JDBC - zapytania SQL

java.sql.PreparedStatement

- ➔ **zapytanie** - prekompilowane zapytanie SQL
- ➔ **dynamiczne** - można wykonywać wiele razy zmieniając parametry “w locie”
- ➔ **ulepszone** - bezpieczeństwo, wydajność, wygoda
- ➔ **PreparedStatement.executeQuery()** - pobranie danych: SELECT
- ➔ **PreparedStatement.executeUpdate()** - aktualizowanie danych: INSERT...
- ➔ **użycie:**

```
int id = 11;
String name = “Adam”;
java.sql.Date date = new java.sql.Date(2018, 6, 10);

String query = "INSERT INTO table1(id, name, date) VALUES(?, ?, ?)";

try(PreparedStatement statement = connection.prepareStatement(query)){
      //parameterIndex zaczyna się od 1!
      statement.setInt(1, id);
      statement.setInt(2, name);
      statement.setDate(3, date);

      statement.executeUpdate();
}
```

- ➔ **bezpieczeństwo** - zapobiega atakom typu **SQLInjections**
- ➔ **SQLInjections** wstrzyknięcie do zapytania SQL złośliwego kodu, który ma na celu zniszczyć dane albo uzyskać dane do których użytkownik nie ma uprawnień

https://www.w3schools.com/sql/sql_injection.asp
https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet

Zadanie nr 3:

1. Przejdź do modułu: **javadb-starter**. Potestuj różne ataki SQLInjection poprzez klasę ***SqlInjectionSample*** z modułu javadb-starter. Zmień metody w klasie ***SqlInjectionSample*** wykorzystując PreparedStatement i sprawdź czy to zapobiega atakom **SQLInjections**.

2. W klasie ***CoursesManager*** stwórz metodę, która doda studenta do bazy danych (użyj PreparedStatement):
   `addStudent(String name, int courseId, String description, String seat)`

   Użyj tej metody w metodzie: *createStudentsTable()* zastępując stary kod dodający studentów.

3. Podobne metody stwórz dla kursów i listy obecności.

4. Stwórz oddzielne metody do każdego wymagania:
    a. Chcę zaktualizować dane studenta o podanym id, tylko: description i seat
    b. Chcę usunąć kurs o podanym id
    c. Chcę zaktualizować nazwę kursu o podanym id
    d. Chcę usunąć obecność studenta o podanym id i podanej dacie
    e. Chcę wyświetlić wszystkich studentów przypisanych do kursu o podanym id
    f. Chcę wyświetlić wszystkie kursy odbywające się w podanym mieście
    g. Chcę wyświetlić obecności wszystkich studentów na podany dzień w kursie o podanym id (użyj funkcji DATE(datetime) do wyciągnięcia daty z pola DATETIME)

5. Uzupełnij listę obecności w bazie danych trenera:
    a. Zmień konfigurację ***ConnectionFactory*** na tę zapisaną w pliku **remote-database.properties.**
    b. Używając wcześniej stworzonej metody wyświetl wszystkie kursy i znajdź id Twojego kursu
    c. Dodaj swoje dane do tabelki ze studentami używając id kursu znalezionego wcześniej
    d. Uzupełnij listę obecności podając swoje id studenta i id kursu oraz dzisiejszą datę

6. **(dla chętnych)** Dodaj metodę która wyświetli listę studentów siedzących w konkretnym miejscu. `listStudentsBySeat(String column, String row, String seat)`

    Metoda powinna obsługiwać takie argumenty:
    a. listStudentsBySeat(A, null, null) - wszyscy studenci z kolumny A
    b. listStudentsBySeat(null, 1, null) - wszyscy studenci z rzędu 1
    c. listStudentsBySeat(null, null, 2) - wszyscy studenci z krzesła 2
    d. listStudentsBySeat(A, 1, 5) - student siedzący w miejscu A.1.5

        Użyj szukania po wzorcu:
        https://dev.mysql.com/doc/refman/8.0/en/pattern-matching.html

7. **(dla chętnych)** Użyj obiektu klasy: ***Connection*** do wyciągnięcia informacji o bazie danych:
    a. Otwórz klasę ***DatabaseDiscovery*** i uruchom
    b. Dodaj kod który wyświetli dodatkowe informacje: nazwa i wersja bazy danych, nazwa użytkownika, nazwa i wersja sterownika do bazy danych - w obiekcie DatabaseMetaData znajdziesz gettery do każdej z tych informacji
    c. Dodaj kod który wyświetli dodatkowe informacje o strukturze tabel w bazie: typ SQL kolumny, rozmiar kolumny, czy kolumna może zawierać NULLe. Nazwy wszystkich parametrów są opisane w dokumentacji metody DatabaseMetaData.getColumns(). Wystarczy najechać myszką na nazwę metody i nacisnąć Ctrl+Q
    d. Spróbuj poeksperymentować z metodą DatabaseMetaData.getColumns(null, null, null, null), sprawdź dokumentację i spróbuj wyciągnąć dane dla konkretnej bazy danych albo dla konkretnej tabeli. W tym celu zamiast null trzeba podać odpowiednie argumenty w metodzie getColumns().
    e. Odczytaj dane bazy danych, której konfiguracja znajduje się w pliku **remote-database.properties.**