**מעבדה מספר 4**

**JDBC ו-MySQL**

במעבדה זו נלמד כיצד לבצע חיבור למסד הנתונים MySQL באמצעות אפליקציה הכתובה ב-Java תוך שימוש ב- JDBC ( Java Database Connectivity).

קבצי המעבדה כוללים קובץ זה עם הסבר ומשימה לביצוע ובנוסף:

קבצי jdbc\_project עם תכנית בסיסית ליצירת connection עם database, קובץ SQL בשם bakupFlights, קובץ בשם mysql\_commands עם פקודות MySQL ואפליקציית JAVA בשם mysql-connector-java-3.1.11-bin .

1. **רקע**

**JDBC**

באמצעות JDBC ניתן לכתוב קוד ב-Java המכיל בתוכו קוד המסוגל לפנות למסד נתונים ולבצע פעולות שונות ומגוונות ב-SQL. הקוד נכתב כולו בשפת Java ובניגוד לשפות Embedded SQL אינו מצריך פעולה נוספת של תרגום מקוד המקור לשפת התכנות (לדוגמא C Embedded SQL). Java מספקת API (Application Programming Interface) שלם לצורך עבודה עם מסדי נתונים. לצורך שימוש ב-API יש ליבא את חבילת java.sql.

באמצעות שימוש ב-JDBC ניתן לבצע מספר התחברויות בו זמנית למספר מסדי נתונים שונים או מספר התחברויות לאותו מסד דרך אותה אפליקציה.

כל ההתקשרות בין האפליקציה למסד נתונים ספציפי מתבצעת דרך תוכנה מתווכת אשר נקראת DBMS Driver (או בקיצור (Driver, כאשר ה-Driver הוא האחראי להעברת פקודות מ-JDBC אל המסד ומהמסד בחזרה ל-JDBC. בדרך כלל טוענים Drivers באופן דינאמי על ידי רישום בזמן ריצה אצל DriverManager שאחראי על ניהול כל ה-Drivers הרצים במחשב המארח.

לאחר שרושמים את ה-Driver שאיתו משתמשים, ניתן ליצור חיבור למסד הנתונים וליצור אובייקט התקשרות למסד ((Connection. מעתה כל הפקודות למסד הנתונים מתבצעות דרך אובייקט ההתחברות עד לסגירתו.

Java משתמשת בתבנית תכן של Object Pooling בכל הקשור לניהול אובייקטי ההתחברות מאחר ויצירת התחברות כרוכה בהשקעת משאבי מחשוב רבים (למשל: פרוטוקול תקשורת או משאבי זכרון).

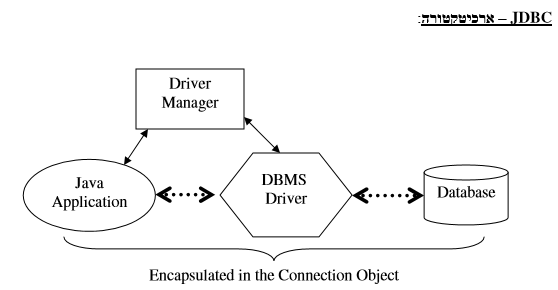
**הסבר על פעולת ה- JDBC**

כדי להשתמש ב-JDBC יש ליצור **connection** ל-database, על ידי שימוש באוביקט מטיפוס .Connectionאוביקטים מטיפוס Connection אופיינים לסוג ה-database, לדוגמא MySQL Connection או Oracle Connection.

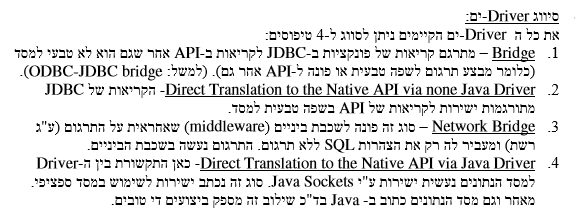
באמצעות אובייקט Connection ניתן ליצור **Statement**. Statement הוא אוביקט פקודה שניתן לשימוש חוזר ובאמצעותו ניתן לבצע פקודות SQL ולקבל תשובות מה-database. ניתן לציין עבור attributes (כמו למשל כמה תוצאות הוא יכול להכיל). בנוסף, Connectionיכול ליצור סוג אחר של Statement שנקרא PreparedStatement עם יכולת ביצוע טובה יותר ועם אבטחת מידע.

השאילתות SQL "SELECT" מחזירות תוצאות באובייקט מטיפוס **ResultSet**, המאפשר למתכנת לעבור של התוצאות בצורת איטרטיבית.

.



**driver** ( בעברית **מנהל התקן** (Driver) הוא [תוכנית מחשב](https://he.wikipedia.org/wiki/%D7%AA%D7%95%D7%9B%D7%A0%D7%99%D7%AA_%D7%9E%D7%97%D7%A9%D7%91) המאפשרת לתוכנית מחשב אחרת, לתקשר עם [חומרה](https://he.wikipedia.org/wiki/%D7%97%D7%95%D7%9E%D7%A8%D7%94) כלשהי או עם תוכנה אחרת הפועלת בפורמט שונה באמצעות מימוש הממשק שלה ומתן API לעבודה מולו. להלן סיווג ה-drivers הקיימים לעבודה עם JDBC.



1. **המעבדה מבוססת על מסד הנתונים הבא:**

**"מסד חברת התעופה"**

זהו מסד אקסס שיושב אצלכם מקומית במחשב.

בתרגול נשתמש בסכמה הבאה:

Aircraft(aid,aname, crusingrange)

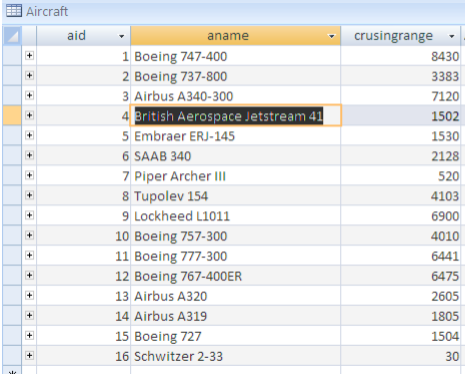
Certifies(edi,aid)

Employees(eid,ename,salary)

Flights(flno,from.to,distance,depart,arrives,price)

לסכימה זו תוסיפו טבלה שמרכזת את מחירי המוצרים הנמכרים בדיוטיפרי.

ולהלן מופע של טבלה במסד:



1. **רישום ה-Driver:**



אנחנו נשתמש ב-Driver שמגעי עם API של MySQL ולכן נרשום:



ניתן גם לרשום Driver ישירות דרך ה-DriverManager:



נתן גם להתחבר למבני נתונים אחרים, למשל Access של Microsoft. רישום דרייבר של אקסס מתבצע דרך המגרש ODBC דבר שמחייב בנוסף לרושמו במערכת ההפעלה.



דוגמא: בצעו התחברות למסד MySQL:



1. **יצירת JDBC Statement**

אובייקט Statement הוא זה ששולח את הצהרת SQL של משתמש אל המסד. ע"מ לבצע הצהרת SQL כל מה שצריך הוא ליצור אובייקט חדש ולהשתמש בפונ' Statement.executeXXX() המתאימה )כלומר יש להחליף את XXX בפעולה הנדרשת: Update או Query) על מנת ליצור אובייקט Statement יש לפנות אל אובייקט ההתקשרות ולקבל ממנו התייחסות לאובייקט(reference).

דוגמא: נניח שטבלת GIFTS לא קיימת במסד ונרצה ליצור אותה (כמובן בהנחה נוספת שיש לנו הרשאה במסד ליצירת טבלאות):



פעולות על מבני נתונים מתחלקות לשני סוגים:

DDL – פעולות יצירה וביטול של טבלאות. ( Data Definition Language)

DML – פעולות לעדכון נתונים בתוך הטבלאות (מחיקה, הוספה, עדכון) (Data Manipulation Language)

שני הסוגים של פעולות מבוצעות על ידי אותה מתודה שבקוד הנ"ל.

דוגמא להכנסת נתונים לטבלה:



1. **שליפת נתונים מתוך טבלה**

בשימוש באובייקט מטיפוס Statement ניתן גם להריץ שאילתות למסד ולקבל את תוצאותיהן.

היות ותוצאת שאילתא ב-SQL היא אוסף של רשומות, התוצאה מוחזרת כאובייקט מסוג ResultSet. אובייקט זה מאפשר בצורה נוחה לגשת לכל רשומה בפלט השאילתא, כמו כן ע"י שימוש ב-Curser ניתן גם לגשת באופן יחסי, אבסולוטי לכל שדה וכו'.

דוגמא: נניח נרצה להריץ שאילתא המוצאת את מוצרי הקפה שמחירם קטן שווה ל-8.99 ₪:

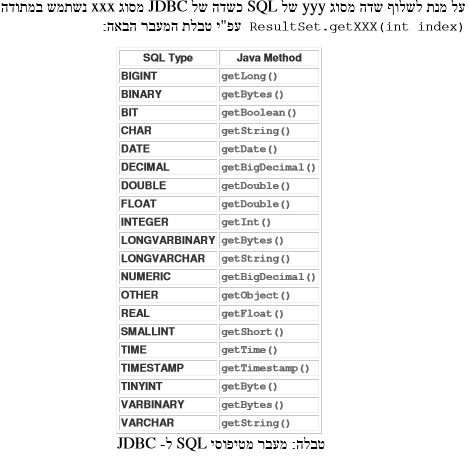


1. **שליפת נתונים מתוך ResultSet**

כפי שנאמר האובייקט ResultSet מכיל את הרשומות של פלט שאילתת SQL שבוצע דרך JDBC. לאובייקט זה קיימות מתודות שימושיות למעבר על הרשומות וכמו כן לגישה לכל אחד מהשדות ברשומות. שימו לב: אובייקט זה מחזיק מצביע לרשומת המידע הנוכחית. כאשר הוא נוצר לראשונה הוא המצביע מאותחל לערך ראשוני, לפני הרשומה הראשונה.

דוגמא: נוציא ל-Standart Output את תוצאת השאילתא מדוגמה קודמת:



1. **מיפוי ביו טיפוסי נתונים של SQL לאלה של JDBC**

על מנת לשוף שדה מסוג yyy של SQL כשדה של JDBC מסוג xxx נשתמש במתודה ResultSet.getxxx(int index) עפ"י טבלת המעבר הבאה:

1. **שימוש ב-** **Prepared Statements**

מסיבות אבטחה **חייבים להשתמש ב-PreparedStatement ע"מ לשלוח הצהרות SQL** למסד הנתונים. הנה סרטון קצר שמסביר את פריצה האבטחה: <https://youtu.be/ciNHn38EyRc>

לא להשתמש במחרוזות אשר נבנות "on the go".

דוגמא:

ועתה נניח שיש לעדכן למתנה Diamond את סך המכירות להיום ל-30(יח' מכירה)

ואם שוב פעם יש להזין נתוני מכירות פשוט נזין ערכים חדשים ונריץ העדכון.

1. **עדכון נתונים דרך ResultsSets**

ניתן ליצור ResultSet של שאילתת SQL שלאחר מכן ניתן להשתמש באתו אובייקט על מנת לבצע פעולות עדכון על תוצאות השאילתא.

דוגמא:



1. **הכנסה ומחיקת רשומות חדשות באופן מתוכנת**

באופן דומה ניתן הכניס רשומות חדשות לטבלה או למחוק רשומות מטבלה.

****

1. **תפיסת חריגות וטיפול בהתראות ממסד הנתונים**

על פי רוב פעולות המתבצעות על אובייקט בחבילת java.sql יכולות לזרוק חריגות SQLException שיש להצהיר עבורן בלוק של try…catch על מנת לתפוס אותן.

למשל בעת רישום Driver בצורה הבאה יתכן ותיזרק חריגה:

התראה (SQL Warning) היא חריגה שאינה נזרקת, אלא יש לברר אם ארעה:



12. תרגיל מעבדה 4

1. תעשה clone לפרויקט מ-GitHub Clasroom כפי שנלמד במעבדה 4:

<https://classroom.github.com/g/ZloyfvaO>

1. תצרו מסד נתונים חינמי ב-<https://remotemysql.com>
2. כנסו ל-<https://remotemysql.com/phpmyadmin/>
3. שחזרו database בשם flights מהקובץ backupFlights.sql מכיל את הטבלה *Flights(num, origin, destination, distance, prices)* עם רשימה של מחירי הטיסות. את הקובץ אפשר למצוא בתקית הפרויקט הורדתם.

בשביל לשחזר את הטבלה תלחצו על מקש SQL בחלק עליו של דף האינטרנט והדביקו בחלון שמתחת את תוכן הקובץ backupFlights.sql, לאחר מכן הלקיוק על GO שמתחת. שרת MySQL כרגע הריץ את הפקודה שתנתנו לו ומילא טבלה בנתונים.

1. כל המשימות הבאות צריכות להתבצע באמצעות אפליקציית Java (ולא ישירות באמצעות MySQL):
   1. הכניסו ל-database מחיר חדש לטיסה מספר 387.
   2. שלפו את הערך החדש והדפיסו את מחיר הטיסה המעודכן.
   3. עדכנו את מחירי הטיסות ב-database. העלו את מחיר כל טיסה במרחק הגדול מ-1000 ק"מ ב-50$. הראו את התוצאות. עדכנו את ה-database בעזרת ResultSet
   4. תחזרו על אותה פעולה בעזרת PreparedStatement.

הוראות הגשה:

1. ב-GitHub:

קוד הפרויקט מעודכן ב-master

1. במודל קובץ זיפ המכיל:
   1. את כל קבציה הפרויקט
   2. מסמך עם 4 צילומי מסך של התוצאות של 4 הסעיפים.
   3. במקום ברור במסמך שיהיה רשום בגדול שם של הקבוצה שתחתיו הגשתם את העבודה בגיטהב.

בהצלחה לכולם,

ועבודה נעימה