23. Java a databáze – DDL, DML

Práce s SQL databází v jazyce Java obvykle probíhá pomocí JDBC (Java Database Connectivity). JDBC poskytuje standardní rozhraní pro připojení k databázím, odesílání dotazů a zpracování výsledků.

Obsah obrázku text, snímek obrazovky, software

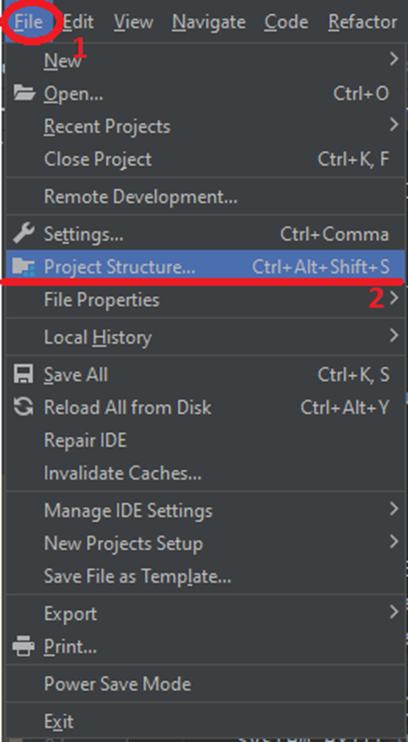
Popis byl vytvořen automatickyDDL – definování schémata

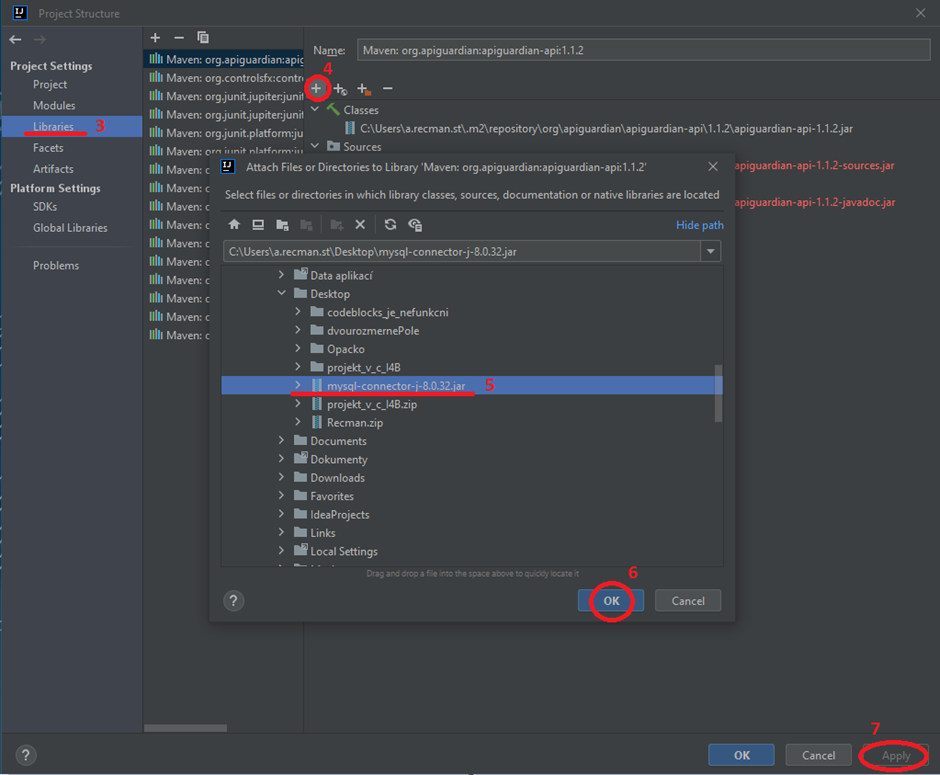
Obsah obrázku text, snímek obrazovky, Písmo, číslo

Popis byl vytvořen automatickyDML – manipulace s daty

**Vytvoření spojení**

**1. Přidání modulu mysql connector**

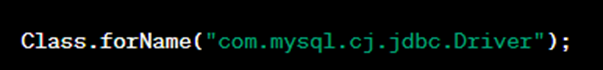




**2. Načtení ovladače (Driver Loading)**

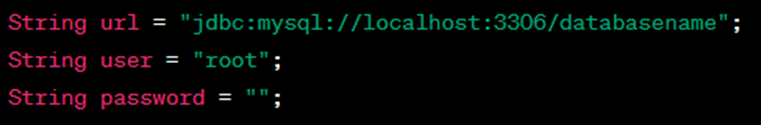
Nejprve musíte načíst ovladač pro konkrétní databázi, se kterou chcete pracovat. Každá databáze má svůj vlastní JDBC ovladač. (pozn. autora: od verze JDBC 4.0 není vždy nutné explicitně načítat ovladač. JDBC 4.0 a novější mají schopnost automaticky načíst ovladač, pokud je dostupný na classpath.)

Př. (pro MySQL)



**3. Vytvoření připojení (Creating a Connection)**

Poté vytvořte připojení k databázi pomocí URL, uživatelského jména a hesla.





**Statement vs. PreparedStatement**

Rozhraní **PreparedStatement** rozšiřuje rozhraní **Statement**

**Statement:**

* *Výhody:*
  + Jednoduché použití pro jednorázové dotazy.
  + Snadné vytváření pro jednoduché SQL dotazy.
* *Nevýhody:*
  + Méně efektivní pro opakované vykonávání dotazů.
  + Bezpečnostní riziko při manipulaci s parametry (SQL injection).

**PreparedStatement:**

* *Výhody:*
  + Předkompilovaný dotaz, což zvyšuje výkon.
  + Umožňuje použití parametrů
  + Bezpečné vložení parametrů pomocí placeholderů, snižuje riziko SQL injection.
* *Nevýhody:*
  + Trochu složitější syntaxe pro vytváření dotazů.
  + Může být příliš zdrcující pro jednorázové a jednoduché dotazy.

**Přehled metod pro provádění SQL dotazů:**

**Rozhraní Statement**

* **boolean execute (String sql,…)**

Provede jakýkoli SQL dotaz (DDL, DML, DQL) a vrací true/false**. true** **vrací** **pro výsledky dotazu, které vrací ResultSet.** Ten poté lze získat pomocí metody **getResultSet().**

* **int executeUpdate (String sql,…)**

Slouží k provedení SQL dotazů INSERT, DELETE, UPDATE. Nebo dotazů, které nic nevrací např. DDL dotazy. Vrací počet ovlivněných řádků.

* **ResultSet executeQuery (String sql)**

Určena pro provádění dotazů, které vrací jeden ResultSet. Převážně se používá pro SELECT.

* **int[ ] executeBatch()**

Slouží k provedení sady příkazů, při úspěšném provedení vrátí pole ovlivněných řádků. Pro přidávání do sady příkazů se používá metoda **addBatch(String sql),** pro vyprázdnění sady metoda **clearBatch()**.

**Rozšiřující rozhraní PreparedStatement**

**PreparedStatemtent** přidává hlavně metody pro práci s parametry.

Metody pro nastavení parametrů, slouží k nastavení hodnoty parametru na určitém indexu.

* **setInt(int parameterIndex, int x)**
* **setString(int parameterIndex, String x)**
* **setDouble(int parameterIndex, double x)**

….

Metoda pro vyprázdnění parametrů

* **clearParameters()**

**DDL (Data Definition Language)**

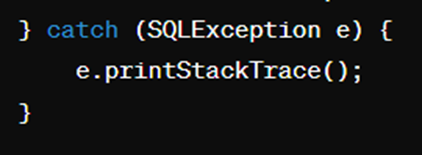
**DDL** je typ SQL příkazu, který pomáhá definovat databázová schémata. Používá se k vytváření, úpravám a mazání databázových struktur, **nikoli** však dat. (CREATE, DROP, ALTER, TRUNCATE, COMMENT, RENAME)

**Př. (pomocí Statement)**

1. try-catch

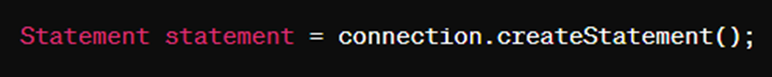
Celý kód musí být obalen v try-catch k zachycení případných výjimek, které mohou vzniknout při vykonání SQL dotazu.





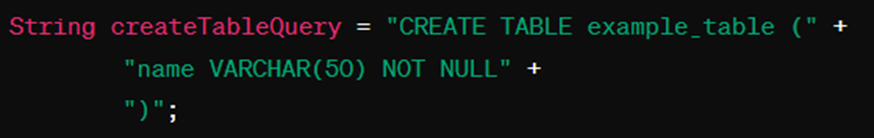
2. Vytvoření objektu statement

Objekt statement umožňuje vykonávat SQL dotazy na připojené databázi.



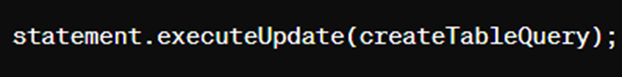
3. Vytvoření DDL dotazu

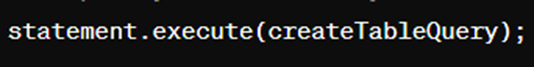
Vytvoření jednoduchého dotazu pro vytvoření tabulky v databázi.



4. Vykonání DDL dotazu

Může být použito více metod pro provedení CREATE příkazu, jelikož CREATE nic nevrací můžeme použít **executeUpdate()**, nebo také lze použít obecnou metodu **execute()**.





5. Uvolnění objektu

Po použití by objekt statement měl být uzavřen, aby byly uvolněny zdroje a bylo zabráněno zbytečnému zatěžování databáze.



**DML (Data Manipulation Language)**

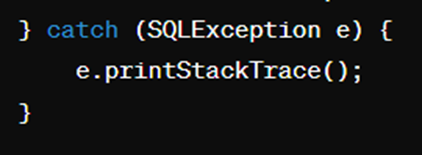
**DML** dotazy se používají k manipulaci s daty v databázi, například pro vkládání, aktualizaci nebo mazání záznamů v tabulkách. (INSERT, UPDATE, DELETE, LOCK)

**Př. (pomocí PreparedStatement)**

1. try-catch

Celý kód musí být obalen v try-catch k zachycení případných výjimek, které mohou vzniknout při vykonání SQL dotazu.





2. Vytvoření DML dotazu

Vytvoření dotazu s parametrem pro úpravu záznamu v databázi. Parametr se zapisuje **“?”**



3. Vytvoření objektu PreparedStatement

Objekt PreparedStatement umožňuje vykonávat SQL dotazy na připojené databázi.



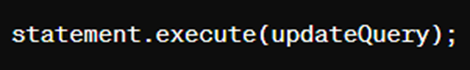
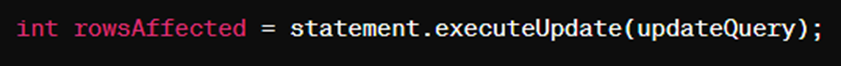
4. Nastavení parametru

Nastaví hodnotu parametru na 1. pozici na 50000.



5. Vykonání DML dotazu

Použijeme metodu **executeUpdate()**, pokud chceme znát počet ovlivněných řádků. Jinak klidně můžeme zase použít obecnou metodu **execute()**.



6. Uvolnění objektu

Po použití by objekt preparedStatement měl být uzavřen, aby byly uvolněny zdroje a bylo zabráněno zbytečnému zatěžování databáze.

