeta 'Tables' de la connexió desitjada, fem clic dret i triem '**Create Table**'



4. ACCES BASES DE DADES DES DE NETBEANS

EXEMPLE01: INSERIR I CONSULTAR DADES

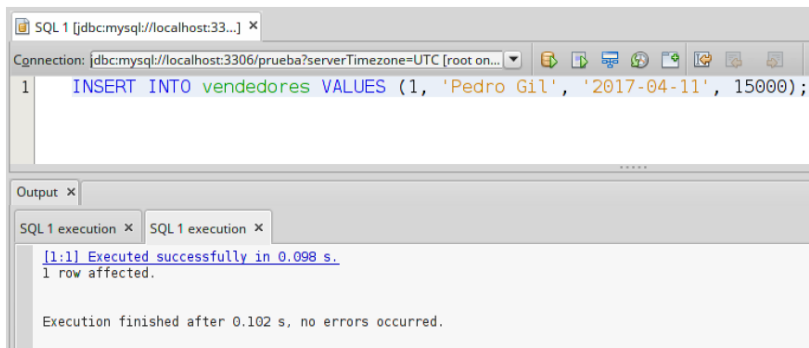


GENERALITAT
VALENCIANA

ceedcv
CENTRE ESPECÍFIC
D'EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

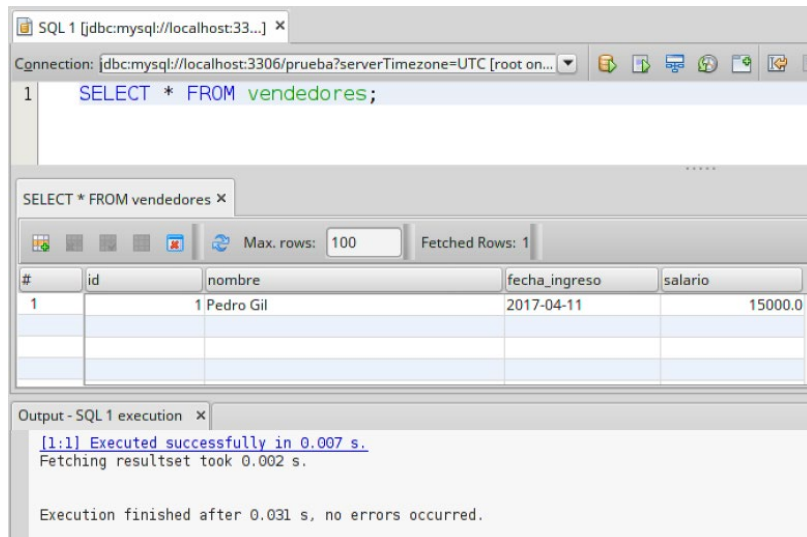
Inserir dades

*INSERT INTO venedors
VALUES (1, 'Pedro Gil', '2017-04-11', 15000);*



Consultar valors

*SELECT * FROM venedors*



4. ACCES BASES DE DADES DES DE NETBEANS

EXEMPLES SQL



GENERALITAT
VALENCIANA



Amb l'Exemple02 es pot practicar SQL.

També, veure exemples més complexos i/o que impliquen dades de diverses taules podeu consultar aquest material complementari:

<https://www.cs.us.es/blogs/bd2013/files/2013/09/consultas-sql.pdf>

5. ACCES BASES DE DADES AMB CODI JAVA

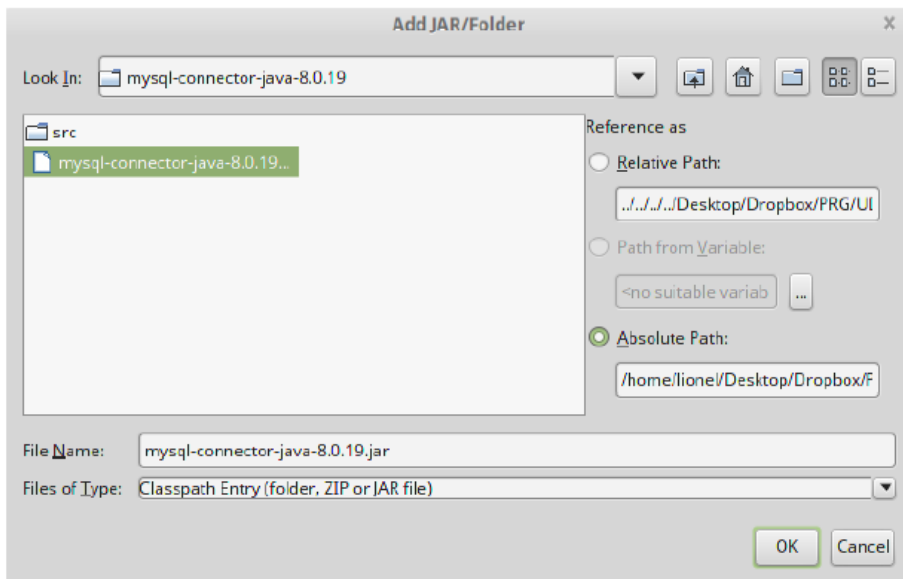
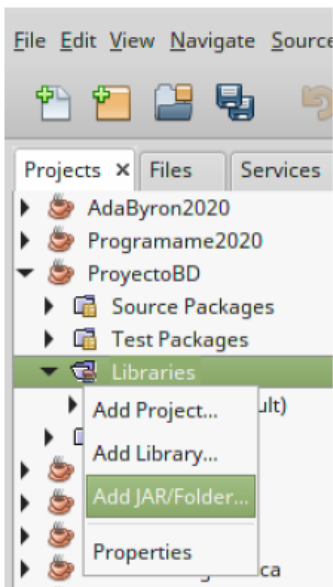
AFEGIR LA LLIBRERIA JDBC AL PROJECTE



GENERALITAT
VALENCIANA

ceedcv
CENTRE ESPECÍFIC
D'EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

Per a poder utilitzar la llibreria JDBC en un projecte Java haurem d'afegir-la al projecte. Farem clic dret sobre la carpeta '**Libraries**' del projecte i seleccionar **[Add JAR/Folder]**. Seleccionar l'arxiu del driver mysql-connector-java-8.0.19.jar i clic en **[OK]**.



5. ACCES BASES DE DADES AMB CODI JAVA

CARREGAR EL DRIVER



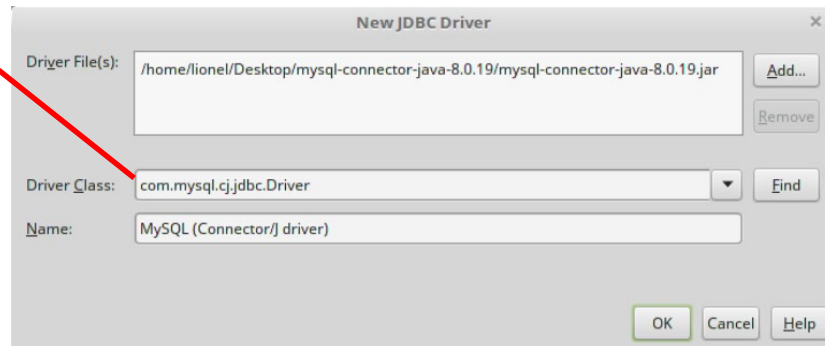
GENERALITAT
VALENCIANA

ceedcv
CENTRE ESPECÍFIC
D'EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

En un projecte Java que realitze connexions a bases de dades és necessari, primer de tot, utilitzar ***Class.forName(...).newInstance()*** per a carregar dinàmicament el Driver que utilitzarem.

```
try {  
    Class.forName("com.mysql.cj.jdbc.Driver").newInstance();  
} catch (Exception e) {  
    // manegem l'error  
}
```

Recorda quan instal·larem el Driver



5. ACCES BASES DE DADES AMB CODI JAVA

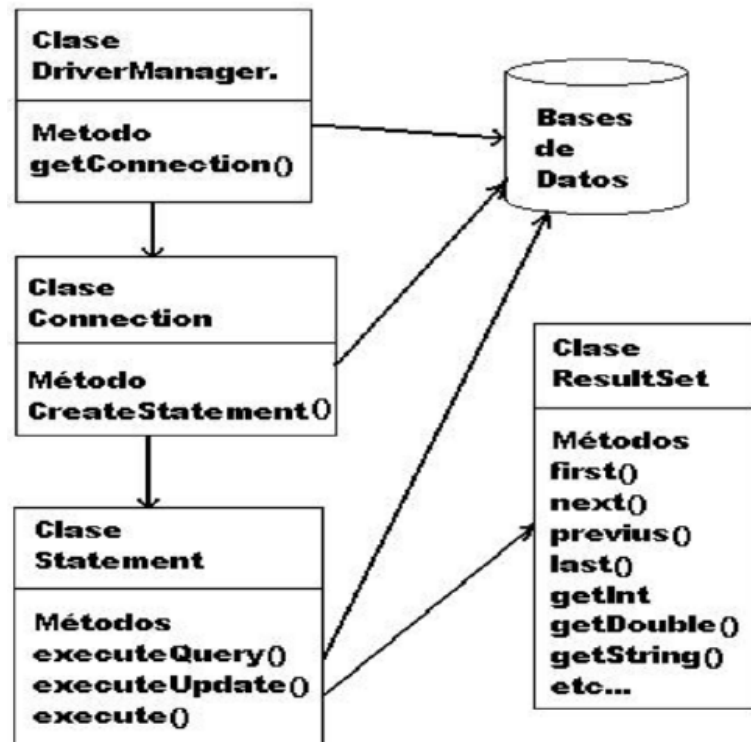
CARREGAR EL DRIVER

Les classes i mètodes utilitzats per a connectar-se a una BBDD funcionen amb tots els drivers de Java (JDBC, etc.). Java només els defineix com a interfícies (interface) i cada llibreria driver els implementa (classes i codi). Per això, s'ha d'utilitzar **Class.forName(...)** per a indicar-li a Java quin driver utilitzarem.

Daquesta forma, si necessitem utilitzar un altre sistema de BBDD sols haurem de canviar la línia de codi on es carrega el driver.

Les quatre classes fonamentals que tota aplicació Java necessita per a connectar-se a una BBDD i executar sentències són:

DriverManager, **Connection**, **Statement** i **ResultSet**.



GENERALITAT
VALENCIANA

ceedcv
CENTRE ESPECÍFIC
D'EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

5. ACCES BASES DE DADES AMB CODI JAVA

CLASSE *DriverManager*



GENERALITAT
VALENCIANA



La classe ***java.sql.DriverManager*** és la capa gestora del driver JDBC corresponent i de **crear connexions amb una BBDD** mitjançant el mètode estàtic ***getConnection(...)*** que té dues variants:

```
DriverManager.getConnection(String url)
```

```
DriverManager.getConnection(String url, String user, String password)
```

```
Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/prova","root","");
```

Aquest mètode pot llançar dos tipus d'excepcions:

- **SQLException**: La connexió no ha pogut produir-se. Pot ser per multitud de motius com una URL mal formada, un error en la xarxa, host o port incorrecte, base de dades no existent, usuari i contrasenya no vàlids, etc.
- **SQLTimeoutException**: S'ha superat el LoginTimeout sense rebre resposta del servidor.

5. ACCES BASES DE DADES AMB CODI JAVA

CLASSE Connection



GENERALITAT
VALENCIANA



Un objecte **java.sql.Connection** representa una sessió de connexió amb una o diverses BBDD.

El mètode més rellevant és **createStatement()** que retorna un objecte Statement associat a aquesta connexió que permet executar sentències SQL. El mètode createStatement() pot llançar excepcions de tipus **SQLException**.

```
Statement stmt = conn.createStatement();
```

Quan finalitzem és aconsellable **tancar la connexió amb close()** per a alliberar recursos.

```
conn.close();
```

5. ACCES BASES DE DADES AMB CODI JAVA

CLASSE Statement



GENERALITAT
VALENCIANA



Un objecte **java.sql.Statement** permet **executar sentències SQL en la base de dades** a través de la connexió amb la qual es va crear el Statement.

Les sentències SQL més comunes són:

executeQuery(...), executeUpdate(...) i **execute(...)**.

Poden llançar excepcions de tipus **SQLException** i **SQLTimeoutException**.

Quan ja no ho necessitem és aconsellable **tancar el statement amb close()** per a alliberar recursos.

stmt.close();

5. ACCES BASES DE DADES AMB CODI JAVA

CLASSE Statement



GENERALITAT
VALENCIANA



- **ResultSet executeQuery(String sql):** Executa la sentència sql indicada (de tipus SELECT). Retorna un objecte ResultSet amb les dades proporcionades pel servidor.

```
ResultSet rs = stmt.executeQuery("SELECT * FROM vendedores");
```

- **int executeUpdate(String sql):** Executa la sentència sql indicada (de tipus DML com per exemple INSERT, UPDATE o DELETE). Retorna un nombre de registres que han sigut inserits, modificats o eliminats.

```
int nr = stmt.executeUpdate ("INSERT INTO vendedores VALUES  
                                (1, 'Pedro Gil', '2017-04-11', 15000);")
```


5. ACCES BASES DE DADES AMB CODI JAVA

CLASSE ResultSet



GENERALITAT
VALENCIANA



CENTRE ESPECÍFIC
D'EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

Un objecte **java.sql.ResultSet** conté un conjunt de resultats (dades) obtinguts després d'executar una sentència SQL (normalment SELECT) en forma de **taula** amb **registres (files)** i **campes (columnes)**.

ResultSet utilitza un cursor que apunta al 'registre actual' sobre el qual podem operar. Inicialment està situat abans de la primera fila i disposem de diversos mètodes per a desplaçar el cursor. El més comú és next():

- **boolean next()**: Mou el cursor al següent registre. Retorna true si llig una fila i false en cas contrari (arribem al final de la taula).

5. ACCES BASES DE DADES AMB CODI JAVA

CLASSE *ResultSet*



GENERALITAT
VALENCIANA



Alguns dels mètodes per a obtindre les dades del registre actual són:

- **String getString(String columnLabel):** Retorna una dada String de la columna indicada pel seu nom. Per exemple: `rs.getString("nom")`
- **String getString(int columnIndex):** Retorna una dada String de la columna indicada per la seua posició (La primera columna és la 1). Per exemple: `rs.getString(2)`

Per a obtindre valors de altre tenim:

- | | |
|-------------------------------------------------|--------------------------------------------|
| • int getInt(String columnLabel) | int getInt(int columnIndex) |
| • double getDouble(String columnLabel) | double getDouble(int columnIndex) |
| • boolean getBoolean(String columnLabel) | boolean getBoolean(int columnIndex) |
| • Date getDate(String columnLabel) | int getDate(int columnIndex) ... |

Nota: Es pot consultar tots els mètodes en la [documentació oficial de Java](#).

5. ACCES BASES DE DADES AMB CODI JAVA

CLASSE ResultSet



GENERALITAT
VALENCIANA



Tots aquests mètodes poden llançar una **SQLException**.

Exemple de com recórrer un ResultSet anomenat rs i mostrar-lo per pantalla:

```
while(rs.next()) {  
    int id = rs.getInt("id");  
    String nom = rs.getString("nom");  
    Date data = rs.getDate("data_ingrés");  
    float salari = rs.getFloat("salari");  
    System.out.println(id + " " + nom + " " + data + " " + salari);  
}
```

Nota: Fer l'Exemple01 corresponent a l'apartat 5.7 de la teoria.

6. NAVEGABILITAT I CONCURRÈNCIA



GENERALITAT
VALENCIANA



Quan invoquem a `createStatement()` **sense arguments**, obtindrem un ***ResultSet*** per defecte en el qual el cursor només pot moure's cap avant i les dades són de només lectura.

El mètode `createStatement()` està sobrecarregat, per la qual cosa, invocant-lo amb arguments ens permet altres funcionalitats.

`Statement createStatement(int resultSetType, int resultSetConcurrency):`

Veiem que signifiquen aquests dos paràmetres.

6. NAVEGABILITAT I CONCURRÈNCIA

ARGUMENT resultSetType



GENERALITAT
VALENCIANA



- **ResultSet.TYPE_FORWARD_ONLY:** ResultSet forward-only i no-actualitzable (per defecte).
 - Només permet moviment cap avant amb next().
 - Les seues dades NO s'actualitzen, són les que hi havia en el moment de llançar la consulta.
- **ResultSet.TYPE_SCROLL_INSENSITIVE:** ResultSet desplaçable i no actualitzable.
 - Permet llibertat de moviment amb mètodes com first(), previous(), last(), etc. a més de next() .
 - Les seues dades NO s'actualitzen.
- **ResultSet.TYPE_SCROLL_SENSITIVE:** ResultSet desplaçable i actualitzable.
 - Permet llibertat de moviments del cursor.
 - Les seues dades SÍ QUE s'actualitzen. És a dir, mentre el ResultSet estiga obert s'actualitzarà automàticament amb els canvis produïts en la base de dades. Això pot succeir fins i tot mentre s'està recorrent el ResultSet, la qual cosa pot ser convenient o contraproduent segons el cas.

6. NAVEGABILITAT I CONCURRENCIA

ARGUMENT resultSetConcurrency



GENERALITAT
VALENCIANA



CENTRE ESPECÍFIC
D'EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

- **ResultSet.CONCUR_READ_ONLY**: Només lectura. És el valor per defecte.
- **ResultSet.CONCUR_UPDATABLE**: Permet modificar les dades emmagatzemades en el ResultSet per a després aplicar els canvis sobre la base de dades.

Nota: El ResultSet per defecte que s'obté amb `createStatement()` sense arguments és el mateix que amb

`createStatement(ResultSet.TYPE_FORWARD_ONLY, ResultSet.CONCUR_READ_ONLY).`

7. CONSULTES

NAVEGACIÓ D'UN *ResultSet*



GENERALITAT
VALENCIANA



Un objecte *ResultSet* conté les files que satisfan les condicions d'una sentència SQL, i ofereix mètodes de navegació pels registres. Si és de tipus `ResultSet.TYPE_SCROLL_INSENSITIVE` o `ResultSet.TYPE_SCROLL_SENSITIVE` a més del mètode `next()` tindrem altres com aquests.

- **`void beforeFirst()`**: Mou el cursor abans de la primera fila.
- **`boolean first()`**: Mou el cursor a la primera fila.
- **`boolean next()`**: Mou el cursor a la següent fila.
- **`boolean previous()`**: Mou el cursor a la fila anterior.
- **`boolean last()`**: Mou el cursor a l'última fila.
- **`void afterLast()`**: Moure el cursor després de l'última fila.
- **`boolean absolute(int row)`**: Posiciona el cursor en el número de registre indicat (el primer registre és l'1. Si valor és negatiu comença a contar pel final (l'últim és el -1).
- **`boolean relative(int registres)`**: Desplaça el cursor un nombre relatiu de registres a partir del registre en que es troba.

Els mètodes booleans retornen “true” si s'ha pogut moure el cursor i “false” en altre cas. Aquests mètodes poden produir una excepció de tipus **`SQLException`**.

7. CONSULTES

NAVEGACIÓ D'UN *ResultSet*



GENERALITAT
VALENCIANA



També existeixen altres mètodes relacionats amb la posició del cursor.

- **int `getRow()`:** Retorna el número de registre actual. Zero si no hi ha registre actual.
- **boolean `isBeforeFirst()`:** Retorna 'true' si el cursor està abans del primer registre.
- **boolean `isFirst()`:** Retorna 'true' si el cursor està en el primer registre.
- **boolean `isLast()`:** Retorna 'true' si el cursor està en l'últim registre.
- **boolean `isAfterLast()`:** Retorna 'true' si el cursor està després de l'últim registre.

7. CONSULTES

OBTENINT DADES DEL ResultSet



GENERALITAT
VALENCIANA



Els mètodes *getXXX()* permeten recuperar els valors de les columnes (camps) de la fila (registre) actual del *ResultSet*. La designació de la columna es pot fer pel nom del camp o pel seu número (posició començant per l'1).

```
String valor = rs.getString(2);  
String valor = rs.getString("titol");
```

*Compte amb la compatibilitat dels tipus.
Es pot fer un getString() sobre un enter,
però no un getDate() sobre un String.*

Les columnes no són case sensitive, no distingeixen entre majúscules i minúscules.

La informació referent a les columnes d'un *ResultSet* es pot obtindre cridant al **mètode *getMetaData()*** que retornarà un objecte *ResultSetMetaData* que contindrà el número, tipus i propietats de les columnes del *ResultSet*.

Si coneguem el nom però no la posició podem utilitzar el mètode *findColumn(columna)* que ens torna l'enter de la posició que ocupa.

8. MODIFICACIÓ



GENERALITAT
VALENCIANA

ceedcv
CENTRE ESPECÍFIC
D'EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

Per a poder modificar les dades que conté un *ResultSet* haurà de ser modificable, es a dir, el mètode *createStatement()* haurà d'utilitzar la constant *ResultSet.CONCUR_UPDATABLE*.

```
Statement stmt = conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,  
ResultSet.CONCUR_UPDATABLE);
```

Per a modificar els valors d'un registre existent s'utilitzen una sèrie de mètodes *updateXXX()*: *updateString()*, *updateInt()*, *updateDouble()*, *updateDate()*, etc. que necessiten dos arguments:

- **La columna que desitgem actualitzar** (nom o número de columna).
- **El valor que volem emmagatzemar** (del tipus que siga).

```
rs.updateInt("edat", 28);
```

8. MODIFICACIÓ



GENERALITAT
VALENCIANA



Si es produeix algun error es llançarà una `SQLException`.

Posteriorment cal cridar a **`updateRow()`** perquè els canvis s'apliquen sobre la BD.

En resum, el procés per a realitzar la modificació d'una fila d'un *ResultSet* és el següent:

1. **Desplacem el cursor al registre** que volem modificar.
2. Cridem a tots els mètodes **`updateXXX(...)`** que necessitem.
3. Cridem a **`updateRow()`** perquè els canvis s'apliquen a la base de dades.

Cal cridar a **`updateRow()`** abans de desplaçar el cursor, sinó es perdran els canvis.

Si volem **cancel·lar** les modificacions d'un registre del **ResultSet** podem cridar a **`cancelRowUpdates()`**, i no utilitzar `updateRow()`.

8. MODIFICACIÓ

EXEMPLE



GENERALITAT
VALENCIANA



// Creem un Statement scrollable i modificable

```
Statement stmt = conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,  
                                         ResultSet.CONCUR_UPDATABLE);
```

// Executem un SELECT i obtenim la taula clients en un ResultSet

```
String sql = "SELECT * FROM clients";  
ResultSet rs = stmt.executeQuery(sql);
```

// Anem a l'últim registre, el modifiquem el camp direcció i actualitzem la base de dades
rs.last();

```
rs.updateString("direccio", "C/ Pepe Ciges, 3");  
rs.updateRow();
```

9. INSERCIÓ



GENERALITAT
VALENCIANA



Per a inserir nous registres necessitarem aquests dos mètodes:

- **void moveToInsertRow():** Desplaça el cursor al 'registre d'inserció', un registre especial utilitzat per a inserir.

Després cridem als mètodes **updateXXX()** per a establir els valors del registre d'inserció.

- **void insertRow():** Insertarà el 'registre d'inserció' en el ResultSet, passant a ser un registre normal més, i també l'insereix en la base de dades.

Finalment, el mètode **moveToCurrentRow()** ens pot tornar a la posició on estava el cursors abans de fer la inserció.

Els camps als que no s'els ha assignat valor amb **updateXXX()** tindran el valor NULL. Si la configuració del camp no admet nulsos produirà una **SQLException**.

9. INSERCIÓ

EXEMPLE



GENERALITAT
VALENCIANA

ceedcv
CENTRE ESPECÍFIC
D'EDUCACIÓ A DISTÀNCIA DE
LA COMUNITAT VALENCIANA

// Creem un Statement scrollable i modificable

```
Statement stmt = conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,  
                                         ResultSet.CONCUR_UPDATABLE);
```

// Executem un SELECT i obtenim la taula clients en un ResultSet

```
String sql = "SELECT * FROM clients";  
ResultSet rs = stmt.executeQuery(sql);
```

// Creem un nou registre i l'inserim

```
rs.moveToInsertRow();  
rs.updateString(2, "Killy Lopez");  
rs.updateString(3, "Wall Street 3674");  
rs.insertRow();
```

10. ESBORRAT

EXEMPLE



GENERALITAT
VALENCIANA



Per a eliminar un registre només cal desplaçar el cursor al registre desitjat i cridar al mètode:

- **void deleteRow():** Elimina el registre actual del ResultSet i també de la base de dades.

// Creem un Statement scrollable i modificable

```
Statement stmt = conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,  
                                         ResultSet.CONCUR_UPDATABLE);
```

// Executem un SELECT i obtenim la taula clients en un ResultSet

```
String sql = "SELECT * FROM clients";  
ResultSet rs = stmt.executeQuery(sql);
```

// Desplacem el cursor al tercer registre

```
rs.absolute(3)  
rs.deleteRow();
```

EXERCICIS PROPOSATS

