Deliverable Session 05 - Information and Communication Technology

Rafael Antonio Echevarria Silva

In this session, we will practice using SQLite from Python to create a database that stores information about sensors, variables, and measurements.

Practice 2: Use of SQLite

A. Create a database

To create a new table in SQLite, we use the connection cursor and the 'execute()' method. The general syntax is:

```
c.execute("CREATE TABLE table_name (field1 TYPE1, field2 TYPE2, …)")
```

To define the necessary tables we have the following code:

```
#Creating a new SQLite table for sensors
c.execute("CREATE TABLE sensors (id INTEGER PRIMARY KEY AUTOINCREMENT, name TEXT NOT NULL, description TEXT, virtual INTEGER)")
#Creating a new table for variables
c.execute("CREATE TABLE variables (id INTEGER PRIMARY KEY AUTOINCREMENT, sensor_id INTEGER, name TEXT NOT NULL, description TEXT, units TEXT)")
#Creating a new table for measures
c.execute("CREATE TABLE measures (id INTEGER PRIMARY KEY AUTOINCREMENT, variable_id INTEGER, measure REAL, date TEXT)")
```

Each table defines an **auto-incrementing** id and the fields needed to relate sensors, variables, and their measurements.

B. Setup “sensors” into table

To insert records into the sensors table, we use INSERT statements. For example:

```
sensors = [
# (name, description, virtual)
("temperature", "Temperature sensor", 0),
("pressure", "Pressure sensor", 0),
("humidity", "Humidity sensor", 0),
('Accelerometer', 'Accelerometer sensor', 0),
('Magnetometer', 'Magnetometer sensor', 0),
('Gyroscope', 'Gyroscope sensor', 0),
('Orientation', 'Orientation sensor', 0)
]

for name, desc, virt in sensors:
c.execute("INSERT INTO sensors (name, description, virtual) VALUES (?, ?, ?)",
(name, desc, virt))
```

In this way we add each sensor with its name, description and “virtual” **Binary** if applicable.

C. Setup “variables” into table

Each variable corresponds to a quantity measured by a sensor. To insert variables:

```
variables = [
# (sensor_id, name, description, units)
('1','Pressure', 'Pressure sensor', 'Pa'),
('2','Humidity', 'Humidity sensor', '%'),
('3','Temperature', 'Temperature sensor', 'ºC'),
('4','Magnetometer', 'Magnetometer sensor', 'Gauss'),
('5','X', 'Accelerometer sensor', 'm/s'),
('5','Y', 'Accelerometer sensor', 'm/s'),
('5','Z', 'Accelerometer sensor', 'm/s'),
('6','X', 'Gyroscope sensor', 'rad'),
('6','Y', 'Gyroscope sensor', 'rad'),
('6','Z', 'Gyroscope sensor', 'rad'),
('7','pitch', 'Orientation sensor', 'º'),
('7','roll', 'Orientation sensor', 'º'),
('7','yaw', 'Orientation sensor', 'º')
]

for sensor_id, name, desc, units in variables:
c.execute("INSERT INTO variables (sensor_id, name, description, units) VALUES (?,
?, ?, ?)",(sensor_id, name, desc, units))
```

Here **sensor_id** links each variable with the sensors table.

D. Write measures

To save a measurement in the **measures** table we collect the value and the current date, and execute:

```
from sense_hat import SenseHat
import sqlite3
import datetime
import time

sense = SenseHat()

sqlite_file = "sensor_data.db"
conn = sqlite3.connect(sqlite_file)
```

```
c = conn.cursor()
for i in range(1,1000):

# c.execute("INSERT INTO measures (variable_id, measure, date) VALUES
(variable_id, {0},'{1:%Y-%m-%d %H:%M:%S.%f}')".format(measure_value,
measure_time))

measure_time = datetime.datetime.utcnow()

pressure = sense.get_pressure()
query = "INSERT INTO measures (variable_id, measure, date) VALUES (1, {0},'{1:%Y-
%m-%d %H:%M:%S.%f}')".format(pressure, measure_time)
c.execute(query)
conn.commit()

...
```

Repeated for each sensor reading, thus recording the time series of values.

E. Access to measures and display the values on the screen

To query the latest values for each variable and display them on the screen, we use a query with JOIN and grouping:

```


query = "select sensors.name, variables.name, measures.measure,
max(measures.date), variables.units from sensors, variables, measures where
sensors.id = variables.sensor_id and variables.id = measures.variable_id group by
variables.id"

c.execute(query)

rows = c.fetchall()

for row in rows:
print(row)

conn.close()
```

This displays the most recent value for each variable along with the corresponding date and unit.