Satellite Database Management Class

# [Setting up the PostGreSQL Database](#_heading=h.dujnybz9phmm)

# [Install jdbc & JavaFX into your Project](#bookmark=id.mcubqpckk1qa)

# [Add new Satellites](#bookmark=id.dz57lv9yaoa5)

# [Retrieve Satellites by ID](#bookmark=id.356vq8laauhy)

# [Update Satellite Details](#bookmark=id.71m6hw3sspu6)

# [Delete Satellites by ID](#bookmark=id.vd5eli848hgq)

# [Get Satellite ID by Name](#bookmark=id.hyohotr3j82)

# [Get an Array of Satellite Names](#bookmark=id.l8n9tz3ygxiu)

# [Create Jar file](#_heading=h.p0vdkrz1fva)

# Setting up the PostGreSQL Database

1. Install PostGreSQL compatible with your OS and pgAdmin from:
   1. <https://www.postgresql.org/download/>
   2. <https://www.pgadmin.org/download/>

and run the following code from the query tool to create the database:

-- Database: SatelliteSimulator

-- DROP DATABASE IF EXISTS "SatelliteSimulator";

CREATE DATABASE "SatelliteSimulator"

WITH

OWNER = postgres

ENCODING = 'UTF8'

LC\_COLLATE = 'en-US'

LC\_CTYPE = 'en-US'

LOCALE\_PROVIDER = 'libc'

TABLESPACE = pg\_default

CONNECTION LIMIT = -1

IS\_TEMPLATE = False;

for OS users:

CREATE DATABASE "SatelliteSimulator"

WITH

OWNER = postgres

ENCODING = 'UTF8'

LOCALE\_PROVIDER = 'libc'

TABLESPACE = pg\_default

CONNECTION LIMIT = -1

IS\_TEMPLATE = False;

1. Create the tables by running the following SQL code:

-- Table: public.Satellite

-- DROP TABLE IF EXISTS public."Satellite";

CREATE TABLE IF NOT EXISTS public."Satellite"

(

"Satellite\_ID" serial primary key,

"SatelliteName" text COLLATE pg\_catalog."default",

"User\_ID" bigint,

"Mass" numeric(10,4),

"Altitude" numeric(10,4),

"Speed" numeric(10,4),

"Area" numeric(10,4)

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."Satellite"

OWNER to postgres;

-- Table: public.Users

-- DROP TABLE IF EXISTS public."Users";

CREATE TABLE IF NOT EXISTS public."Users"

(

"USER\_ID" serial primary key,

"Name" text COLLATE pg\_catalog."default" NOT NULL,

"Email" text COLLATE pg\_catalog."default" NOT NULL,

"Password" text COLLATE pg\_catalog."default" NOT NULL

)

TABLESPACE pg\_default;

ALTER TABLE IF EXISTS public."Users"

OWNER to postgres;

1. To add add a user run the follow sql code in the the Query Tool in pgAdmin:

Create user Satellite\_User with password '123456789'

– Next:

GRANT USAGE, SELECT ON SEQUENCE "Satellite\_Satellite\_ID\_seq" TO Satellite\_User;

GRANT CONNECT ON DATABASE "SatelliteSimulator" TO Satellite\_User;

GRANT USAGE ON SCHEMA public TO Satellite\_User;

GRANT SELECT, INSERT, UPDATE, DELETE ON ALL TABLES IN SCHEMA public TO Satellite\_User;

ALTER DEFAULT PRIVILEGES IN SCHEMA public

GRANT SELECT, INSERT, UPDATE, DELETE ON TABLES TO Satellite\_User;

Run this command next:

Create extension pgcrypto;

Insert into the user table a default user:

INSERT INTO "Users" ("Name","Email", "Password")

VALUES ('Default\_User','nowhere@mail', crypt('sfhot45822nRFe3554?#$d', gen\_salt('bf')));

To see User Table run:

Select \* from "Users"

# Install jdbc & JavaFX into your project

1. Download the jdbc driver from :

<https://learn.microsoft.com/en-us/sql/connect/jdbc/download-microsoft-jdbc-driver-for-sql-server?view=sql-server-ver16>

&

<https://jdbc.postgresql.org/download/> (Java 8)

1. Download JavaFX from Oracle:

<https://www.oracle.com/java/technologies/install-javafx-sdk.html> (Please note VSCode will not work with JavaFX)

1. In your IDE make a reference in the class path for the jdbc Jar file. In IntelliJ IDEA go to file> Project Structure,
2. Click on the first upper left “+” symbol and select Java then browser for the jdbc jar file, postgres jar, JavaFX jar files and select it then select OK.
3. To use Maven ad the following code to your pom.xml file

<dependency>

<groupId>org.postgresql</groupId>

<artifactId>postgresql</artifactId>

<version>42.5.0</version>

</dependency>

1. Be sure to have install postgressql on your Desktop
   1. <https://www.postgresql.org/download/windows/> (Windows)
   2. For OS users in your terminal window run: brew install postgresql
2. Next set up your Datasource on your IDEA:
   1. Open IntelliJ IDEA
      1. Go to View > Tool Windows > Database
      2. In the Database window, click the + ➜ Data Source > PostgreSQL
      3. Fill in the following:
         1. Name: SatelliteSimulator@localhost
         2. Host: localhost
         3. Port: 5432 (default for PostgreSQL)
         4. Database: SatelliteSimulator
         5. User: postgres
         6. Password: your DB password (the one you personnally created on postgres account)
         7. IntelliJ may prompt you to Download Driver — accept it
         8. Click Test Connection
         9. Once successful, click OK
   2. Open Eclipse:
      1. Install the DBeaver Plugin:
         1. In Eclipse, go to:  
             Help > Eclipse Marketplace
         2. Search for "DBeaver"
         3. Click Install and follow the steps to restart Eclipse
      2. Open the DBeaver Database Perspective:
         1. Go to:  
             Window > Perspective > Open Perspective > Other…
         2. Select Database Development or DBeaver
      3. Create a New Database Connection:
         1. In the Database Navigator, right-click and choose  
             New Database Connection
         2. Select PostgreSQL (or your DBMS) from the list
         3. Click Next
      4. Fill in Connection Details:
         1. Host: localhost
         2. Port: 5432 (for PostgreSQL)
         3. Database: SatelliteSimulator
         4. User: postgres
         5. Password: your password (the one you personnally created on postgres account)
         6. Click Test Connection
            1. If the driver is not installed, DBeaver will prompt you to download the JDBC driver — approve it
         7. Click Finish
      5. Now you'll see your database in the Database Navigator
3. Set your connection in the SatelliteDatabaseManager class:

private static final String URL = "jdbc:postgresql://localhost:5432/SatelliteSimulator";

private static final String USER = " satellite\_user ";

private static final String PASSWORD = "your\_password";

# Add new Satellites

1. Initialize the SatelliteDatabaseManager class:

SatelliteDataBaseManager manager = new SatelliteDataBaseManager();

1. Add a Satellite:

Satellite satellite = new Satellite(1,"spy",11110.5,4.7,547.12);

try {

manager.addSatellite(satellite);

System.out.println("Satellite added successfully!");

} catch (ValidationError | DatabaseError e) {

System.err.println(e.toString());

}

# Retrieve Satellites by ID

1. Initialize the SatelliteDatabaseManager class:

SatelliteDataBaseManager manager = new SatelliteDataBaseManager();

1. Retrieve Satellite from database by its Satellite\_ID:

try {

Satellite satellite = manager.getSatelliteById(1);

System.out.println("Satellite Name: " + satellite.getSatelliteName());

} catch (DatabaseError e) {

System.err.println(e.toString());

}

# Update Satellite Details

1. Initialize the SatelliteDatabaseManager class:

SatelliteDataBaseManager manager = new SatelliteDataBaseManager();

1. To update an existing Satellite

Satellite satellite = new Satellite(1,"spy",12000.5,5.5,600.0);

satellite.setSatelliteID(1); // Existing ID

try {

manager.updateSatellite(satellite);

System.out.println("Satellite updated successfully!");

} catch (ValidationError | DatabaseError e) {

System.err.println(e.toString());

}

# Delete Satellites by ID

1. Initialize the SatelliteDatabaseManager class:

SatelliteDataBaseManager manager = new SatelliteDataBaseManager();

1. To delete a satellite by it’s IS call deleteSatelliteByID()

try {

manager.deleteSatelliteById(1); // Existing ID

System.out.println("Satellite deleted successfully!");

} catch (DatabaseError e) {

System.err.println(e.toString());

}

# Get Satellite ID by Name

1. Use method getSatelliteIdByName(“<StringName>”) example:

try {

SatelliteDataBaseManager manager = new SatelliteDataBaseManager();

int satelliteId = manager.getSatelliteIdByName("Hubble");

System.out.println("Satellite ID: " + satelliteId);

} catch (DatabaseError e) {

System.err.println(e.toString());

}

# Get an Array of Satellite Names

1. Use getSatelliteNamesAndIds() example:

try {

SatelliteDataBaseManager manager = new SatelliteDataBaseManager();

String[] satelliteNames = manager.getSatelliteNamesAndIds();

for (int i = 0; i < satelliteNames.length; i++) {

if (satelliteNames[i] != null) {

System.out.println("ID: " + i + ", Name: " + satelliteNames[i]);

}

}

} catch (DatabaseError e) {

System.err.println(e.toString());

}

# Create Jar file

1. Install maven from:

<https://maven.apache.org/download.cgi> (download the Binary tar.gz archive under LINK)

2. Once you've saved the file to your desktop, extract or unzip it.

3. Collect your downloaded Maven file path (ie. /Users/yourusername/Downloads/apache-maven-3.9.6/)

4. Open the terminal window and perform the following per OS:

For mac OS:

* First type:

nano ~/.zshrc

* Then:

export MAVEN\_HOME="Users/yourusername/Downloads/apache-maven-3.9.6"

export PATH="$MAVEN\_HOME/bin:$PATH"

* Then save (CTRL+X, then Y, then Enter), and apply it:
* Next type:

source ~/.zshrc

* Finally:
  + mvn -v

For Windows:

**a. Add MAVEN\_HOME:**

* Open **System Properties** → Advanced → Environment Variables.
* Under **System Variables**, click **New**:  
  + 1. **Name:** MAVEN\_HOME
    2. **Value:** C:\Program Files\Apache\Maven\apache-maven-3.9.9

**b. Add to PATH:** Edit the Path variable:

* Add: C:\Program Files\Apache\Maven\apache-maven-3.9.9\bin

3. Make sure your project contains the pom.xml file

4. execute: in IDE command window(Change path locations accordingly) :

mvn clean package

5. After running the command above, the jar name and location with be displayed, then execute

jar cfm <Path><filename>.jar manifest.txt -C src/ .

for example :

jar cfm C:\Users\rgord\IdeaProjects\LEOSatellite\_Simulator\target\LEOSatelliteSimulator-1.0-SNAPSHOT.jar manifest.txt -C src/ .

6. test by running:

java --module-path "<java Lib>" --add-modules javafx.controls,javafx.fxml -jar <Path><filename>.jar

For example:

java --module-path "C:\Program Files\openjfx-24\_windows-x64\_bin-sdk\javafx-sdk-24\lib" --add-modules javafx.controls,javafx.fxml -jar C:\Users\rgord\IdeaProjects\LEOSatellite\_Simulator\target\LEOSatelliteSimulator-1.0-SNAPSHOT.jar