**איך לעבוד עם JDBC**

**Java Database Connectivity**

**שלב א – הכנת הסביבה**

1. יש לבחור את מוצר ה DBMS (Database Management System) שאיתו נעבוד. קיימים בשוק מוצרים רבים מאוד. דוגמאות מוכרות:
   1. Apache Derby של חברת IBM, מוכר גם בשם Java DB. [קישור לאתר](https://db.apache.org/derby/)
   2. MySQL של חברת Oracle. [קישור לאתר](https://www.mysql.com/)
   3. Microsoft SQL Server [קישור לאתר](https://www.microsoft.com/en-us/sql-server)
2. בדוגמה הנוכחית נעבוד עם Apache Derby שאותו ניתן להוריד מהאתר המופיע בקישור לעיל. את קובץ ה ZIP שהורדנו יש לחלץ לתיקייה שיהיה לנו נוח לעבוד ממנה. למשל C:\java\servers
3. נפעיל את המערכת באמצעות לחיצה כפולה על הקובץ startNetworkServer.bat שנמצא בתיקייה bin. יופיע חלון DOS שחור שמציין started and ready to accept connections on port 1527
4. כעת יש לעבור ל Eclipse וליצור Java Project
5. יש לצרף לפרויקט את קובץ ה jar שנקרא derbyclient.jar ונמצא בתיקייה lib. עושים זאת באמצעות לחצן ימני על הפרויקט, בחירה ב properties, בחירה בחלונית libraries, ולחיצה על הכפתור Add External Jars.
6. סיימנו את שלב ההכנות – יש לנו מערכת database שפועלת וממתינה לבקשות. כמו כן יש לנו פרויקט Java שבו נוכל לכתוב תכניות בשפת Java שעובדות עם ה database ושולחות לו בקשות כגון יצירת database (קבוצת טבלאות שמכונסת תחת שם אחד), יצירת טבלאות וכן עבודה עם נתונים – הוספה, קריאה, עדכון ומחיקה המוכרות כ CRUD actions (Create, Read, Update, Delete).

**שלב ב – עבודה מתוך Java**

ישנם מספר טיפוסים ב Java שאותם נצרך להכיר על מנת לעבוד עם JDBC. טיפוסים אלה נמצאים ב package שעוסק ב SQL ושמו: java.sql

* java.sql.**Driver** – ממשק שמייצג דרייבר להפעלת database. המימוש לטיפוס זה מגיע מיצרן ה DBMS.
* java.sql.**DriverManager** – מחלקה שמספקת שירות של ניהול כל הדרייברים הפעילים בתכנית. באמצעותה ניתן לקבל Connection.
* java.sql.**Connection** – ממשק שמייצג קישוריות ל database צפציפי ב DBMS. המימוש לטיפוס זה מגיע מיצרן ה DBMS. באמצעות טיפוס זה ניתן לקבל Statement.
* java.sql.**Statement** – ממשק שמייצג אובייקט שיכול לקבל פקודות בשפת SQL ולהריץ אותן. המימוש לטיפוס זה מגיע מיצרן ה DBMS.
* java.lang.**ResultSet** – ממשק שמייצג טבלת תוצאות שהתקבלה כתוצאה מפעולת select של SQL.

**בהמשך דוגמאות קוד לאופן העבודה עם הטיפוסים לעיל**

**שלבים בסיסיים בעבודה**

try {

String driverName = "org.apache.derby.jdbc.ClientDriver";

**Class.forName(driverName); // טעינת הדרייבר**

} catch (ClassNotFoundException e) {

e.printStackTrace();

}

String url = "jdbc:derby://localhost:1527/db1";

try (

**Connection con = DriverManager.getConnection(url); // Connection יצירת**

**Statement stmt = con.createStatement(); // Statement יצירת**

) {

...

} catch (SQLException e) {

e.printStackTrace();

}

**הטיפוס ResultSet**

**ResultSet rs = stmt.executeQuery(sql); // ResultSet יצירת**

**דוגמה לעבודה עם ResultSet**

String url = "jdbc:derby://localhost:1527/db1";

try ( // ARM resources:

Connection con = DriverManager.getConnection(url);

Statement stmt = con.createStatement();

**ResultSet rs = stmt.executeQuery("select \* from persons");**

) {

// go over the result set

while (**rs.next()**) {

int id = **rs.getInt("id");**

String name = **rs.getString("name");**

int age = **rs.getInt("age");**

System.out.println(id + ", " + name + ", " + age);

}

} catch (SQLException e) {

e.printStackTrace();

}