

# Connecting with Java

This guide explains how to establish a connection between a **Java** application and a **PostgreSQL** database using the **JDBC driver**. It walks through the necessary setup, configuration, and execution of a simple SQL query.

## Variables

Certain parameters must be provided to establish a successful connection to a PostgreSQL database. Below is a breakdown of each required variable, its purpose, and where to find it. Here’s what each variable represents:

Variable	Description	Purpose
USER	PostgreSQL username, from the Elestio service overview page	Identifies the database user who has permission to access the PostgreSQL database.
PASSWORD	PostgreSQL password, from the Elestio service overview page	The authentication key required for the specified USER to access the database
HOST	Hostname for PostgreSQL connection, from the Elestio service overview page	The address of the server hosting the PostgreSQL database.
PORT	Port for PostgreSQL connection, from the Elestio service overview page	The network port is used to connect to PostgreSQL. The default port is 5432 .
DATABASE	Database Name for PostgreSQL connection, from the Elestio service overview page	The name of the database being accessed. A PostgreSQL instance can contain multiple databases.

These values can usually be found in the Elestio service overview details, as shown in the image below. Make sure to take a copy of these details and add them to the code moving ahead.



postgresql-2p7j1

PostgreSQL

Running

Open terminal

Delete service

Clone this service

Overview

Tools

Backups

Metrics

Monitoring

Logs

Audit

Security

Alerts

Termination protection

Disabled. VM can be powered off and terminated.

Protection deactivated



Database Admin

Display your database credentials

Hide DB Credentials

Host postgresql-2p7j1-u7774.vm.elestio.app



Port 25432



User postgres



Password \*\*\*\*\*

Show password



CLI PGPASSWORD=\*\*\*\*\* psql --host=postgresql-2p7j1-u7774.vm.elestio.app --port=25432 --username=postgres

Show password



# Prerequisites

## Install Java & JDBC driver

Check if Java is installed by running:

```
java -version
```

If not installed, install it first and then download and install **JDBC** driver from

<https://jdbc.postgresql.org/download/> or if you have Maven installed, run the following command with updated version of the driver:

```
mvn org.apache.maven.plugins:maven-dependency-plugin:2.8:get -Dartifact=org.postgresql:postgresql:42.7.5:jar -Ddest=postgresql-42.7.5.jar
```

# Code

Once all prerequisites are set up, create a new file named `Pg.java` and add the following code:

```

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.HashMap;
import java.util.Map;

public class Pg {
    private static class ConnectionConfig {
        private final String host;
        private final String port;
        private final String database;
        private final String username;
        private final String password;

        public ConnectionConfig(String host, String port, String database, String username, String password) {
            this.host = host;
            this.port = port;
            this.database = database;
            this.username = username;
            this.password = password;
        }

        public String getConnectionUrl() {
            return String.format("jdbc:postgresql://%s:%s/%s?sslmode=require", host, port, database);
        }

        public boolean isValid() {
            return host != null && !host.isEmpty() &&
                port != null && !port.isEmpty() &&
                database != null && !database.isEmpty();
        }
    }

    private static Map<String, String> parseArguments(String[] args) {
        Map<String, String> config = new HashMap<>();
        for (int i = 0; i < args.length - 1; i++) {
            String key = args[i].toLowerCase();
            String value = args[++i];

```

```
        config.put(key, value);
    }
    return config;
}
```

```
private static ConnectionConfig createConfig(Map<String, String> args) {
    return new ConnectionConfig(
        args.get("-host"),
        args.get("-port"),
        args.get("-database"),
        args.get("-username"),
        args.get("-password")
    );
}
```

```
private static void validateConnection(Connection connection) throws SQLException {
    try (Statement stmt = connection.createStatement();
        ResultSet rs = stmt.executeQuery("SELECT version()")) {
        if (rs.next()) {
            System.out.println("Database Version: " + rs.getString("version"));
        }
    }
}
```

```
public static void main(String[] args) {
    try {
        // Load PostgreSQL driver
        Class.forName("org.postgresql.Driver");

        // Parse and validate configuration
        Map<String, String> parsedArgs = parseArguments(args);
        ConnectionConfig config = createConfig(parsedArgs);

        if (!config.isValid()) {
            System.err.println("Error: Missing required connection parameters (host, port, database)");
            return;
        }

        // Establish connection and validate
        try (Connection conn = DriverManager.getConnection(
```

```

        config.getConnectionUrl(),
        config.username,
        config.password)) {

        System.out.println("Successfully connected to the database!");
        validateConnection(conn);
    }

} catch (ClassNotFoundException e) {
    System.err.println("Error: PostgreSQL JDBC Driver not found");
    e.printStackTrace();
} catch (SQLException e) {
    System.err.println("Database connection error:");
    e.printStackTrace();
}
}
}
}

```

To execute the script, open the terminal or command prompt and navigate to the directory where `Pg.java`. Once in the correct directory, run the script with the command (Update the variables with actual values acquired from previous steps.

```

javac Pg.java && java -cp postgresql-42.7.5.jar:. Pg -host HOST -port PORT -database DATABASE -username
avnadmin -password PASSWORD

```

If the connection is successful, the terminal will display output similar to:

```

Successfully connected to the database!
Database Version: PostgreSQL 16.8 (Debian 16.8-1.pgdg120+1) on x86_64-pc-linux-gnu, compiled by gcc
(Debian 12.2.0-14) 12.2.0, 64-bit

```

---

Revision #1

Created 1 April 2025 06:11:51 by kaiwalya

Updated 1 April 2025 08:35:48 by kaiwalya