Sqlite3_tutorial

May 6, 2018

0.1 Sqlite3 Tutorial

0.1.1 Author: Jiawen Yan

0.1.2 Date: 2018-04-23

0.1.3 Version: 1.0

```
In [2]: # by default, python3 has installed sqlite3, so you could simply import it
        import sqlite3
```

```
In [ ]: # however, to actually browse database, you need to execute following commands in termin
        '''
        sudo add-apt-repository -y ppa:linuxgndu/sqlitebrowser
        sudo apt-get update
        sudo apt-get install sqlitebrowser
        '''
```

0.1.4 Part 1, Basics

```
In [23]: # to connect to database, we need to establish a connection and cursor
         conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
         c = conn.cursor()
         #### your codes ####
         c.close()
         conn.close()
```

1. Create a table in a database

```
In [24]: '''
         Datatypes:
             NULL. The value is a NULL value.
             INTEGER. The value is a signed integer, stored in 1, 2, 3, 4, 6, or 8 bytes dependi
             REAL. The value is a floating point value, stored as an 8-byte IEEE floating point
             TEXT. The value is a text string, stored using the database encoding (UTF-8, UTF-16
             BLOB. The value is a blob of data, stored exactly as it was input.
         '''
         def create_table(conn, c):
             c.execute("""CREATE TABLE IF NOT EXISTS Products(
```

```
                        id INTEGER PRIMARY KEY AUTOINCREMENT,
                        _id INTEGER,
                        url TEXT,
                        sku_id TEXT,
                        domain TEXT,
                        name TEXT,
                        item_name TEXT,
                        parameter1 TEXT,
                        parameter2 TEXT,
                        price REAL ,
                        brand TEXT ,
                        other TEXT) ;""")
            conn.commit()
        conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
        c = conn.cursor()
        create_table(conn, c)
        c.close()
        conn.close()
```

2. Insert Statement

```
In [25]: def insert_data(conn, c, num):
             import pandas as pd
             fin = pd.read_csv("sample_jd_products.csv")
             for i in range(1, num+1):
                 c.execute('''INSERT INTO Products
                 (_id, url, sku_id, domain, name, item_name, parameter1, parameter2, price, bran
                 VALUES ("{}", "{}", "{}", "{}", "{}", "{}", "{}", "{}", "{}", "{}", "{}") ; '''
                 conn.commit()


         conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
         c = conn.cursor()
         insert_data(conn, c, 2000)
         c.close()
         conn.close()
```

3. Select Statement

```
In [5]: def select_data(conn, c, Id):
            ret = c.execute("SELECT sku_id, name, price FROM Products WHERE id = {}".format(Id))
            ret = ret.fetchall()
            if ret:
                print(ret[0])
        conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
        c = conn.cursor()
        select_data(conn, c, Id = 128)
        c.close()
        conn.close()
```

```
('1718026732', "['   15g  ']", '109.0')
```

4. Update statement

```
In [8]: def update_data(conn, c, Id, new_price):
            ret = c.execute("UPDATE Products SET price = {} WHERE id = {}".format(new_price, Id)

        conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
        c = conn.cursor()
        select_data(conn, c, Id = 128)
        update_data(conn, c, Id = 128, new_price=12.0)
        select_data(conn, c, Id = 128)
        c.close()
        conn.close()
```

```
('1718026732', "['   15g  ']", '109.0')
('1718026732', "['   15g  ']", '12.0')
```

5. Delete Statement

```
In [26]: def delete_data(conn, c, Id):
             c.execute("DELETE FROM Products WHERE id = {} ;".format(Id) )

         conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
         c = conn.cursor()

         select_data(conn, c, Id = 999)
         delete_data(conn, c, Id = 999)
         conn.commit()
         select_data(conn, c, Id = 999)

         c.close()
         conn.close()
```

*6. Time efficience

```
In [4]: # how to quickly select data from database?
        from datetime import datetime
        import random

        def many_selects(conn, c, times):
            sku_ids = ["1106257762","1055818589","1434612168","1339190822","1050895058",
                       "1077734754","1391081652","1101196302","1435256636","1222981002"]
            time_start = datetime.now()
            for i in range(times):
                Id = random.choice(sku_ids)
```

```
            ret = c.execute("SELECT sku_id, name, price FROM Products WHERE sku_id = {}".for
        time_end = datetime.now()
        print(time_end-time_start)

    conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
    c = conn.cursor()
    many_selects(conn, c, 200000)
    c.close()
    conn.close()
```

```
0:00:10.874353
```

```
In [14]: # trick: create index
         def remove_index(conn, c):
             c.execute("DROP INDEX sku_id_index ;")
             conn.commit()
         def create_index(conn, c):
             c.execute("CREATE UNIQUE INDEX IF NOT EXISTS sku_id_index ON Products (sku_id);")
             conn.commit()
         conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
         c = conn.cursor()
         create_index(conn, c)
         #remove_index(conn, c)
         c.close()
         conn.close()
```

```
In [15]: conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
         c = conn.cursor()
         many_selects(conn, c, 200000)
         c.close()
         conn.close()
```

```
0:00:05.495829
```

0.1.5 Part2, Intermediate

1. Select Order and Limit

```
In [ ]: '''
        SQL SYNTAX:
        SELECT column-names
          FROM table-name
         ORDER BY column-names
        LIMIT X;
        '''
```

```
In [33]: # select ... order by statement
         # select ... limit 5
```

```
        conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
        c = conn.cursor()
        select_data_order(conn, c)
        ret = c.execute("SELECT sku_id, name, price FROM Products WHERE price > 1000.0 ORDER BY
        ret = ret.fetchall()
        if ret:
            for r in ret:
                print(r)
        c.close()
        conn.close()
```

```
('6100165', "SK-II''30ml+10ml", 1040.0)
('4380118', "SK-II''30ml+10ml", 1040.0)
('2223524', 'SK-II10p ', 1060.0)
('5981507', 'Olay(+++++++)', 1090.0)
('1452961433', 'ST.HERB 15ml/  1-30', 1160.0)
('5203591', 'ST.HERB  ', 1160.0)
('1717795340', "['  ']", 1176.0)
('6100187', "SK-II PITERA''+'' 75ml+15g ", 1180.0)
('1366031', 'Whoo 6315ml6 ', 1210.0)
('2562502', 'Whoo 7336ml+++      ', 1220.0)
```

2. Select Offset top X rows

```
In [35]: # Offset top X rows
```

```
        conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
        c = conn.cursor()
        ret = c.execute("SELECT sku_id, name, price FROM Products WHERE price > 1000.0 ORDER BY
        ret = ret.fetchall()
        if ret:
            for r in ret:
                print(r)
        c.close()
        conn.close()
```

```
('1032200369', 'ST.HERB  1', 1299.0)
('234261', 'Sisley125ml  (', 1299.0)
('5878510', "SK-II''230ml+20g+2p+0.8g+2gx2", 1370.0)
('6100177', "SK-II PITERA''230ml ", 1370.0)
('6088552', "SK-II PITERA''230ml   ", 1370.0)
('5878494', "SK-II''230ml+20g+2p+0.8g+2gx2", 1370.0)
('2574022', "SK-II230mlPITERA''    ", 1370.0)
('16676298276', 'DEWOS  ', 1380.0)
('6100191', 'SK-II50g+15g ', 1450.0)
('1526049919', '  12', 1500.0)
```

3.Select Distinct

```
In [ ]: '''
        SQL SYNTAX:
        SELECT DISTINCT column-name
          FROM table-name
        '''

In [39]: # distinct select, to get unique value(s)
         conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
         c = conn.cursor()

         ret = c.execute("SELECT DISTINCT name, price FROM Products WHERE price > 1000.0 ORDER B
         ret = ret.fetchall()
         if ret:
             for r in ret:
                 print(r)

         c.close()
         conn.close()
```

```
("SK-II''30ml+10ml", 1040.0)
('SK-II10p ', 1060.0)
('Olay(+++++++)', 1090.0)
('ST.HERB 15ml/  1-30', 1160.0)
('ST.HERB  ', 1160.0)
("['  ']", 1176.0)
("SK-II PITERA''+'' 75ml+15g ", 1180.0)
('Whoo 6315ml6 ', 1210.0)
('Whoo 7336ml+++      ', 1220.0)
('ST.HERB  1', 1299.0)
```

4. Select MIN/MAX

```
In [ ]: '''
        SQL SYNTAX:
        SELECT MIN/MAX(column-name)
          FROM table-name
        '''

In [ ]: # select MIN/MAX

        conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
        c = conn.cursor()

        ret = c.execute("SELECT MAX (price) FROM Products WHERE price < 9000")
        ret = ret.fetchall()
        if ret:
```

```
            for r in ret:
                print(r)

        c.close()
        conn.close()
```

5. Select COUNT/SUM/AVG

```
In [ ]: '''
        SQL SYNTAX:
        SELECT COUNT/SUM/AVG(column-name)
          FROM table-name
        '''
```

```
In [46]: # Count
         conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
         c = conn.cursor()
         ret = c.execute("SELECT COUNT (sku_id) FROM Products")
         ret = ret.fetchall()
         if ret:
             for r in ret:
                 print(r)
         c.close()
         conn.close()
```

```
(2000,)
```

```
In [47]: # Count
         conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
         c = conn.cursor()
         ret = c.execute("SELECT SUM (price) FROM Products")
         ret = ret.fetchall()
         if ret:
             for r in ret:
                 print(r)
         c.close()
         conn.close()
```

```
(312363.20000000135,)
```

```
In [48]: # AVG
         conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
         c = conn.cursor()
         ret = c.execute("SELECT AVG (price) FROM Products")
         ret = ret.fetchall()
         if ret:
             for r in ret:
```

```
                print(r)
        c.close()
        conn.close()
```

```
(156.18160000000069,)
```

0.1.6 Part3, Conditions and Logics

1. And/Or/Not logics

```
In [8]: # AND
        conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
        c = conn.cursor()

        ret = c.execute("SELECT sku_id, name, price FROM Products WHERE price < 1000 AND price >
        ret = ret.fetchall()
        if ret:
            for r in ret:
                print(r)

        c.close()
        conn.close()
```

```
('6100179', "SK-II''30ml+10ml", 960.0)
('1304764', 'Whoo 6350ml   ', 958.0)
```

```
In [3]: # NOT
        conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
        c = conn.cursor()

        ret = c.execute("SELECT sku_id, name, price FROM Products WHERE NOT price <1000 ")
        ret = ret.fetchall()
        if ret:
            for r in ret:
                print(r)
                break

        c.close()
        conn.close()
```

```
('2562502', 'Whoo 7336ml+++      ', 1220.0)
```

2. Between

```
In [11]: # NOT
         conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
```

```
        c = conn.cursor()

        ret = c.execute("SELECT sku_id, name, price FROM Products WHERE price BETWEEN 900 AND 1
        ret = ret.fetchall()
        if ret:
            for r in ret:
                print(r)

        c.close()
        conn.close()
```

```
('6100179', "SK-II''30ml+10ml", 960.0)
('1304764', 'Whoo 6350ml   ', 958.0)
```

3.time efficiency

```
In [15]: from datetime import datetime
         # Between ... and
         conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
         c = conn.cursor()
         st = datetime.now()
         for _ in range(20000):
             ret = c.execute("SELECT sku_id, name, price FROM Products WHERE price BETWEEN 900 A
             ret = ret.fetchall()
         et = datetime.now()
         print(et-st)
         c.close()
         conn.close()
         # AND
         conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
         c = conn.cursor()
         st = datetime.now()
         for _ in range(20000):
             ret = c.execute("SELECT sku_id, name, price FROM Products WHERE price >= 900 AND pr
             ret = ret.fetchall()
         et = datetime.now()
         print(et-st)
         c.close()
         conn.close()
```

```
0:00:06.378568
0:00:06.283135
```

4. In statement

```
In [24]: conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
         c = conn.cursor()
```

```
        ret = c.execute("SELECT sku_id, name, price FROM Products WHERE price IN ('100');")
        ret = ret.fetchall()
        print(ret)
        c.close()
        conn.close()
```

```
[('1951395', 'AUPRES  150ml ', 100.0), ('1463206373', "[' +']", 100.0), ('1463206374', "[' +']",
```

```
In [3]: conn = sqlite3.connect("sqlite3_tutorial_JD_data.db")
        c = conn.cursor()
        ret = c.execute("SELECT sku_id, name, price FROM Products WHERE name LIKE '%SK-II%';")
        ret = ret.fetchall()
        for r in ret:
            print(r)
            break
        c.close()
        conn.close()
```

```
('3342868', 'SK-II160ml     ', 560.0)
```