

JPA – Java Persistence API

JPA oferă facilități ORM (Object/Relational Mapping) pentru a gestiona date relaționate în aplicații Java, în principal pentru a memora datele pe un suport persistent (de obicei bază de date relațională) și pentru a efectua interogări asupra datelor persistante. Java Persistence cuprinde:

- *Java Persistence API.*
- *Limbajul de interogare.*
- *Java Persistence Criteria API.*
- *Metadate de mapare.*

Unitatea elementară care se persistă se numește *entitate*. De obicei o entitate este asociată cu un tabel din baza de date, o linie (înregistrare) din tabel fiind o instanță a entității. O entitate are câmpuri sau proprietăți persistente, care sunt asociate prin mapări cu elementele specifice suportului de stocare.

O clasă trebuie să îndeplinească anumite condiții pentru a fi entitate, ea va fi în plus marcată cu adnotarea `@Entity`.

Se pot impune anumite validări prin adnotări specifice. Astfel, un câmp obligatoriu se marchează cu `@NotNull` iar un format particular de introducere este precizat prin `@Pattern(regexp="...")`.

Entitățile posedă elemente unice de identificare numite *chei primare*, care pot fi simple (un singur câmp) sau compuse (mai multe câmpuri).

Relațiile dintre entități pot avea următoarele *multiplicități*:

- **One-to-one**, în care o instanță a unei entități este legată (relaționată) cu o singură instanță a altrei entități.
- **One-to-many**, în care o instanță a unei entități poate fi legată (relaționată) la mai multe instanțe ale celeilalte entități. Exemplul clasic este cel al unei comenzi (order) care poate avea mai multe linii.
- **Many-to-one**, este relația inversă față de one-to-many, când mai multe instanțe ale unei entități pot fi legate (relaționate) la o instanță a celeilalte entități.
- **Many-to-many**, când instanțele unei entități pot fi legate la mai multe instanțe ale celeilalte entități. De exemplu, un curs are mai mulți studenți și fiecare student urmează mai multe cursuri.

Relațiile pot fi:

- **Bidirectionale**, în care fiecare entitate are un câmp prin care se referă la cealaltă entitate (fiecare entitate "știe" de cealaltă).
- **Unidirectionale**, în care doar o entitate "știe" de cealaltă.

Aceste relații crează anumite dependențe, care impun anumite reguli la realizarea anumitor operații (ștergere).

Entitățile sunt gestionate de un manager de entități, o instanță a `javax.persistence.EntityManager`, care este asociat cu un context de persistență (o mulțime de entități care există într-un anumit depozit de informații).

Instanțele de entități pot fi în una din stările următoare:

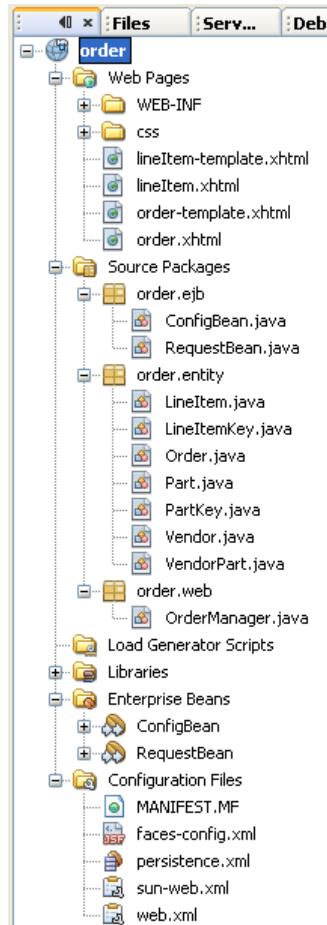
- *New*
- *Managed*
- *Detached*
- *Removed*

1. Aplicația "order"

Deschideți în NetBeans proiectul "**order**", care se găsește în tutorialul JEE6 (examples/persistence) sau în arhiva cu surse de pe site. Aceasta este o aplicație simplă de gestiune a unor produse și de realizare de comenzi pe aceste produse.

Analizați fișierele componente și modul lor de organizare.

Identificați entitățile aplicației (produs, furnizor, comandă, linie de comandă).



Identificați relațiile dintre entități și multiplicitatea acestora.

Rulați proiectul din NetBeans, identificând pagina de pornire (în fișierul de configurare).

Identificați și analizați cele 2 şabloane folosite.

Aplicați modificările necesare pentru a schimba aspectul implicit al paginii de plecare:

Order Java Persistence Example DAI

View All Orders

Order ID	Shipment Info	Status	Last Updated	Discount	Actions
1111	333 New Court, New City, CA 90000	N	Fri Nov 12 02:39:19 EET 2010	10%	Delete
4312	333 New Court, New City, CA 90000	N	Fri Nov 12 02:39:19 EET 2010	0%	Delete

Create New Order

Order ID:

Shipment Info:

Status:

Discount:

Find Vendors

Search for Vendors:

Vendor

Adăugați o comandă nouă, completând valori corecte în câmpurile comenzii:

Create New Order

Order ID:	1234
Shipment Info:	abcd
Status:	Complete
Discount:	0 %
<input type="button" value="Submit"/>	

Observați efectul imediat:

Order Java Persistence Example DAI

View All Orders

Order ID	Shipment Info	Status	Last Updated	Discount	Actions
1111	333 New Court, New City, CA 90000	N	Fri Nov 12 02:39:19 EET 2010	10%	Delete
1234	abcd	Y	Fri Nov 12 03:26:23 EET 2010	0%	Delete
4312	333 New Court, New City, CA 90000	N	Fri Nov 12 02:39:19 EET 2010	0%	Delete

Adăugați o comandă nouă, completând valori incorecte în câmpurile comenzii:

Create New Order

Order ID:	qqq
Shipment Info:	abcd
Status:	Complete
Discount:	0 %
<input type="button" value="Submit"/>	

Identificați de ce nu se adaugă înregistrarea dorită. Verificați unde se semnalează problemele.

Identificați cum se face "încărcarea" inițială a datelor.

Adăugați un buton de ștergere produs de pe comandă.

Adăugați un atribut nou pe comandă, numărul de produse conținute. Acesta va fi (re)calculat automat la adăugarea sau ștergerea unui produs.

2. Aplicația "address-book"

Deschideți în NetBeans proiectul "address-book", care se găsește în tutorialul JEE6 (examples/persistence) sau în arhiva cu surse de pe site. Aceasta este o aplicație simplă de gestiune a unei liste de contacte.

Analizați fișierele componente și modul lor de organizare.

Identificați entitatele aplicației (contact).

Modificați mesajele afișate la introducerea incorectă a datelor.

Create New Contact

First Name:	a	
Last Name:	a	
Birthday:	12/12/0012	
Home Phone:	000	Not a valid phone number.
Mobile Phone:	000	Not a valid phone number.
Email:	000	Not a valid email address.

[Save](#)

[Show All Contact Items](#)

[Index](#)

Adăugați un câmp nou pe entitate, care să memoreze ocuparea persoanei și propagați acest câmp în toate paginile necesare.

3. Aplicația "criteriaQuery"*

Deschideți în NetBeans proiectul "criteriaQuery", care se găsește în exemplele JPA (<https://glassfish-samples.dev.java.net/servlets/ProjectDocumentList?folderID=5214&expandFolder=5214&folderID=0>) sau în arhiva cu surse de pe site. Descrierea acestei aplicații se găsește la adresa https://glassfish-samples.dev.java.net/source/browse/*checkout*/glassfish-samples/tags/JAVAEE6_SAMPLES-0_9-b16/ws/javaee6/jpa/criteriaQuery/docs/index.html.

Identificați entitățile acestei aplicații.

Rulați aplicația, mai întâi direct, apoi pas cu pas (depanare).

4. Resurse JPA

Resurse utile:

- <http://download.oracle.com/javaee/6/tutorial/doc/>
- <http://download.oracle.com/javaee/6/tutorial/doc/bnbpy.html>
- <http://download.oracle.com/javaee/6/javaserverfaces/2.0/docs/pdldocs/facelets/h/tld-summary.html>
- <http://www.coreservlets.com/JSF-Tutorial/jsf1/>
- <http://www.coreservlets.com/JSF-Tutorial/jsf2/>
- <https://javaeetutorial.dev.java.net/servlets/ProjectDocumentList>
- <http://download.oracle.com/javaee/6/javaserverfaces/2.0/docs/pdldocs/facelets/>
- <http://netbeans.org/kb/docs/web/jsf20-intro.html>

JDBC – Java Database Connectivity

JDBC reprezintă un API pentru realizarea operațiilor cu baze de date (relaționale) din aplicații Java. Clasele JDBC se găsesc în pachetul `java.sql`, existând și o extensie în `javax.sql`.

În general prelucrarea unei operații SQL cu JDBC urmează o succesiune de pași:

- Stabilirea unei conexiuni la sursa de date (bază de date, sistem de fișiere etc).
- Crearea unei fraze SQL (un obiect `Statement` se crează cu ajutorul unui obiect `Connection`).
- Executarea interogării (regăsire date, apel procedură stocată, modificare date).
- Prelucrarea obiectului `ResultSet` rezultat.
- Închiderea conexiunii, pentru eliberarea resurselor folosite.

Există 3 feluri de fraze SQL:

- `Statement`: fraze SQL simple, fără parametri.
- `PreparedStatement`: extind `Statement` și pot conține parametri de intrare.
- `CallableStatement`: extind `PreparedStatement` și se folosesc pentru a executa proceduri stocate ce pot avea atât parametri de intrare cât și parametri de ieșire.

Execuția unei interogări se face prin apelul unor metode ale obiectului `Statement`:

- `execute`: poate returna unul sau mai multe obiecte `ResultSet`, care pot fi obținute prin apelarea repetată a metodei `Statement.getResultSet`.
- `executeQuery`: returnează un obiect `ResultSet`.
- `executeUpdate`: returnează numărul de linii afectate de fraza SQL de tip `INSERT`, `DELETE`, `UPDATE`.

Operațiile uzuale sunt descrise de următoarea secvență de cod:

```
try {
    stmt = con.createStatement();
    ResultSet rs = stmt.executeQuery(query);
    while (rs.next()) {
        String coffeeName = rs.getString("COF_NAME");
        int supplierID = rs.getInt("SUP_ID");
        float price = rs.getFloat("PRICE");
        int sales = rs.getInt("SALES");
        int total = rs.getInt("TOTAL");
        System.out.println(coffeeName + "\t" + supplierID + "\t" + price + "\t" + sales +
                           "\t" + total);
    }
}
```

Specificarea sursei de date se face prin indicarea unui șir de caractere de tip URL:

- Pentru MySQL:

`jdbc:mysql://[host][,failoverhost...][:port]/[database][?propertyName1][=PropertyValue1][&propertyName2][=PropertyValue2]...`

- unde host:port implicit 127.0.0.1:3306.

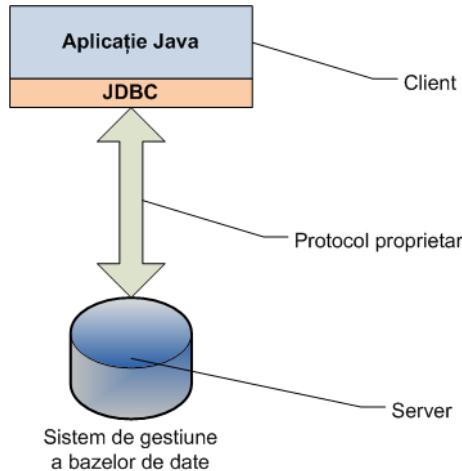
- Pentru Java DB:

`jdbc:derby:[subsubprotocol:] [databaseName] [;attribute=value]*`

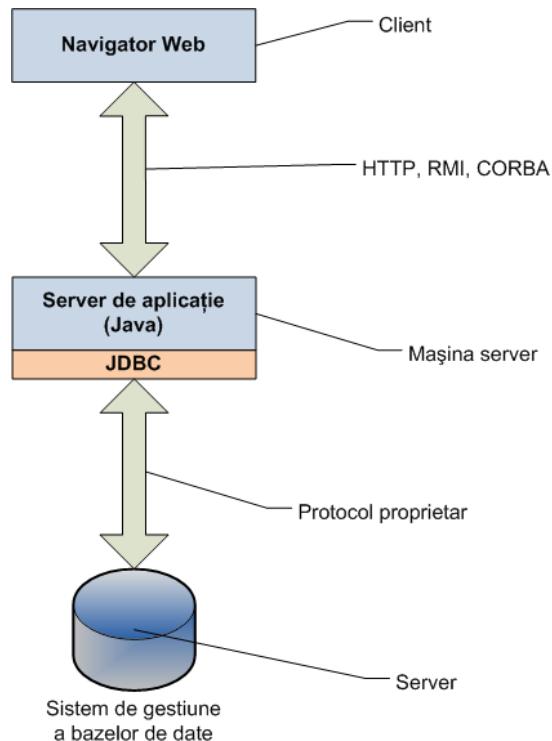
- `subsubprotocol` (unde e baza de date: intr-un director, în memory, in class path, în JAR).
- `databaseName` is the name of the database to connect to.
- `attribute=value` listă optională de atribute, pentru indicarea unor task-uri:

- Crearea bazei de date din URL.
- Criptarea bazei de date din URL.
- Director pentru log.
- Utilizator și parolă de conectare.

Arhitectura JDBC în modelul cu 2 straturi este următoarea:



Arhitectura JDBC în modelul cu 3 straturi este următoarea:



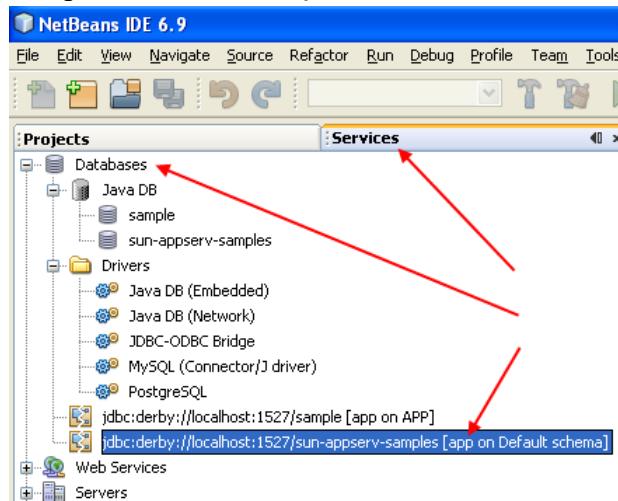
1. Derby (Java DB)

Derby este un motor de bază de date relațională pur Java, care folosește standardul SQL și JDBC ca API, oferind o instanță încorporată în aplicație sau de server de rețea și o serie de utilitare în linie de comandă:

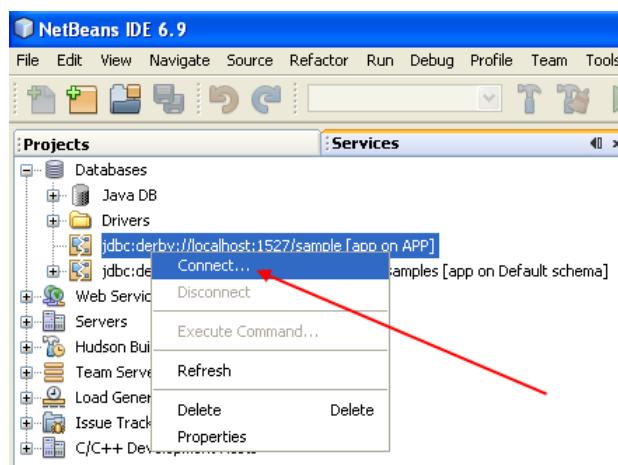
- ij (SQL scripting).
- dblook (schema dump).
- sysinfo (system info).

JDK 1.6 instalează un astfel de server în locația "C:\Program Files\Sun\JavaDB".

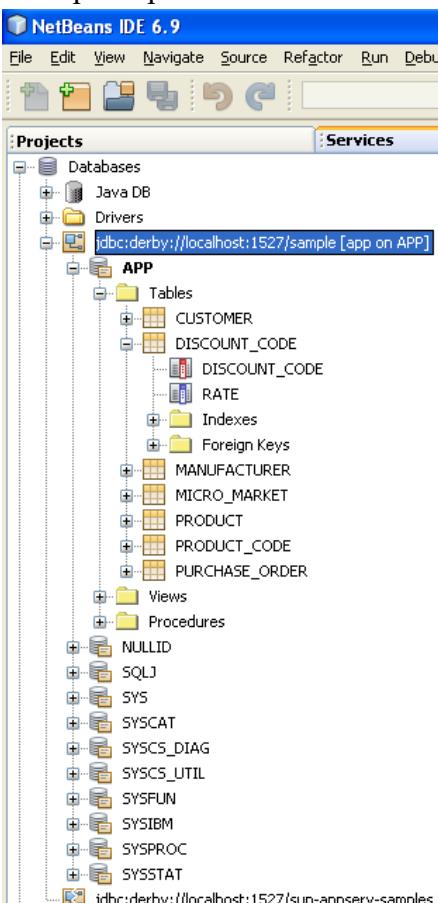
Mediul de dezvoltare NetBeans permite accesarea și controlarea acestui server:



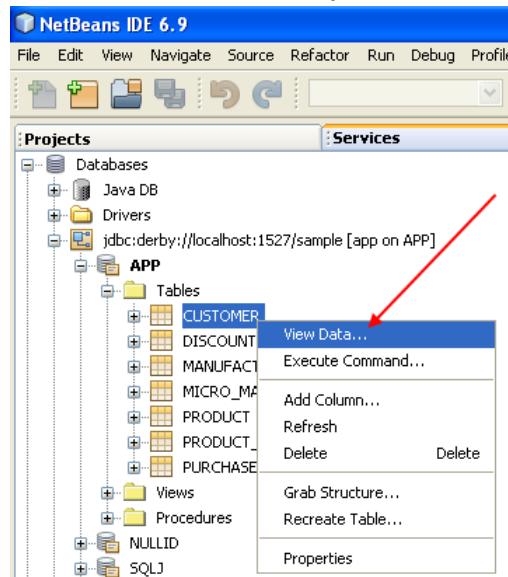
Startați manual acest server:



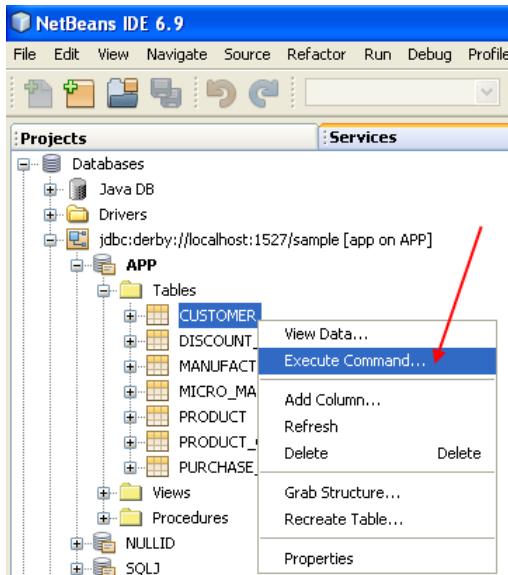
Analizați schemele care apar și tabelele principale ale acestora:



Observați modalitatea de vizualizare a datelor din tabele și analizați conținutul acestora:



Execuția manuală a unei serie de interogări asupra datelor existente:



2. IJ*

Realizați setările de mediu necesare (variabilele DERBY_HOME, PATH) și lansați utilitarul ij:

```
C:\WINDOWS\system32\cmd.exe - ij
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\GG>ij
ij version 10.4
ij> -
```

Comanda help oferă o serie de informații suplimentare:

```
C:\WINDOWS\system32\cmd.exe - ij

C:\Documents and Settings\GG>ij
ij version 10.4
ij> help;

Supported commands include:

PROTOCOL 'JDBC protocol' [ AS ident ];
-- sets a default or named protocol
DRIVER 'class for driver'; -- loads the named class
CONNECT 'url for database' [ PROTOCOL namedProtocol ] [ AS connectionName ];
-- connects to database URL
-- and may assign identifier
SET CONNECTION connectionName; -- switches to the specified connection
SHOW CONNECTIONS; -- lists all connections
AUTOCOMMIT [ ON | OFF ]; -- sets autocommit mode for the connection
DISCONNECT [ CURRENT | connectionName | ALL ];
-- drop current, named, or all connections;
-- the default is CURRENT

SHOW SCHEMAS; -- lists all schemas in the current database
SHOW [ TABLES | VIEWS | PROCEDURES | SYNONYMS ] < IN schema >;
-- lists tables, views, procedures or synonyms
SHOW INDEXES < IN schema | FROM table >;
-- lists indexes in a schema, or for a table
DESCRIBE name; -- lists columns in the named table

COMMIT; -- commits the current transaction
ROLLBACK; -- rolls back the current transaction

PREPARE name AS 'SQL-J text'; -- prepares the SQL-J text
EXECUTE < name | 'SQL-J text' > [ USING < name | 'SQL-J text' > ] ;
-- executes the statement with parameter
-- values from the USING result set row
REMOVE name; -- removes the named previously prepared statement

RUN 'filename'; -- run commands from the named file

ELAPSEDTIME [ ON | OFF ]; -- sets elapsed time mode for ij
MAXIMUMDISPLAYWIDTH integerValue; -- sets the maximum display width for
-- each column to integerValue

ASYNC name 'SQL-J text'; -- run the command in another thread
WAIT FOR name; -- wait for result of ASYNC'd command

GET [SCROLL INSENSITIVE] CURSOR name AS 'SQL-J query';
-- gets a cursor <JDBC result set> on the query
-- SCROLL cursors are only available
-- in JDBC 2.0 and higher.
-- <Cursor scroll type is ignored in JDBC 1.X.>
NEXT name; -- gets the next row from the named cursor
FIRST name; -- gets the first row from the named scroll cursor
LAST name; -- gets the last row from the named scroll cursor
PREVIOUS name; -- gets the previous row from the named scroll cursor
ABSOLUTE integer name; -- positions the named scroll cursor at the absolute row number
-- (A negative number denotes position from the last row.)
RELATIVE integer name; -- positions the named scroll cursor relative to the current row
-- (integer is number of rows)
AFTER LAST name; -- positions the named scroll cursor after the last row
BEFORE FIRST name; -- positions the named scroll cursor before the first row
GETCURRENTROWNUMBER name; -- returns the row number for the current position of the named scroll cursor
-- (0 is returned when the cursor is not positioned on a row.)
CLOSE name; -- closes the named cursor
LOCALIZEDDISPLAY [ ON | OFF ];
-- controls locale sensitive data representation

EXIT; -- exits ij
HELP; -- shows this message

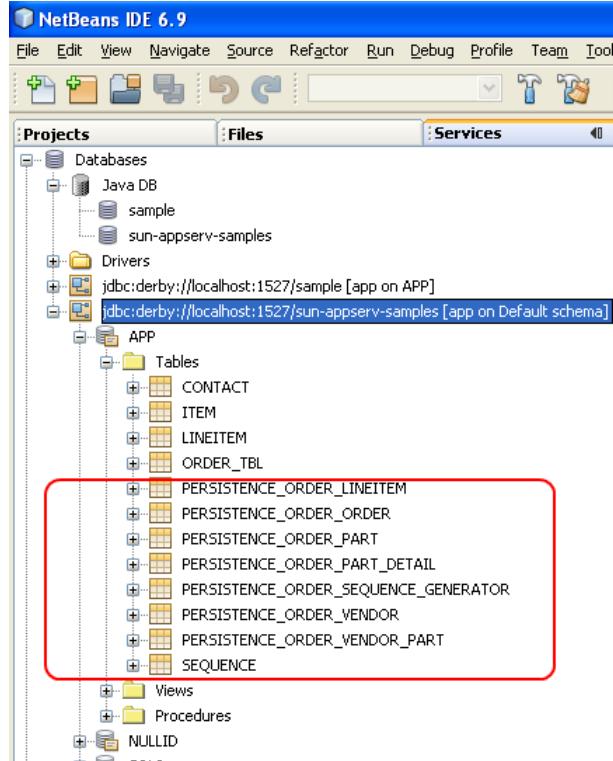
Any unrecognized commands are treated as potential SQL-J commands and executed directly.

ij> -
```

3. Aplicația "order"

Deschideți în NetBeans aplicația "order" folosită în secțiunea anterioară.

Identificați tabelele asociate entităților aplicației.



Lansați aplicația și adăugați comenzi și produse.

Order Java Persistence Example - DAI Lab

View All Orders

Order ID	Shipment Info	Status	Last Updated	Items	Value	Discount	Sum	Actions
1111	333 New Court, New City, CA 90000	N	Fri Nov 19 05:18:11 EET 2010	3	738.53	10%	664.677	Delete
1234	Test lab DAI	N	Fri Nov 19 05:21:19 EET 2010	2	117.990000000000001	20%	94.392000000000001	Delete
4312	333 New Court, New City, CA 90000	N	Fri Nov 19 05:18:11 EET 2010	3	2011.44	0%	2011.44	Delete

Create New Order

Order fields

Order ID:

Shipment Info:

Status: Pending

Discount: 0 %

Find Vendors

Search for vendors:

Line Items for Order 1234

Item ID	Part Number	Vendor Part Number	Revision	Quantity	Remove From Order
1	9876-4321-02	2	2	4	<button style="border: none; padding: 0;">Remove 1</button>
2	5456-6789-03	3	3	1	<button style="border: none; padding: 0;">Remove 1</button>

Available items				
Part Number	Vendor Part Number	Revision	Add To Order	
1234-5678-01	1	1	<button style="border: none; padding: 0;">Add</button>	
5456-6789-03	3	3	<button style="border: none; padding: 0;">Add</button>	
9876-4321-02	2	2	<button style="border: none; padding: 0;">Add</button>	
ABCD-XYZW-FF	4	5	<button style="border: none; padding: 0;">Add</button>	
SDFG-ERTY-BN	5	7	<button style="border: none; padding: 0;">Add</button>	

[Back to Orders](#)

Comparați cele afișate de aplicație cu informațiile existente în baza de date.

The screenshot shows the NetBeans IDE interface. On the left, the Projects panel displays a database connection named 'APP' which contains tables like CONTACT, ITEM, LINEITEM, ORDER_TBL, and several persistence-related tables such as PERSISTENCE_ORDER_LINEITEM, PERSISTENCE_ORDER_ORDER, PERSISTENCE_ORDER_PART, etc. In the center, there's a 'SQL Command 5' window with the following SQL query:

```
1 select * from APP.PERSISTENCE_ORDER_ORDER
```

Below the command window, a results table is shown for the query:

#	ORDERID	SHIPMENTINFO	STATUS	LASTUPDATE	ITEMCOUNT	DISCOUNT
1	4312 333 New Court, New City, CA 90000	N	2010-11-19 05:25:28.89		3	0
2	1111 333 New Court, New City, CA 90000	N	2010-11-19 05:25:28.89		3	10
3	1234 Test lab DAI	N	2010-11-19 05:25:46.531		3	20

4. Resurse JDBC

Resurse utile:

- <http://download.oracle.com/javase/tutorial/jdbc/index.html>
- <http://download.oracle.com/javase/tutorial/jdbc/overview/index.html>
- <http://download.oracle.com/javase/tutorial/jdbc/basics/index.html>
- <http://www.oracle.com/technetwork/java/overview-141217.html>
- <http://www.jdbc-tutorial.com/>
- <http://www.herongyang.com/JDBC/>
- <http://onjava.com/pub/a/onjava/2006/08/02/jjdbc-4-enhancements-in-java-se-6.html>
- <file:///C:/Program%20Files/Sun/JavaDB/demo/programs/readme.html>
- <http://download.oracle.com/javaee/6/tutorial/doc/>