DAISI

Driving Assistance Interface for Simulated Interventions

User Guide

Universiteit Utrecht

Contents

Chapter 1

Setup

1.1 Connecting the hardware

The steering wheel has three parts: The wheel itself, the pedals and the shifter. For our purposes, only the pedal and steering wheel are necessary. To connect the steering wheel you need to:

1. Connect the pedals to the steering wheel. (See figure 1.1.)
   (a) plug the cable of the pedals in the connector on the underside of the steering wheel (connector with the pedal icon).
   (b) place the pedals on the ground in reach for the participant.
   (c) to avoid the pedals sliding around, something might need to be placed behind the pedals.
2. Connect the power supply to the steering wheel. (See figure 1.2.)
   (a) plug the power supply cable in the connector on the underside of the steering wheel.
3. Mount the steering wheel to a sturdy surface.
   (a) Screw the 2 knobs on top of the steering wheel loose (counter-clockwise) until it can fit on the edge of the sturdy surface.
   (b) mount it on the edge of the sturdy surface
   (c) Tighten the 2 knobs until they are snug (clockwise).
4. Plug in the USB-cable and plug the power supply in the outlet.
   (a) On startup the steering wheel will turn completely to the left and right, so be prepared for that.

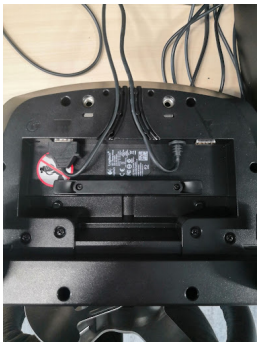

Figure 1.1: Connecting the pedals

Figure 1.2: Connecting the power supply

1.2 Windows

1.2.1 Installing DAISI

In this section we will explain how to install the DAISI software.

1. Go to `https://github.com/red-panda-productions/DAISI/releases/latest`.
2. Download `DAISI-X.X.X-win32-Release.exe` (see figure 1.3.
3. Run the installer executable.
   (a) Windows may give a Defender SmartScreen popup blocking the installer, if this happens click *More info* (figure 1.4) and then *Run anyway* (figure 1.5)

(b) Windows may give a "User Account Control" popup, click *yes* to give the installer permissions.

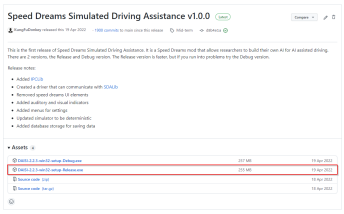

Figure 1.3: Download page for version 1.0.0

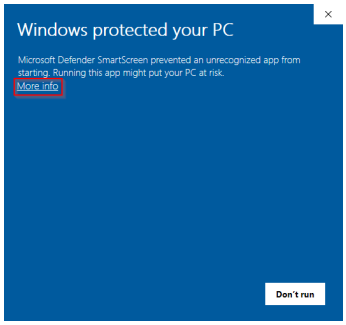

Figure 1.4: Microsoft Defender SmartScreen popup

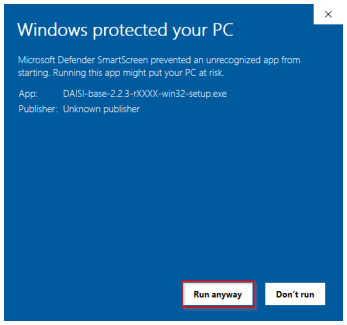

Figure 1.5: Microsoft Defender SmartScreen popup with more info

4. Click *Next*.
5. Check *I accept the terms of the License Agreement* and press *Next* (figure 1.6).
6. Check that the *Base System* is selected and press *Next* (figure 1.7).

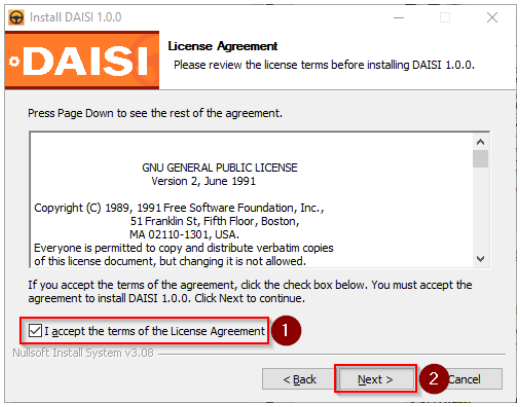

Figure 1.6: License Screen

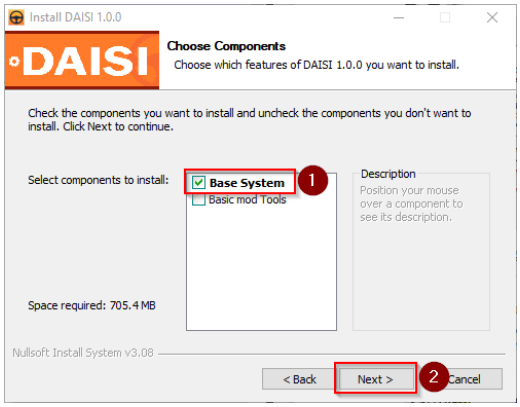

Figure 1.7: Component Selection Screen

7. Choose the directory to install DAISI to. The default is `C:\Program Files (x86)\DAISI-2.2.3` (figure 1.8).
8. Press *Install*
9. Wait for the installation to finish.
10. Press *Finish*, optionally checking *Run DAISI `X.X.X`* to automatically start DAISI after the installer is closed or *Show Readme* to open the readme in your preferred text editor (figure 1.9).

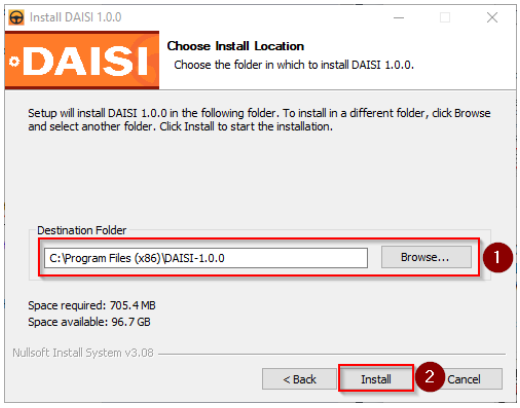

Figure 1.8: Install Location Screen

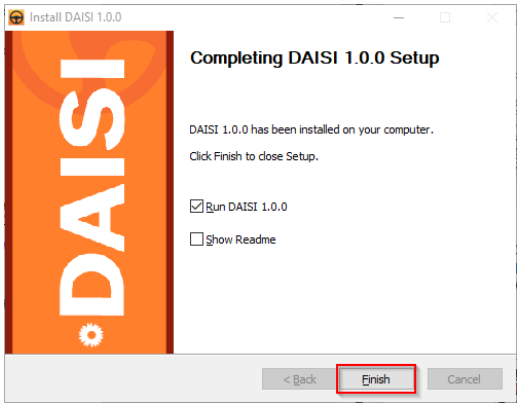

Figure 1.9: Finished Installation Screen

1.2.2 Installing DBeaver

To be able to use DBeaver, you need to have OpenJDK installed on your system, otherwise it will not be able to run. To install OpenJDK, follow the following steps:

1. Go to https://adoptium.net/.
2. Download the latest release version (see Figure 1.10).
3. Open the `OpenJDK{version}.msi` installer, where version is the OpenJDK version you downloaded (see Figure 1.11).
4. Click on *Next*.
5. Make sure that in the next window, you select `"set JAVA_HOME variable"` to `"will be installed on local hard drive"` (see Figure 1.12).
6. Click on *Next*.
7. Click on *Install*.
8. After the installation is done, click on *Finish*.

Figure 1.10: OpenJDK latest release

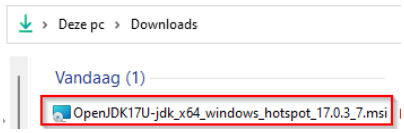

Figure 1.11: OpenJDK installer

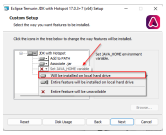

Figure 1.12: set JAVA_HOME

From here, you can start installing DBeaver

1. Go to https://github.com/red-panda-productions/dbeaver/releases.
2. Click on DBeaver_installer.exe to download the .exe file (see Figure 1.13).
3. If asked by your system, choose the desired directory and click *Save* (see Figure 1.14).
4. Alternatively, if the download gives a warning that the installer can be dangerous, press the arrow next to the warning and press *Keep* (see Figure 1.15).
5. Find the DBeaver_installer.exe in the saved directory and double click the .exe file (see Figure 1.16).

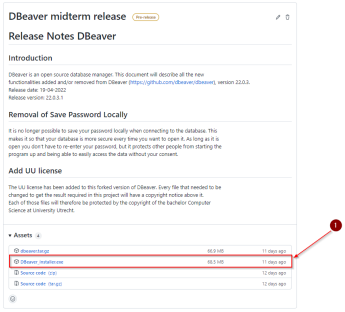

Figure 1.13: DBeaver Release page

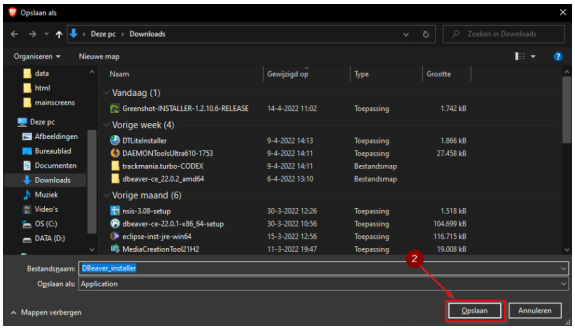

Figure 1.14: Save in windows directory

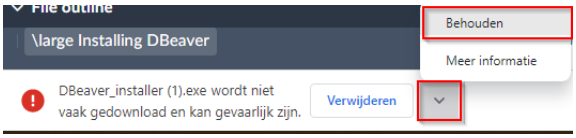

Figure 1.15: Allow DBeaver to be installed

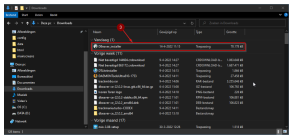

Figure 1.16: DBeaver.exe file

6. Windows will probably warn you that it does not trust the installer. Click on *More info* and press *Run anyway* (see Figures 1.17 and 1.18).

7. Windows will now ask if the installer is allowed to make changes to your system, press yes (see Figure 1.19).

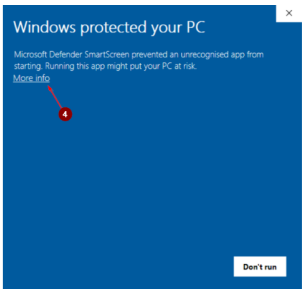

Figure 1.17: Press More info

Figure 1.18: Press run anyway

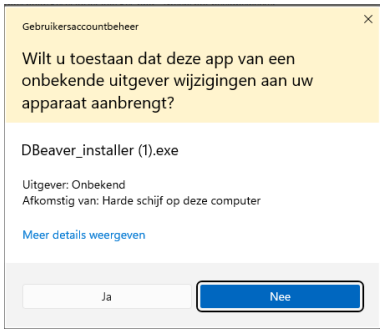

Figure 1.19: Allow it to make changes on your system

8. In *Destination Folder*, set your desired installation path and press *Install* (see Figure 1.20).

9. Wait until the installation is finished, then click *Close* (see Figure 1.21).

10. Go to the directory where you have installed DBeaver.

11. Optional: right click the .exe file and create a shortcut to your desktop (or somewhere else) for easy access.

12. Double click dbeaver.exe to run DBeaver. (see Figure 1.22).

13. DBeaver will now start up and you will be greeted with a popup that asks you if you want to create a sample database. You can decide if you want to do that or not. From there, you can start working with DBeaver.

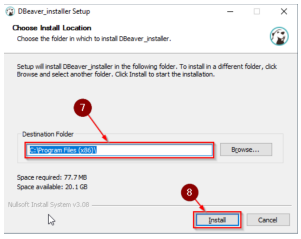

Figure 1.20: Set desired installation path

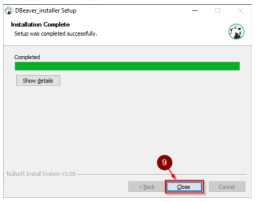

Figure 1.21: Close the installer

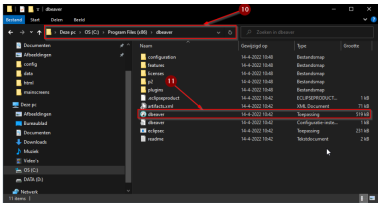

Figure 1.22: DBeaver.exe

1.2.3 Connecting Steering Wheel

Prerequisites

1. Steering wheel hardware is set up (see section 1.1).

Downloading the Driver

Although the steering wheel might work without a driver on Windows, a driver needs to be installed to guarantee the proper functionalities. If you are using the "Logitech G27 Racing Wheel" on Windows 10, you can follow the next part of the guide. To connect the steering wheel on Linux, you can follow the guide in section 1.3.3.

1. For the "Logitech G27 Racing Wheel" driver go to: https://support.logi.com/hc/en-hk/articles/360024155954 or lookup "Logitech G27 driver" (see figure 1.23).
2. Click on *Downloads* (1), choose the correct version of Windows that you own (2) and download the driver installer (3) (see figure 1.24).
3. After opening the installer you will be greeted with the screen shown in figure 1.25.
4. Click on *next* (1), accept the license agreement (2) and click on *install* (3).

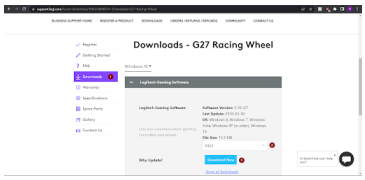

Figure 1.23: Driver website homescreen

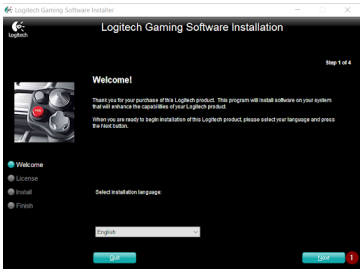

Figure 1.24: Driver installer screen

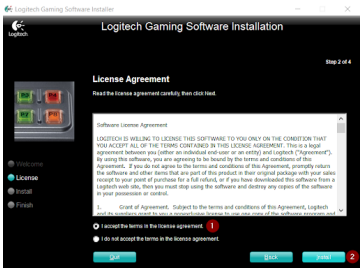

Figure 1.25: Driver installer license agreement

5. The driver will now install. Continue to follow the Logitech Gaming Software setup wizard, shown in figure 1.26
6. After following the setup wizard, you will be greeted with the screen shown in figure 1.27. In this screen you can test the functionality of the steering wheel and pedals to make sure they work:
   (a) Test the steering wheel and pedals.
   (b) Click **settings** (1) (see figure 1.27).
   (c) Set degrees of rotation to 900° (1) (see figure 1.28).
   (d) Enable Centering Spring (2).
   (e) Close (3).
7. The steering wheel should now correctly be connected to your computer.

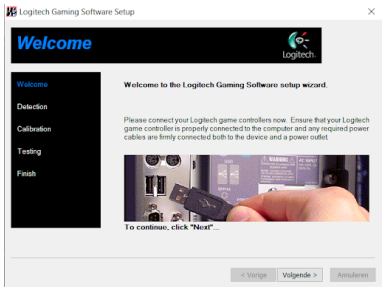

Figure 1.26: Driver start screen

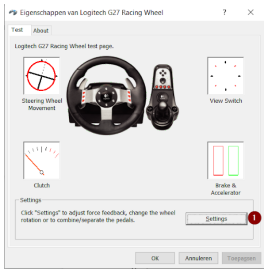

Figure 1.27: Driver functionalities

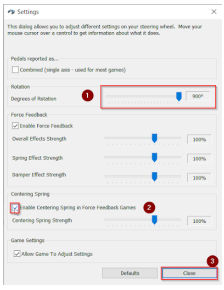

Figure 1.28: Driver settings screen

1.2.4 Setup Database

This guide will show you how to set up a MySQL database on a Windows machine. The first part of the guide will show how to set up an internal database (only accessible from your local machine). The second part of the guide will show you how to set up an external database (accessible from other devices than just your local machine).

Internal Database

To set up an internal database on your Windows machine, first download the MySQL installer for Windows(x86, 32-bit), MSI Installer (mysql-installer-web-community-8.0.28.0.msi) from `https://dev.mysql.com/downloads/installer/` (1). You will be sent to a page suggesting you create an account. Unless you want to create an account, click on *No thanks, just start my download.* (2).

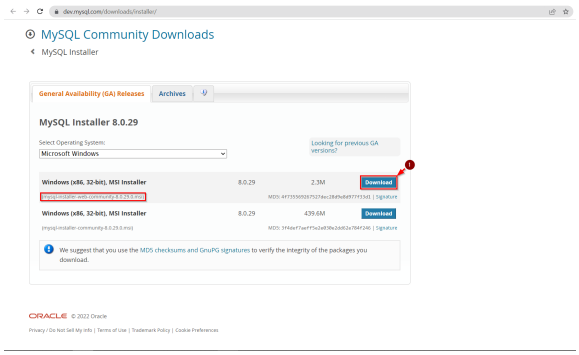

Figure 1.29: MySQL download page

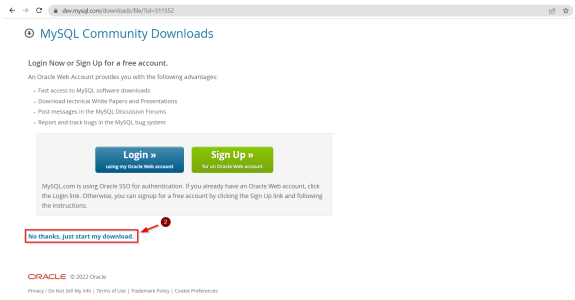

Figure 1.30: MySQL log-in page

Start the installer by opening the downloaded file. If Windows asks for permission to install the files, make sure to click *Yes* or *Allow*.

When the MySQL installer asks for a setup type, choose the *Server only* setup type (3) and click *Next* (4). If everything goes correctly, you will see an overview of the selected downloads which should only include the latest version of MySQL Server. If this is the case, click *Execute* (5) and wait for installation to finish, then click *Next* (6).

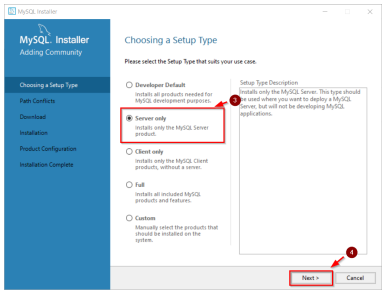

Figure 1.31: MySQL *Choose Setup Type* page

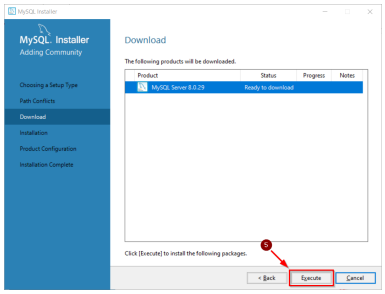

Figure 1.32: MySQL *Download* page before download

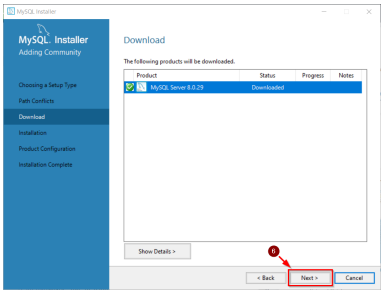

Figure 1.33: MySQL *Download* page after download

In the *Type and Networking* menu, select the *Development Computer* Config Type (7) and click *Next* (8). In the *Authentication Method* menu, ensure strong password encryption is selected (9) and click *Next* (10). In the *Accounts and Roles* menu, you will be prompted for a root password. Come up with a password and make sure to remember it before clicking *Next*. On the *Windows Service* menu, continue with the default values by pressing *Next*. On the *Apply Configuration* menu, click *Execute* and wait for configuration to finish before clicking *Finish*. Finally, you should see the *Installation Complete* screen, where you can click *Finish* to finish and close the installer (11). This finishes the installation for an internal database.

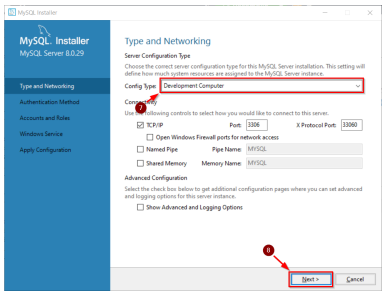

Figure 1.34: MySQL Type and Networking page

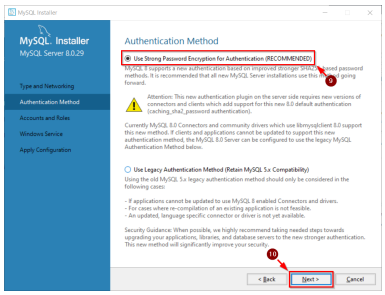

Figure 1.35: MySQL Authentication Method page

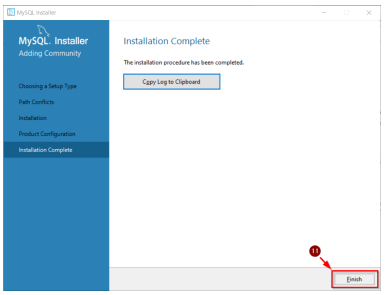

Figure 1.36: MySQL Installation Complete screen

A connection to this internal database can now be made using the username "root", the password you have configured, the address "localhost" and port 3306.

External Database

For a database that has to be accessed from sources other than your local machine, additional steps have to be completed.

First, ensure you have set up an internal database on the machine you want your external database to run on. To do this, follow the guide at 1.3.4 with the following changes:

- On the screen shown in Figure 1.34, enable the checkbox for *Open Windows Firewall ports for network access*. Additionally, select a *Server Computer Config Type* if the machine is dedicated to running your database, otherwise continue with *Development Computer*.

Next, encryption has to be added to the database. To do this, first locate the installation folder of MySQL, which should be at `C:\ProgramData\MySQL\MySQL Server 8.0` (or a higher number if you are using a more recent version of MySQL). This folder will be referred to as the Install folder from here on.

Copy your Certificate Authority's public key (as a PEM file), server's public key (as a PEM file), and server's private key (as a PEM or KEY file) into the Data folder located inside the Install folder.

In the file `my.cnf`, located inside the Install folder, find the following line (or add it at the bottom of the file if it does not yet exist): `[mysqld]`. Directly under this line, add the following lines to the file:

```
ssl_ca={ca.pem}
ssl_cert={cert.pem}
ssl_key={server-key.pem or .key}
require_secure_transport=ON
```

Replace {ca.pem} with the filename of your Certificate Authority's public key file, {cert.pem} with the filename of your server's public key file, and {server-key.pem or .key} with the filename of your server's private key file.
Next, a new user has to be created on the database that can be used from outside the current machine. To create this user, open a terminal window and enter the command `mysql -p`. Then, enter the following MySQL commands, replacing [username] with the desired username and [password] with the desired password for the user:

```
CREATE USER '[username]'@'%' IDENTIFIED BY '[password]';
GRANT ALL PRIVILEGES ON *.* TO '[username]'@'%' WITH GRANT OPTION;
```

Lastly, restart your machine to ensure all changes are processed and the database is booted up with the correct settings. A connection to this internal database can now be made using the username and password you have just configured, along with the IP address of your machine and port 3306.

Beware that the user created in this guide can do anything with the database. As such, make sure to use a secure password and do not share the credentials of this user with anyone else. Avoid using this user when connecting to the database normally as well. Instead, you may create new users using the `CREATE USER` statement or the DBeaver User Management window.

1.3 Linux

1.3.1 Installing DAISI

For Linux there are two ways of installing DAISI. If you have a debian based system the easiest way is by installing via the debian package. However if you do not have a debian system you have to install it by building the program.

Prerequisites

First we need to install the dependencies of the DAISI simulator, these can be installed via the terminal by using your OS packaging command. For instance for Ubuntu use `sudo apt-get install` {package name}. The full commands for ubuntu that need to be installed are listed below:

sudo apt-get install libplib-dev
sudo apt-get install libexpat1-dev
sudo apt-get install openscenegraph-3.4
sudo apt-get install freeglut3-dev
sudo apt-get install libvorbis-dev
sudo apt-get install libsdl2-dev
sudo apt-get install libopenal-dev
sudo apt-get install libenet-dev
sudo apt-get install libjpeg-dev
sudo apt-get install libpng-dev
sudo apt-get install libcurl4-openssl-dev
sudo apt-get install libmysqlcppconn-dev
sudo apt-get install libmsgpack-dev
sudo apt-get install libboost-all-dev

Installing DAISI via debian packages

After installation of the dependencies you can begin to install DAISI via the debian package. First you will need to install IPCLib. Go to `https://github.com/red-panda-productions/ipc-lib/releases`

and download the lastest release of IPCLib that is listed with the .deb extension. After downloading go to the folder where the download was placed and run `dpkg -i ipclib.deb`. This will install IPCLib on your system. After that you can install DAISI. Go to `https://github.com/red-panda-productions/DAISI/releases` and download the latest release of DAISI that is listed with the .deb extension. After downloading go to the folder where the download was placed and run `dpkg -i DAISI.deb`. This will install DAISI on your system. You can then run DAISI by typing daisi in your console.

Installing DAISI via CMake

If you do not have a debian based system you can install DAISI via CMake. First install cmake and ninja by typing `sudo snap install cmake --classic` and `sudo apt-get install ninja-build` to install cmake and ninja. Secondly you will need to install a newer compiler version of gcc and g++, which can be done by `sudo apt install gcc-11 g++-11` followed by `sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-11 110 --slave /usr/bin/g++ g++ /usr/bin/g++-11 --slave /usr/bin/gcov gcov /usr/bin/gcov-11`. Then you will need to install IPCLib. Go to a folder, open a terminal and type `git clone https://github.com/red-panda-productions/ipc-lib.git`, this will put a folder in your directory. Go into the folder and run the command `cmake -DCMAKE_BUILD_TYPE=Release -S . -B build -G ninja`, this will setup the cmake tool. Then type `cmake --build ./build --config Release` to build IPCLib. After that you can type `sudo cmake --install ./build`, which will install IPCLib on your system. Next we will install solid, which is a library that DAISI uses. Go to `https://sourceforge.net/projects/freesolid/` and download the latest release of solid. Unzip the file and move into the folder. Open a terminal and type `sudo chmod +x ./configure` and type `./configure`, which configures solid for your system. Then type `cmake -DCMAKE_BUILD_TYPE=Release -S . -B build -G ninja` to configure CMake. Then type `cmake --build ./build --config Release` to build solid and after that type `sudo cmake --install ./build`, which will install solid on your system. Lastly we can install DAISI. Go to a folder, open a terminal and type `git clone https://github.com/red-panda-productions/DAISI.git`, this will put the DAISI repository on your system. Go to `DAISI/speed-dreams/source-2.2.3`. Type `cmake -DCMAKE_BUILD_TYPE=Release -S . -B build -G ninja` which will configure CMake. After that type `cmake --build ./build --config Release`, which builds the DAISI simulater. After building you can install by typing `sudo cmake --install ./build`.

1.3.2 Installing DBeaver

To be able to use DBeaver, you need to have OpenJDK installed on your system, otherwise it will not be able to run. You can install OpenJDK simply by:

1. Opening a terminal.
2. Running the command `sudo apt-get install openjdk-11-jdk`.

From there, you can start installing DBeaver:

1. Go to https://github.com/red-panda-productions/dbeaver/releases.
2. Double click the `dbeaver.tar.gz` file to download the file (see Figure 1.37).

3. Either open a terminal and navigate to where you have downloaded the .tar.gz file or right click the directory where the file is downloaded and choose `Open in Terminal` (see Figure 1.38).
4. To install the DBeaver software run the command: `tar -xzvf dbeaver.tar.gz` or `tar -xzvf dbeaver.tar.gz -C [path where dbeaver should be installed]` if you want DBeaver to be installed in a specific location.

Figure 1.37: DBeaver Linux installer

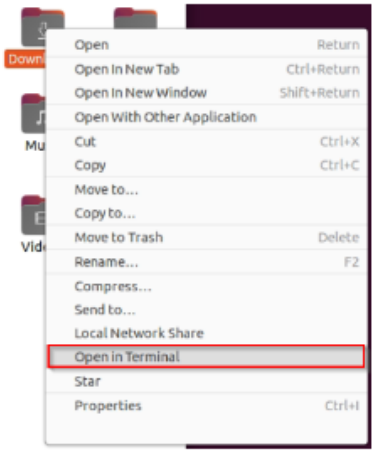

Figure 1.38: Open in Terminal

5. If you use the first command, then run the command: `cd dbeaver`, otherwise, go to the directory where you have installed DBeaver inside the terminal. You should have a path like shown in Figure 1.39.
6. Run the command `./dbeaver` inside the terminal to start DBeaver.
7. An alternative way to start up DBeaver is by opening the directory where DBeaver is installed and double clicking the `dbeaver` file or clicking the `dbeaver` file and pressing *Enter* (not the `dbeaver.inl` or the `dbeaver.png` files) (see Figure 1.40).
8. DBeaver will now start up and you will be greeted with a popup that asks you if you want to create a sample database. You can decide if you want to do that or not. From there, you can start working with DBeaver.

Figure 1.39: terminal path

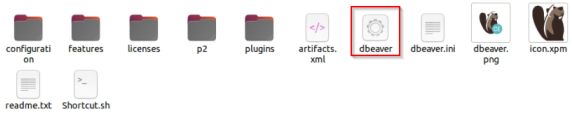

Figure 1.40: dbeaver file

1.3.3 Connecting Steering Wheel

Prerequisites

1. Steering wheel hardware is set up (see section 1.1).
2. Have git installed.

Downloading Configuration Software

When connecting the steering wheel to Linux, you will need to download a software that can configure the settings of the steering wheel. This guide will use https://github.com/berarma/oversteer and the "Logitech G27 Racing Wheel". To connect the steering wheel on Windows you can follow the guide in section 1.2.3.

1. To install the dependencies of Oversteer run the following command (copy and pasting might add newlines, make sure to run the commands without them):
```
sudo apt install python3 python3-distutils python3-gi python3-pyudev python3-xdg python3-evdev
gettext meson appstream-util desktop-file-utils python3-matplotlib python3-scipy
```
2. then run the following commands to build and install Oversteer:
```
git clone https://github.com/berarma/oversteer.git
cd oversteer
```

```
meson build
cd build
sudo ninja install
```

3. lastly, reboot your computer.

After rebooting, you can run the Oversteer program. Here, you can make sure the *Rotation rate* is set to "900" (see figure 1.41). After that navigate to the *Force Feedback* menu and turn the *autocenter strength* to "100" (see figure 1.42).

The steering wheel should now be connected to your computer.

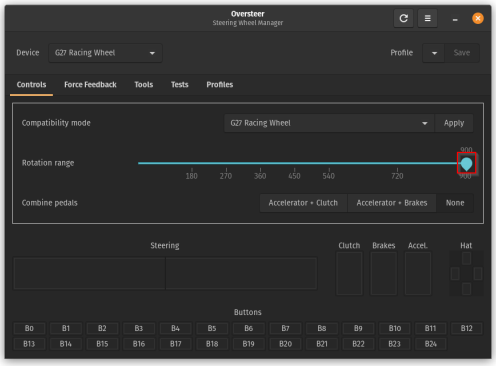

Figure 1.41: Oversteer Controls Menu

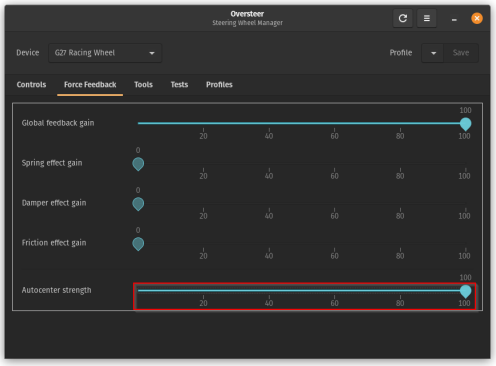

Figure 1.42: Oversteer Force Feedback Menu

1.3.4 Setup Database

This guide will show you how to set up a MySQL database on a Linux machine. The first part of the guide will show how to set up an internal database (only accessible from your local machine). The second part of the guide will show you how to set up an external database (accessible from other devices than just your local machine).

Internal Database

To set up an internal database on your Linux machine, first download the MySQL Debian Package (`mysql-apt-config_w.x.y-z_all.deb`, where `w.x.y-z` is replaced with the latest MySQL version number) from `https://dev.mysql.com/downloads/repo/apt/` (1). You will be sent to a page suggesting you create an account. Unless you want to create an account, click on *No thanks, just start my download.* (2).

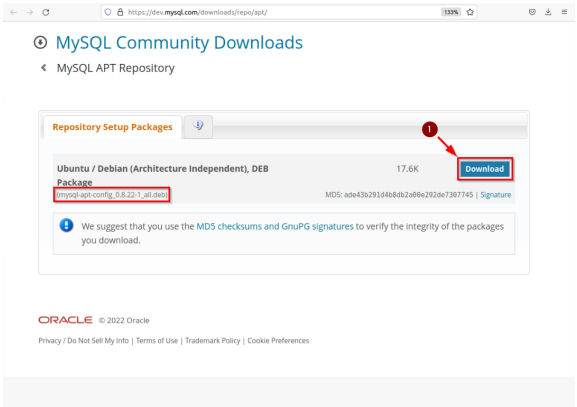

Figure 1.43: MySQL download page

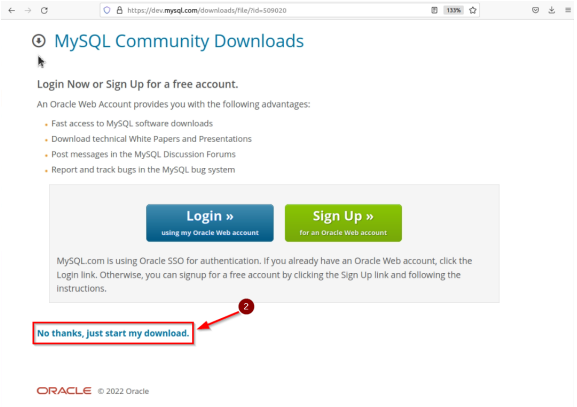

Figure 1.44: MySQL log-in page

Either open a terminal and navigate to where you have downloaded the Debian package or right-click the directory where the package is downloaded and choose *open in terminal* (see Figure 1.38). To then install the MySQL release package, run the command: `sudo dpkg -i mysql-apt-config\_w.x.y-z\_all.deb`.

This will open the installer window. Ensure the MySQL Server Cluster product is set to at least mysql-8.0, MySQL Tools & Connectors are Enabled, and MySQL Preview packages are Disabled. Then navigate to *Ok* and press `Enter` (3).

Next, run the following commands:

```
sudo apt-get update
sudo apt-get build-dep mysql-server
sudo apt-get install mysql-apt-config
sudo apt-get install mysql-shell
sudo apt-get install mysql-server
```

You will be prompted for a root password (4). Come up with a password and make sure to remember it before pressing `Enter`. When asked which authentication method to use, make sure to select *Use Strong Password Encryption* (5) before pressing `Enter`. This finishes the installation for an internal database.

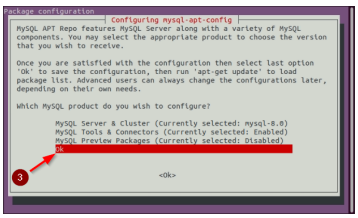

Figure 1.45: Configure mysql-apt-config

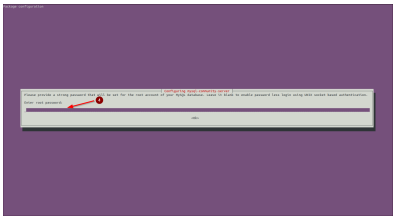

Figure 1.46: Configure MySQL password

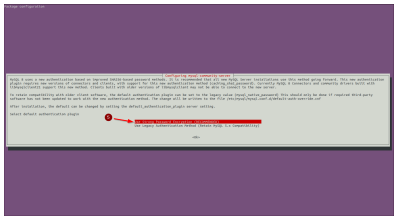

Figure 1.47: Authentication Method menu

External Database

For a database that has to be accessed from sources other than your local machine, additional steps have to be completed.

First, ensure you have set up an internal database on the machine you want your external database to run on. To do this, follow the guide at 1.3.4.

Next, encryption has to be added to the database. To do this, first locate the installation folder of MySQL, which should be at `/etc/mysql/`. Create a new folder inside this folder and copy your Certificate Authority's public key (as a PEM file), server's public key (as a PEM file), and server's private key (as a PEM or KEY file) into this folder. For this example, we will name the folder `certificates`.

In the file `my.ini`, located inside the Install folder, find the following line (or add it if it does not yet exist): `[mysqld]`. Directly under this line, add the following lines to the file:

```
ssl_ca=/etc/mysql/certificates/{ca.pem}
ssl_cert=/etc/mysql/certificates/{cert.pem}
ssl_key=/etc/mysql/certificates/{server-key.pem or .key}
require_secure_transport=ON
```

Replace {`ca.pem`} with the filename of your Certificate Authority's public key file, {`cert.pem`} with the filename of your server's public key file, and {`server-key.pem or .key`} with the filename of your server's private key file.

Next, a new user has to be created on the database that can be used from outside the current machine. To create this user, open a terminal window and enter the command `mysql -p`. Then, enter the following MySQL commands, replacing `[username]` with the desired username and `[password]` with the desired password for the user:

```
CREATE USER '[username]'@'%' IDENTIFIED BY '[password]';
GRANT ALL PRIVILEGES ON *.* TO '[username]'@'%' WITH GRANT OPTION;
```

Lastly, restart your machine to ensure all changes are processed and the database is booted up with the correct settings. A connection to this internal database can now be made using the username and

password you have just configured, along with the IP address of your machine and port 3306.

Beware that the user created in this guide can do anything with the database. As such, make sure to use a secure password and do not share the credentials of this user with anyone else. Avoid using this user when connecting to the database normally as well. Instead, you may create new users using the `CREATE USER` statement or the DBeaver User Management window.

Chapter 2

Using DAISI

2.1 Starting The Program

To start DAISI, one must run it as an administrator. After starting the program you will be greeted with the *Main Menu* (see Figure 2.1. On the *Main Menu* you can choose to either *start* an experiment (1), change the general *options* of the program (2), open the *credits* (3) or *quit* the program (4). Clicking on the *Options* button opens the *Options Menu* (see Figure 2.2). In the *options menu* three sub-menus with different option categories are available: The *Control* settings (1), The *Display* configuration settings (2) and the *Sound* settings (3).

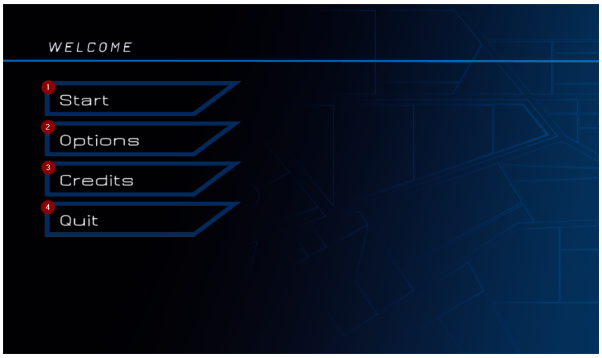

Figure 2.1: DAISI Main Menu

Figure 2.2: DAISI Options Menu

2.2 Configure Controls

2.2.1 Prerequisites

1. Followed the *Connecting Steering Wheel* guides for Windows (section 1.2.3) of Linux (section 1.3.3), if the steering wheel is used.

To change the controls of DAISI you will need to navigate to the *Controls* menu, shown in figure 2.3. Here you can click on the text next to the action you want to configure (e.g MOUSE_LEFT in figure 2.3, if you want to change the *left steer* control). This text will then disappear, after which the next input will be the configured control. The next section will show how to do this for a steering wheel, but controllers, joysticks and keyboards can also be used.

2.2.2 Configuring Steering Wheel

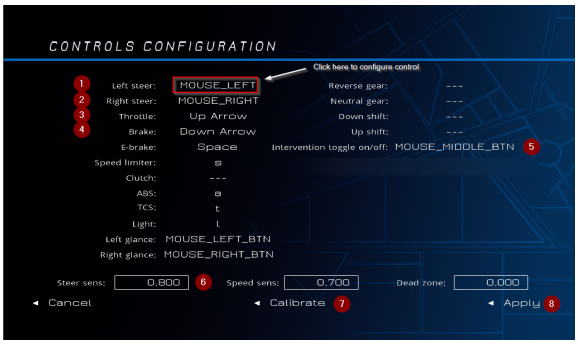

Figure 2.3: DAISI controls configuration screen

Figure 2.4: Calibration Menu

After connecting the steering wheel to your computer, you can start DAISI and change the controls to use the steering wheel.

1. Open DAISI and navigate to the *Options* (see Figure 2.2).
2. Click on *Controls*(1).
3. See Figure 2.3:

   (a) Click on the text next to *Left steer* (1) (in this case MOUSE_LEFT) and move the steering wheel to the left. It should now display `"AXIS0-0"`. (make sure after clicking to not move your mouse as this will choose your mouse as the input. If this happens click on the text again).
   (b) Click on the text next to *Right steer* (2) and move the steering wheel to the right. It should now display `"AXIS0-0"`.
   (c) Click on the text next to *Throttle* (3) and press the gas-pedal (the most right one). It should now display `"AXIS1-0"`.
   (d) Click on the text next to *Brake* (4) and press the brake-pedal (the middle one). It should now display `"AXIS2-0"`.
   (e) Click on the text next to *Intervention toggle on/off* (5) and press the button you want to use to turn the intervention on and off. If it is a button on the steering wheel it will display something like `"BTN7-0"`.
   (f) Set the *Steer sens* to ”0.800” (6) (this can be changed if the sensitivity is too much or too little).
   (g) Click on *Calibrate* (7) and follow the instruction of the calibration menu (see figure 2.4), given in the bottom of the screen (1). If something goes wrong you can press the *Reset* button (2). Once done you can press *Done*
   (h) Click on *Apply* (8).

4. The steering wheel should now be able to be used in DAISI.

2.3 Other Configuration Menus

The two other option menus, the *Display Configuration Menu* (see Figure 2.5) and the *Sound Configuration Menu* (see Figure 2.6), in the *Options Menu* contain the rest of the settings for the program (not including parameters to set for experiments).

The *Display Configuration Menu* has options for the following:

1. The window size of the program;
2. The color depth (amount of colors that a pixel can display);
3. The display mode (windowed or fullscreen);
4. The video mode detection method (either automatic or manual);
5. The video initialization method (either compatible or best possible).

It is recommended to use 'Auto' and 'Compatible' for the last two options.

The *Sound Configuration Menu* has options for the *sound effects volume*, the *music volume* (with a quick toggle) and the *intervention volume* of the AI. A volume of '100' means the volume is set as loud as possible (100%).

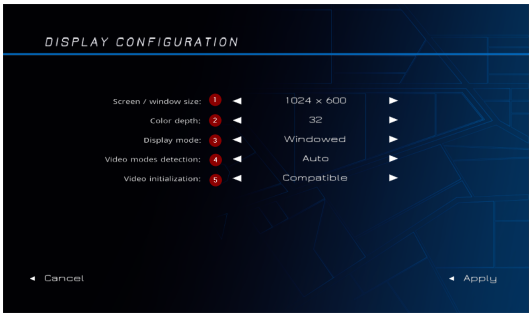

Figure 2.5: Display Configuration Menu

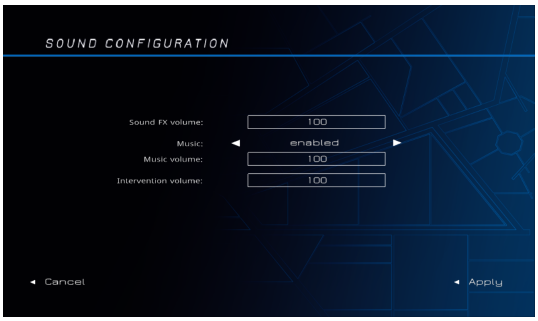

Figure 2.6: Sound Configuration Menu

2.4 Setting Up An Experiment

Selecting *Start* (1) in the *Main Menu* opens the *Researcher Menu* (see Figure 2.7).

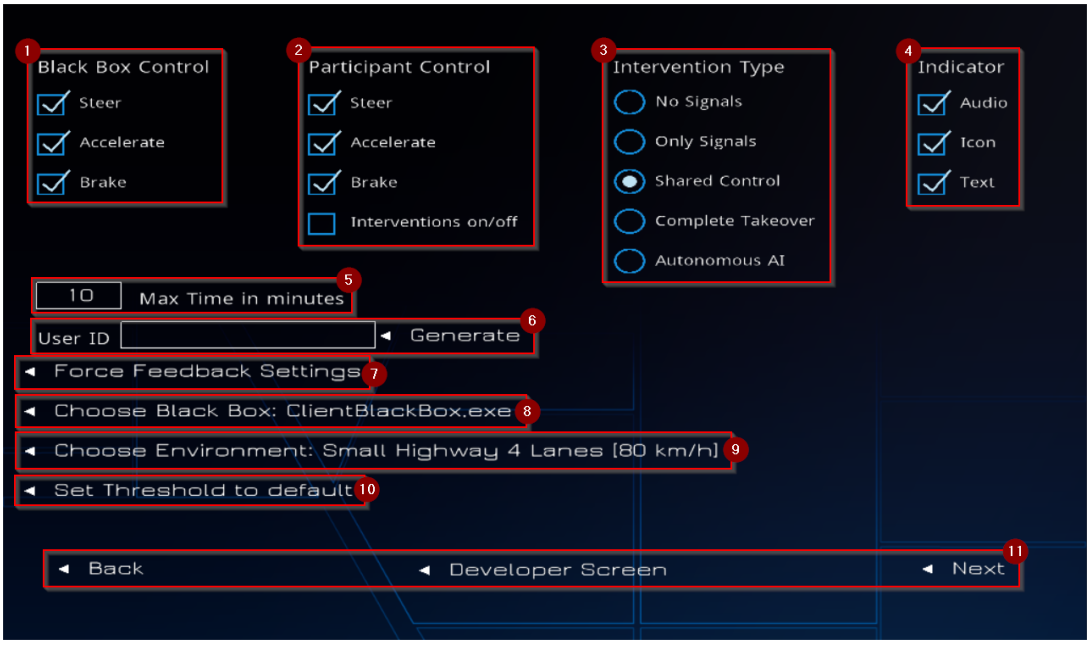

Figure 2.7: Researcher Menu

In this menu, researchers select many different options to use in their experiments. Below is a list of what every option in the menu changes.

1. Checkboxes for changing the permissions of the black box over the car.
2. Checkboxes for changing the permissions of the user/participant.
3. Radiobuttons for selecting how the AI will intervene with the user.
4. Checkboxes for selecting what type of indication the AI will use when it intervenes with the user.
5. Input field for setting the *Maximum Time* the experiment can run for (in minutes).
6. Input field for filling in a participant *User Id*. Has a button next to it to *generate* a random number with four digits.
7. Button to open the *Force Feedback* settings.
8. Button to select a *Black Box* that will intervene with the user driving in the simulation.
9. Button to open the *Environment Selection Menu*.
10. Button to set the *Threshold* values in the *Developer Screen* to their default values.
11. Buttons to go to the previous (*Main*) menu, the next (*Data Selection*) menu and the *Developer Screen*.

Figure 2.8: Force Feedback Configuration Menu

Clicking on the *Force Feedback Settings* button, opens the *Force Feedback Configuration Menu* (see Figure 2.8). Here you can:

1. Toggle the *Force feedback* and the *Reverse Force feedback*.
2. Reduce the global strength of the force feedback by making wheel strength higher.
3. Change the *align effect* of the force feedback.
4. Change the *engine revving effect* settings (vibrations from engine).
5. Change the *lowSpeedConstantForce effect* settings (resistance to turning at low speeds).

2.4.1 Loading Black Boxes

Before a run, you must select the black box you want to use. This can be done by pressing the *Choose Black Box* button in the *Researcher Menu* (see figure 2.9).

On windows, a windows explorer pop-up will appear when you press this button. Inside this explorer, navigate to the EXE file of your black box. The location of this file depends on your black box. A black box made in the SDALib repository will be located inside your build folder (figure 2.10).

On Linux, a similar process is required. A pop-up screen will be opened, and one must locate the black box executable. Keep in mind that Linux does not have the EXE file extension (figure 2.11).

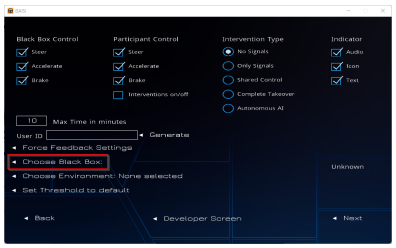

Figure 2.9: *Choose Black Box* button

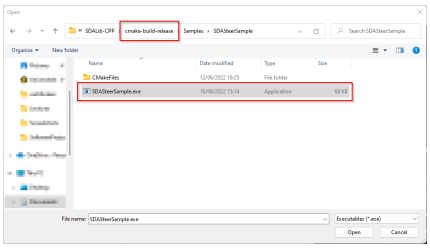

Figure 2.10: Selecting black box on Windows

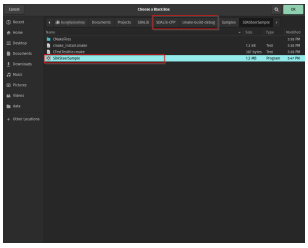

Figure 2.11: Selecting black box on Linux

After selecting your black box, DAISI will do a performance check to test the speed of the selected black box. A speed indication (Fast, Moderate, or Slow) is shown on the screen. The file name of your selected black box can also be seen (figure 2.12).

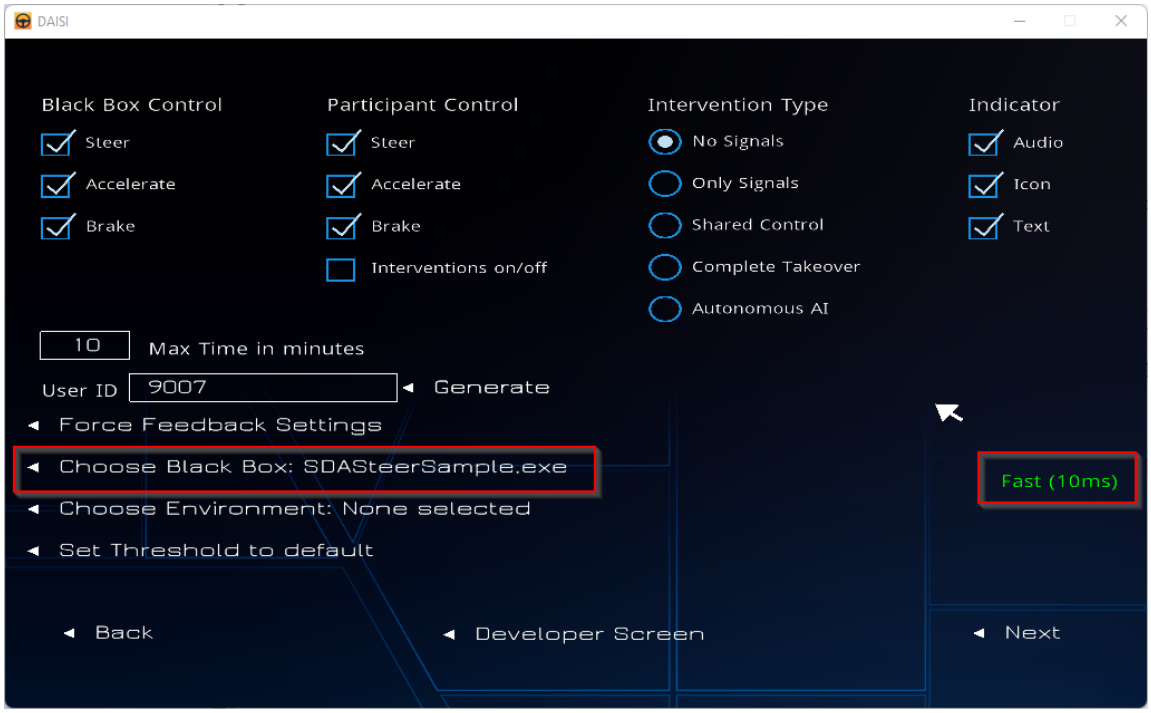

Figure 2.12: Selected black box tested

2.4.2 Loading Environments

To select the environment to load, press the *Choose Environment* button on the Researcher Menu, as shown in Figure 2.13. This will bring you to the Environment Selection Menu. This menu displays a variety of statistics about the currently selected environment, as well as options for changing this environment.

In Figure 2.14, the Environment Selection Menu is laid out. Section 1 in this figure displays the outline of the currently selected environment. Section 2 lists its category, which is the option that determines the shape of the environment. Section 3 lists the actual environment, which will determine the specifics of the environment. Section 4 has a description of the currently selected environment, while Section 5

will tell you some of the statistics of the environment, such as the length and road width, as well as the estimated time it will take to drive one lap in the environment.

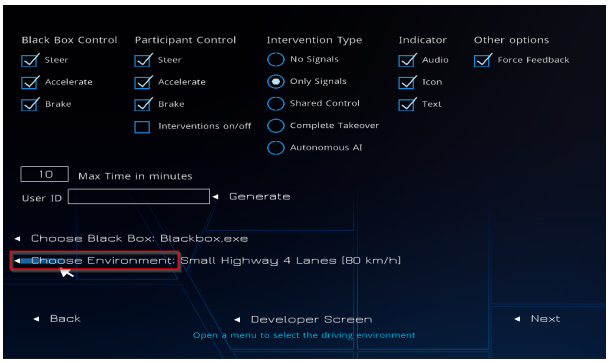

Figure 2.13: Location of the *Choose Environment* button on the Researcher Menu

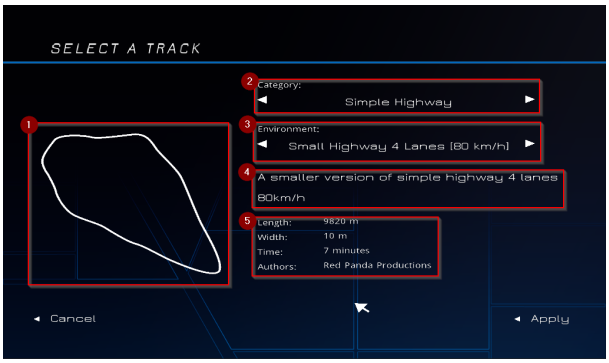

Figure 2.14: Overview of the Environment Selection Menu

To change between environment categories, press either of the buttons highlighted in Figure 2.15. One can notice the outline displayed has changed compared to Figure 2.14, along with all the other information on the menu.

To change between different environments within a category, press either of the buttons highlighted in Figure 2.16. One can notice the description and statistics of the environment have changed.

Once you are satisfied with your choice of environment, you can press the *Apply* button, which is highlighted in Figure 2.17. Pressing the *Cancel* button will not set the used environment to the one currently displayed in the Environment Selection Menu.

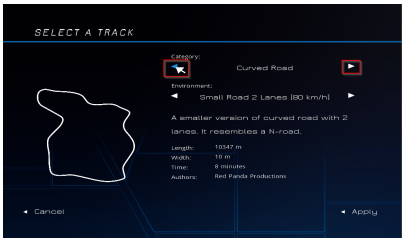

Figure 2.15: Buttons for changing the environment category

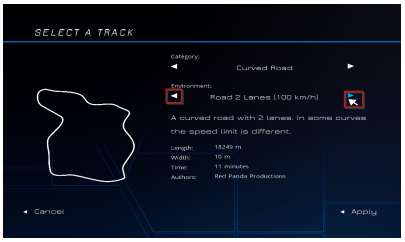

Figure 2.16: Buttons for changing the selected environment

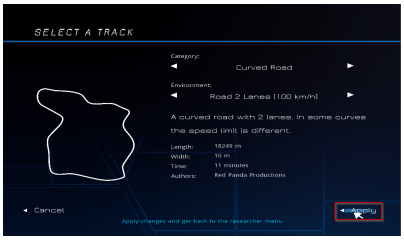

Figure 2.17: Apply Button for confirming environment selection

2.4.3 Connect To Database

Clicking on *Database settings* in the *Data Selection Menu* will bring you to the screen shown in figure 2.18.

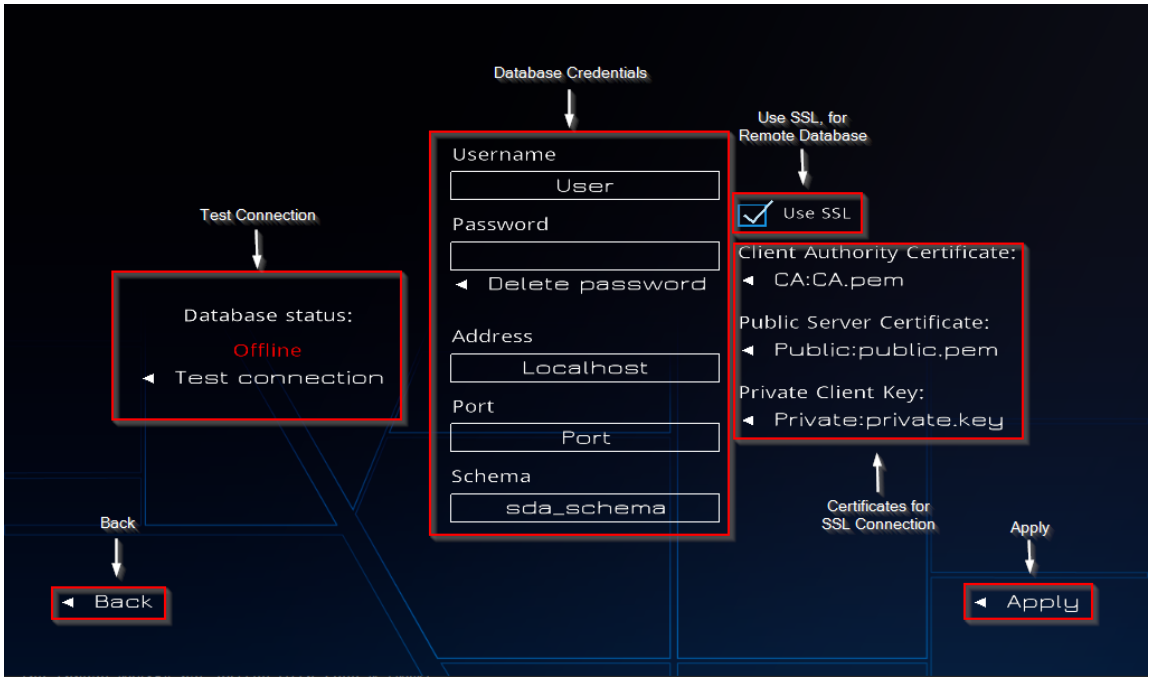

Figure 2.18: DAISI Database Connection screen

- **Database Credentials**: Here you need to fill in the database credentials for your specific MySQL database.
  - *Username* is the username of your MySQL user, for example "root".
  - The *Password* is the password for this MySQL user, for example "PASSWORD123". This password is stored locally and you can press the *Delete password* button to delete it after you are done.
  - *Address* is the address of your database, this can be "localhost" or "127.0.0.1" for a local database. For a remote database it needs to be the url or IP-address of the remote database.
  - *Port* is the port number to connect to you database. For local databases this will probably be "3306".
  - *Schema* is the specific database schema that you want the data to be stored in, for example "DAISI_scheme". This will create a new database schema if there is not one already with that name.
- **Use SSL, for Remote Database**: The remote database uses SSL to ensure a secure connection between DAISI and the remote database. So to be able to use the remote database this needs to be checked. Now the *Certificates for SSL Connection* buttons should appear.
- **Certificates for SSL Connection**: Once *Use SSL* is checked this appears. These buttons will open file dialogs for you to select the correct certificates.
  - *Client Authority Certificate* requires you to select a Client Authority (CA) certificate in PEM format
  - *Public Server Certificate* is the public certificate of your server, again in PEM format.
  - *Private Client Key* is the private key of the client that wants to connect to the server, in this case DAISI. This needs to be formatted in the KEY format.
- **Test Connection**: once everything is filled in to connect to the local or remote database, you can test the connection by pressing the button. The status will now turn green and say *online*. If it stays offline, one of your settings is probably incorrect, or your database is actually offline.
- **Apply**: Press *Apply* to apply the settings.
- **Back**: Press back to go back to the previous menu, without applying the settings.

2.5 Saving Data

The last menu before starting a simulation allows you to configure what data you want to save during the simulation. This menu is shown in figure 2.19. This menu has four check boxes, each of which controls the collection of a part of all the data. During a simulation this the selected data is buffered in TXT and CSV files, which are temporarily stored in your users `AppData/Local/DAISI-data-buffers` folder.

Figure 2.19: DAISI Data Collection Menu: (1) Label displaying the current status of the database connection; (2) Button to go to the database settings menu, explained in section 2.4.3; (3) Check-boxes to customize data collection. These are explained in detail in the sections below; (4) Button to go to the advanced data settings menu; (5) Button to go back to the Researcher Menu; (6) Button to start the simulation.

2.5.1 General Meta Data

This is data that is always stored for every experiment. It contains information about the black box that was used, the environment that was used and the intervention settings that were used. This data is then all connected together in the Trial table:

- **trial_id**: The unique identifier belonging to this trial.
- **trial_time**: The (start) time at which this trial was performed. This is in `yyyy-mm-dd hh-mm-ss` format.
- **participant_id**: The unique identifier of the participant that took this trial.
- **blackbox_id**: A foreign key, linking to an unique identifier in the Blackbox table.
- **environment_id**: A foreign key, linking to an unique identifier in the Environment table.
- **settings_id**: A foreign key, linking to an unique identifier in the Settings table.

The following data is stored in the Blackbox table:

- **blackbox_id**: The unique identifier belonging to this black box data.
- **filename**: The filename of the black box with file extension.
- **version**: The version is a date-time, which is the last time this file has been changed.

- **name**: The name of the black box file, without file extension.

The following data is stored in the Environment table:

- **environment_id**: The unique identifier belonging to this environment data.
- **filename**: The file path to the XML file representing the environment, starting from the data folder in the project.
- **version**: The version is an integer representing the version of the environment. This version is specified under the 'version' attribute in the environments XML file.
- **name**: The name of the environment as given in the 'name' attribute in the environments XML file.

The following data is stored in the Settings table:

- **environment_id**: The unique identifier belonging to this settings data.
- **intervention_mode**: An enum value corresponding to the selected intervention type: 'Off' = no signals, 'Suggest' = signals only, 'Shared' = shared control, 'Force' = complete takeover, 'Drive' = autonomous AI.

2.5.2 Environment Data

This checkbox exists to indicate that the environment data is saved, as described in 2.5.1. However, this data will always be saved and is thus not configurable. Therefore checkbox cannot be deselected.

2.5.3 Car Data

This checkbox controls whether all data about the car is stored every time step. This is data about the car's position, direction, speed, acceleration and gear. The database stores the following values in the GameState table:

- **game_state_id**: The unique identifier of this game state data.
- **tick**: The time step of the game at which this data was stored.
- **x**: The car's x position.
- **y**: The car's y position.
- **z**: The car's z position.
- **direction_x**: The car's angle along the x-axis.
- **direction_y**: The car's angle along the y-axis.
- **direction_z**: The car's angle along the z-axis.
- **speed**: The car's current velocity.
- **acceleration**: The car's current acceleration.
- **gear**: The car's current gear.

2.5.4 Human User Data

This checkbox controls whether all data about the user input is stored every time step. This is data about the user's steer, brake, gas and clutch input. The database stores the following values in the UserInput table:

- **user_input_id**: The unique identifier of this user input data.
- **tick**: The time step of the game at which this data was stored.

- **steer**: The user's steering amount at this time step. This is a value between about -1 and +1, where -1 means that the user is steering fully to the right, and +1 means that the user is steering fully to the left.
- **brake**: The user's braking amount at this time step. This is a value between 0 and 1, where 0 means that the brake is not pressed, and 1 means that the brake is fully pressed.
- **gas**: The user's throttle amount at this time step. This is a value between 0 and 1, where 0 means that the throttle is not pressed at all, and 1 means that it is fully pressed.
- **clutch**: The user's clutch value at this time step. This is a value ranging from about 0 and 1, where 0 means that the clutch is not pressed, and 1 means that the clutch is pressed.

2.5.5 Intervention Data

This checkbox controls whether all data about the black box decisions is stored every time step. This is data about the black box steer, brake and acceleration decisions. The database stores an intervention entry for each time step in the Intervention table. Each decision is stored in its own table, linking back to the Intervention table. Decisions are only stored if there was a decision made by the black box:

- **intervention_id**: The unique identifier of this intervention data.
- **tick**: The time step of the game at which this data was stored.
- **steer_decision**: The black box' steering decision amount at this time step. This is a value between about -1 and +1, where -1 means that the decision is to fully steer to the right, and +1 means that the decision is to fully steer to the left. This is stored as 'amount' in the SteerDecision table.
- **brake_decision**: The black box' braking decision amount at this time step. This is a value between 0 and 1, where 0 means that the decision is to not brake, and 1 means that the decision is to fully brake. This is stored as 'amount' in the BrakeDecision table.
- **accel_decision**: The black box' acceleration decision amount at this time step. This is a value between 0 and 1, where 0 means that the decision is to not press the throttle at all, and 1 means that the throttle should be fully pressed. This is stored as 'amount' in the AccelDecision table.

2.5.6 Advanced Settings

The advanced settings menu allows the user to setup the data compression. The menu is shown in figure 2.20.

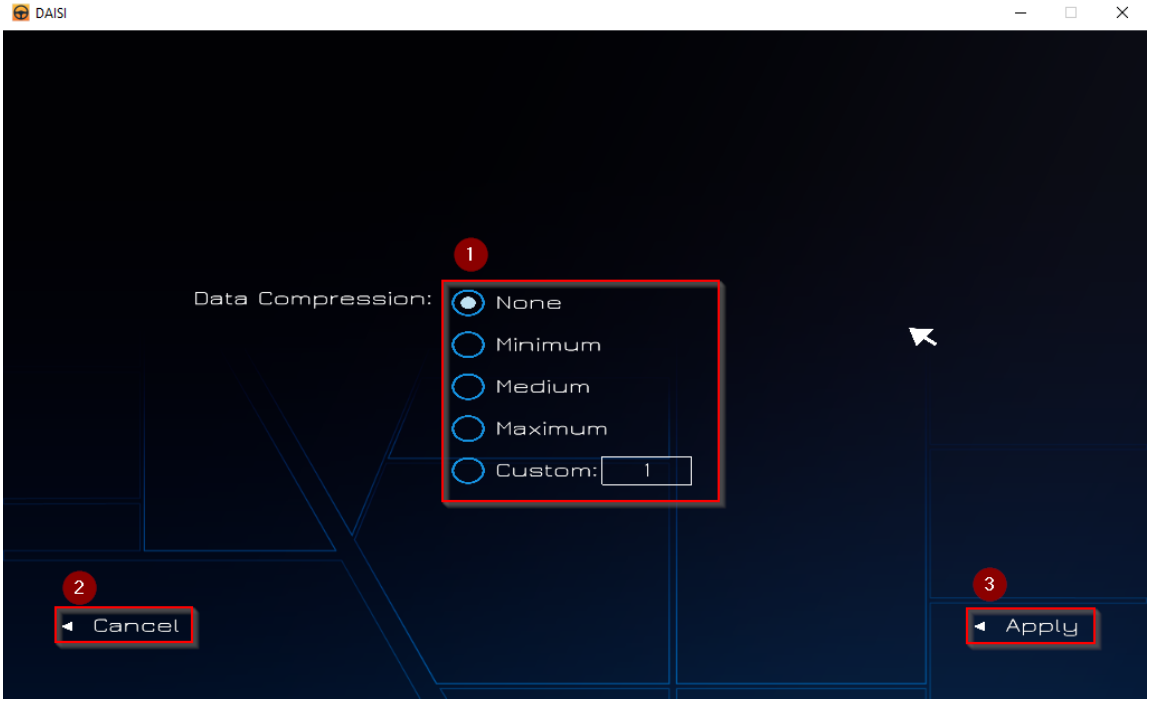

Figure 2.20: DAISI Advanced Data Settings Menu: (1) Options for data compression options. *None* stores data every time step, *Minimum* compresses data from three time steps into a single data point. *Medium* compresses data from five time steps and *Maximum* compresses data from 9 time steps. The *Custom* option uses the number in the text-box to determine how many time steps will be compressed into a single data point. This value can at most be 49 and should always be an odd number; (2) Button to cancel your changes and go back to the Data Selection Menu; (3) Button to apply your changes and go back to the Data Selection Menu.

2.6 The Simulation

Clicking on *Start* in the *Data Collection Menu* (?), starts the race and brings you to a loading screen (see Figure 2.21) after which it opens the simulation (see Figure 2.22).

Figure 2.21: Loading Screen

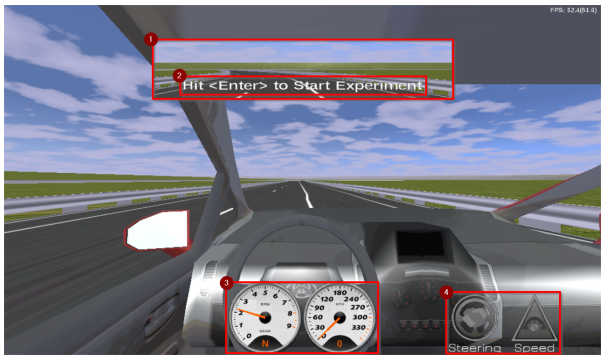

Figure 2.22: The Start of the Experiment

In the simulation, you can see the rear view mirror at the top (1), a line of text telling the researcher to start the experiment (2), The gear (left) and speed (right) at the bottom (3) and at the bottom right the interventions of the AI (4). The interventions will light up once the AI intervenes with the driver. Different intervention types have different icons.
After the experiment duration—set in the *Maxt Time* in the *Researcher Menu*—ends, the *End of Experiment Screen* is opened (see Figure 2.23). It is also possible for researchers to pause the experiment. Doing this opens the *Pause Menu* (see Figure 2.24). Here researchers can choose to *Resume* the experiment (1), *Restart* the experiment (2) or *Stop* the experiment (3).

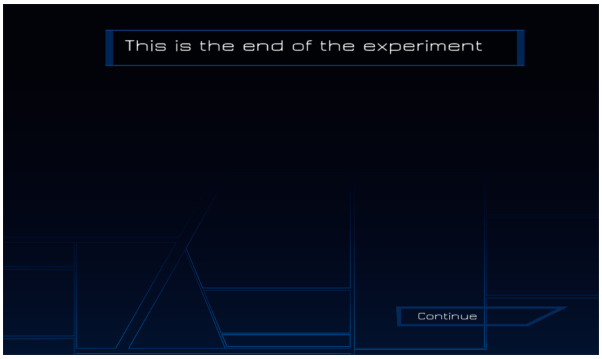

Figure 2.23: The End of the Experiment

Figure 2.24: The Pause Menu

Clicking on the *Continue* button in the *End of Experiment Screen* or on *Restart* or *Stop* in the *Pause Menu* opens the *Save Data to Database Screen* (see Figure 2.25). If you choose to not save the data a confirmation screen pops up (see Figure 2.26). After choosing to save or not, researchers are done and are brought back to the *Main Menu*.

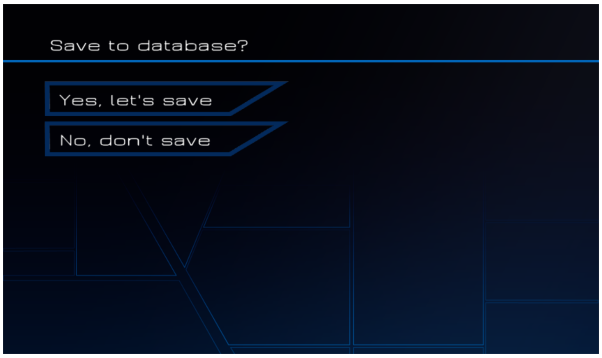

Figure 2.25: The Save the Data to the Database screen

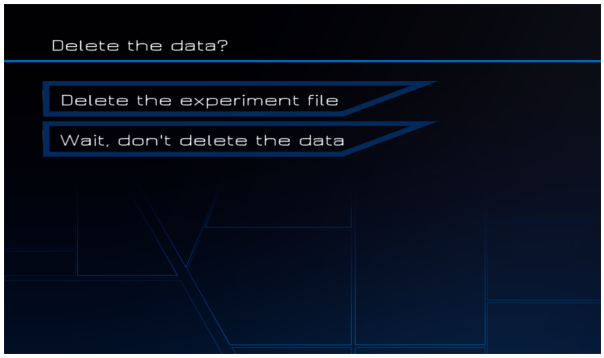

Figure 2.26: The 'Do Not Save' Confirmation Screen

2.6.1 Static Use Cases

Studying Particular Intervention Indicators

No indicators

With DAISI, it is possible to study the different intervention types without directly communicating to the user what interventions the AI is making. For each intervention type, it is possible to turn off all types of indicators. In Figure 2.27, it is shown what the 'indicator' settings should look like in the researcher menu in case a researcher does not want the black box to communicate its decisions under any circumstance. However, note that even though it is possible to turn off the indicators for the only signals intervention type, it is not logical to do so, as this would have a similar functionality to the no signals intervention type with all indicators turned off.

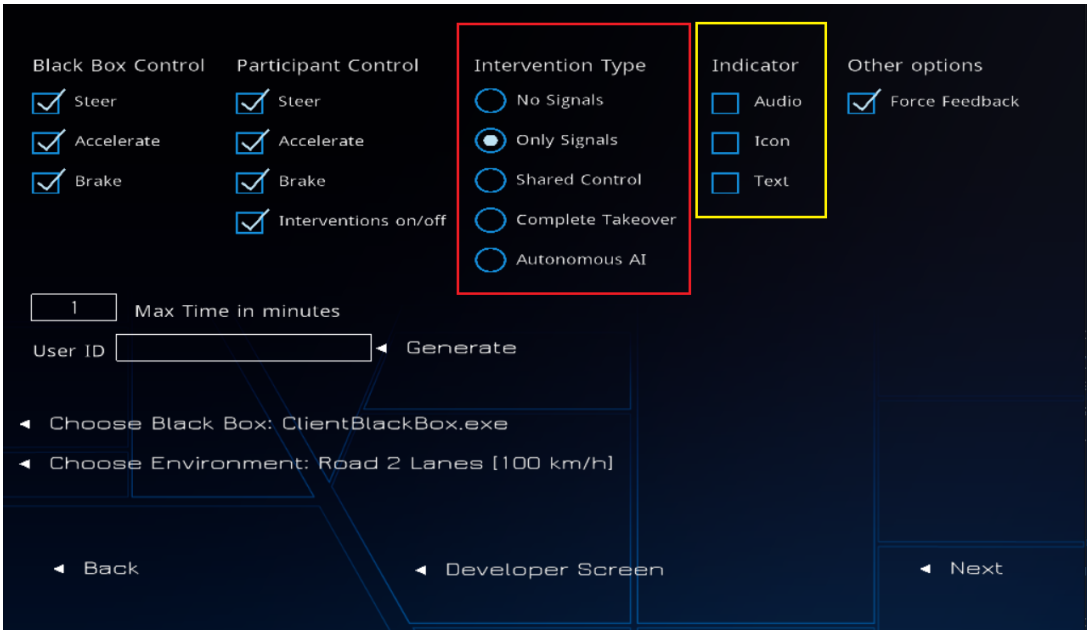

Figure 2.27: Researcher menu settings in a scenario with the only signals intervention type (red box), though without actually having indicators (yellow box).

Solely icons

Besides that, it is also possible to study how specific types of intervention indicators influence the participant in their own way. For example, it is possible to allow the system to solely communicate the AI's decision via icon signals. As an example, we have a scenario wherein a researcher wants to use the following settings:

1. Shared Control intervention type.
2. All black box controls.
3. All participant controls.
4. Force feedback for the steering wheel.
5. Solely icon indicators

To activate the settings from the list above, the researcher menu would need to have the checkboxes (un)ticked as shown in Figure 2.28.

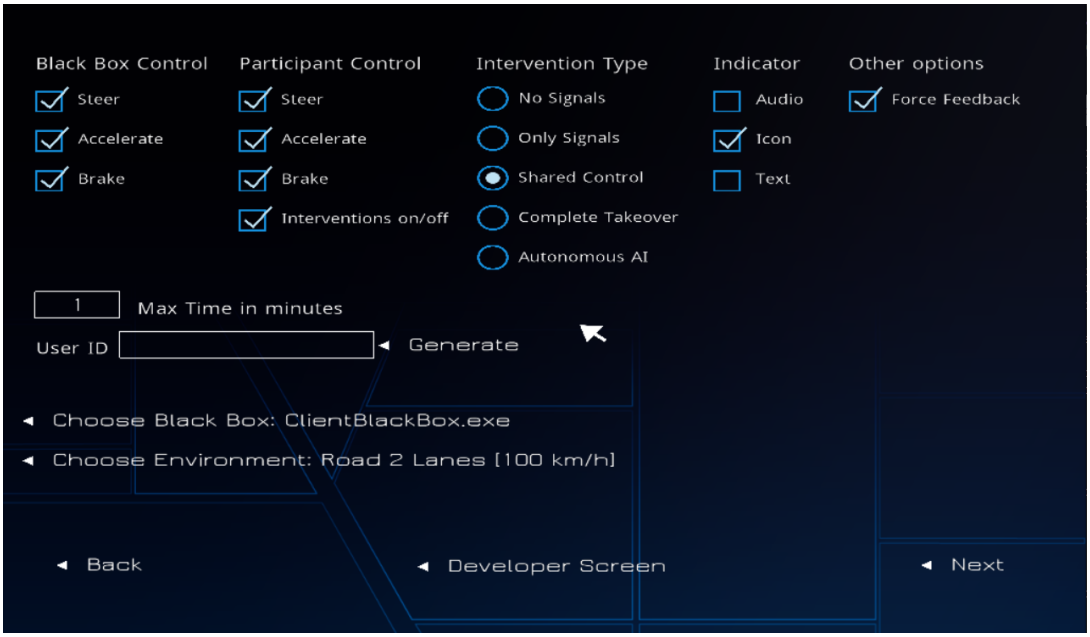

Figure 2.28: Researcher menu settings in a scenario with the shared control intervention type and solely icon indicators.

No Signals & Autonomous AI intervention type

Even though the no signals intervention type is no different from a regular driving simulator in terms of its controls, it can still be used to research participants' reactions to driving assistance mechanisms by turning on at least one indicator option. Figure 2.29 displays what the icons in the driving simulation look like with the no signals intervention type, including icon and text indicators, including the researcher menu settings in the top left corner.

Figure 2.29: Driving simulation in a scenario with the no signals intervention type, and icon and text indicators. In the top left corner, the researcher menu settings are displayed. Even though the driver is far from the center of the road, the icons are not 'activated' but remain idle.

For the autonomous AI intervention type, it is also possible to turn on indicators. Figure 2.30 displays the driving simulation in the autonomous AI intervention type with icon and text indicators turned on.

Figure 2.30: Driving simulation in a scenario with the autonomous AI intervention type, and icon and text indicators. In the top left corner, the researcher menu settings are displayed. The icons are always blue, because they should indicate that the interventions are always active.

Study Specific Participant and Black Box Controls

Participant Control

With DAISI, it is also possible to study the specific types of AI decisions and their influence on drivers. In the researcher menu, the researcher can choose to turn on or off specific types of 'participant controls'. With this settings option, it is possible to turn off the option for a participant to steer, accelerate or brake during the simulation.

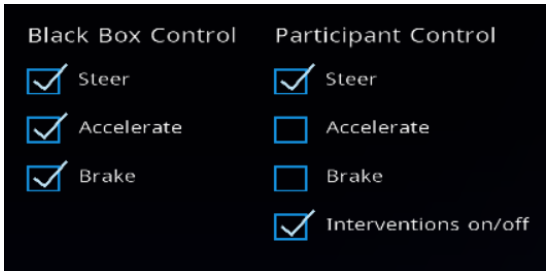

Figure 2.31: Researcher menu settings when the black box can make decisions regarding steering, accelerating and braking, whilst the participant only has control over the steering.

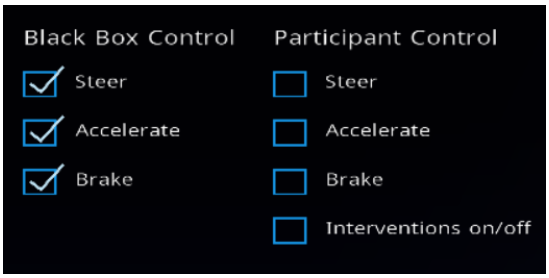

Figure 2.32: Researcher menu settings when the black box can make decisions regarding steering, accelerating and braking, whilst the participant has no control over any decision types.

Even though a black box may have the functionality to interfere with a participant's steering, accelerating or braking, a researcher can study how a participant responds to a black box's decision once their own controls for that particular 'decision' are turned off. In Figure 2.31 is displayed the researcher menu's settings when allowing the black box to interfere with steering, accelerating and braking, whilst solely allowing the participant to be able to steer.

As shown in Figure 2.32, all black box controls are turned on, whilst all participant controls are turned off. With the no signals or only signals intervention type in combination with these settings will result in a simulation scenario where nothing will be happening in the simulation as neither the participant nor the AI can drive. In the other intervention types, the functionality will be similar to the autonomous AI

intervention type.

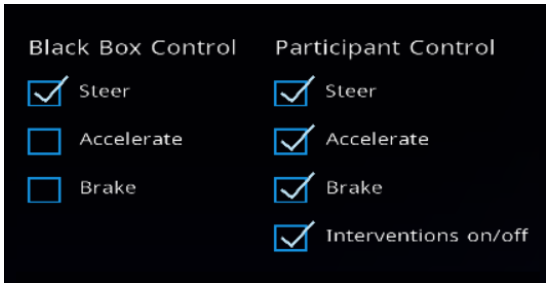

Figure 2.33: Researcher menu settings when the black box can make decisions regarding steering only, whilst the participant has all controls (steering, accelerating, braking).

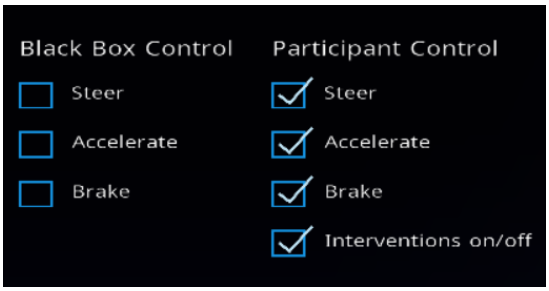

Figure 2.34: Researcher menu settings when the black box cannot interfere with any of the decision types, whilst the participant has control over all decision types.

Black Box Control

A reversed research scenario is also feasible in DAISI: to study how a participant responds to a driving situation where the black box interferes with one particular decision, but not with others.

Even though a black box may have the functionality to interfere with all decision types, namely steering, accelerating and braking, a researcher can choose to turn off one or more specific decision types for a simulation. In Figure 2.33, we display the researcher settings for which the black box is only allowed to interfere with the steering, even though the participant can control everything.

As shown in Figure 2.34, the participant is given full control over all car controls, whilst the black box is given none. This combination of settings will result in a simulation context similar to the no signals intervention type, regardless of what intervention type is actually chosen in the researcher menu. The black box cannot interfere with any of the decision types with these settings. Note that in this scenario, checking the *interventions on/off* checkbox will not make a difference.

Interventions On/Off

Researchers can also allow participants to accept or reject interventions from the AI. By checking the *interventions on/off* checkbox, the participant is given the ability to turn an AI's decision on or off with a button on the steering wheel. If the checkbox in the researcher menu is not checked, the participant will not be able to turn the AI's interventions off.

Chapter 3

Using DBeaver

3.1 Basic Functionalities

3.1.1 Application View

When opening DBeaver, there should be an application view similar to Figure 3.1. The DBeaver wiki provides more explanation on DBeaver's functionalities (see `https://github.com/dbeaver/dbeaver/wiki`).

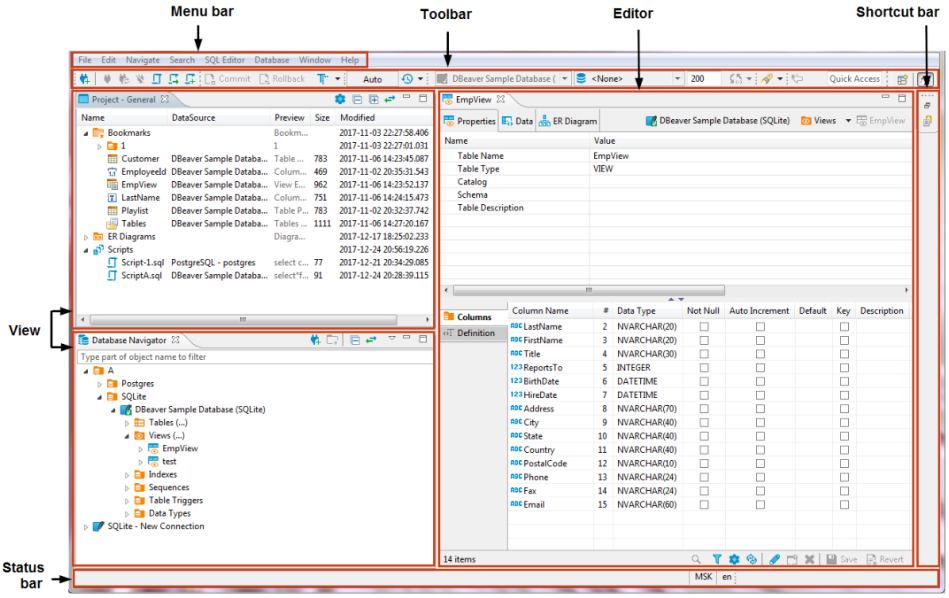

Figure 3.1: Overview of the DBeaver interface. Picture is taken from the DBeaver wiki (see `https://github.com/dbeaver/dbeaver/wiki/Application-Window-Overview` for the exact image).

- *Menu Bar*: the main toolbar which shows the following:
  - The *File* menu contains menu items for the creation of files, folders, projects, database connections, database projects, and ER diagrams as well as to *Import* and *Export* items.
  - The *Edit* menu contains global commands like *Cut*, *Copy*, *Paste*, and *Delete* targeted at the active element.
  - The *Navigate* menu allows navigation through scripts and database objects.
  - The *Search* menu provides options to search among files, database objects and across data.
  - The *SQL Editor* menu is for opening SQL Editor and managing its appearance. An SQL Editor is a tool that contains SQL text-editing features, which include editing functions, triggers and SQL scripts.
  - The *Database* menu allows management of database drivers, connections and transactions, as well as reconnecting to and disconnecting from a database.

  - The *Window* menu includes items to manage the look of DBeaver window: show/hide and minimize/maximize views and editors, display bars, split editors, and manage other preferences.
  - The *Help* menu contains links to information and help resources, as well as menu items to check the version number and availability of updates.

- *Toolbar*: Contains buttons for most basic and frequently used functionalities. The toolbar uses icons which will be described in more detail later in the document.
- *Shortcut bar*: There is one shortcut bar on the right side of the workspace zone. Shortcut bars host shortcuts of views and editors and appear if at least one view or editor is minimized, otherwise they are hidden.
- *View in the workspace*: The most important views are the *Database Navigator*, *Projects* and *Project Explorer*. Open any of these windows by choosing **Window** in the main toolbar (top bar). It is possible to select from all Project Views by then choosing **Show View** and from there choosing the required view. See Figure 3.2 for the view workspace in DBeaver.

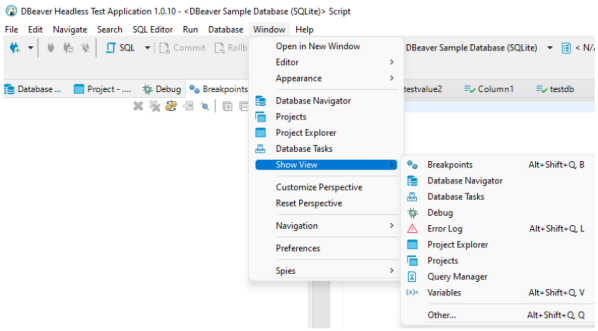

Figure 3.2: View (workspace) in DBeaver.

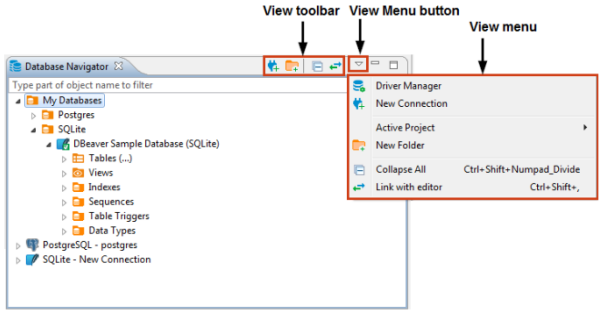

Figure 3.3: View the toolbar and the menu in DBeaver.

Each view offers their own view toolbar and view menu. See Figure 3.3 for the view toolbar and view menu.

3.1.2 Search

In the main toolbar, choose **Search → Search...**, as shown in Figure 3.4. This will open a new window in which to search for files (in the whole application), names inside a database or table, metadata of databases, etc. There are also different options on how to search (whole string, case sensitive, etc.).

3.1.3 Bookmarks

Any Database Object (table, column, variable, etc.) can be bookmarked by selecting it and then pressing `Ctrl+Alt+Shift+D`. This will open the bookmark window in which you can save any bookmarks (see Figure 3.5), which allows you to quickly open Database objects from the *Project* or *Projects explorer view*.

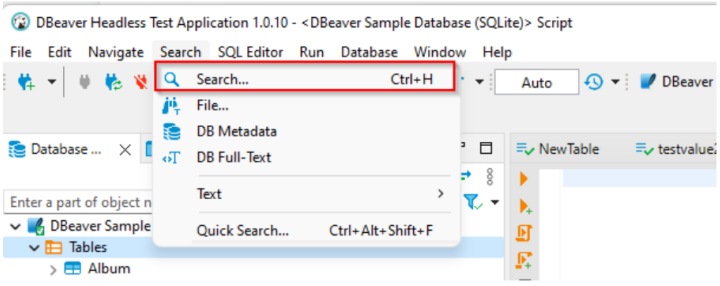

Figure 3.4: Search... option in the main toolbar in DBeaver.

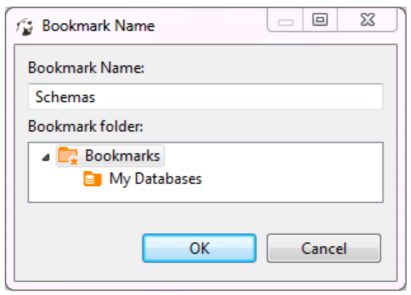

Figure 3.5: Bookmark option in DBeaver.

3.2 Database Connections

In this section we will explain how to create and modify a connection with your database to the DBeaver software, such that you can visualize and modify the data in your database (which is explained in section ??).

3.2.1 Adding a new database connection

To add a new database connection in DBeaver, first click the *New Database Connection* button in the toolbar. Figure 3.6 displays where the *New Database Connection* button is positioned in DBeaver. This will prompt you to the menu displayed in Figure 3.7, where you can determine the connection settings for your database. First, the driver of the database should be selected. When interacting with the Simulated Driving Assistance database, this should always be MySQL (1). After selecting the database driver, you can move on to the next screen (2).

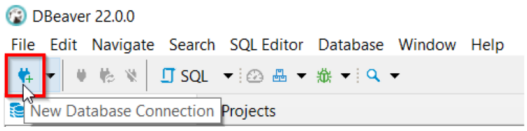

Figure 3.6: New database connection.

Next, the connection settings for the new database should be selected, see Figure 3.8.

- Set the server host by inserting the address of the database in the *Server Host* field (1). For a local database, this should be "localhost". For an external database, this could be an IP address or a domain name.
- Set the port to connect to on the server host by entering the port number in the *Port* field (2).
- Set the username by entering your username in the *Username* field (3). If no additional users have been granted access to the database, the default username is `root`. If additional database accounts have been created, the username of these accounts could be used.
- To ensure a secure connection, go to the *SSL* tab (4) for additional encryption settings.
- Click on the *Finish* button (5) to establish the connection to your database.

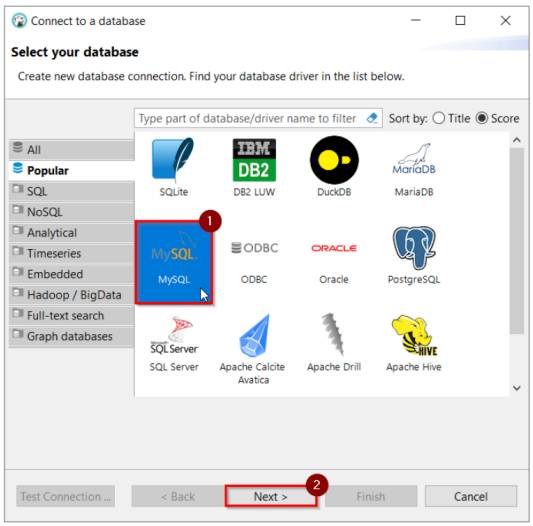

Figure 3.7: Select your database in DBeaver.

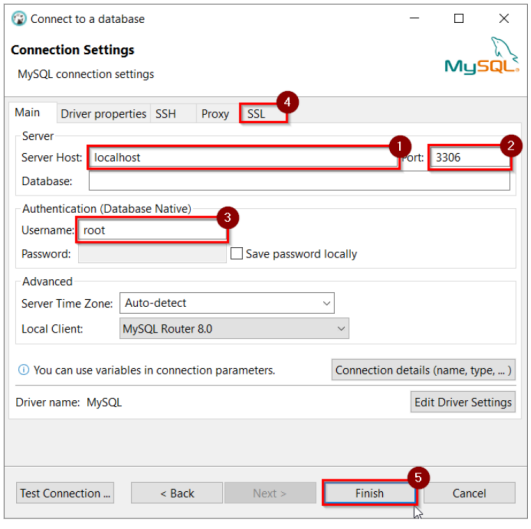

Figure 3.8: Connecting to your database in DBeaver.

3.2.2 Editing an existing database connection

You can edit the connection settings of an existing database by selecting your database in the *database navigator* (see Figure 3.9). Right-clicking this will show you the *Edit Connection* option. Alternatively, you could select the database in the *database navigator* and press the `F4` key.

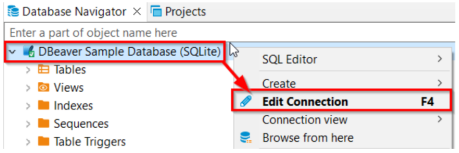

Figure 3.9: Editing an existing database.

3.2.3 Connecting to a database

You can connect to a database that had already been saved by you in DBeaver. First, select the database you want to connect to in the *Database Navigator* menu (see Figure 3.10). Click on the *Connect* button in the toolbar or right-click the database in the *database navigator* and select *Connect*. If the database does not have a password, a connection to the database will be established immediately.

If your database is password-protected, a new menu will show up in order to ask for one's password. Type in the password of the corresponding account and press the *OK* button to establish a connection. If necessary, a different username could also be set in this menu.

3.2.4 Trouble Shooting

If for some reason you are having trouble with your connection to the database, it may help to invalidate the current connection and reconnect. To do this, select the database to reconnect to in the *Database Navigator menu*. Next, click the *Invalidate/Reconnect* button in the toolbar (see Figure 3.11). Alternatively, you could right-click the database in the database navigator and select *Invalidate/Reconnect*. When reconnecting this way, it is not required to re-enter the password.

3.2.5 Disconnecting from a database

After you have finished working with the database, disconnect DBeaver from the database. First, select the database to disconnect to in the *Database Navigator menu*. Next, click the *Disconnect* button in the toolbar (see Figure 3.12). Alternatively, you could right-click the database in the database navigator and select *Disconnect*. A running database connection will be automatically disconnected when the program is closed.

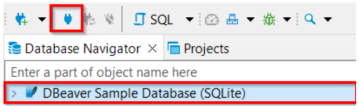

Figure 3.10: Database Navigator menu.

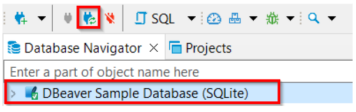

Figure 3.11: Invalidating the current connection.

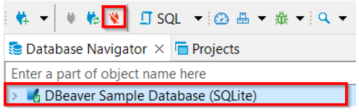

Figure 3.12: Disconnecting from a database.

3.3 Connection Functionalities

In this section we will explain what functionalities are available in DBeaver after establishing a connection with your database. This guide will go into detail about how to visualise and change your data. To establish a connection with a database follow the guide in section **??**.

3.3.1 Database Object Editor

In the *Database Navigator* panel you can open any database object (e.g. the table, a column, or a variable). To do this, either double click your preferred object or select the object and press *Enter*. This opens the *database object editor* (see figure 3.13). It contains a database object path, the contents of your database object, and a toolbar. The toolbar contains different icons based on your selection in the content area. Figure 4.1 in appendix A gives an overview of the names and functionality of the icons. Furthermore, the database object editor has several visualisation tabs for showing the *properties*, the *data*, and the *ER-Diagram* of your database.

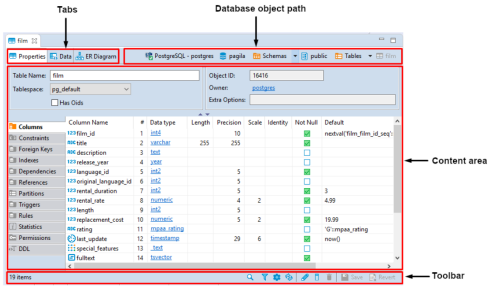

Figure 3.13: Database object Editor in DBeaver.

3.3.2 Data-editor

When selecting the *Data* tab from figure 3.13, you will arrive at the data editor (see figure 3.14). This editor has three different toolbars: a top toolbar, a side toolbar and a bottom toolbar. Appendix A provides an overview of the names and functionalities of all icons in these toolbars.

The bottom toolbar shows the number of rows fetches. By default, this number is set to 200, but it could also be disabled by pressing `Windows → Preferences`. The *fetch all rows* button and *fetch next page for results* button allow you to retrieve more data points. Furthermore, you can go to a specific line in a table by right clicking it and choosing `navigate → Go to line` on the context menu.

To quickly navigate to tables using foreign keys either right-click a cell and select `Navigate → Referencing tables → <table name>`, or choose the cell with a navigate link icon and click `Navigate → Navigate link`.

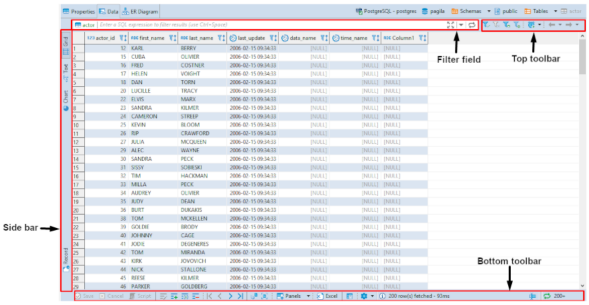

Figure 3.14: Database editor

In the side toolbar it is possible to change the view of the data to either a *Grid* or *Text*. See Figure 3.15 and Figure 3.16. If you have a large amount of columns, than you can also press the *Record* button (at the bottom of the sidebar) to get a view like in Figure 3.17

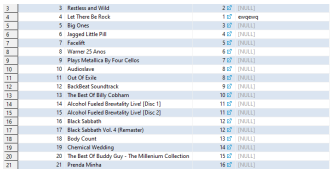

Figure 3.15: Grid view of a Database table

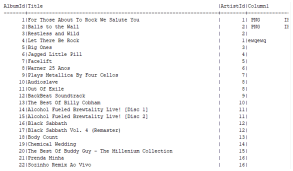

Figure 3.16: Text view of a Database table

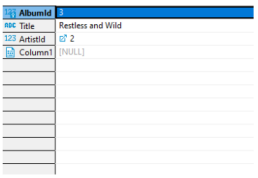

Figure 3.17: Record view of a Database table

Refreshing the data can be done by either right clicking the data table and choosing Refresh or by pressing the refresh button. Alternatively, you can choose or set a custom auto-refresh by pressing the refresh button shown in Figure 3.18, which will refresh the data every x seconds. To generate an SQL statement, select one or more rows then right click it (see Figure 3.19). From here, you can choose `Generate SQL → <your SQL command>` to generate a valid SQL command.

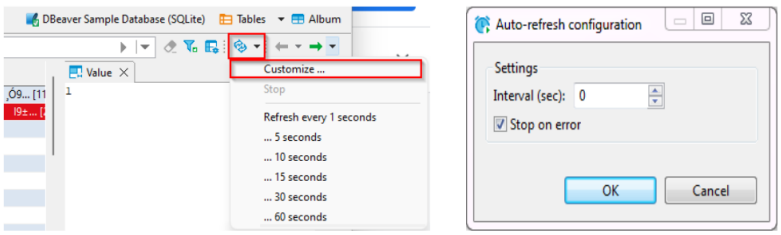

Figure 3.18: Custom auto refresh function

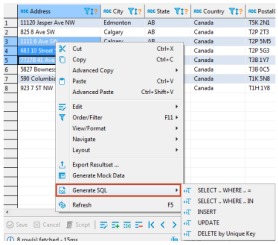

Figure 3.19: Generating SQL scripts

3.3.3 Context Window

Every data point (cell) in the data table can be right clicked to get a context window as shown in Figure 3.20. As some parts in this window are not intuitive, an explanation is provided:

- **Advance Copy**: opens advanced copy sub menu that allows copying data with preset formatting parameters
- **Advance Paste**: pastes several valued delimited with a tabulation or line break;
- **Edit**: for CLOB/BLOB data format it opens the contents of the cell in a new tab, otherwise, it opens a properties window for the cell.
- **Navigate**: Opens a sub menu that helps users navigate throughout the data table.
- **Order/Filter**: Displays a sub menu that allows selecting filter criteria for the data. The sub menu contains the most common filters that can be applied to the cell in focus. By default, DBeaver filters data by sending a request to the server (the Server-side results ordering checkbox selected). To filter data on the client side using DBeaver‘s internal algorithm, clear the checkbox.
- **Export Data**: Opens the data Transfer Wizard that guides you through the steps to select a format and export data. NOTE: The system exports the whole result set including records that are not visible in the screen and preserves all applied data filters and ordering.

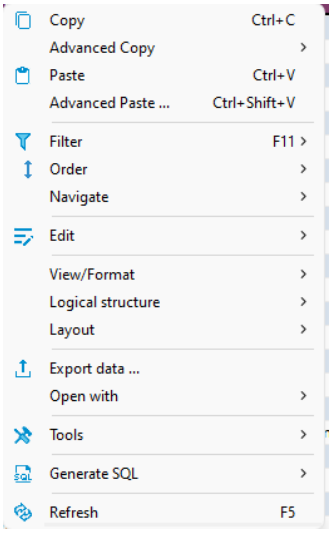

Figure 3.20: Context window of a data point

3.3.4 SQL editor

It is possible to use SQL scripts in DBeaver. To open one either press `F3` in the database editor or click on the SQL-editor in the main toolbar and choose any of the script (or console) commands (Figure 3.21). The script editor allows for simple auto-complete when writing a known SQL command or a known database variable (tablename, columnname, etc.).

You can auto generate standard SQL commands (e.g. count, condition, order) by pressing `Ctrl+Alt+Space` and then choosing the template SQL code you want to use. After you choose a template, you can change the initializing variables. Furthermore, by pressing `Ctrl+Space` you are able to search for objects in the database based on their name (see Figure 3.22).

You can change the format of SQL commands in a script by right clicking in an SQL script and choosing `Format → Format SQL` (see Figure 3.23). Moreover, you are able to comment out a line by writing – at the start of a line/command. If you want to comment out a block of SQL lines/commands, then you can encase the code inside `/* */`.

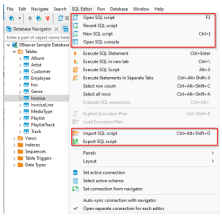

Figure 3.21: different options of opening scripts

Figure 3.22: An example how you could use the SQL assist function

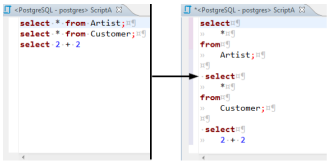

Figure 3.23: Format change of the SQL commands

It is possible to have multiple SQL commands in one file. For this reason, you have multiple execution possibilities, which can be quickly accessed by the sidebar (see Figure 3.24). In the result panel, the result of the SQL command will be written as `Result <A> (<B>)`, where A is the index of the query (redoing the query in a different result tab will increase the index number), and B is the index of the result set of the query (if you execute a whole script, in what order they get returned). By either right clicking in a script and selecting `execute → export from Query` or by pressing the *Export* button in the result tab (bottom toolbar) you can export the result data with your preferred settings.

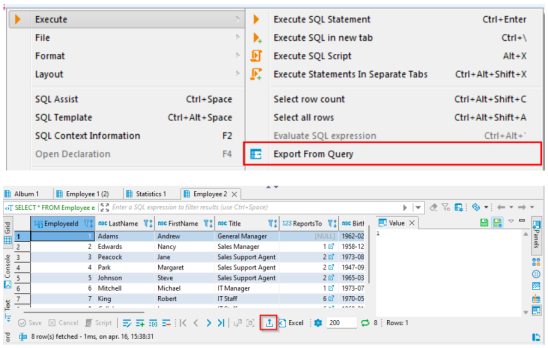

Figure 3.24: Two different ways how you can export the data gotten from SQL commands

3.4 Extra Functionalities

DBeaver has a set of interesting extra functionalities that allow for more flexibility while working with a Database.

3.4.1 Query manager

The query manager is a view that keeps track of all queries executed in the current session. It can be activated through clicking on *Window*, hovering over *Show View*, then selecting **Query Manager** (see figure 3.25). By default, it is positioned in the bottom right of the program. It displays statistics about the executed queries (see figure 3.26).

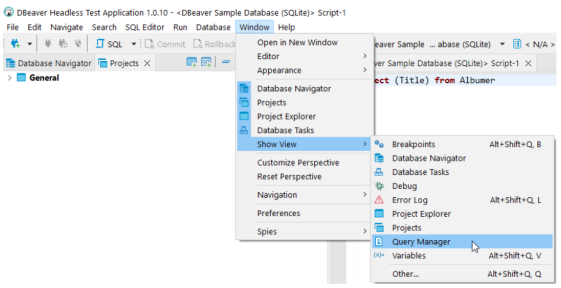

Figure 3.25: Button on how to open up the Query Manager

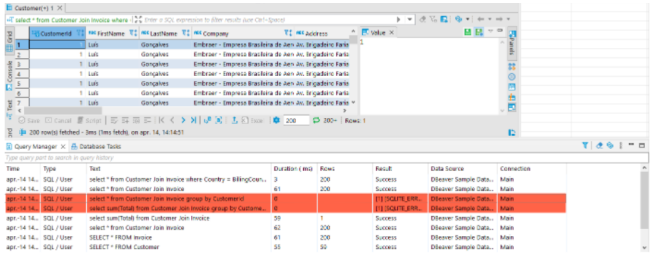

Figure 3.26: Panel with statistics of the recently executed queries on the database

The query manager can be edited by clicking on the **Filter** button in the bottom right. In the menu that pops up you can set the properties of the Query Manager, such as the types of queries and objects shown in the Query Manager, as well as where logs should be stored (if at all) (see figure 3.27).

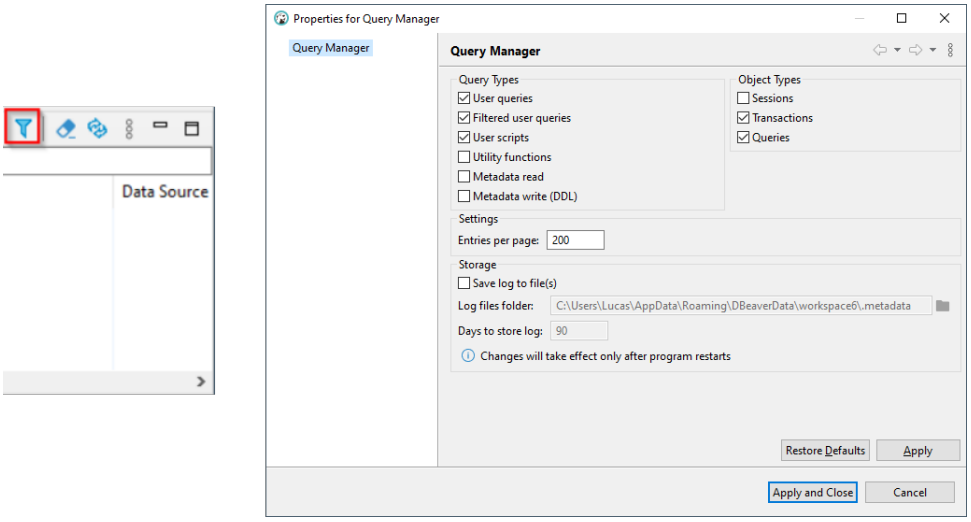

Figure 3.27: How to open and use the query filter to filter the query statistics

3.4.2 Entity-Relationship diagram

The entity-relationship diagram view is accessible in the database object editor. Double click a database table in the database navigator to open this view. The ER-diagram view is in the third tab of this view. In this view, the currently selected table is displayed, as well as any tables with which this table has a relationship.

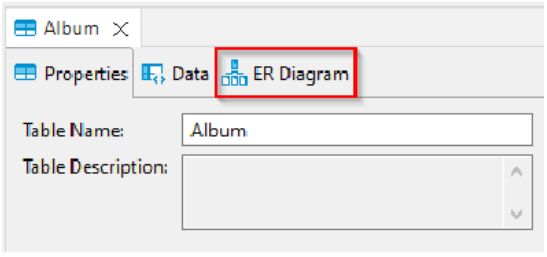

Figure 3.28: Button to open the ER-diagram

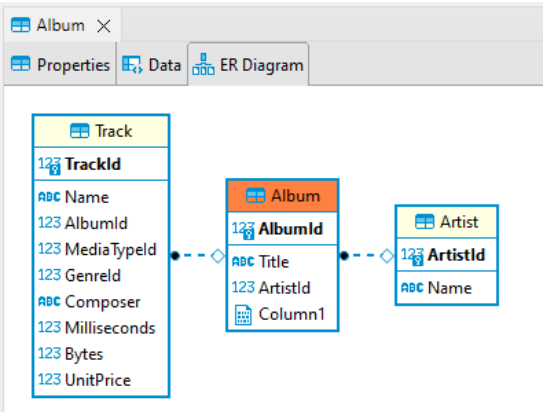

Figure 3.29: An example of an ER-diagram

Selecting an element of a table displays properties about this element on the right of the database object editor. The same holds for the relationships between tables. You can move through the ER-diagram by double clicking another table in the view. The diagram can be exported in a multitude of formats, or printed, by using the three buttons on the bottom of the view (see figure 3.30).

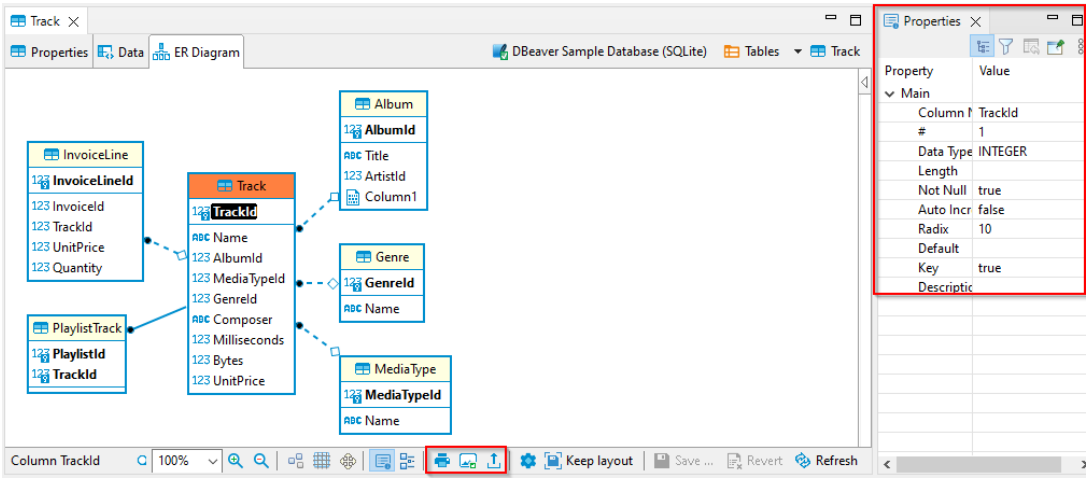

Figure 3.30: An example of an ER Diagram after you double click on another table in the ER Diagram.

By clicking on the tables button in the top right, you can reach two tabs detailing the entire database, the rightmost one of which is the ER-diagram for the entire database, not just a table and its connected tables (see figure 3.31). Note that this functionality is for exploratory purposes. You can move elements of the diagram, but you cannot make changes to the underlying structure, as these diagrams represent the actual database that you're working with. The SQL editor provides that functionality if you want to make a change manually, while the database navigator provides a visual interface that generates the required SQL statements.

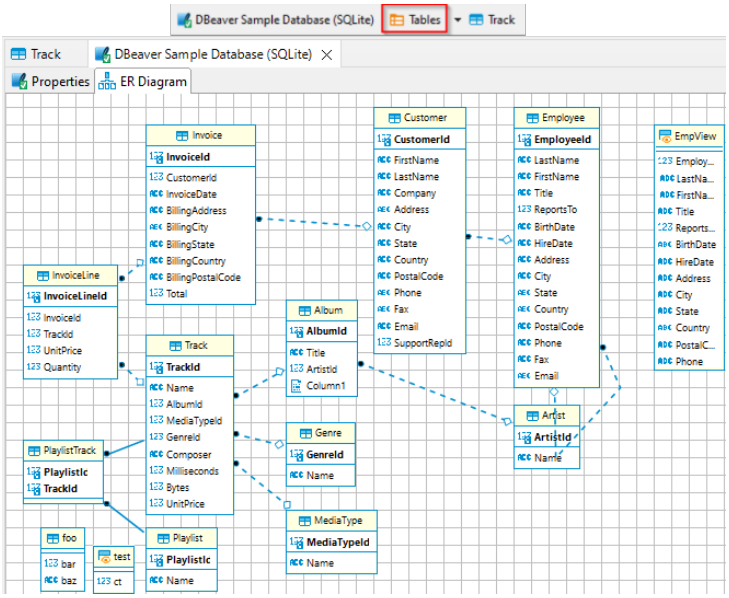

Figure 3.31: An example of a complete ER diagram with an entire database

3.4.3 Tasks

DBeaver provides functionality for automation of tasks, allowing you to press only one button as opposed to typing/selecting relevant values again and again. These tasks are accessed through clicking on **Database**, then hovering over **Tasks** (see figure 3.32).

In this sub-menu, you can select any of your defined tasks to automatically run them. Additionally, you can open the Database Tasks view, which opens in the bottom right (see figure 3.33). This view lists your tasks, as well as previous execution logs. Double clicking a task here allows you to edit the task. You can run a task by clicking on the Proceed button (see figure 3.34).

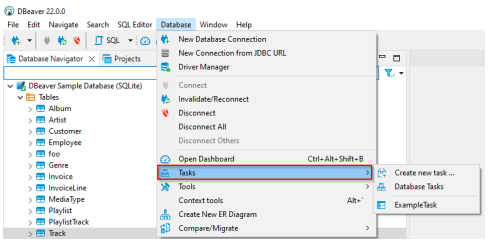

Figure 3.32: How to open the tasks panel

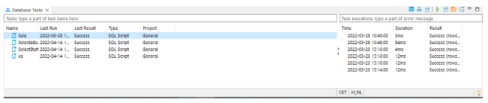

Figure 3.33: An example of how the Database Task view looks like

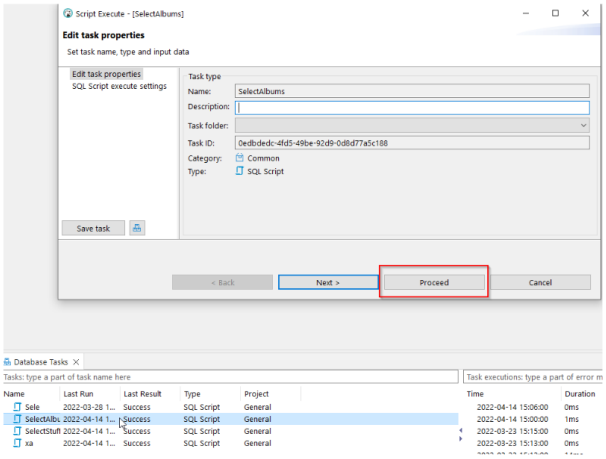

Figure 3.34: Window in which you can edit a task

Tasks have three possible types. Imports, exports and scripts. Imports allow you to choose a number of tables. You must then choose a file-type or database and a location where the content is placed. The task will import whatever is placed there into the table(s).

Exports allow you to choose a number of tables. You must then choose an export type and a location to save the export. The task exports the tables into this location. You can supply queries on this data before export.

Scripts are placed in (a) project folder(s) similar to database connections. Script tasks are given a number of these scripts, which are then executed. Note that scripts that purely select elements from tables do not display their results in a result view, as would happen upon running a script from the SQL console. If you wish to see such results, open the script in the console instead of running the task.

Extensions

DBeaver can be extended with downloadable extensions. To install extensions, click **Help**, then **Install New Software** (see figure 3.35). Extensions have a P2 repository url. Paste this url into the Work with field and select the packages you want to install (see figure 3.36).

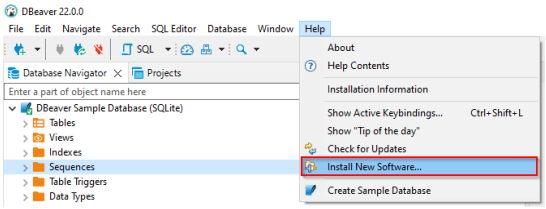

Figure 3.35: Button to add extensions in DBeaver

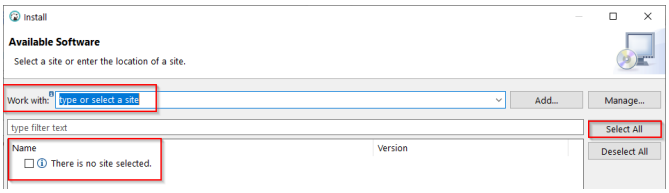

Figure 3.36: Install new software menu

You can then select the specific components in the selection field, or alternatively, select all components by using the button on the right. Finish installation by clicking **Next**, followed by **Finish**. Restart DBeaver to have the changes take effect.

Extensions include:

- Office Integration: `https://dbeaver.io/update/office/latest/`, adding support for exporting in Office formats.

- Debugger support: `https://dbeaver.io/update/debug/latest/`, adding support for SQL debugger in PostgreSQL
- SVG support: `https://dbeaver.io/update/svg/latest/`, adding support for exporting entity-relationship diagrams in SVG format.
- Advanced SSH libraries: `https://dbeaver.io/update/sshj/latest/`, adding support for more SSH key formats.
- Git Integration: Not any specifically hosted by DBeaver. Their wiki recommends Oxygen: `http://download.eclipse.org/releases/oxygen/` or other Eclipse version items.

Simple view

In the database navigator, a couple database properties are visible, as shown in figure 3.37. If instead one is not interested in displaying some of these, the view can be simplified: right click the database connection (highlighted in red), then hover over **Connection View** and select **Simple** (see figure 3.39). This results in a view where only the tables and views are visible (see figure 3.38).

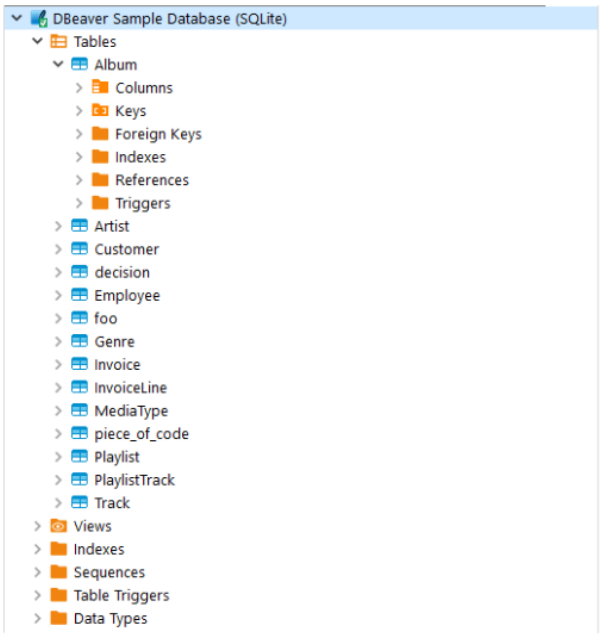

Figure 3.37: The standard view DBeaver shows when you use the Database Navigator

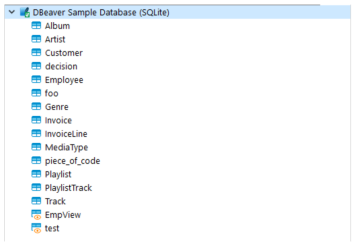

Figure 3.38: A simple view that DBeaver can show when using the Database Navigator

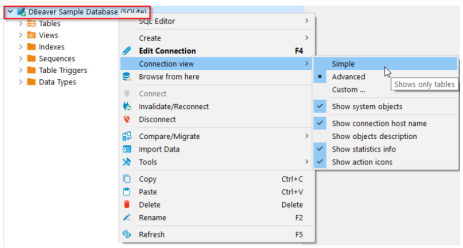

Figure 3.39: The way you can change the normal (advanced) view to a more simple view

3.5 Updating DBeaver

Guide for updating DBeaver

3.6 Database Structure

In this guide, the structure of the DAISI database will be introduced. Figure 3.40 shows an entity relationship diagram of the database.

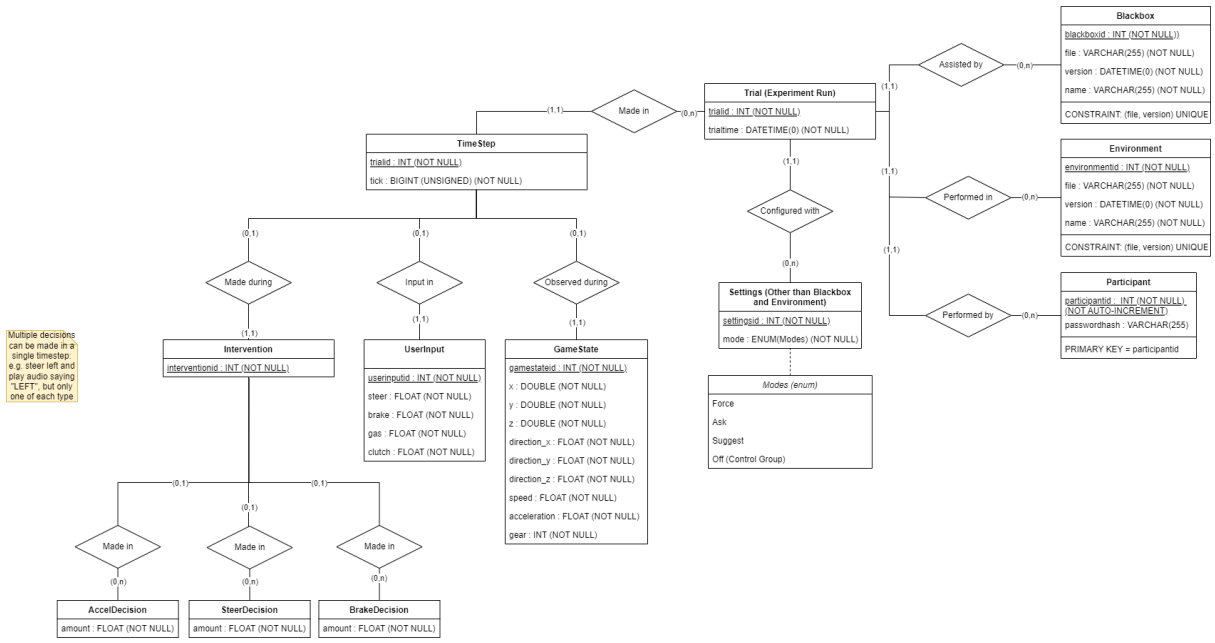

Figure 3.40: Entity Relationship Diagram of the DAISI database

Every experiment is saved with a Trial at the root. A trial references the black box, environment, participant and settings used for the experiment. Each of these is only saved once per unique entry, and referenced with the same ID across different trials if it is identical.

Besides the Blackbox, Environment, and Participant entities which should be mostly self-explanatory, a trial also includes a reference to a single Settings entity. These settings refer to specific trial settings that a researcher may have configured for the experiment, such as the intervention type. Once again, a Settings entity is always unique and its ID may be used across different trials if they used the same settings.

A TimeStep entity represents a single data time step saved to the database, and saves only the ID of the trial it was made in and the tick in the simulation the data was saved at. All data saved during a tick refers to one of the entries in the TimeStep table to show when it was saved and in which trial. The main purpose of this table is to enforce the constraint that every data entry should reference a real tick.

Three different types of data can be saved during a trial: interventions, user input, and game state data. Every time step can have at most one of each stored, but a time step may be stored without one of the data types as well. This may happen in certain scenarios such as when the storage of user input is disabled during a trial or when the AI does not make a decision on a certain time step.

A User Input entity includes attributes for each of the user input values that is saved during the simulation. These values represent the original user input before being influenced by the decision of the driving assistance algorithm. The meaning of every variable is further explained in Guide 2.5.4.

A Game State entity includes attributes for each of the game state values that is saved during the simulation and mostly refer to the current state of the car. This includes the position of the car in the environment, the direction the car is moving in, the speed and acceleration of the car, and the gear of the car. The meaning of every variable is further explained in Guide 2.5.3.

An Intervention entity represents a decision made by a black box algorithm. On its own it does not contain any values, but merely represents the fact that an algorithm replied during this time step. If an Intervention entity (represented by the intervention_id) is not referenced by a Decision, this means that the black-box algorithm decided to not make any interventions.

The AccelDecision, SteerDecision and BrakeDecision represent the algorithm's decision to accelerate,

steer or brake respectively. All three reference an Intervention entity to show when they were made. Each of them include a value to represent how much the algorithm wants to accelerate or brake, or which direction to steer (as further explained in Guide 2.5.5).

3.6.1 Example Queries

Using the database structure outlined above, some examples of database queries are provided here.

This query will select all made decisions during a certain trial. The amount of each decision will be placed in a separated column, with a result of `NULL` if this decision was not present. By using a `LEFT JOIN` on the `Intervention` table, the times where a black box returned an empty decision will also be returned.

```
SELECT i.intervention_id AS intervention_id   ,
           sd.amount                  AS steer_amount      ,
           ad.amount                  AS accelerate_amount ,
           bd.amount                  AS brake_amount
        FROM Intervention i
        LEFT JOIN SteerDecision sd
            # The primary key of Intervention is intervention_id, and can thus be used
            ↪  for joining
               ON sd.intervention_id = i.intervention_id
        LEFT JOIN AccelDecision ad
               ON ad.intervention_id = i.intervention_id
        LEFT JOIN BrakeDecision bd
               ON bd.intervention_id = i.intervention_id
        WHERE i.trial_id = <trial_id> # Replace <trial_id> with the ID of the trial
        ↪  you want to investigate
```

This query will select all instances where the user steered the opposite way compared to the black-box algorithm's decision.

```
SELECT
        ts.trial_id AS trial_id ,
        ts.tick     AS tick     ,
        sd1.steer   AS decision_steer ,
        ui.steer    AS user_steer
        FROM TimeStep ts
        # Use inner join to discard any time steps that did not save steering
        ↪  decisions
        INNER JOIN (
                # Select all steering decisions
                SELECT
                        i.tick AS tick ,
                        i.trial_id AS trial_id ,
                        sd.amount AS steer
                        FROM Intervention i
                        # Use inner join to discard any interventions that did not
                        ↪  save a steering decision
                        INNER JOIN SteerDecision sd
                        ON i.intervention_id = sd.intervention_id
                ) sd1
                # The primary key of TimeStep is a combination of trial_id and tick.
                ↪  Use both to join.
            ON sd1.tick = ts.tick AND sd1.trial_id = ts.trial_id
        # Use inner join to discard any time steps that did not save user input as
        ↪  well
        INNER JOIN UserInput ui
            ON ui.tick = ts.tick AND ui.trial_id = ts.trial_id
```

```
        WHERE sd1.steer * ui.steer < 0 # Multiplying two numbers where both are
        ↪  negative returns positive again. Negative numbers are only if one of the
        ↪  two values is negative.
```

This query will select the trial id, starting time, black-box name, environment name, and (hashed) participant ID from all trials ran between the start and end date.

```
SELECT
        t.trial_id ,
        t.trial_time ,
        p.participant_id ,
        e.name AS environment_name ,
        b.name AS blackbox_name
        FROM
        Trial t
        LEFT JOIN Participant p
            ON p.participant_id = t.participant_id
        LEFT JOIN Environment e
            ON e.environment_id = t.environment_id
        LEFT JOIN Blackbox b
            ON b.blackbox_id = t.blackbox_id
        WHERE trial_time BETWEEN '<YYYY-MM-DD hh:mm:ss>' AND '<YYYY-MM-DD hh:mm:ss>' #
        ↪  Replace <YYYY-MM-DD hh:mm:ss> with the dates and times you want to receive
        ↪  data between
```

This statement will create a table for participant data if no table with the name `ParticipantData` exists yet. In this example, we save the participant's name, birth year, gender, and whether they have a driving license. As participant data is not saved by DAISI, a researcher would have to insert this data themselves using the participant's ID.

```
CREATE TABLE IF NOT EXISTS ParticipantData (
    participant_id VARCHAR(255) NOT NULL,
    first_name TEXT,
    last_name TEXT,
    birth_year YEAR,
    gender ENUM('M', 'F', 'X'),
    has_driving_license BOOL,
    CONSTRAINT participant_id_primary_key PRIMARY KEY (participant_id),
    CONSTRAINT data_participant_id_foreign_key FOREIGN KEY (participant_id) REFERENCES
    ↪  Participant(participant_id)
)
```

This query will select all trial IDs, along with the last name of the participant, for trials performed by participants born before the year 2000 with a driving license. This query uses the `ParticipantData` table as shown in the previous example.

```
SELECT t.trial_id, pd.last_name FROM Trial t
        LEFT JOIN ParticipantData pd
        ON t.participant_id = pd.participant_id
        WHERE pd.birth_year < 2000 AND pd.has_driving_license
```

Chapter 4

Appendix A: DBeaver Icons

4.1 DBeaver Icons

This section shows explanations of the icons used in DBeaver. Figure 4.1 explains the toobar icons, Figure 4.2 shows the bottom toolbar icons, Figure 4.3 shows the top toolbar icons, Figure 4.4 shows the side bar icons, and Figure 4.5 shows the SQL side bar icons.

Button	Name	Description
	Search items	Displays a search field next to the button: - Type in the search combination - the content updates dynamically - To remove the filter, click the cross icon next to the search field
	Filter settings	Opens the Filtering window which allows setting a custom filter
	Configure columns	Opens the Configure columns dialog box in which you can select the columns to display or hide in the current view
	Refresh the selected items	Depending on the database type, refreshes either the current item or its parent or the whole database object – reloading data from the database
	View	Opens an editor/viewer for the item currently in focus
	Create new [items]	Creates a new item of the same type as currently displayed in the open view, for example, a column
	Delete database object	Deletes the item currently in focus
Save	Save the current contents	- Same as the Save button on the application main toolbar - Same as Ctrl+S - Opens the Persist Changes window that allows saving changes in the currently open sub-entity NOTE: DBeaver recommends saving work after each change.
Revert	Revert to the last saved state	Reverts all changes made to the whole database object to the last saved state
	Load from file	- Allows selecting a file from the file system - Disabled if the SQL code is read-only
	Save to file	Allows saving the current SQL code to a file
	Open in SQL console	Opens the SQL code in an SQL Editor

Figure 4.1: Property icons

Button	Name	Description Chart_button
Save	Save	Saves all unsaved changes to the data such as adding, duplicating, deleting rows, inline editing of values.
Cancel	Cancel	Discards all unsaved changes to the data
Script	Script	Opens the Preview Changes window in which you can see changes that you have made to the data.
	Edit cell value in separate dialog/editor	Opens the cell in focus for editing in a separate editor or dialog box.
	Add new row	Adds a new empty row below the current row.
	Duplicate current row	Copies the current rows and pastes the copy below the current row.
	Delete current row	Colors the rows in focus in red to mark them for deletion.
	Move to first row	Moves the focus (highlighting) from the current to the first row of the table
	Move to previous row	Moves the focus (highlighting) from the current to the previous row of the table
	Move to next row	Moves the focus (highlighting) from the current to the next row of the table
	Move to last row	Moves the focus (highlighting) from the current to the last row of the table
	Fetch next page of results	Fetches the next portion of data (next N rows) making it ready for display.
	Fetch all rows	Fetches the whole result set making it ready for display, see the Scrolling Results Page section of the Navigation article for more information
Panels	Panels	Opens panels on the right side of the Data Editor, see the Panels for information
	Configure	Opens a dropdown menu with settings
JSON	JSON	- Available in EE version only for MongoDB documents and JSON tables - Switches to JSON view of data
XML	XML	- Available in EE version only for XML tables - Switches to XML view of data
	Generate Mock Data	Available in EE version only. Opens the Mock Data Generator window
	Rows count details	Opens the Status details dialog box showing the timing details of fetching table rows
	Calculate total row count	Calculates the total number of rows in the table

Figure 4.2: Bottom toolbar icons

Button	Name	Description
	Apply filter criteria	Applies filter criteria entered in the filter field above the data table.
	Remove all filters/orderings	Removes all filters and orderings applied to the data
	Save filter settings for current object	Saves the current filter settings for the database object to apply next time when you reopen it in the editor.
	Custom Filters	Opens the Result Set Order/Filter Settings window.
	Configure auto-refresh	Allows configuring data auto-refresh settings.
	Forward and backward - history navigation buttons	Navigate forward and backward in the Data Editor history. The buttons are equivalent to pressing the key combinations: Alt+Left (backward) and Alt+right (forward).

Figure 4.3: Top toolbar icons

Button	Name	Description Chart_button
Grid	Grid	Switches to grid view of data
Text	Text	Switches to plain text view of data
Record	Record	- Same as pressing Tab - Switches the positions of rows and columns so that the columns appear as rows, and the rows hide in one Value column.

Figure 4.4: Side bar icons

icon	keyboard shortcut	Description
	ctrl + enter	executes the SQL command on which the cursor is placed, otherwise will execute the SQL command below the cursor.
	ctrl + / (while not highlighting)	executes the SQL command on which the cursor is placed, otherwise will execute the SQL command below the cursor. The new result will be saved on a new tab in the results panel.
	alt + x	Will execute all SQL commands in a SQL script (different SQL commands will be separated by a ;). All results will be placed in the same result panel under multiple tabs.
	ctrl + alt + shift + x	Will execute all SQL commands in a SQL script and every result will be placed in its separate result panel. (could cause problems with large scripts)

Figure 4.5: SQL side bar icons