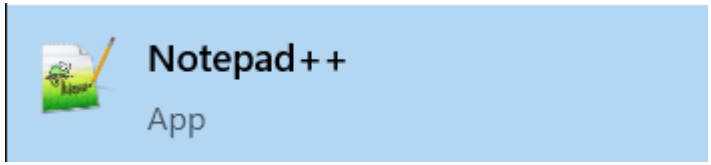
Creating a Database and Inserting Data

Creating a Database

- A database is typically created only once through the lifecycle of the database.
- This can be done from a GUI or on the SQL-level.
  - On this course you will practice creating databases using SQL, because:
    - You will later have to write SQL inside Python code.
    - This way you will have a ready-made script for creating your database. If something goes terribly wrong, it is possible to destroy the whole database and quickly rebuild it.

Creating a Database

- Notepad++ is handy for writing SQL code.
- The ready-written SQL commands can be copied to the MariaDB console.



- It is also possible to write SQL on a graphical user interface.

Database Creation and Deployment

- **create database** statement creates a new database
- **use duckburg** selects the database you want to work with

```
create database duckburg;
use duckburg;
```

Creating Tables

- You can create a new table using the **create table** statement:
  - The following information is defined in the statement:
    - Primary key
    - Foreign keys, if any
    - Data types
    - Extras, can be used for example to define if:
      - a column accepts empty values.
      - the auto_increment feature should be used. Auto_increment creates unique numbers for each row in a column automatically. This makes sure that there are no duplicate values.
- As foreign keys depend on other tables, the tables that foreign keys point to must be created first.

Creating Tables

```
create table duckburger(
ID int not null auto_increment,
first_name varchar(40),
last_name varchar(40),
primary key (id)
);
create table pet(
ID int not null auto_increment,
name varchar(40),
primary key(id)
);
create table owns(
pet_ID int,
duckburger_ID int,
primary key (pet_ID, duckburger_ID),
foreign key (pet_ID) references pet(ID),
foreign key (duckburger_ID) references duckburger(ID)
);
```

Inserting Data

- The **insert into** statement is used for inserting data into the database
  - Because of auto_increment we must write a bit more code to specify which columns the data should be inserted into.
  - The auto_increment column should not be listed.
  - You should not insert any data into an auto_increment column.

Inserting Data

```
insert into duckburger(first_name, last_name)
values("Donald", "Duck"),("Scrooge", "McDuck"),
("Huey", "Duck"),("Magica", "De Spell"), ("Mickey", "Mouse");

insert into pet(name)
values("Bolivar"), ("Pluto"), ("Ratface");

insert into owns(pet_ID, duckburger_ID)
values(1,1),(1,3),(2,5),(3,4);
```

Database Creation Script

You can use this script to quickly create your own duckburg database for practicing:

```
create database duckburg;
use duckburg;

create table duckburger(
ID int not null auto_increment,
first_name varchar(40),
last_name varchar(40),
primary key (id)
);
create table pet(
ID int not null auto_increment,
name varchar(40),
primary key(id)
);
create table owns(
pet_ID int,
duckburger_ID int,
primary key (pet_ID, duckburger_ID),
foreign key (pet_ID) references pet(ID),
foreign key (duckburger_ID) references duckburger(ID)
);

insert into duckburger(first_name, last_name)
values("Donald", "Duck"),("Scrooge", "McDuck"),
("Huey", "Duck"),("Magica", "De Spell"), ("Mickey", "Mouse");

insert into pet(name)
values("Bolivar"), ("Pluto"), ("Ratface");

insert into owns(pet_ID, duckburger_ID)
values(1,1),(1,3),(2,5),(3,4);
```