notebook.community

EDIT AND RUN

Lesson 3: Create MySQL Database Table for Sensor Data

Learning Objectives:

1. Learn how to access MySQL from Jupyter Notebook to create tables and uses
2. Prepare location to store data collected from IoT devices

Exercise 1: Create a MySQL table to hold the sensor data from our Raspberry Pi

Note: This exercise will use an existing MySQL database server to expedite the process. You can use these notes to create the setup in your own server.

Logon using an admin account and create a table called temps3 to hold sensor data:

Table Design Schema:

```
Field          Description
device        -- VARCHAR, Name of the device that logged the data
datetime      -- DATETIME, Date time in ISO 8601 format YYYY-MM-DD HH:MM:SS
temp          -- FLOAT, temperature data
hum           -- FLOAT, humidity data
```

Load the sql notebook extension:

```
In [3]:

%load_ext sql
```

Connect to the MySQL database instance using and account that has admin access and run SQL to drop/create table:

```
In [6]:

%%sql mysql://admin:admin@172.20.101.81/pidata
DROP TABLE if exists temps3;

CREATE TABLE temps3 (
    device varchar(20) DEFAULT NULL,
    datetime datetime DEFAULT NULL,
    temp float DEFAULT NULL,
    hum float DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
0 rows affected.
0 rows affected.


Out[6]:

[]
```

Check to see that the table was created:

```
In [7]:

%sql show tables;
```

```
2 rows affected.


Out[7]:
```

Tables_in_pidata
temps
temps3

Exercise 2: Create a MySQL user that has access to the new table:

We will create a user called **piuser** that will have limited access to the **temps3** table on the **pidata** database.

We will create a user account called

```
'piuser'@'%'
```

This means the **piuser** account can access the MySQL server from any hostname

Login to the MySQL database using an account that has admin access, then run the code to drop/create the new user:

Note: We will create a new user called **piuser3** with a password **logger** that will have access to the **temps3**

```
In [29]:

%%sql mysql://admin:admin@172.20.101.81

DROP USER IF EXISTS piuser3;
CREATE USER 'piuser3'@'%' IDENTIFIED BY 'logger';
GRANT SELECT, INSERT, DELETE, UPDATE ON pidata.temps3 TO 'piuser3'@'%';
FLUSH PRIVILEGES;
```

```
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.

Out[29]:

[]
```

```
In [30]:

%sql select * from mysql.user;
```

```
12 rows affected.
```

Out[30]:

Host	User	Select_priv	Insert_priv	Update_priv
localhost	root	Y	Y	Y
localhost	mysql.sys	N	N	N
localhost	debian-sys-maint	Y	Y	Y
localhost	rmj	Y	Y	Y
localhost	phpmyadmin	N	N	N
192.168.8.131	pilogger	N	N	N
192.168.8.131	rmj	Y	Y	Y
%	pilogger	N	N	N
%	rmj	Y	Y	Y
%	admin	Y	Y	Y
%	user1	N	N	N
%	piuser3	N	N	N

Check to be sure the new user has access to the new table:

In [32]:

```
%%sql mysql://piuser3:logger@172.20.101.81/pidata
select * from temps3;
```

```
0 rows affected.

Out[32]:

device    datetime    temp    hum
```

Exercise 3: Add test data to new table to confirm the new piuser account can add records to the table

Note: We are still connected to the pidata database using the piuser3@'%' account

```
In [34]:

for x in range(10):
    %sql INSERT INTO temps3 (device,datetime,temp,hum) VALUES('pi-003',date(n
```

```
1 rows affected.
1 rows affected.
1 rows affected.
1 rows affected.
1 rows affected.
1 rows affected.
1 rows affected.
1 rows affected.
1 rows affected.
1 rows affected.
```

Now we can check to see if the data was inserted as expected:

```
In [35]:

%sql SELECT * from temps3;
```

```
10 rows affected.
```

Out[35]:

device	datetime	temp	hum
pi-003	2017-07-23 00:00:00	73.2	22.0
pi-003	2017-07-23 00:00:00	73.2	22.0
pi-003	2017-07-23 00:00:00	73.2	22.0
pi-003	2017-07-23 00:00:00	73.2	22.0
pi-003	2017-07-23 00:00:00	73.2	22.0
pi-003	2017-07-23 00:00:00	73.2	22.0
pi-003	2017-07-23 00:00:00	73.2	22.0
pi-003	2017-07-23 00:00:00	73.2	22.0
pi-003	2017-07-23 00:00:00	73.2	22.0
pi-003	2017-07-23 00:00:00	73.2	22.0

Next we will empty the table so it will be ready for live sensor data:

In [37]:

```
%sql DELETE FROM temps3;
```

```
10 rows affected.
```

Out[37]:

```
[]
```

In [38]:

```
%sql SELECT * FROM temps3;
```

```
0 rows affected.

Out[38]:

device    datetime    temp    hum
```

Content source: richjimenez/mysql-data-raspberry-pi

Similar notebooks:

- lesson-3-mysql-database-for-raspberry-pi-sensor-ata
- lesson-2-jupyter-notebook-for-data-analysis
- add-data-to-table
- Python3_tutorial-checkpoint
- Snippets_MySQL
- lesson-4-save-raspberry-pi-sensor-data-to-mysql-database
- Proto DDL with Core_B-checkpoint
- Python for MySQL_2jan15
- Python3_tutorial
- CSV Data to MySQL for use in VISDOM

notebook.community | gallery | about