Vývoj Aplikácií s Viacvrstvovou Architektúrou

JDBC (Java Database Connectivity)

IT ACADEMY

STU FIIT

Lektor: PhDr. Ing. Mgr. Miroslav Reiter, DiS., MBA, MPA
Kontakt: miroslav.reiter@it-academy.sk

Čo nás čaká a neminie...

1. časť

- Úvod do Javy
- Štruktúra platformy
- Vývojové technológie
- Kolekcie
- Logovanie
- Lokalizácia

2. časť

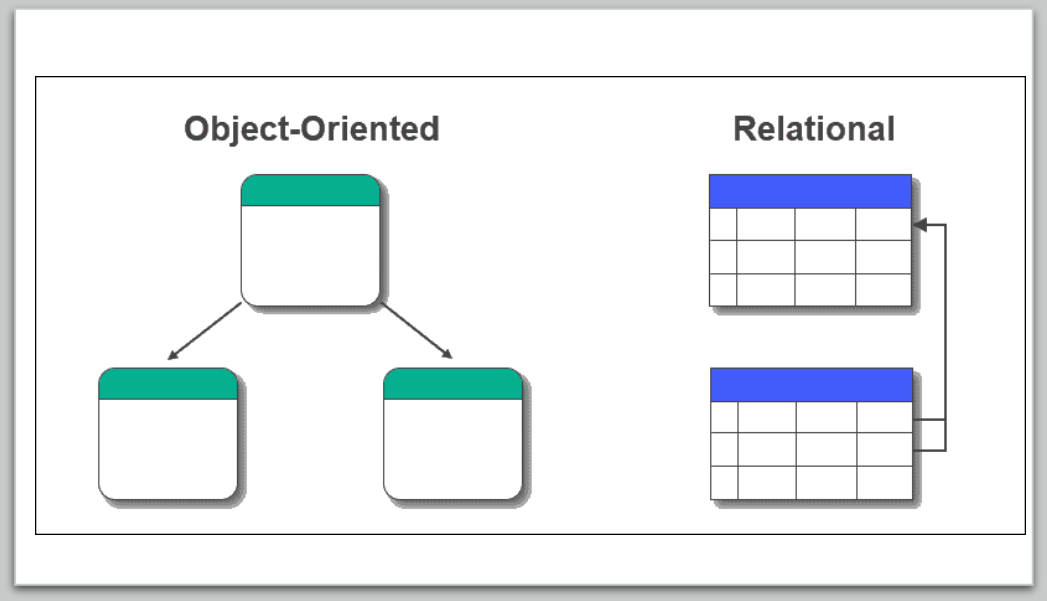
- NIO.2, IO, XML
- Regulárne výrazy
- Modularita
- JDBC
- Bezpečnosť
- Prehľad JEE a .NET

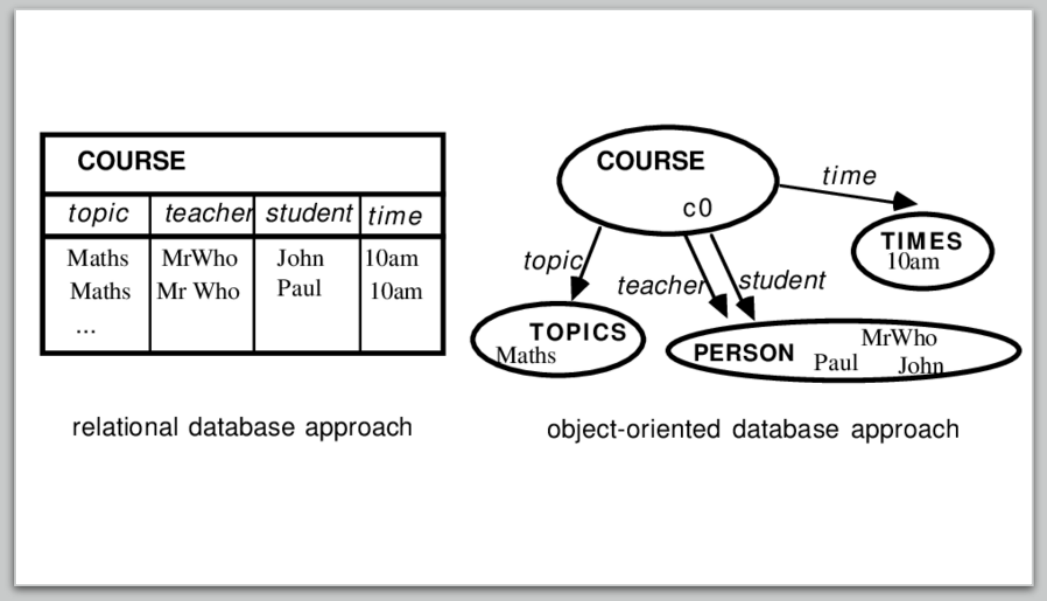
Best practices | Faily | Fuckupy

Ako pracovať v Jave s databázami?

Objektový vs relačný prístup

- Objektový svet a svet relačných databáz sú rozdielne, **ide o 2 odlišné prístupy**
- **Relačné databázy** sú overený **spôsob** ako **pracovať s dátami a ukladať ich** aj keď existujú aj objektovo orientované databázy
- **Relačné databázy objektovo nefungujú** a objekty ukladať (väčšinou) nevie
- Existuje niekoľko možností, ako sa s týmto vysporiadať





Dôležité vlastnosti

OOP a OOP databázy

OOP rozhrania

Identifikátor objektu

Dedičnosť

Polymorfizmus

Enkapsulácia

Relačné databázy

Dátová integrita

Relačná algebra

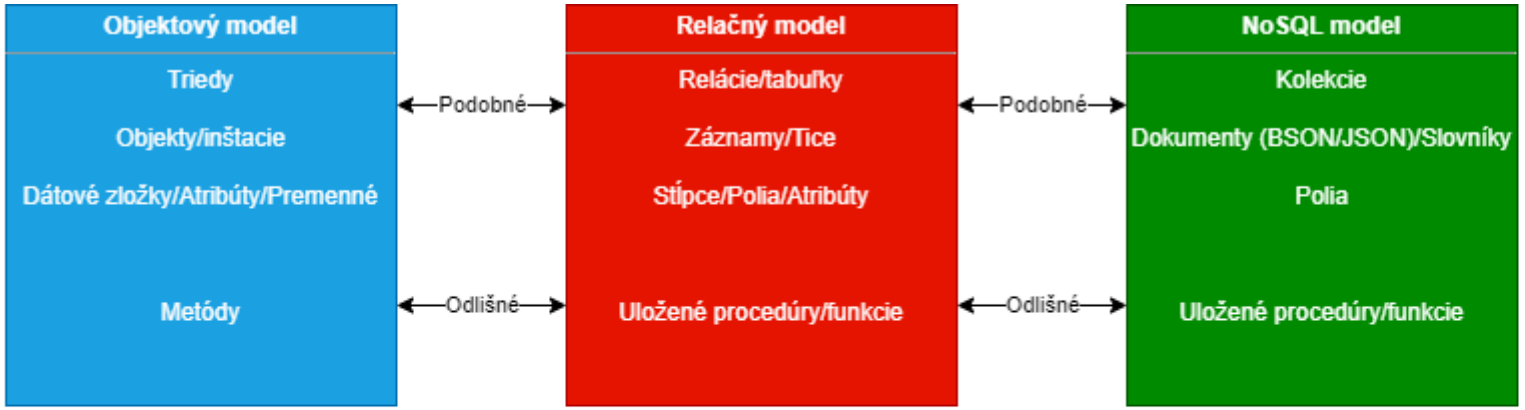
Perzistentnosť

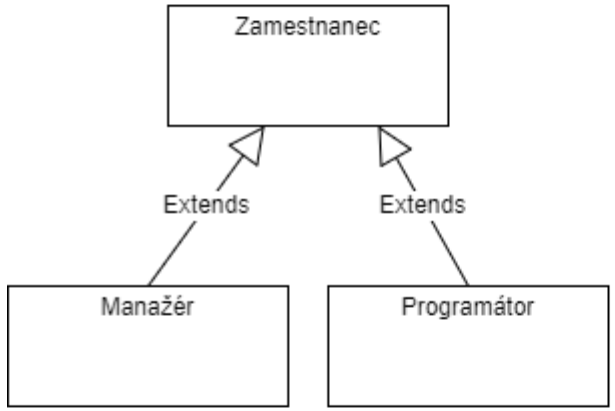
Integrita

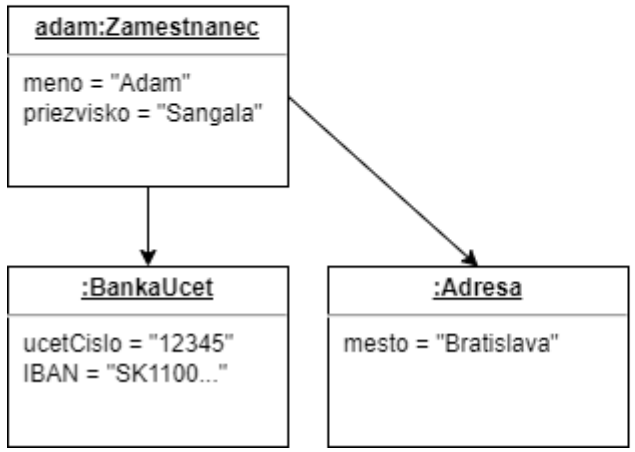
Bezpečnosť

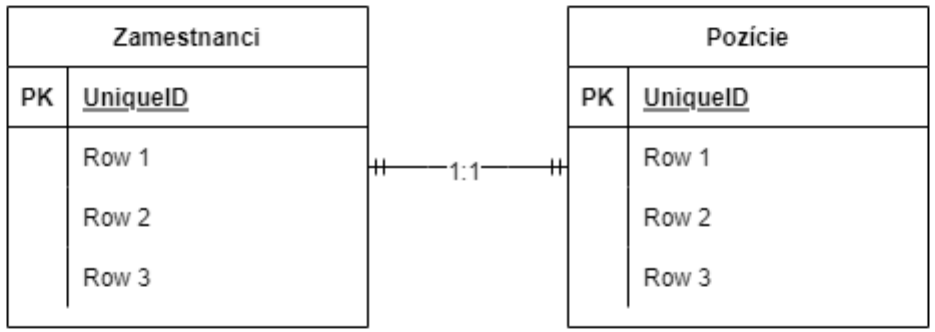
Výkon

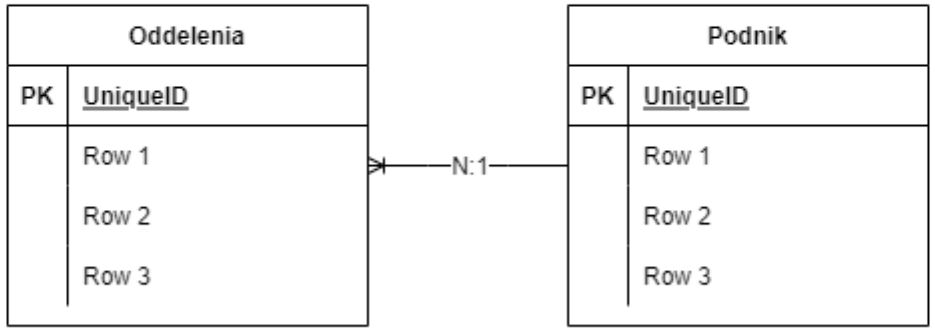
OODBMS	RDBMS
Main Objectives: data encapsulations and independence.	Main Objective: ensuring data independence from application programs.
Independence of classes: classes can be recognized without affecting the mode of using it.	Data independence: Data can be recognized without affecting the mode of using it.
OODBMS store data and methods.	RDBMS store only data.
Encapsulations: the data can be used only through their class methods.	Data partitioning: data can be partitioned depending on the requirements of the users and on the specific user's applications.
Complexity: the structure of data may be complex, involving types of data.	Simplicity: users perceive data as columns, rows/tuples and tables.











Kolekcia Zamestnanec

```
{
        _id : "a8df6hre6s3",
        krstneMeno : "Adam",
        priezvisko : "Sangala",
        bankovyUcet : [
                { ucetCislo : "12345"},
                { ucetCislo: "78946"}
        ],
        adresy : [
                {mesto : "Bratislava"},
                {mesto : "Kosice"}
        ]
}
```

1 zamestnanec - 1 Dokument

Aké máme možnosti v Jave pre prácu s DB?

Neobjektové programovanie

- V Jave veľmi ťažké/skoro nemožné...
- Chceme používať komponenty 3. strán
- Nekvalita kódu a porušovanie OOP princípov

Databázový wrapper

- Umožňuje pracovať s DB ako s objektom
- Pracujeme v jazyku SQL
- Kombinueme objektový a relačný kód

Objektovo relačné mapovania (ORM)

- Java Persistence API
- Hibernate

Objektové databázy

- MongoDB
- Neo4j
- ObjectDB

- PostgreSQL

- Má daná relačná DB aj možnosť práce s objektami?

Ako na (relačné) databázy v Jave?

Java Database Connectivity (JDBC)

1. Pristupuje k **údajom** ako k **riadkom** a **stĺpcom**
2. JDBC je **API**
3. **Bude na skúške** (je aj na certifikačnej skúške)

Java Persistence API (JPA)

1. Pristupuje k **údajom** prostredníctvom **Java objektov** pomocou objektovo-relačné mapovania (**ORM**)
2. Myšlienkou je, že nemusíte písať toľko kódu a svoje údaje získate v objektoch Java
3. **Nebude na skúške** (nie je ani na certifikačnej skúške)

Architektúra JDBC

Dvojvrstvová (Two-tier) architektúra

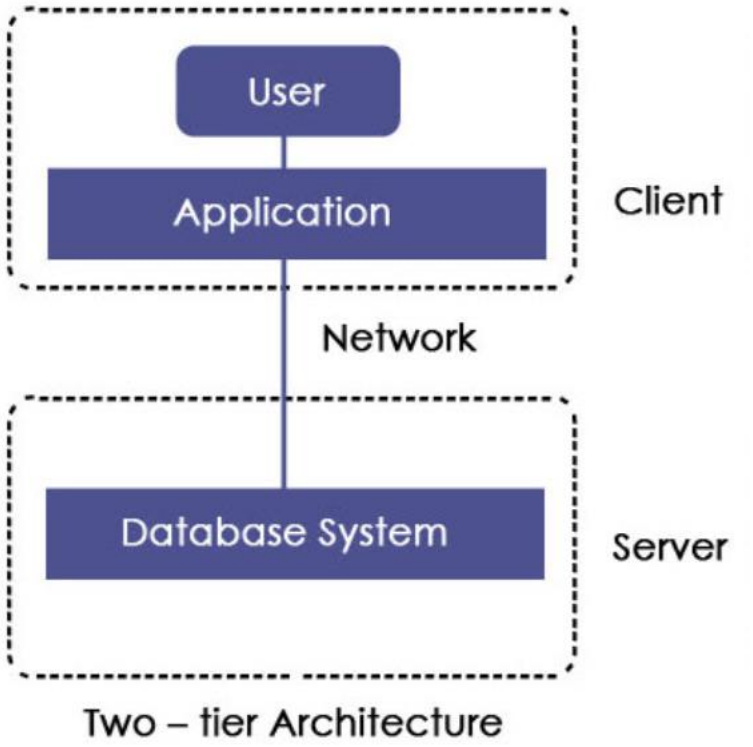
- Táto architektúra pomáha java programu alebo aplikácii priamo komunikovať s databázou
- Na komunikáciu s konkrétnou databázou potrebuje ovládač JDBC
- Dopyt alebo požiadavku odošle používateľ do databázy a výsledky dostane späť
- Databáza môže byť prítomná na rovnakom počítači alebo akomkoľvek vzdialenom počítači pripojenom cez sieť
- Tento prístup sa nazýva architektúra klient-server alebo konfigurácia

Trojvrstvová (Three-tier) architektúra

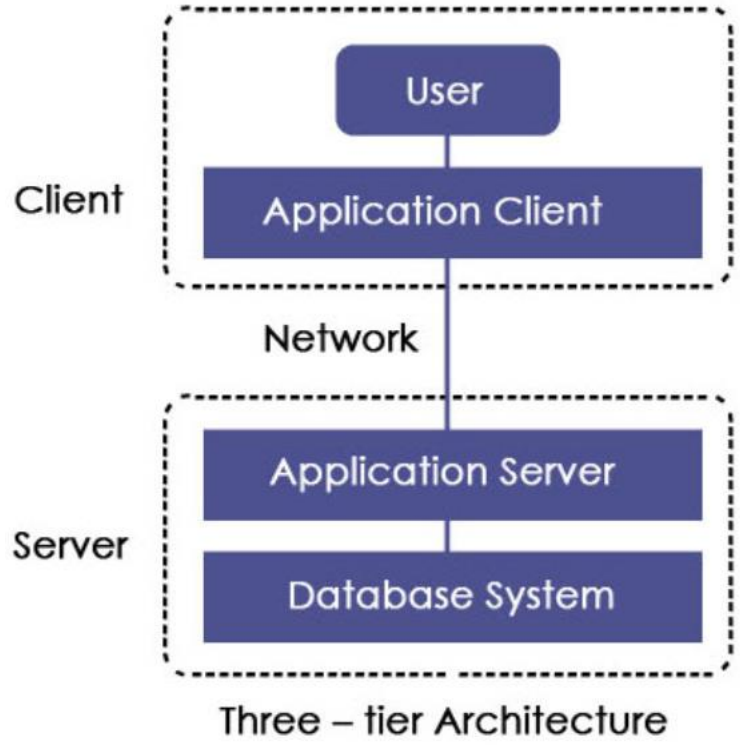
- V tomto neexistuje priama komunikácia
- Požiadavky sa odosielajú do strednej vrstvy, t. j. HTML prehliadač odošle požiadavku do java aplikácie, ktorá sa potom ďalej posiela do databázy
- Databáza spracuje požiadavku a pošle výsledok späť do strednej vrstvy, ktorá následne komunikuje s používateľom
- Zvyšuje výkon a zjednodušuje nasadenie aplikácie

Architektúra s JDBC

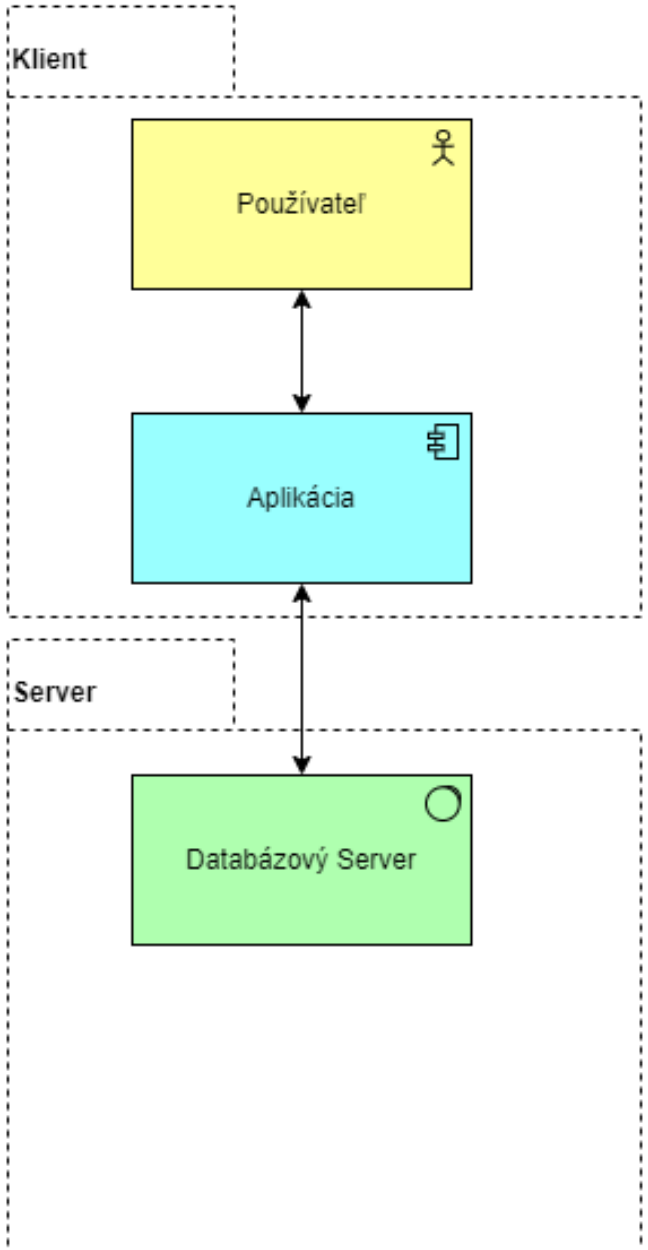
Dvojvrstvová (Two-tier) architektúra



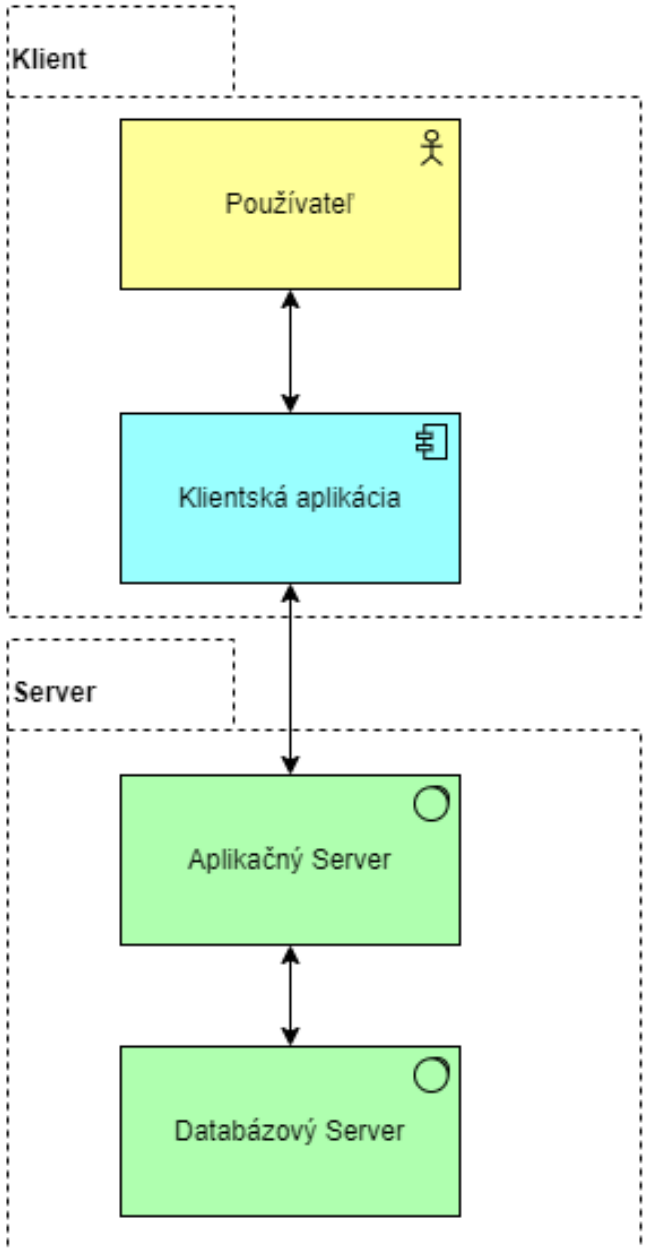
Trojvrstvová (Three-tier) architektúra

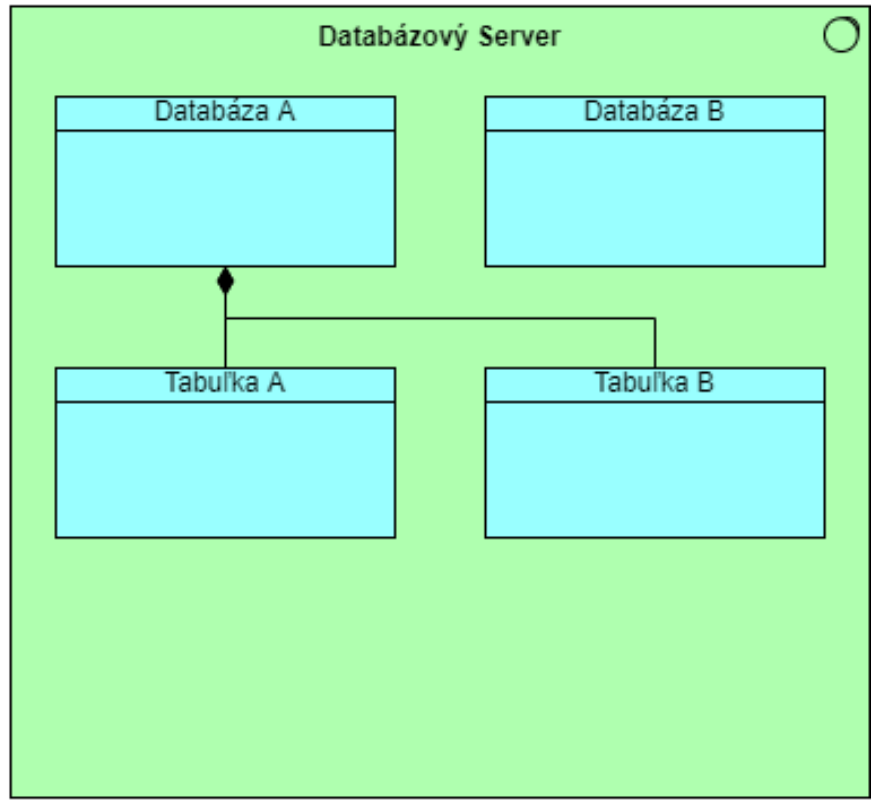


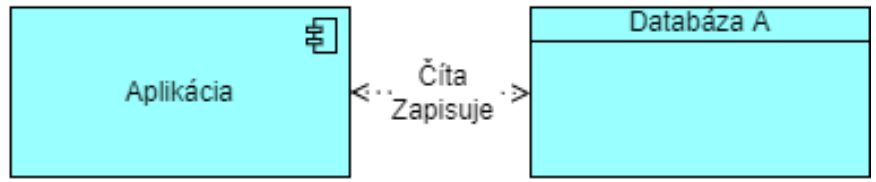
Dvojvrstvová (Two-tier) architektúra

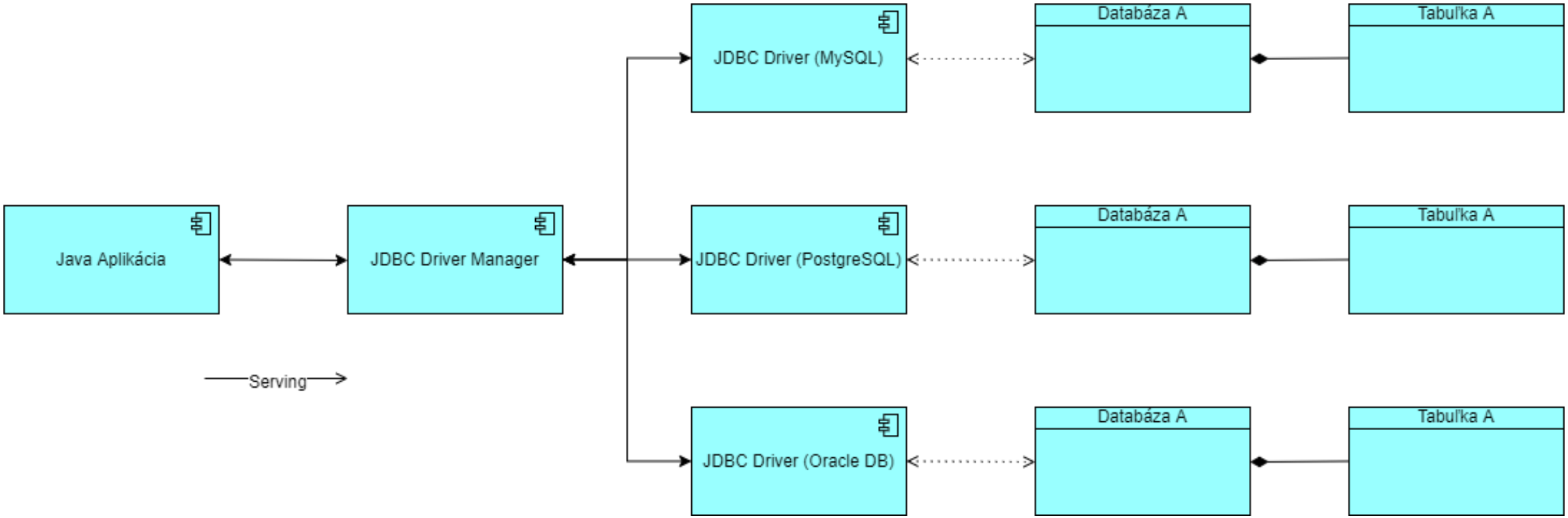


Trojvrstvová (Three-tier) architektúra



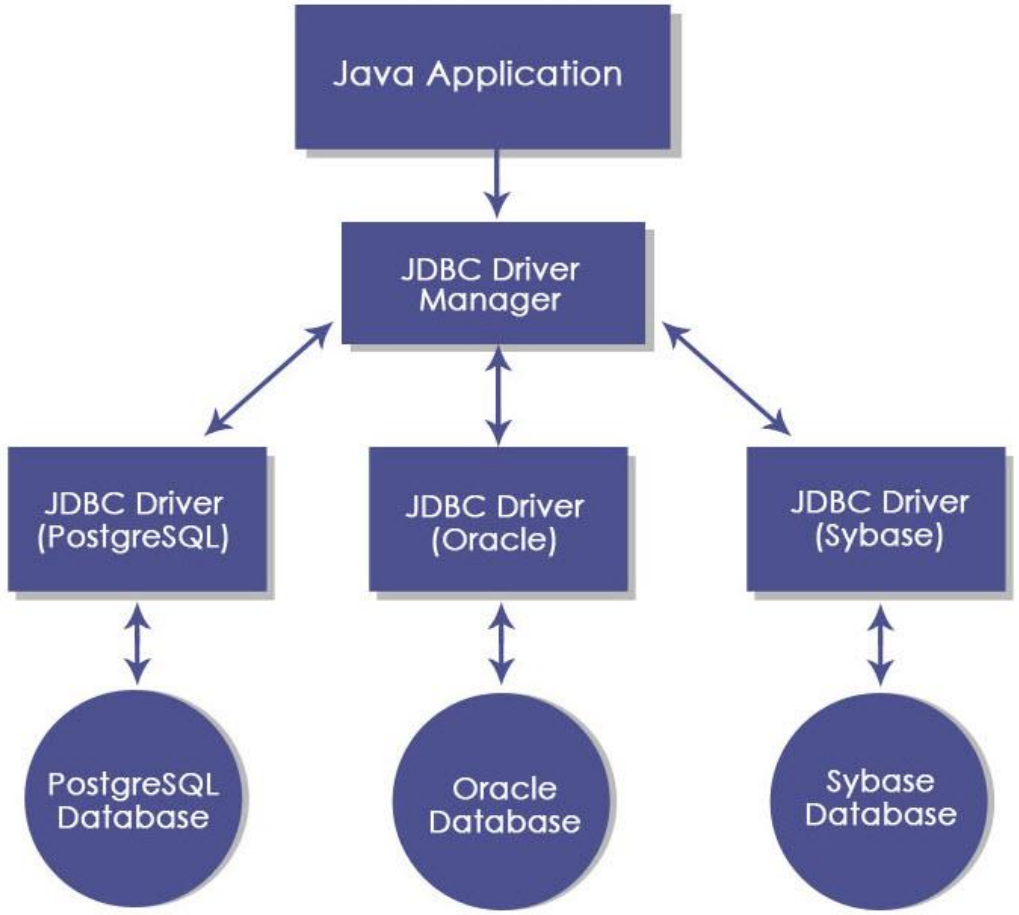





Java Aplikácia
JDBC Driver Manager
JDBC Driver (MySQL)
JDBC Driver (PostgreSQL)
JDBC Driver (Oracle DB)
Databáza A
Databáza A
Databáza A
Tabuľka A
Tabuľka A
Tabuľka A
Serving

Komponenty JDBC architektúry

- **Driver Manager**: **Trieda**, ktorá obsahuje **zoznam všetkých ovládačov**. Keď je prijatá požiadavka na pripojenie, spáruje ju s príslušným databázovým ovládačom pomocou protokolu nazývaného komunikačný subprotokol. Na nadviazanie spojenia sa použije zodpovedajúci ovládač.
- **Driver**: Je **rozhranie**, **ktoré riadi komunikáciu s databázovým serverom**. Na komunikáciu sa používajú objekty DriverManager
- **Connection**: Je to **rozhranie**, ktoré obsahuje **metódy** na **kontaktovanie databázy**
- **Statement**: Toto **rozhranie** vytvára **objekt** na **odosielanie** SQL **dopytov** alebo **príkazov** do **databázy**
- **ResultSet**: Obsahuje **výsledky získané** po **vykonaní príkazov** SQL alebo **dopytov**
- **SQLException**: **Všetky chyby**, ktoré sa vyskytnú v databázovej aplikácii, sú spracované touto triedou

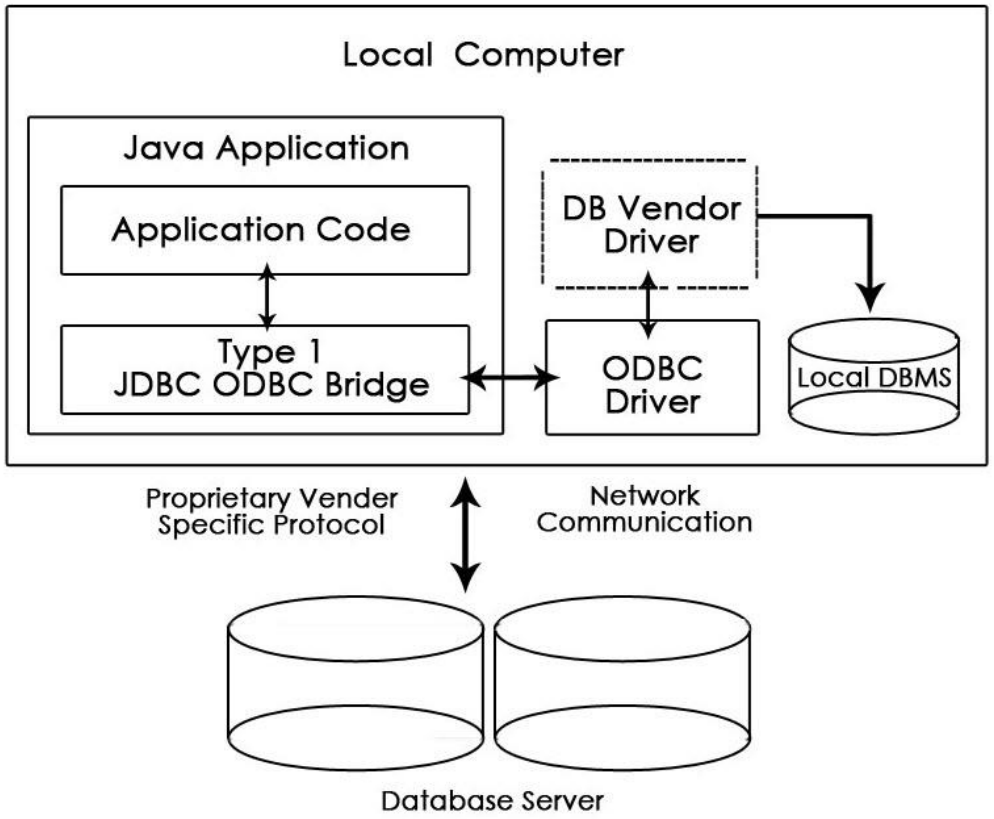


Rozhrania v balíčku java.sql

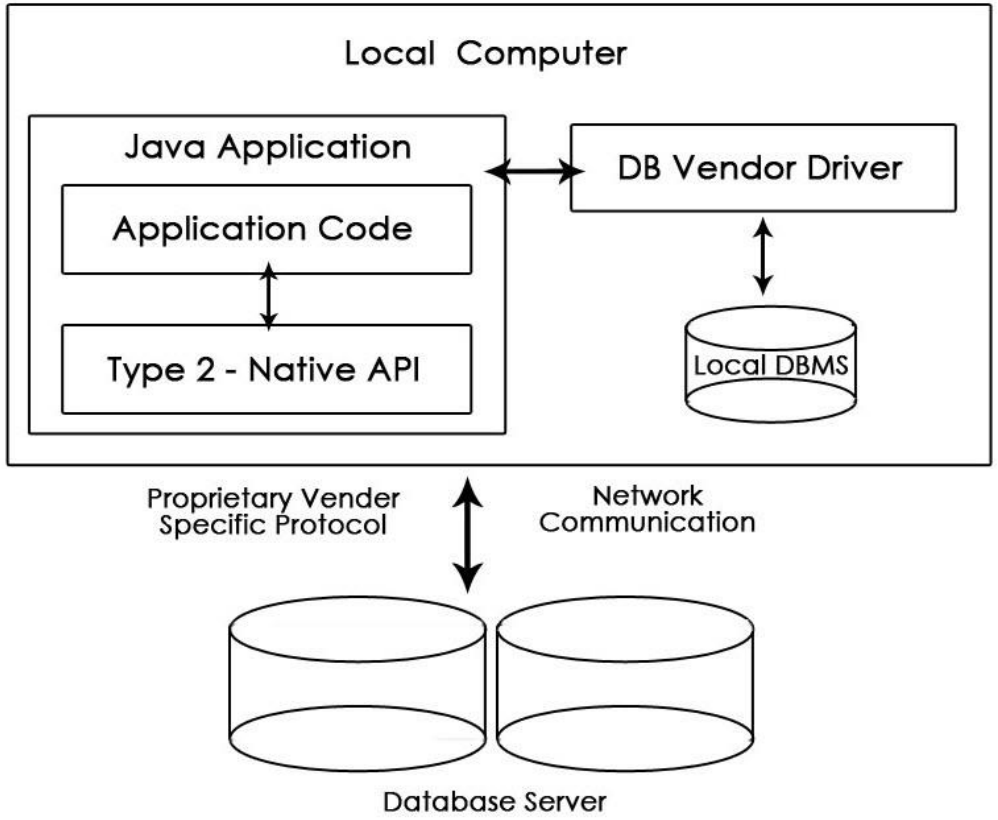
1. **Driver**: **povoľuje viacero databázových ovládačov**. Objekty DriverManager sú vytvorené na komunikáciu s databázou. Tieto objekty vytvára DriverManager.registerDriver();
2. **Connection**: Rozhranie pripojenia **vytvára spojenie**, tj **reláciu medzi Java programom a databázou**. Má veľa metód ako **rollback**(), **close**()
3. **Statement**: Poskytuje **metódy** na **vykonávanie SQL dopytov**. Poskytuje továrenské metódy na získanie objektu ResultSet. Niektoré metódy rozhrania príkazov sú **executeQuery**(), **executeUpdate**()
4. **PreparedStatement**: Pomáha, keď je potrebné **mnohokrát implementovať dopyty SQL**. Prijíma vstupné parametre počas behu.
5. **CallableStatement**: Používa sa, keď sa má **pristupovať k uloženým procedúram**. Prijíma aj **parametre počas behu**.
6. ResultSet: Pomáha ukladať výsledok vrátený po vykonaní SQL dopytov.

4 typy ovládačov JDBC

1. Driver/JDBC-ODBC Bridge

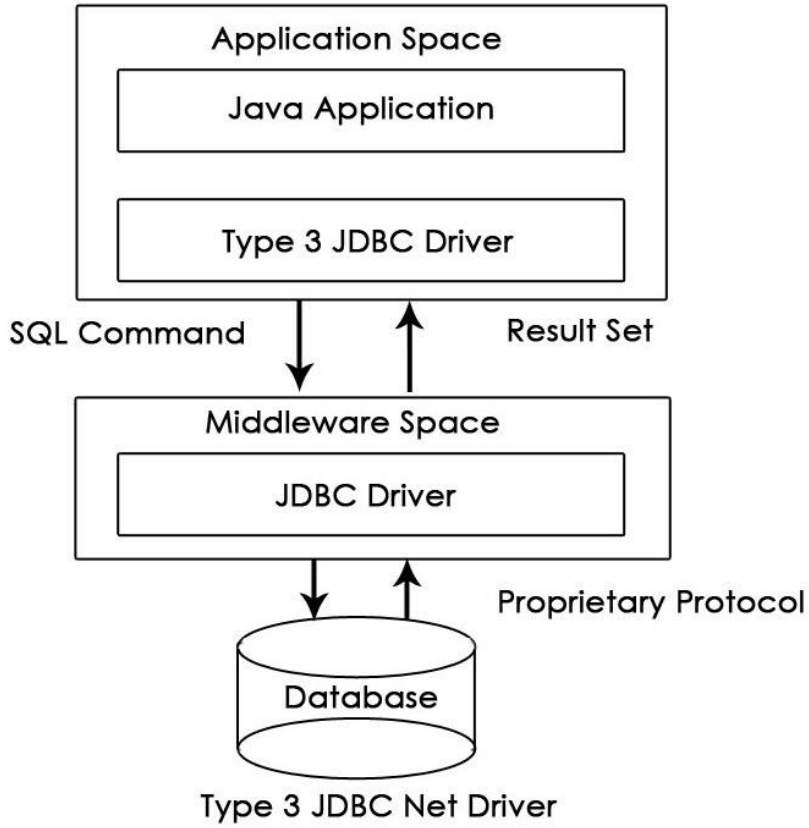


2. Driver/Nat. API Part. Java Driver

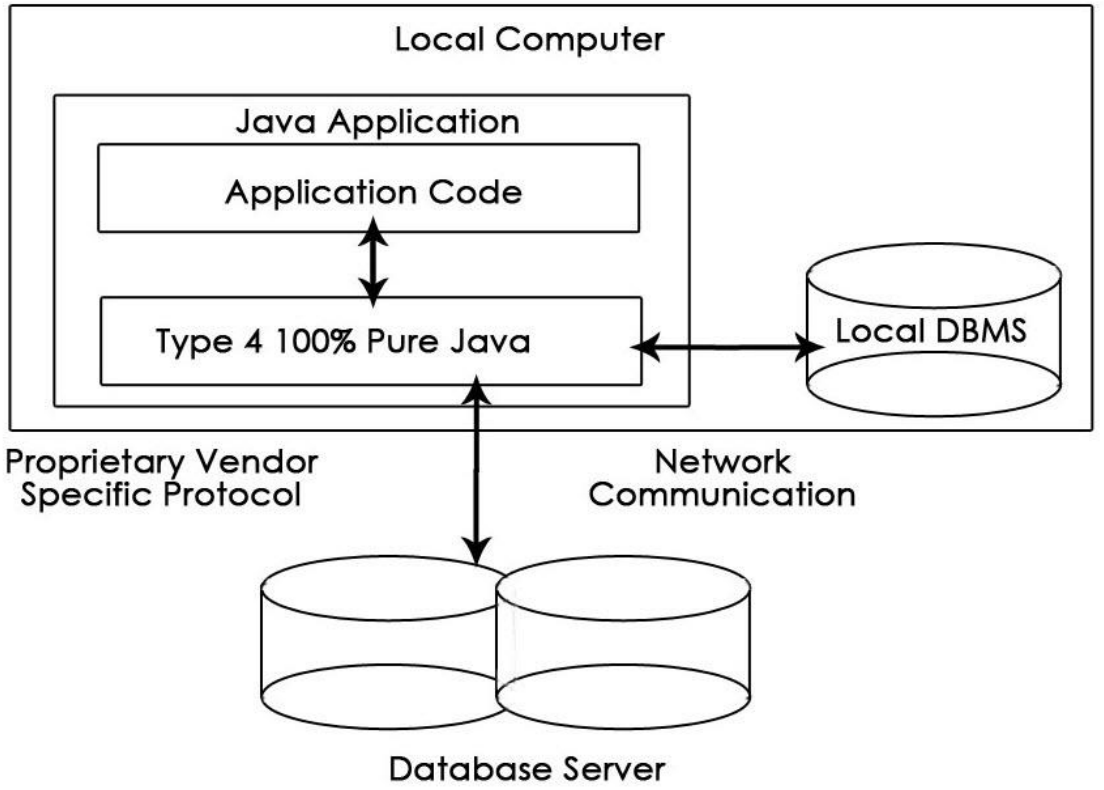


4 typy ovládačov JDBC

3. Driver/Network Protocol Driver



4. Thin Driver



Java Database Connectivity (JDBC)

- **Aplikačné programové rozhranie (API)** pre programovací **jazyk Java**, ktoré definuje, **ako môže klient pristupovať k databáze**
- Ide o **technológiu prístupu** k **údajom** na báze Java, ktorá sa používa **na pripojenie k databáze Java**
- Je **súčasťou platformy Java Standard Edition** od Oracle
- Poskytuje **metódy** na **vyhľadávanie** a **aktualizáciu** údajov v databáze a je orientovaný **na relačné databázy**
- Bridge JDBC-ODBC umožňuje pripojenia k akémukoľvek zdroju údajov s prístupom ODBC v hostiteľskom prostredí Java Virtual Machine (JVM)

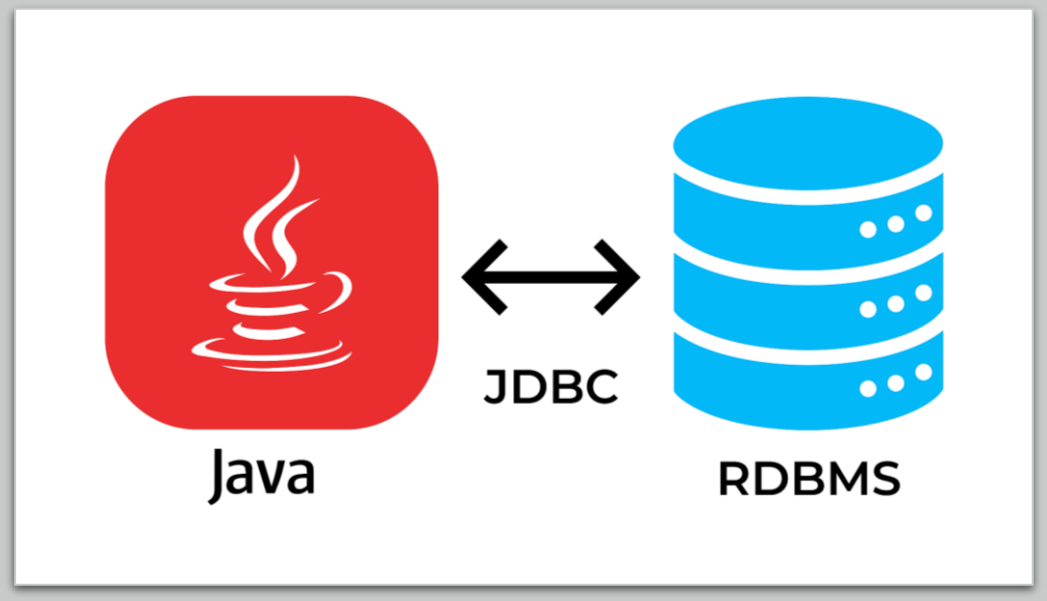
JDBC

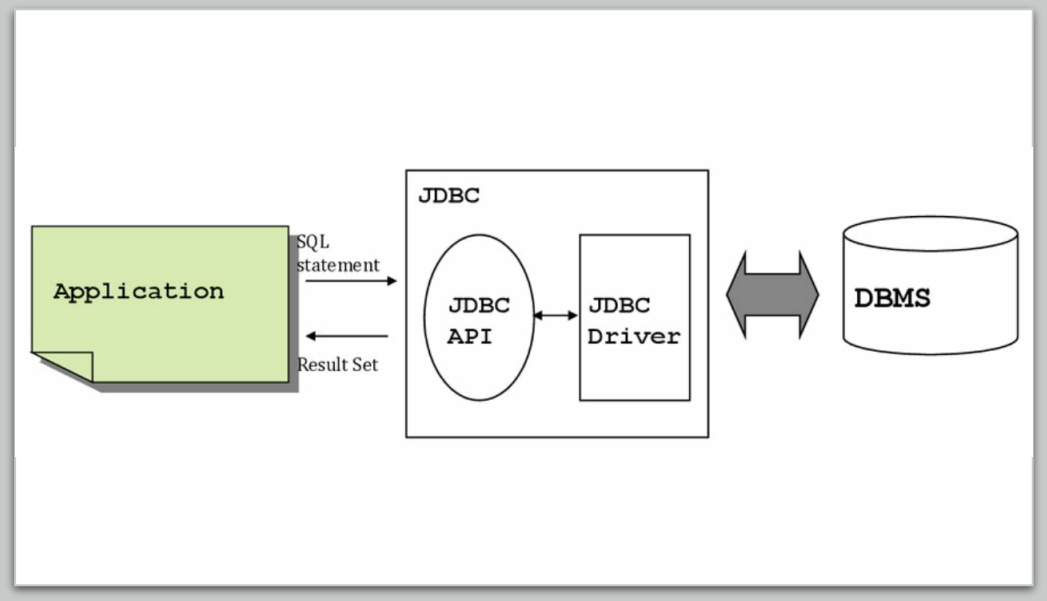
Developer(s)	Oracle Corporation
Stable release	JDBC 4.3 / September 21, 2017
Operating system	Cross-platform
Type	Data access API
Website	JDBC API Guide

Od JDK 1.1. (02-1997)

JDBC
(Java DB Connectivity)

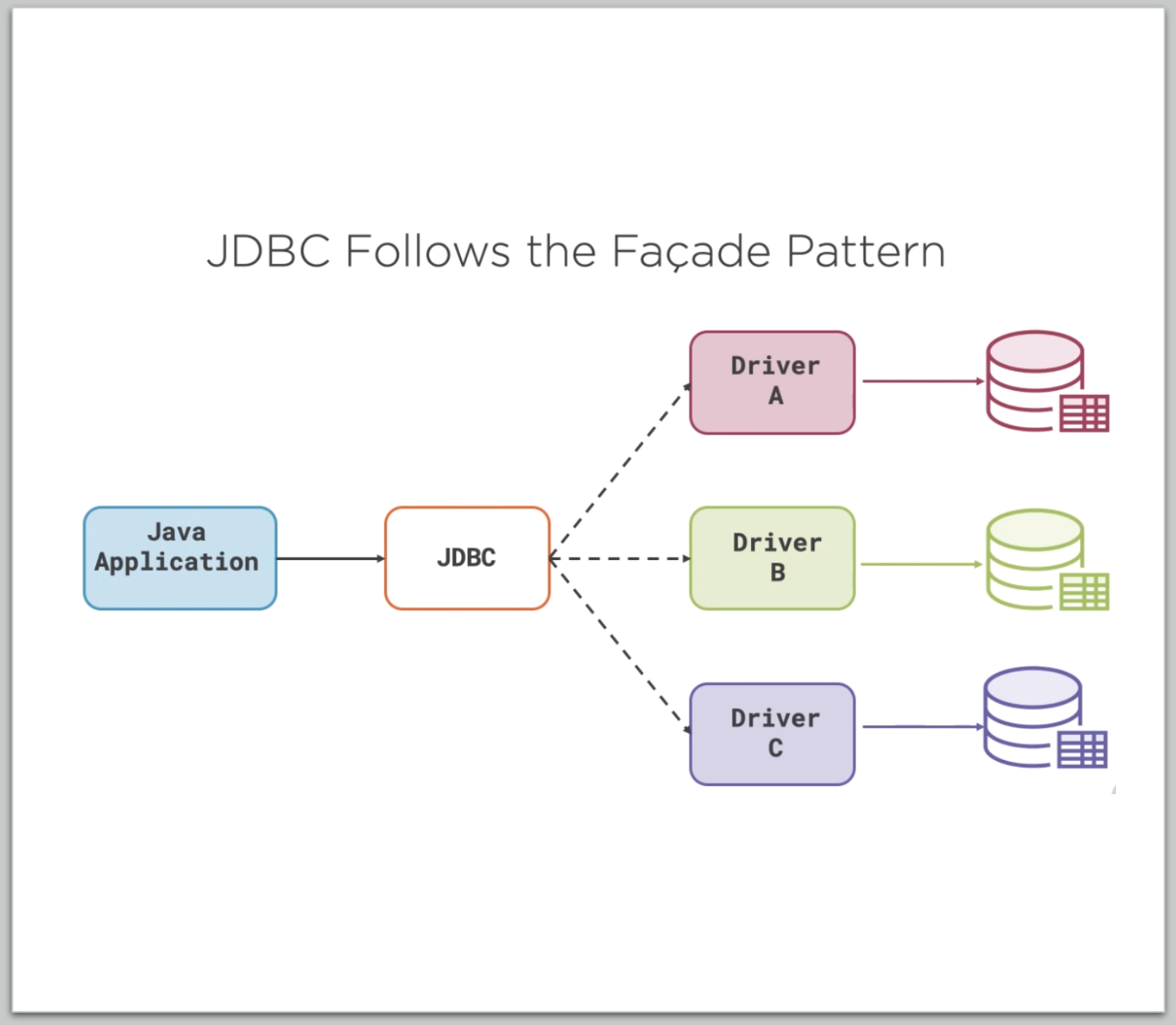
- Vytvorenie spojenia s databázou
- Posielanie SQL dopytov
- Získavanie a spracovanie resultov



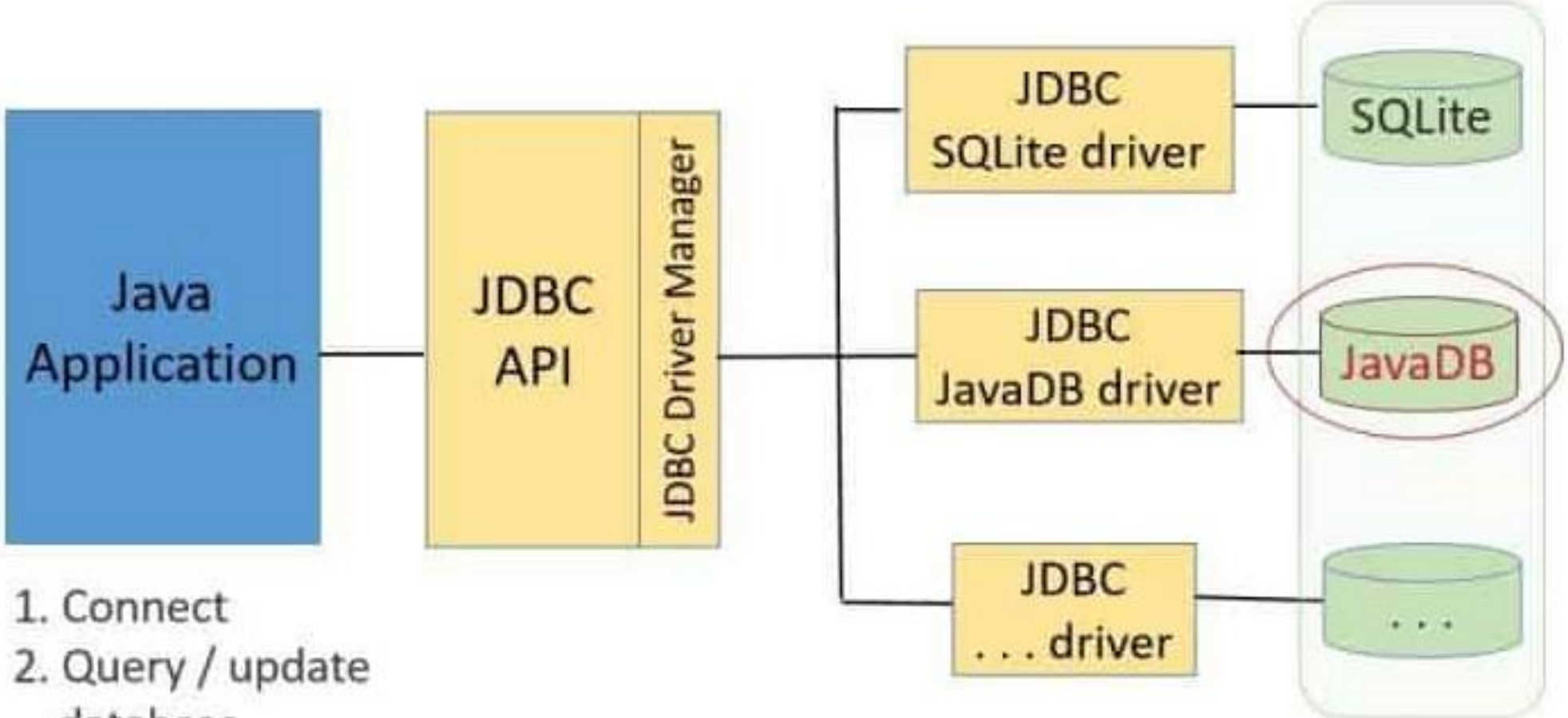


Java Database Connectivity (JDBC)

- **Všetly triedy JDBC** sú obsiahnuté v java **balík**och **java.sql** a **javax.sql**
- **Posledná verzia JDBC 4.3** (2017)
- Umožňuje existenciu viacerých implementácií a ich použitie tou istou aplikáciou
- API poskytuje **mechanizmus** na **dynamické načítanie správnych Java balíkov** a ich registráciu v JDBC Driver Manager
- Správca ovládačov sa používa vzor **továreň pripojenia na vytváranie pripojení JDBC** (**vzor Fasáda**)



Ako funguje JDBC?



1. Connect
2. Query / update database
3. Receive results

Java Database Connectivity (JDBC)

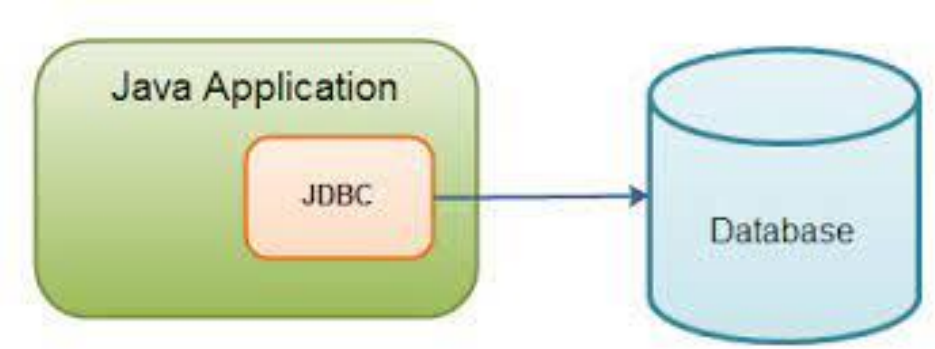
- Pripojenia JDBC podporujú vytváranie a vykonávanie príkazov
- Môžu to byť aktualizačné príkazy, ako napríklad SQL CREATE, INSERT, UPDATE a DELETE, alebo to môžu byť príkazy dopytu, ako napríklad SELECT
- JDBC predstavuje príkazy využívajúce jednu z nasledujúcich tried:
  1. **Statement** – **pokaždé sa posiela na databázový server**
  2. **PreparedStatement** – príkaz je **uložený do vyrovnávacej pamäte** a potom je cesta vykonania vopred určená na databázovom serveri, čo **umožňuje jeho vykonanie viackrát efektívnym spôsobom**.
  3. **CallableStatement** – používa sa na vykonávanie **uložených procedúr v databáze**.
- Aktualizačné príkazy ako INSERT, UPDATE a DELETE vrátia počet aktualizácií, ktorý indikuje, koľko riadkov bolo ovplyvnených v databáze. Tieto príkazy neposkytujú žiadne ďalšie informácie
- Príkazy dopytu vrátia sadu výsledkov riadkov JDBC (**ResultSet**)
- **Jednotlivé stĺpce v riadku** sa **získavajú** buď **podľa názvu** alebo **podľa čísla stĺpca**
- V množine výsledkov môže byť ľubovoľný počet riadkov.
- Sada výsledkov riadkov **obsahuje metadáta**, ktoré popisujú **názvy stĺpcov** a ich **typy**

Oracle Datatype	`setXXX()` Methods
CHAR	`setString()`
VARCHAR2	`setString()`
NUMBER	`setBigDecimal()`
	`setBoolean()`
	`setByte()`
	`setShort()`
	`setInt()`
	`setLong()`
	`setFloat()`
	`setDouble()`
INTEGER	`setInt()`
FLOAT	`setDouble()`
CLOB	`setClob()`
BLOB	`setBlob()`
RAW	`setBytes()`
LONGRAW	`setBytes()`
DATE	`setDate()`
	`setTime()`
	`setTimestamp()`

Inštalácia MySQL/PostgreSQL/... drivera

- Stiahnuť driver napr. MySQL JDBC Driver : http://www.mysql.com/downloads/api-jdbc-stable.html
- Pri použití v aplikácii **stačí pridať jar súbor** mysql-connector-java-X.X.X-stable-bin.jar **systémovej premennej CLASSPATH**

```
cmd>set
CLASSPATH=%CLASSPATH%;drive:\path\to\my
sql-connector-java-X.X.X-stable-bin.jar
```



- Je možné **spustiť aplikáciu** použitím **parametra classpath**

```
java -classpath=drive:\path\to\mysql-
connector-java-X.X.X-stable-bin.jar
aplikacia.class
```

Java JDBC API

The Java Database Connectivity (JDBC) API provides universal data access from the Java programming language. Using the JDBC API, you can access virtually any data source, from relational databases to spreadsheets and flat files. JDBC technology also provides a common base on which tools and alternate interfaces can be built.

The JDBC API is comprised of two packages:

- `java.sql`
- `javax.sql`

You automatically get both packages when you download the Java Platform Standard Edition (Java SE) 8.

To use the JDBC API with a particular database management system, you need a JDBC technology-based driver to mediate between JDBC technology and the database. Depending on various factors, a driver might be written purely in the Java programming language or in a mixture of the Java programming language and Java Native Interface (JNI) native methods. To obtain a JDBC driver for a particular database management system, see JDBC Data Access API.

Enhancements in Java SE 8

Component: core-libs
Sub-Component: java.sql:bridge
Synopsis: The JDBC-ODBC Bridge has been removed.
RFE: 7176225

Component: core-libs
Sub-Component: java.sql
Synopsis: JDBC 4.2 introduces the following features:

- Addition of REF_CURSOR support

MySQL Community Downloads

Connector/J

General Availability (GA) Releases | Archives

Connector/J 8.0.28

Select Operating System:

Microsoft Windows

Recommended Windows Download:

Windows (x86, 32 & 64-bit), MySQL Installer MSI — Go to Download Page >

15 April 2022

PostgreSQL JDBC Driver 42.3.4 Released

Notable changes

Changed

- fix: change name of build cache PR 2471
- feat: add support for ResultSet#getObject(OffsetTime.class) and PreparedStatement#setObject(OffsetTime.class) PR 2467
- fix: Use non-synchronized getTimeZone in TimestampUtils PR 2451
- docs: Fix CHANGELOG.md misformatted markdown headings PR 2461
- docs: remove loggerLevel and loggerFile from docs and issues #2489
- feat: use direct wire format -> LocalDate conversion without resorting to java.util.Date, java.util.Calendar, and default timezones PR 2464 fixes Issue #2221

Added

Fixed

- docs: Update testing documentation PR 2446
- fix: Throw an exception if the driver cannot parse the URL instead of returning NULL fixes Issue PR 2421 (#2441)
- fix: Use PGProperty instead of the property names directly PR 2444
- docs: update changelog, missing links at bottom and formatting PR 2460
- fix: Remove isDeprecated from PGProperty. It was originally intended to help produce automated docs. Fixes #Issue 2479 PR 2480
- fix: change PGInterval parseISO8601Format to support fractional second PR 2457
- fix: More test and fix for issues discovered by PR #2476 PR #2488

See full changelog for 42.3.4

---

15 February 2022

PostgreSQL JDBC Driver 42.3.3 Released

Notable changes

LATEST RELEASES

42.3.4 · 15 Apr 2022 · Notes
42.3.3 · 15 Feb 2022 · Notes
42.3.2 · 01 Feb 2022 · Notes
42.2.25 · 01 Feb 2022 · Notes
all · 22 Dec 2021 · Notes

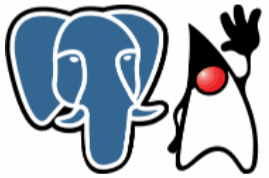
Downloads | Snapshots

SHORTCUTS

» GitHub project
» Documentation
» Mailing list
» Report a security issue
» Report a bug
» FAQ

SUPPORT US

PostgreSQL is free. Please support our work by making a donation.



Oracle Database JDBC driver and Companion Jars Downloads

Governed by the No-clickthrough FUTC license

PLEASE READ :

- Use the latest version of the Oracle JDBC driver that supports the JDK and Database version that you use and is compatible with your support requirements.
- Get all new and older versions of Oracle JDBC drivers from Central Maven Repository and Refer to Maven Central Guide for details.
- 21c is an Innovation Release. 21c has premier support until April 2024. Refer to page#6 of Lifetime Support Policy for more details.
- 19c is a Long Term Release. 19c has premier support until April 2024 and extended support until April 2027.
- You can also get the older release and quarterly updates of Oracle JDBC drivers from support.oracle.com. Refer to **MOS note 2849223.1** for more details.

Refer to these 21c related documents for more information.

Oracle JDBC Developer's Guide

UCP Developer's Guide

Online JDBC Javadoc

Online UCP Javadoc

Online Reactive Streams Ingestion (RSI) Javadoc

Oracle Database 21c (21.5) JDBC Driver & UCP Downloads

Supports Oracle Database versions - 21c, 19c, 18c, and 12.2

Name	Download	JDK Supported	Description
Oracle JDBC driver	ojdbc11.jar	Implements JDBC 4.3 spec and certified with JDK11 and JDK17	Oracle JDBC driver except classes for NLS support in Oracle Object and Collection types. (5,173,200 bytes) - (SHA1: 3e588fed4618a39c7d655bfc1159ecb57fabc217)
Oracle JDBC driver	ojdbc8.jar	Implements JDBC 4.2 spec and certified with JDK8 and JDK11	Oracle JDBC driver except classes for NLS support in Oracle Object and Collection types. (5,080,094 bytes) - (SHA1: eb13f0948ca8675f71d59b5e8a1129d4c6189e7a)
Universal Connection Pool - ucp11.jar	ucp11.jar	Certified with JDK11 and JDK17	Universal Connection Pool **to be used with ojdbc11.jar** (1,799,268 bytes) - (SHA1: 207a260b482423effd7b0bb8c04695932cf4992d)

Version

SQL Server 2019

Filter by title

Programming to interact with SQL Server
Welcome to SQL Server >
SQL Server drivers
Driver feature support matrix
SQL Server driver history
SQL data developer
ADO.NET
JDBC
Microsoft JDBC Driver for SQL Server
Getting started
Overview
Overview
Download
Release notes
System requirements
Support matrix
Using the driver
Parsing results
Using useFMTOnly
Understanding Java EE support
Deploying the driver
Finding additional information

Download PDF

Download Microsoft JDBC Driver for SQL Server

Article • 04/05/2022 • 2 minutes to read • 13 contributors

The Microsoft JDBC Driver for SQL Server is a Type 4 JDBC driver that provides database connectivity through the standard JDBC application program interfaces (APIs) available on the Java platform. The driver downloads are available to all users at no extra charge. They provide access to SQL Server from any Java application, application server, or Java-enabled applet.

Download

Version 10.2 is the latest general availability (GA) version. It supports Java 8, 11, and 17. If you need to use an older Java runtime, see the Java and JDBC specification support matrix to see if there's a supported driver version you can use. We're continually improving Java connectivity support. As such we highly recommend that you work with the latest version of Microsoft JDBC driver.

Download Microsoft JDBC Driver 10.2 for SQL Server (zip)
Download Microsoft JDBC Driver 10.2 for SQL Server (tar.gz)

Version information

- Release number: 10.2.0
- Released: January 31, 2022

When you download the driver, there are multiple JAR files. The name of the JAR file indicates the version of Java that it supports.

Note

In this article

Download
Available languages
Using the JDBC driver with Maven Central
Unsupported drivers

Show more

Príklad na pripojenie k DB

```
Connection conn = DriverManager.getConnection(
     "jdbc:somejdbcvendor:other data needed by some jdbc vendor",
     "myLogin",
     "myPassword");
try {
     /* you use the connection here */
} finally {
    //It's important to close the connection when you are done with it
    try {
        conn.close();
    } catch (Throwable e) { /* Propagate the original exception
                               instead of this one that you want just logged */
        logger.warn("Could not close JDBC Connection", e);
    }
}
```

Try with Resources od JDK 1.7

- RowSet 1.2: Lists the enhancements for JDBC RowSet.

RFE: 8005080

Enhancements in Java SE 7

Component: core-libs
Sub-Component: java.sql
Synopsis: JDBC 4.1 introduces the following features:

- The ability to use a `try`-with-resources statement to automatically close resources of type `Connection`, `ResultSet`, and `Statement`
- RowSet 1.1: The introduction of the `RowSetFactory` interface and the `RowSetProvider` class, which enable you to create all types of row sets supported by your JDBC driver.

RFE: 6589685

Component: docs
Sub-Component: release_notes
Synopsis: The JDBC-ODBC Bridge will be removed in JDK 8.
RFE: 8001747

More Information

- JDBC Basics Tutorial
- JDBC Home Page
- Frequently Asked Questions about JDBC

Príklad na pripojenie k DB a vkladanie dát

```
try (Connection conn = DriverManager.getConnection(
     "jdbc:somejdbcvendor:other data needed by some jdbc vendor",
     "myLogin",
     "myPassword")) {
     /* you use the connection here */
} // the VM will take care of closing the connection
```

```
try (Statement stmt = conn.createStatement()) {
    stmt.executeUpdate("INSERT INTO MyTable(name) VALUES ('my name')");
}
```

Try with Resources od JDK 1.7

Príklad na výber dát z DB/tabuľky

```
import java.sql.*;

class Test {
   public static void main(String[] args) {
      Connection con = null;
      Statement stmt = null;

      try {
         Class.forName("oracle.jdbc.driver.OracleDriver");
         con = DriverManager.getConnection( "jdbc:oracle:thin:@dbclick", "user", "passwd");
         stmt = con.createStatement();
         ResultSet rs = stmt.executeQuery("select * from product");

         while (rs.next()) {
            String product = rs.getString(1);
            System.out.println(product);
         }
      } catch (Exception e) {
         e.printStackTrace();
      } finally {
         try {
            if (stmt != null) stmt.close();
            if (con != null) con.close();
         } catch (Exception e) { }
      }
   }
}
```

Try with Resources od JDK 1.7

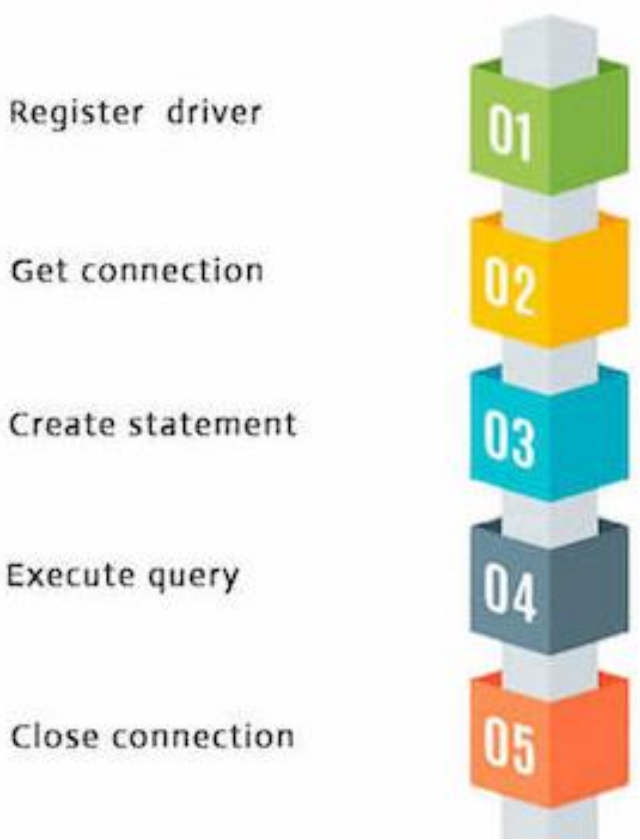
Príklad na výber dát z DB/tabuľky

```
try (Statement stmt = conn.createStatement();
     ResultSet rs = stmt.executeQuery("SELECT * FROM MyTable")
) {
     while (rs.next()) {
          int numColumns = rs.getMetaData().getColumnCount();
          for (int i = 1; i <= numColumns; i++) {
             // Column numbers start at 1.
             // Also there are many methods on the result set to return
             //  the column as a particular type. Refer to the Sun documentation
             //  for the list of valid conversions.
             System.out.println("COLUMN " + i + " = " + rs.getObject(i));
          }
     }
}
```

Try with Resources od JDK 1.7

1. Načítanie ovládača (Loading driver)

```
Class.forName("myDriver.ClassName");
Class.forName("com.mysql.jdbc.Driver");
Class.forName("org.postgresql.Driver");
Class.forName("oracle.jdbc.driver.OracleDriver");

DriverManager.registerDriver(new
oracle.jdbc.driver.OracleDriver());
```



ClassNotFoundException, ak ovládač nie je k dispozícii

2. Vytváranie spojení (Connecting to DB)

- Po načítaní ovládača sa vytvorí pripojenie
- Objekt **pripojenia používa meno používateľa**, **heslo** a adresu **URL** na nastavenie pripojenia
- URL má preddefinovaný formát, ktorý obsahuje názov databázy, použitý ovládač, IP adresu, kde je databáza uložená, číslo portu a poskytovateľa služieb

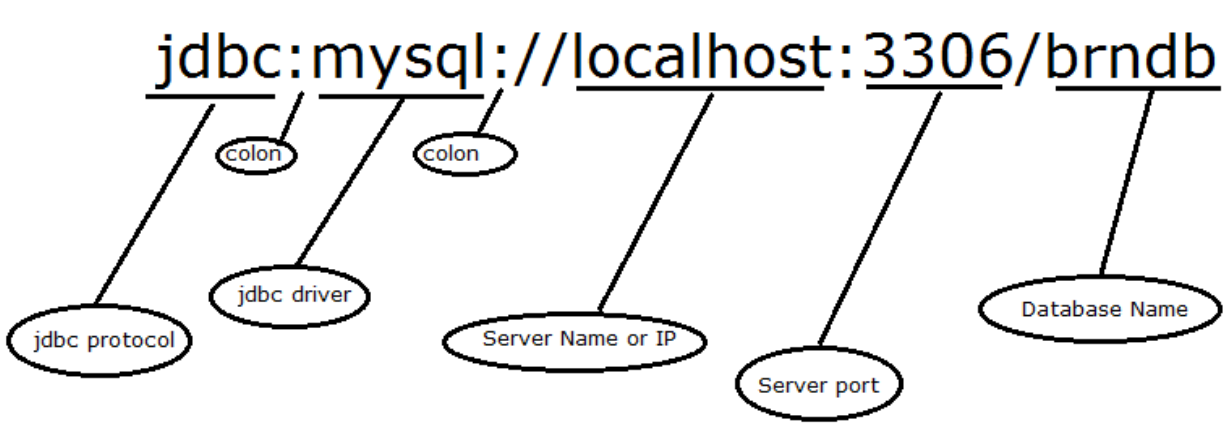
```
Connection con =
DriverManager.getConnection(URL,
user, password);
```

```
Connection con =
DriverManager.getConnection(
"jdbc:mySubprotocol:myDataSource",
username, password);
```

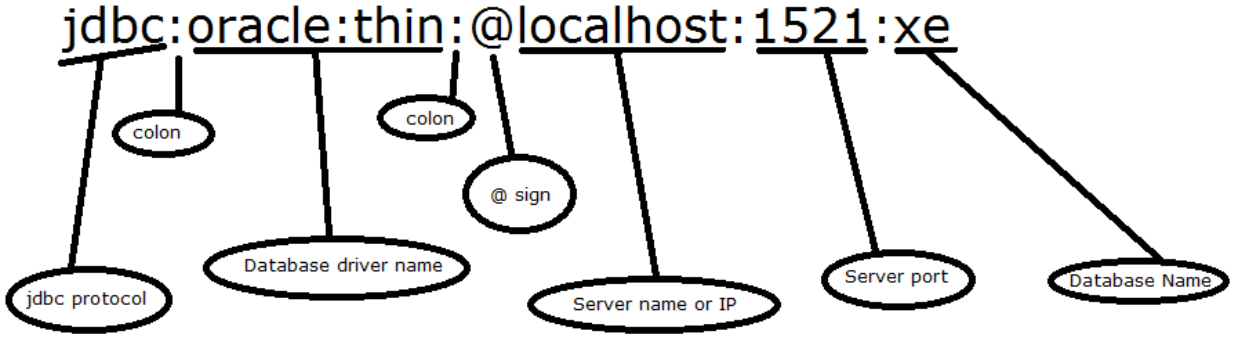
```
Connenction con = DriverManager
.getConenction(
"jdbc:mysql://hostname:3306/database"
, username, password);
```

Vytvorenie spojenia

MySQL



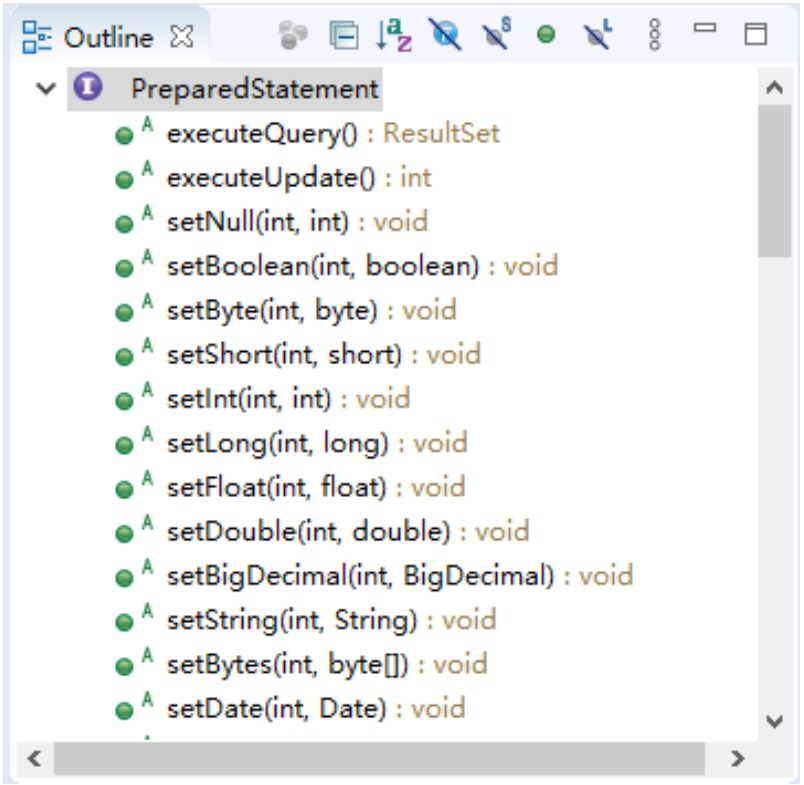
Oracle DB



3. Vytvorenie príkazu (Create statement)

- Po nadviazaní spojenia môže používateľ interagovať s databázou
- Rozhrania, ako sú príkazy JDBC, **PreparedStatement**, **CallableStatement**, poskytujú metódy, ktoré **umožňujú používateľovi odosielať príkazy SQL** a **získavať údaje** z **databázy**

```
Statement stmt =
con.createStatement();
```



PreparedStatement a CallableStatment

- ```
  Statement statement =
  connection.createStatement();
  ```

- ```
  PreparedStatement
  preparedStatement =
  connection.prepareStatement("INSE
  RT INTO EMPLOYEE(Id, Name)
  VALUES(?,?)");
  ```

- ```
  CallableStatement
  callableStatement =
  connection.prepareCall("{call
  Employee_procedure(?,?,?)}");
  ```

- ```
  CallableStatement
  callableStatement =
  connection.prepareCall("{ ? =
  call Employee_function(?,?,?)}");
  ```

4. Vykonanie dopytu (Executing queries)

- Dopyt SQL sa vykoná na interakciu s databázou
- **Dopyt** môže slúžiť **na aktualizáciu/vloženie** do databázy alebo na **získanie údajov**
- **Rozhranie Statement** poskytuje 2 metódy:

1. **executeQuery**() na **vykonávanie dopytov** na **získanie údajov**, zatiaľ čo
2. **executeUpdate**() na **vykonávanie dopytov** na **aktualizáciu** alebo **vkladanie (insert, update, delete)**

```
ResultSet resultSetObject =
stmt.executeQuery();
```

```
String query = "Update Employee SET
type= ? WHERE empId = ?";
PreparedStatement psObject =
connObj.prepareStatement(query);
ResultSet rsObject =
psObject.executeQuery();
```

```
stmt.executeUpdate("CREATE TABLE
users(username VARCHAR(50), password
VARCHAR(20));");
stmt.executeUpdate("INSERT INTO users
VALUES('Adam', 'TajN3*');");
ResultSet rs =
stmt.executeQuery("SELECT * FROM
users;");
```

4. Vykonanie dopytu (Executing queries)

Používanie Statement

Používanie PreparedStatement

```
Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery("SELECT * FROM product");
```

```
PreparedStatement pstmt =
   con.prepareStatement("INSERT INTO product values(?, ?)");
pstmt.setString(1, "mp3");
pstmt.setInt(2, 150);
pstmt.executeUpdate();
```

compact2, compact3
java.sql

Interface PreparedStatement

All Superinterfaces:

AutoCloseable, Statement, Wrapper

All Known Subinterfaces:

CallableStatement

```
public interface PreparedStatement
extends Statement
```

An object that represents a precompiled SQL statement.

A SQL statement is precompiled and stored in a `PreparedStatement` object. This object can then be used to efficiently execute this statement multiple times.

Note: The setter methods (`setShort`, `setString`, and so on) for setting IN parameter values must specify types that are compatible with the defined SQL type of the input parameter. For instance, if the IN parameter has SQL type `INTEGER`, then the method `setInt` should be used.

If arbitrary parameter type conversions are required, the method `setObject` should be used with a target SQL type.

In the following example of setting a parameter, `con` represents an active connection:

```
PreparedStatement pstmt = con.prepareStatement("UPDATE EMPLOYEES
                                SET SALARY = ? WHERE ID = ?");
pstmt.setBigDecimal(1, 153833.00)
pstmt.setInt(2, 110592)
```

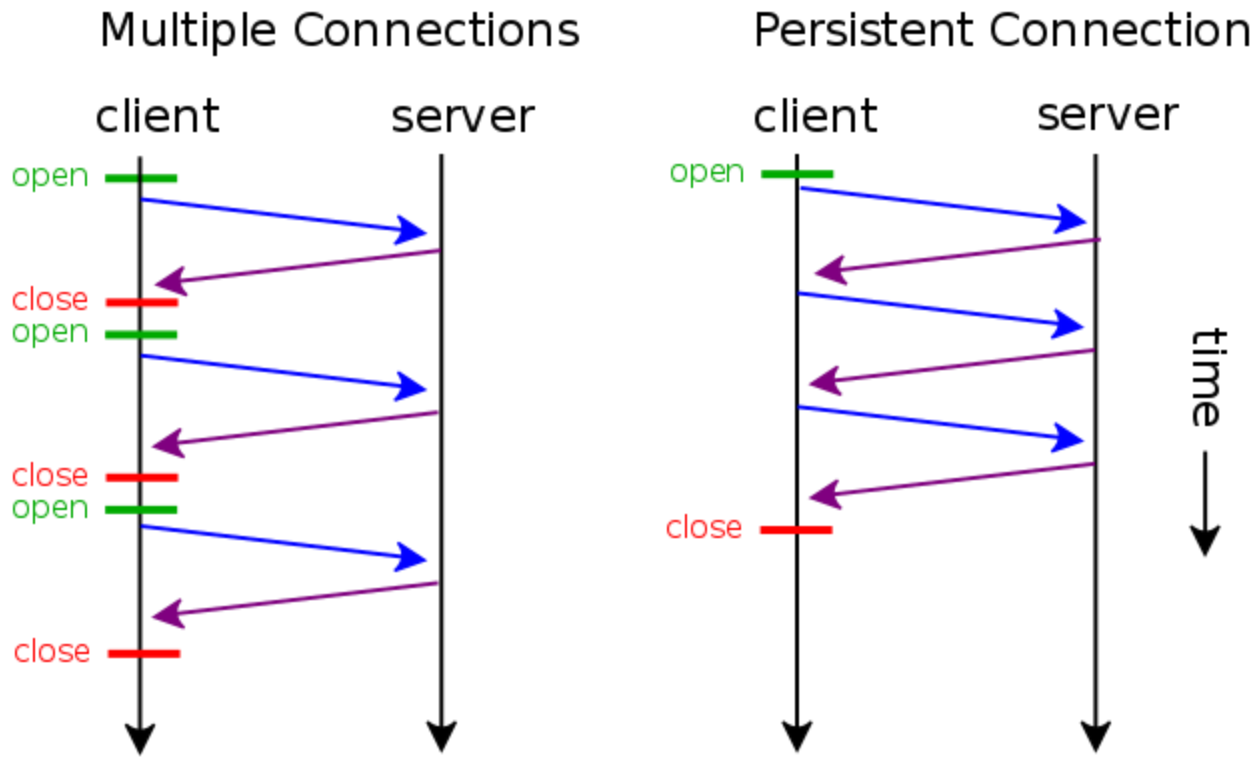
See Also:

Connection.prepareStatement(java.lang.String), ResultSet

5. Zatvorenie pripojenia (Close connect.)

- Po vykonaní nášho dopytu sa údaje, ktoré chcel používateľ aktualizovať alebo načítať, vykonali, takže teraz je čas zatvoriť vytvorené pripojenie
- **Rozhranie Connection** poskytuje metódu **close**() na **zatvorenie pripojenia**

```
connection.close();
con.close();
```



Zatvorenie pripojenia (Close connect.)

```
try {
    …
   con = DriverManager.getConnection( … );
   stmt = con.createStatement();
    …
} catch (Exception e) {
   e.printStackTrace();
} finally {
   try {
        if (stmt != null) stmt.close();
        if (con != null) con.close();
   } catch (Exception e) {}
}
```

```
try(Connection con = getConnection(url,
username, password, "org.postgresql.Driver");
    Statement stmt = con.createStatement();
    ResultSet rs = stmt.executeQuery(sql);
) {
    // Statements
}
catch(....){}
```

Objekty con, stmt a rs sa stanú súčasťou bloku try a Java tieto prostriedky po použití automaticky zatvorí

Zatvorenie pripojenia (Close connect.)

```
try (Connection connection = getDatabaseConnection(); Statement
statement = connection.createStatement()) {
    String sqlToExecute = "SELECT * FROM persons";
    try (ResultSet resultSet = statement.execute(sqlToExecute)) {
        if (resultSet.next()) {
            System.out.println(resultSet.getString("name");
        }
    }
} catch (SQLException e) {
    System.out.println("Failed to select persons.");
}
```

1: import Packages

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
```

2: Register Database Driver to JDBC

```
Class.forName("Database Driver");
```

3: Open Connection

```
Connection connection=DriverManager.getConnection("jdbc:Database url", "username", "password");
```

4: Create Statement

```
Statement smt = conn.createStatement();
```

5: Execute Query

```
ResultSet rs=smt.executeQuery("SQL statement");
```

6: Extract Database

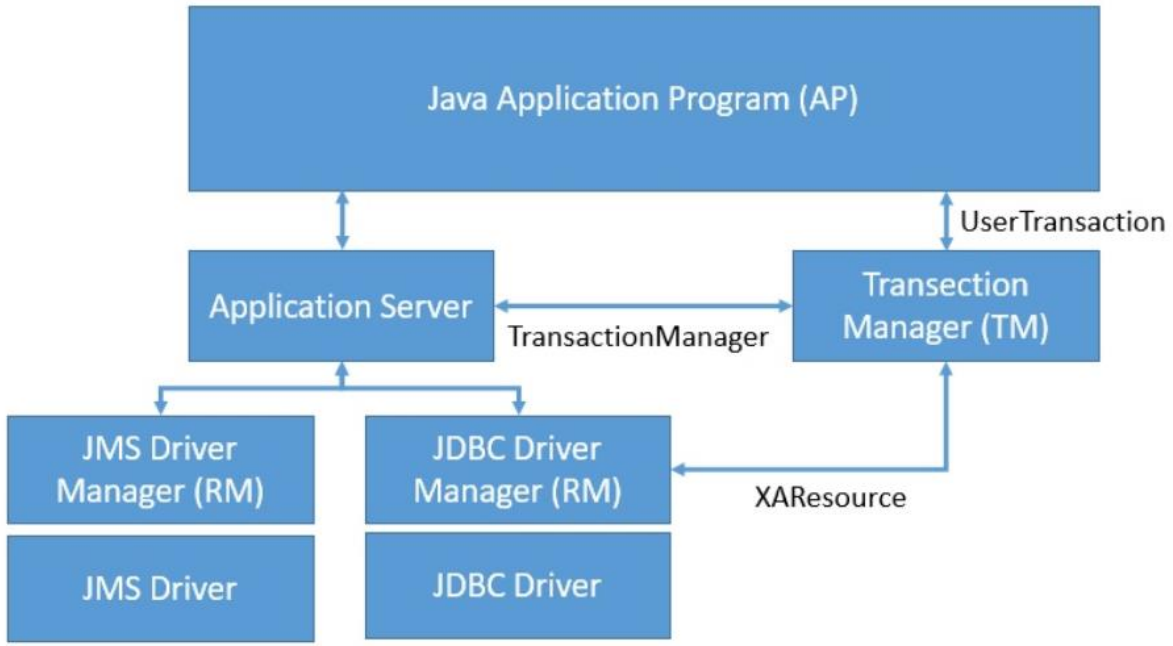
```
 rs.next()
 int i=rs.getInt(index);
 int i=rs.getInt(columnName);
```

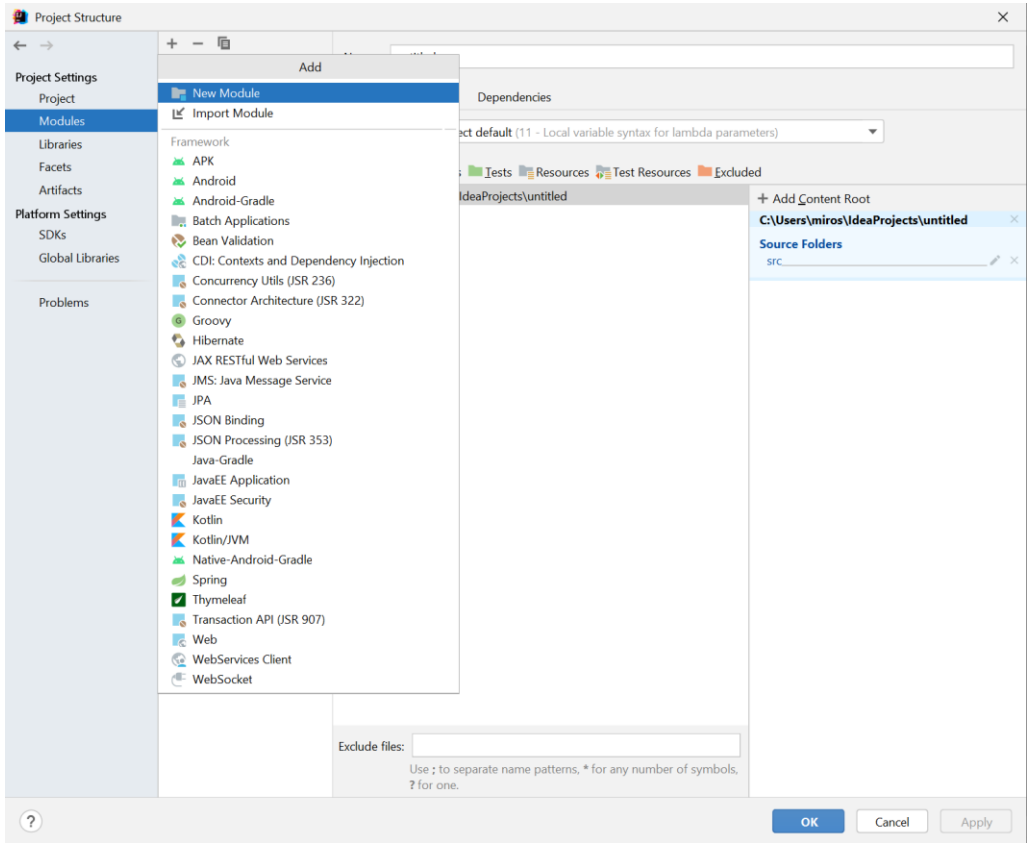
7: Close connection

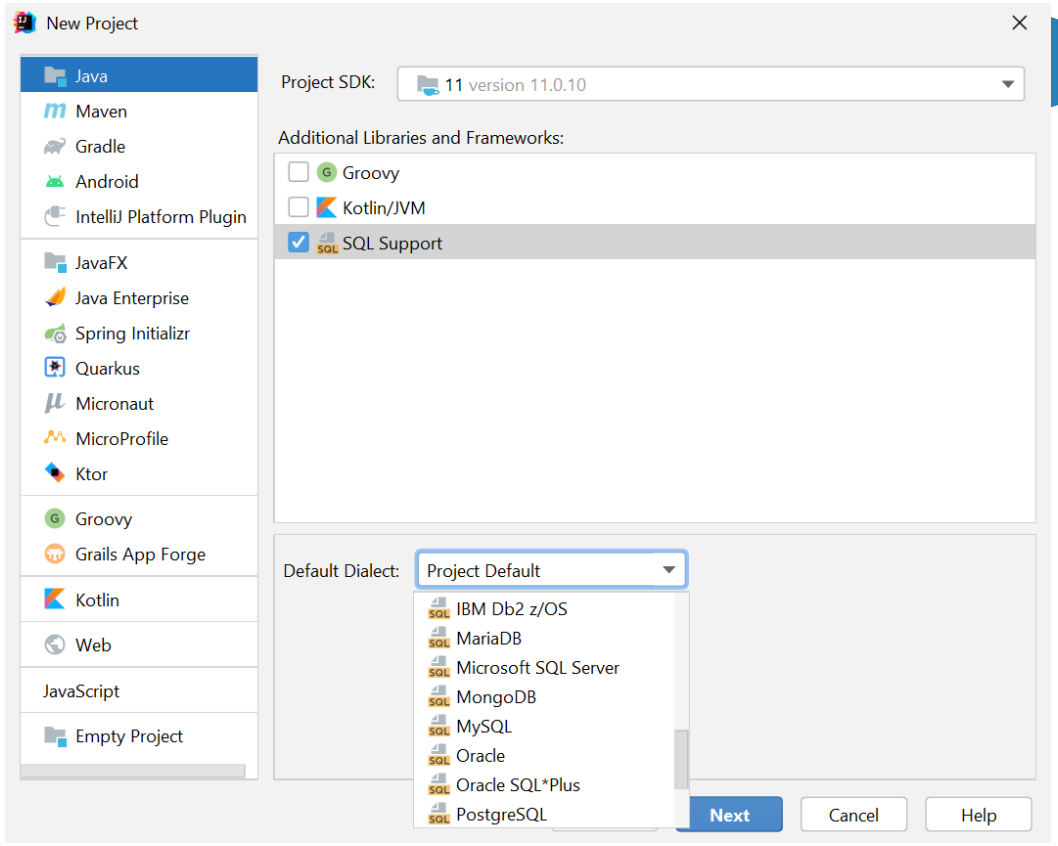
```
 connection.close()
```

Transakcie

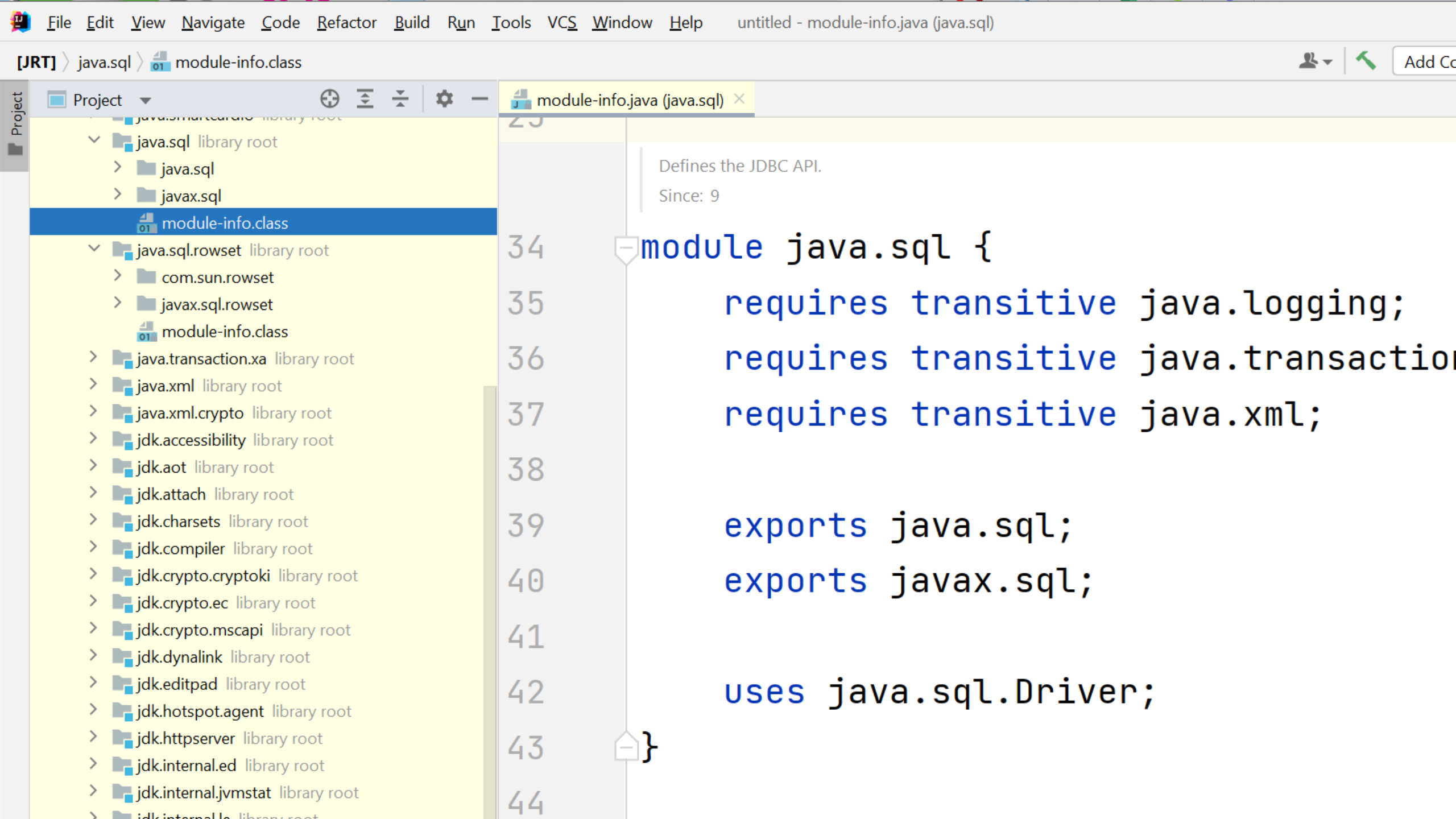
```
try {
     ...
    con = DriverManager.getConnection( … );
    con.setAutoCommit(false);

    stmt = con.createStatement();
    stmt.executeQuery( … );
    stmt.executeQuery( … );
    …
    conn.commit();
} catch (SQLException e) {
    conn.rollback();
    …
}
```

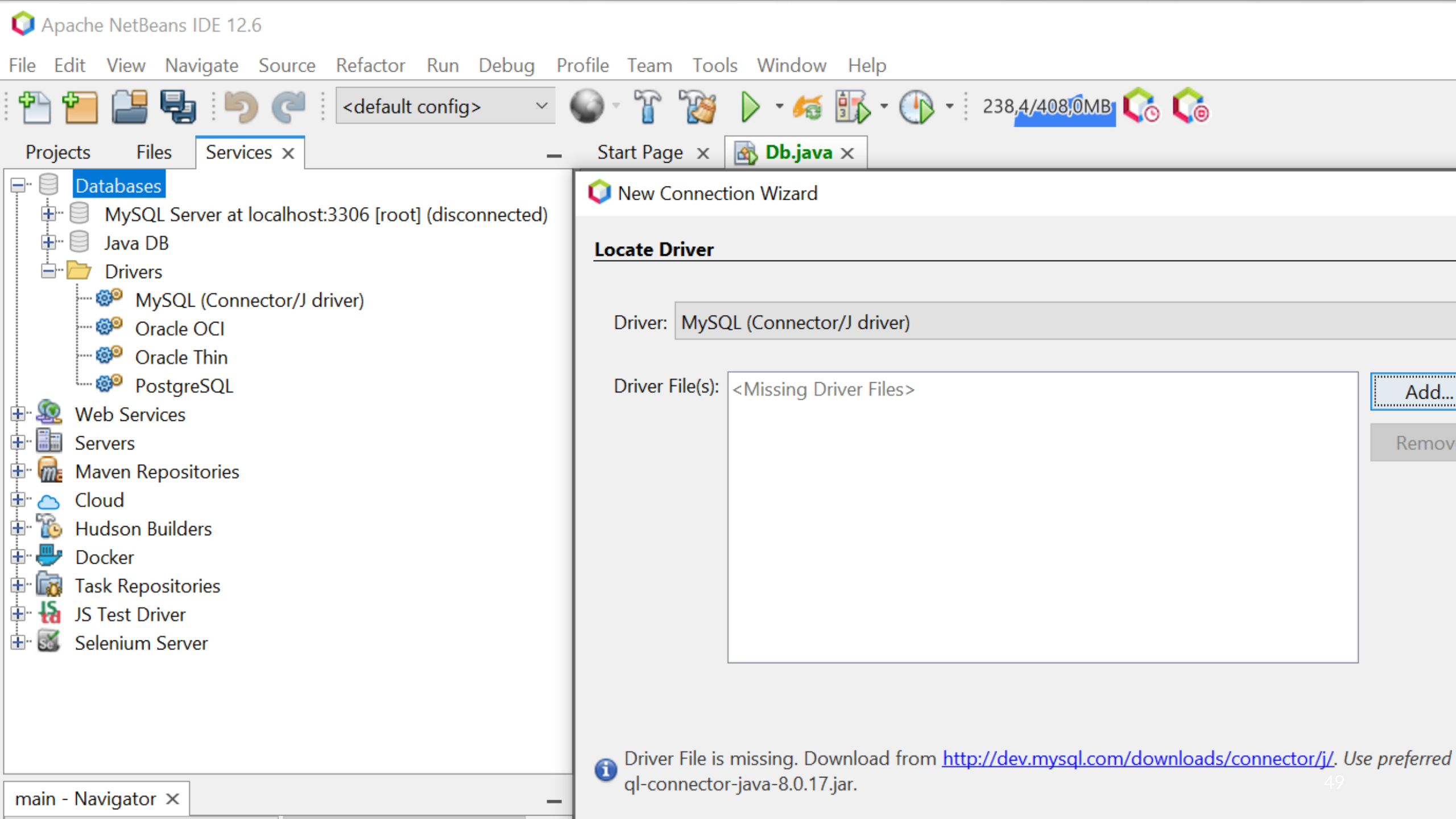






Podpora SQL a ORM


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
untitled - module-info.java (java.sql)
[JRT] java.sql module-info.class
Add Co
Project
module-info.java (java.sql)
java.sql library root
java.sql
javax.sql
module-info.class
java.sql.rowset library root
com.sun.rowset
javax.sql.rowset
module-info.class
java.transaction.xa library root
java.xml library root
java.xml.crypto library root
jdk.accessibility library root
jdk.aot library root
jdk.attach library root
jdk.charsets library root
jdk.compiler library root
jdk.crypto.cryptoki library root
jdk.crypto.ec library root
jdk.crypto.mscapi library root
jdk.dynalink library root
jdk.editpad library root
jdk.hotspot.agent library root
jdk.httpserver library root
jdk.internal.ed library root
jdk.internal.jvmstat library root
Defines the JDBC API.
Since: 9
34 module java.sql {
35 requires transitive java.logging;
36 requires transitive java.transactio
37 requires transitive java.xml;
38
39 exports java.sql;
40 exports javax.sql;
41
42 uses java.sql.Driver;
43 }
44


Apache NetBeans IDE 12.6
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
<default config>
238,4/408,0MB
Projects Files Services
Start Page
Db.java
Databases
MySQL Server at localhost:3306 [root] (disconnected)
Java DB
Drivers
MySQL (Connector/J driver)
Oracle OCI
Oracle Thin
PostgreSQL
Web Services
Servers
Maven Repositories
Cloud
Hudson Builders
Docker
Task Repositories
JS Test Driver
Selenium Server
main - Navigator
New Connection Wizard
Locate Driver
Driver: MySQL (Connector/J driver)
Driver File(s): <Missing Driver Files>
Add...
Remov
Driver File is missing. Download from http://dev.mysql.com/downloads/connector/j/. Use preferred
ql-connector-java-8.0.17.jar.

Čo si mám nainštalovať pre prácu s DB v Jave (JDBC)?

Výber nástrojov a DB

- **Microsoft Office**
  - Excel -> Access -> PowerBI -> SQL Server + OLAP + IS + BI + SharePoint + Windows Server + Azure
- **Google Suite**
  - Google Tabuľky (Spreadsheets) -> Google Cloud Platform + Google Analytics (Premium) + Google BigQuery + Google Kubernetes Engine + Cloud SQL (MySQL, PostgreSQL)
- **Apache OpenOffice/Libre Office**
  - R, Python, Java, JavaScript -> Apache Hadoop + MongoDB + PIG + HIVE + Spark + MySQL/MariaDB + Oracle DB + PostgreSQL + GIS
- **SAP**
  - SAP NetWeaver + Moduly + SAP Lumira + SAP Fiori + SAP Business One + SAP Hana
- **Amazon Web Service, Heroku, Firebase**
- **Vlastné (custom) riešenia**

Musí mať podporu JDBC

Čo budeme potrebovať

PostgreSQL

phpPgAdmin

ORACLE DATABASE

DG

DbVisualizer
The Universal Database Tool

MySQL

IT ACADEMY

TeamSQL

DBeaver

DBeaver Community

Free Universal Database Tool

Star 23,525 | Follow @dbeaver_news

search here … Go

Home | About | Download | Sources | Documentation | News | Support | Enterprise Edition | CloudBeaver

Download

Community Edition 21.3.0

Released on November 29, 2021 (Milestones).
It is free and open source (license).
Also you can get it from the GitHub mirror.

Enterprise Edition 21.2

Released on September 6, 2021
EE version web site: dbeaver.com
Trial version is available.

Windows

- **Windows 64 bit (installer)**
- Windows 64 bit (zip)

- Install from Microsoft Store
- Chocolatey (`choco install dbeaver`)

Mac OS X

Enterprise Edition features:

- Support of NoSQL databases:
  - MongoDB
  - Cassandra
  - InfluxDB
  - Redis
  - Amazon DynamoDB
  - Amazon DocumentDB
  - Amazon Keyspaces

15 April 2022

PostgreSQL JDBC Driver 42.3.4 Released

Notable changes

Changed

- fix: change name of build cache PR 2471
- feat: add support for ResultSet#getObject(OffsetTime.class) and PreparedStatement#setObject(OffsetTime.class) PR 2467
- fix: Use non-synchronized getTimeZone in TimestampUtils PR 2451
- docs: Fix CHANGELOG.md misformatted markdown headings PR 2461
- docs: remove loggerLevel and loggerFile from docs and issues #2489
- feat: use direct wire format -> LocalDate conversion without resorting to java.util.Date, java.util.Calendar, and default timezones PR 2464 fixes Issue #2221

Added

Fixed

- docs: Update testing documentation PR 2446
- fix: Throw an exception if the driver cannot parse the URL instead of returning NULL fixes Issue PR 2421 (#2441)
- fix: Use PGProperty instead of the property names directly PR 2444
- docs: update changelog, missing links at bottom and formatting PR 2460
- fix: Remove isDeprecated from PGProperty. It was originally intended to help produce automated docs. Fixes #Issue 2479 PR 2480
- fix: change PGInterval parseISO8601Format to support fractional second PR 2457
- fix: More test and fix for issues discovered by PR #2476 PR #2488

See full changelog for 42.3.4

---

15 February 2022

PostgreSQL JDBC Driver 42.3.3 Released

Notable changes

LATEST RELEASES

42.3.4 · 15 Apr 2022 · Notes
42.3.3 · 15 Feb 2022 · Notes
42.3.2 · 01 Feb 2022 · Notes
42.2.25 · 01 Feb 2022 · Notes
all · 22 Dec 2021 · Notes

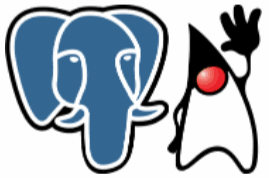
Downloads | Snapshots

SHORTCUTS

» GitHub project
» Documentation
» Mailing list
» Report a security issue
» Report a bug
» FAQ

SUPPORT US

PostgreSQL is free. Please support our work by making a donation.



SQL Tutorial

Learn SQL using: SQL Server, Oracle, MySQL, DB2, and PostgreSQL.

Language: English • 中文

Tutorials: Learn SQL in stages

0 SELECT basics

Some simple queries to get you started

1 SELECT name

Some pattern matching queries

2 SELECT from World

In which we query the World country profile table.

3 SELECT from Nobel

Additional practice of the basic features using a table of Nobel Prize winners.

4 SELECT within SELECT

In which we form queries using other queries.

5 SUM and COUNT

In which we apply aggregate functions. more the same

6 JOIN

In which we join two tables; game and goals. previously music tutorial

7 More JOIN operations

In which we join actors to movies in the Movie Database.

8 Using Null

In which we look at teachers in departments. previously Scottish Parliament

8+ Numeric Examples

In which we look at a survey and deal with some more complex calculations.

9 Self join

```sql
-- borrowed from https://stackoverflow.com/q/7745609/808921

CREATE TABLE IF NOT EXISTS `docs` (
  `id` int(6) unsigned NOT NULL,
  `rev` int(3) unsigned NOT NULL,
  `content` varchar(200) NOT NULL,
  PRIMARY KEY (`id`,`rev`)
) DEFAULT CHARSET=utf8;
INSERT INTO `docs` (`id`, `rev`, `content`) VALUES
  ('1', '1', 'The earth is flat'),
  ('2', '1', 'One hundred angels can dance on the head of a pin'),
  ('1', '2', 'The earth is flat and rests on a bull\'s horn'),
  ('1', '3', 'The earth is like a ball.');
```

Build Schema Edit Fullscreen Browser [;]

```sql
-- based on answer https://stackoverflow.com/a/7745635/808921

SELECT a.id, a.rev, a.content
FROM `docs` a
INNER JOIN (
    SELECT id, MAX(rev) rev
    FROM `docs`
    GROUP BY id
) b ON a.id = b.id AND a.rev = b.rev;


SELECT a.*
FROM `docs` a
LEFT OUTER JOIN `docs` b
    ON a.id = b.id AND a.rev < b.rev
WHERE b.id IS NULL;
```

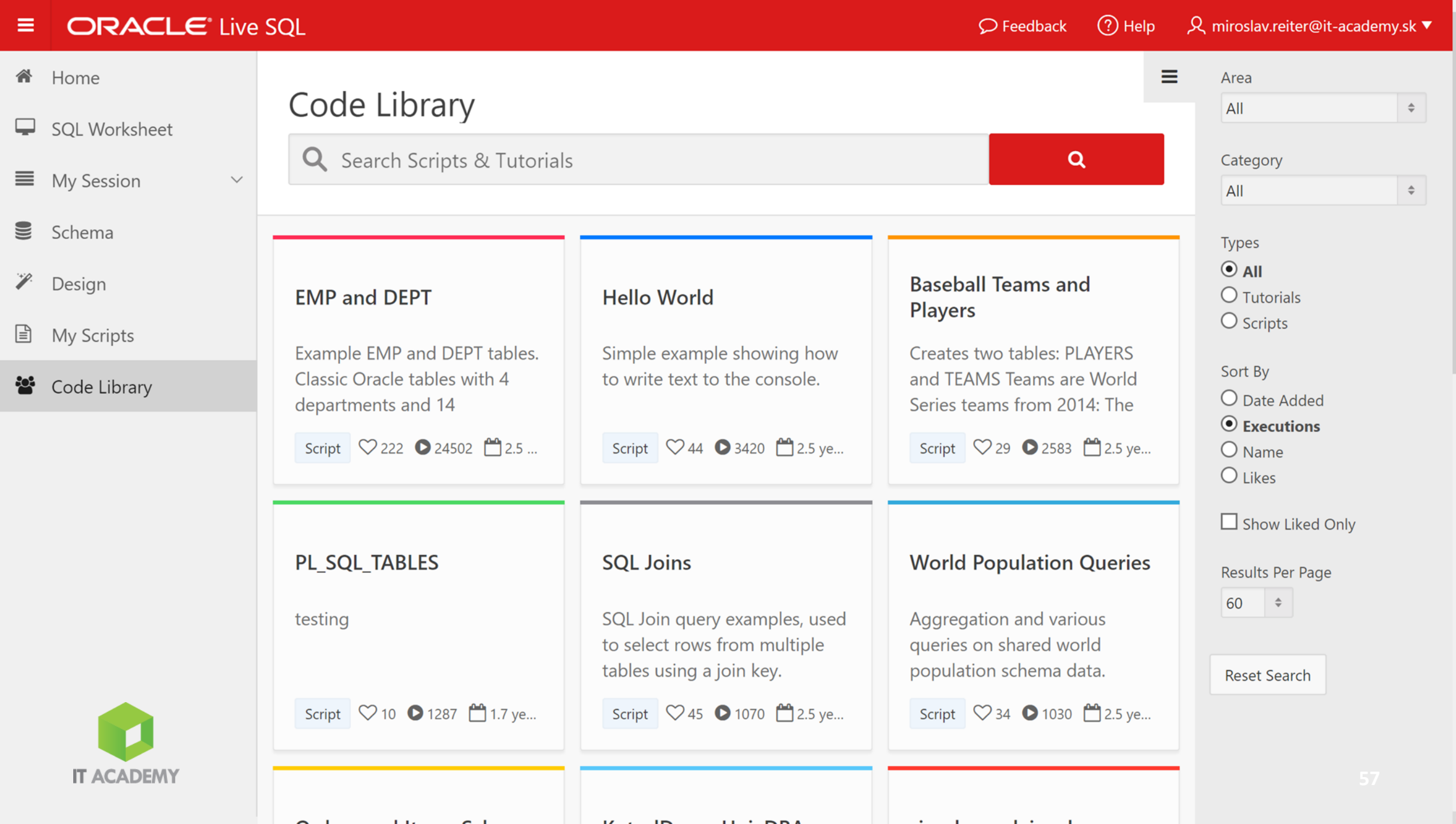
Run SQL Edit Fullscreen [;]

id	rev	content
1	3	The earth is like a ball.
2	1	One hundred angels can dance on the head of a pin

Record Count: 2; Execution Time: 4ms View Execution Plan link

id	rev	content
1	3	The earth is like a ball.
2	1	One hundred angels can dance on the head of a pin

Record Count: 2; Execution Time: 0ms View Execution Plan link


ORACLE Live SQL
Feedback
Help
miroslav.reiter@it-academy.sk
Home
SQL Worksheet
My Session
Schema
Design
My Scripts
Code Library
Code Library
Search Scripts & Tutorials
EMP and DEPT
Example EMP and DEPT tables. Classic Oracle tables with 4 departments and 14
Script 222 24502 2.5 ...
Hello World
Simple example showing how to write text to the console.
Script 44 3420 2.5 ye...
Baseball Teams and Players
Creates two tables: PLAYERS and TEAMS Teams are World Series teams from 2014: The
Script 29 2583 2.5 ye...
PL_SQL_TABLES
testing
Script 10 1287 1.7 ye...
SQL Joins
SQL Join query examples, used to select rows from multiple tables using a join key.
Script 45 1070 2.5 ye...
World Population Queries
Aggregation and various queries on shared world population schema data.
Script 34 1030 2.5 ye...
Area
All
Category
All
Types
All
Tutorials
Scripts
Sort By
Date Added
Executions
Name
Likes
Show Liked Only
Results Per Page
60
Reset Search
IT ACADEMY
57

Generate test data for your database

Quick recipes to test real applications with random data

Table Structure: User Account | Export Format: CSV | Generated rows: Number of entries

Use an existing data model and customize it to mimick your table structure or create one from scratch.

#	Column title	Data type	Delete
1	Id	Auto-increment	
2	Full Name	Full Name	
3	Country	Country	
4	Email	Email	
5	Created At	DateTime	

Add Another Column | Clear table | Load recipe | Save recipe | Generate data

Why do I need to fill a database with random data?

When developing an application, you would be wise to test it. You might test it for correctness and you might test it for load. Both of these situations benefit from having a large body of data that is semi-coherent (so you can kind of inspect it) but that is automatically generated. Manually inserting 3 or 4 rows in each table just isn't good enough. This is an attempt at making the problem smaller. Generating fixtures has never been easier.

IT ACADEMY

generatedata.com

Login | English

Generate | About | News | Donate

Your data set name here... SAVE

COUNTRY-SPECIFIC DATA

All countries

DATA SET

Order	Table Column	Data Type	Examples	Options	Help	Del
1		Select Data Type				
2		Select Data Type				
3		Select Data Type				
4		Select Data Type				
Order	Table Column	Data Type	Examples	Options	Help	Del

Add 1 Row(s)

EXPORT TYPES

CSV | Excel | HTML | JSON | LDIF | Programming Language | SQL | XML

- hide data format options

Database table name: myTable

Database Type: MySQL

Misc Options:
- Include CREATE TABLE query
- Include DROP TABLE query
- Enclose table and field names with backquotes

Statement Type: INSERT | INSERT IGNORE | UPDATE

INSERT batch size: 10

Primary Key:
- None
- Add default auto-increment column

Generate 100 rows | Generate in-page | New window/tab | Prompt to download | Zip? | Generate

MySQL Connector C++ 8.0	Oracle Corporation	16. 8. 2021	130 MB	8.0.26
MySQL Connector J	Oracle Corporation	16. 8. 2021	2,70 MB	8.0.26
MySQL Connector Net 8.0.26	Oracle	16. 8. 2021	84,6 MB	8.0.26
MySQL Connector Python v8.0.26	Oracle	16. 8. 2021	25,4 MB	8.0.26
MySQL Connector/ODBC 8.0	Oracle Corporation	16. 8. 2021	69,2 MB	8.0.26
MySQL Documents 5.7	Oracle Corporation	16. 8. 2021	72,6 MB	5.7.35
MySQL Examples and Samples 5.7	Oracle Corporation	16. 8. 2021	3,68 MB	5.7.35
MySQL for Visual Studio 1.2.9	Oracle	16. 8. 2021	19,9 MB	1.2.9
MySQL Router 8.0	Oracle Corporation	16. 8. 2021	148 MB	8.0.26
MySQL Shell 8.0.26	Oracle and/or its affiliates	16. 8. 2021	239 MB	8.0.26
MySQL Workbench 8.0 CE	Oracle Corporation	16. 8. 2021	143 MB	8.0.26
psqlODBC	PostgreSQL Global Development Gro...	21. 8. 2019	4,00 MB	08.02.0200
psqlODBC_x64	PostgreSQL Global Development Gro...	10. 3. 2022	7,69 MB	09.06.0504

Čo sa oplatí prečítať?

Slovensko a Česko

- Albatrosmedia
- Kopp
- Grada
- Wolters Kluwer
- BEN
- Veda

Zahraničie

- O'Reilly
- Manning
- Packt
- Apress
- Wiley
- No Starch Press

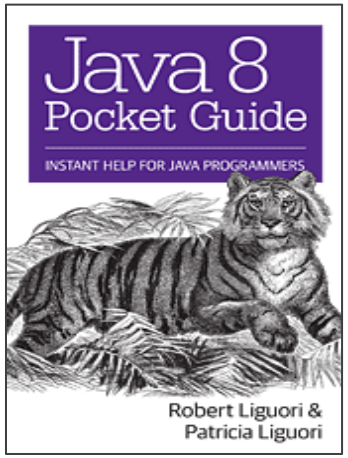
YouTube tutorialy

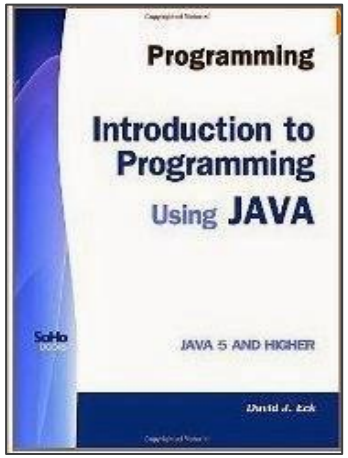
Packt Publishing

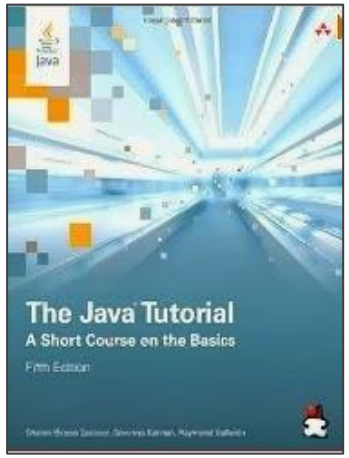
Čo sa oplatí/neoplatí prečítať SK/CZ

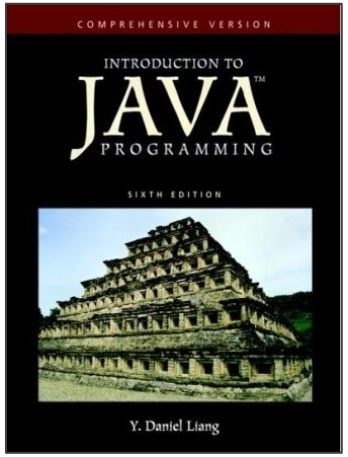
Mistrovství a Kuchárka

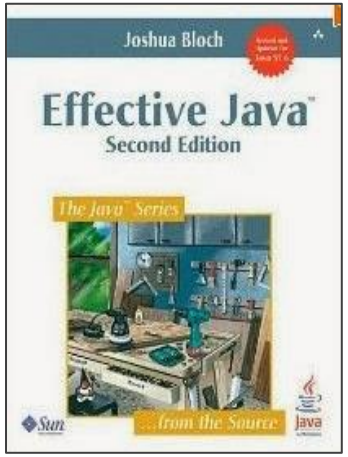
Čo sa oplatí/neoplatí prečítať EN

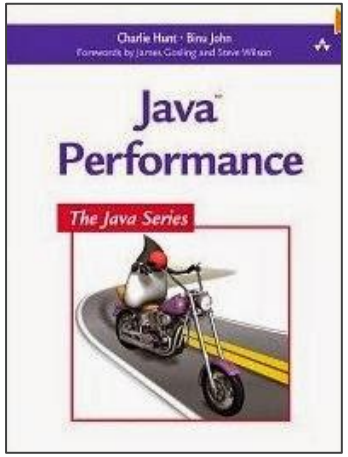


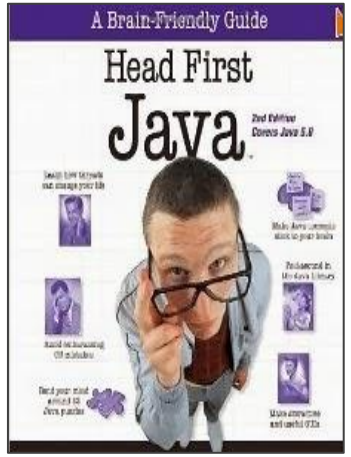












Head First

Vývojári

Miroslav | Domov | Vytvoriť

Vývojári

Verejná skupina

- Informácie
- **Diskusia**
- Oznámenia
- Členovia
- Podujatia
- Videá
- Fotky
- Súbory

Hľadať v tejto skupine

Skratky

- Podnikanie na Slove… 2
- UK Manazment Externe…
- Testovacia firma
- VITA - Virtual It Academy
- Startupisti 2
- Rubyslava 2
- Vývojári

Zobraziť viac

Ste člen | Upozornenia | Zdieľať | Viac

Napísať príspe… | Pridať fotku/vi… | Živé video | Viac

Napíšte niečo...

Fotka/video | Divácka párty | Označiť priat… | •••

NOVÁ AKTIVITA

Roland Mondek
10 h

POZVAŤ ČLENOV

Zadajte meno alebo e-mailovú adresu...

ČLENOVIA 5 505 členov

POPIS

Skupina softvérových vývojárov. Táto skupina by mala byť miestom... Zobraziť viac

TYP SKUPINY

Všeobecné

VAŠE STRÁNKY

- IT Academy
- VITA - Virtual It Academy

KONTAKTY

- Evka Rybárska
- Jarmila Palenčárová
- Stefan Orosi
- Ivana Ivka Jasaňov
  Hrá Word Blitz
- Ivana Pavlíková
- Martin Vanko
- Lucia Kovačičová 4 h
- Lošák Filip
- Andrej Nejedlik
- Gabika Zubrikova

SKUPINOVÉ KONVERZÁCIE

Vytvoriť novú skupinu

ĎALŠIE KONTAKTY (82)

Hľadať

Home

PUBLIC

Stack Overflow

Tags

Users

FIND A JOB

Jobs

Companies

TEAMS What's this?

Free 30 Day Trial

Tags

A tag is a keyword or label that categorizes your question with other, similar questions. Using the right tags makes it easier for others to find and answer your question.

Show all tag synonyms

tomcat

Popular Name New

tomcat

for questions about Apache Tomcat (or simply Tomcat, formerly also Jakarta Tomcat) which is an open source Servlet Container developed by the…

40936 questions — 11 asked today, 47 this week

tomcat7

Version 7.x (June 2010) of the Apache Tomcat servlet container. Use only if your question is specifically related to features of this version.

5541 questions — 10 asked this month, 143 this year

tomcat8

Version 8.x (August 2013 onwards) of the Apache Tomcat servlet container. Use only if your question is specifically related to features of this version.

2576 questions — 6 asked this week, 20 this month

tomcat6

Version 6.x (December 2006) of the Apache Tomcat servlet container. Use only if your question is specifically related to features of this version.

1877 questions — 11 asked this year

tomcat9

Version 9.x (August 2017 onwards) of the Apache Tomcat servlet container. Use only if your question is specifically related to features of this version.

784 questions — 8 asked this week, 32 this month

tomcat5.5

Version 5.5.x (August 2004) of the Apache Tomcat servlet container. Use only if your question is specifically related to features of this version.

252 questions — 4 asked this year

embedded-tomcat-8

Embedded Apache Tomcat 8

209 questions — 21 asked this year

maven-tomcat-plugin

The Tomcat Maven Plugin provides goals to manipulate WAR projects within the Tomcat servlet container.

202 questions — 2 asked this year

embedded-tomcat-7

Questions about running Apache

tomcat8.5

Version 8.5.x (June 2018 onwards) of

tomcat-valve

a type of component that can be

tomcat-jdbc

about tomcat and jdbc working

Mrkni na náš YouTube kanál a daj odber

➡ WWW.YOUTUBE.COM/C/IT-ACADEMYSK ⬅

IT ACADEMY