
SQL Compiler Documentation

Π16107

---

1. Συνάρτηση execSQL (database.py)

Δέχεται ως είσοδο ένα SQL ερώτημα, το αναλύει, και στη συνέχεια στέλνει τις παραμέτρους του στην αντίστοιχη συνάρτηση της miniDB.

Φυσικά, κάθε εντολή της SQL χρησιμοποιεί διαφορετικές παραμέτρους. Ο έλεγχος για το είδος της εντολής γίνεται με τη βοήθεια των Regular Expressions.

2. Αρχείο sql_helper.py

Εδώ γίνεται η ανάλυση του ερωτήματος σε παραμέτρους.

Η βασική συνάρτηση που χρησιμοποιείται είναι η parameter_config. Δέχεται ως είσοδο ένα string και 2 λέξεις (word1, word2) και επιστρέφει μια λίστα που περιέχει όλες τις λέξεις ανάμεσα σε word1 και word2.

πχ:

parameter_config("SELECT ID,dept_name FROM instructor WHERE salary>8000", 'SELECT', 'FROM')

Σημασία: Βρες τις στήλες που ζητούνται στο ερώτημα.

Επιστρέφει: ['ID', 'dept_name']

Αντί για «word2», μπορούμε να εισάγουμε μια λίστα από λέξεις. Μερικές φορές δεν μπορούμε να είμαστε σίγουροι για τη $2^η$ λέξη. Για παράδειγμα, σε εντολή SELECT μπορεί να υπάρχει όρος 'WHERE', 'ORDER BY', 'LIMIT' ή συνδυασμός των παραπάνω. Επομένως στην περίπτωση αυτή ψάχνουμε τις λέξεις ανάμεσα σε word1 και οποιοδήποτε keyword στο word2.

Για να επιτευχθεί αυτό, χρησιμοποιείται και η συνάρτηση create_pattern που επιστρέφει το RegEx pattern που θα χρησιμοποιηθεί.

πχ:

parameter_config("SELECT ID,dept_name FROM instructor WHERE salary>8000 ORDER BY name LIMIT 3", 'FROM', ['INNER JOIN','WHERE','ORDER BY','LIMIT'])

Σημασία: Βρες το όνομα του πίνακα στο ερώτημα.

Επιστρέφει: ['instructor']

Σημειώσεις:

- Οι λέξεις word1 και word2 δεν είναι case sensitive μιας και είναι SQL keywords.
- Αν το word2 είναι κενό σημαίνει «Βρες όλες τις λέξεις μετά το word1 (μέχρι το τέλος)».
- Εάν υπάρχουν επιπλέον κενά/tabs ανάμεσα στις λέξεις, αγνοούνται.
- Σε αντίθεση με τους κανονικούς SQL Compilers, τα ονόματα των στηλών μπορούν να χωρίζονται είτε με κόμμα είτε με κενά.

  πχ: SELECT ID dept_name FROM ...

  Ένας SQL compiler επιστρέφει μόνο τη στήλη 'ID'. Αυτή η συνάρτηση αναγνωρίζει και την 'dept_name'.

- Αν το word2 είναι λίστα, έχει σημασία η σειρά των λέξεων. Το RegEx pattern που δημιουργείται, δίνει προτεραιότητα στην πρώτη λέξη που θα εντοπιστεί (με τη σειρά που δίνεται στη λίστα).
  πχ: SELECT * FROM instructor WHERE salary>8000 ORDER BY name

  Χρησιμοποιώντας "parameter_config(query, 'FROM', ['WHERE', 'ORDER BY'])" θα επιστραφεί ['instructor'] (το αποτέλεσμα που θέλουμε).

  Χρησιμοποιώντας "parameter_config(query, 'FROM', ['ORDER BY', 'WHERE'])" θα επιστραφεί ['instructor', 'WHERE', 'salary>8000'].

  Επομένως, πρέπει να προσέχουμε τη σειρά των λέξεων στη λίστα word2.

Επιπλέον, το αρχείο sql_helper.py περιέχει βοηθητικές συναρτήσεις για την ανάλυση κάθε εντολής ξεχωριστά. Κάθε τέτοια συνάρτηση δημιουργείται με τέτοιο τρόπο ώστε να επιστρέφει τις παραμέτρους που χρειάζονται στην αντίστοιχη συνάρτηση της miniDB.

πχ: Αν η execSQL εντοπίσει ότι πρόκειται για εντολή SELECT, οι παράμετροι του ερωτήματος θα βρεθούν μέσω της ‘parameters_select’. Αν είναι DELETE, τότε ‘parameters_delete’ κ.ο.κ.

3. Αρχείο sql_compiler.py

Εδώ περιέχονται οι εντολές που αφορούν την ίδια τη βάση δεδομένων (CREATE/DROP DATABASE) και όχι τους πίνακές της. Η λειτουργία είναι παρόμοια με τις εντολές του αρχείου database.py.

4. Παραδείγματα εκτέλεσης

SELECT-FROM

```
>>> db.execSQL('SELECT * FROM instructor')
SELECT: *
FROM: ['instructor']
INNER JOIN (TABLE 2): None
ON: None
WHERE: [None]
ORDER BY: [None]
LIMIT: None
ASC: True

## instructor ##
  ID (str) #PK#  name (str)    dept_name (str)      salary (int)
---------------  ------------  -----------------  --------------
          10101  Srinivasan    Comp. Sci.                  65000
          12121  Wu            Finance                     90000
          15151  Mozart        Music                       40000
          22222  Einstein      Physics                     95000
          32343  El Said       History                     60000
          33456  Gold          Physics                     87000
          45565  Katz          Comp. Sci.                  75000
          58583  Califieri     History                     62000
          76543  Singh         Finance                     80000
          76766  Crick         Biology                     72000
          83821  Brandt        Comp. Sci.                  92000
          98345  Kim           Elec. Eng.                  80000
```

SELECT-FROM-WHERE

```
>>> db.execSQL('SELECT ID,name,salary FROM instructor WHERE salary>80000')
SELECT: ['ID', 'name', 'salary']
FROM: ['instructor']
INNER JOIN (TABLE 2): None
ON: None
WHERE: ['salary>80000']
ORDER BY: [None]
LIMIT: None
ASC: True

## instructor ##
  ID (str) #PK#  name (str)      salary (int)
---------------  ------------  --------------
          12121  Wu                     90000
          22222  Einstein               95000
          33456  Gold                   87000
          83821  Brandt                 92000
```

SELECT (με INNER JOIN)

```
>>> db.execSQL('SELECT * FROM instructor INNER JOIN student ON dept_name==dept_name WHERE instructor_salary>80000')
## Select ops no. -> 156
# Left table size -> 12
# Right table size -> 13

## instructor_join_student ##
  instructor_ID (str)  instructor_name (str)    instructor_dept_name (str)      instructor_salary (int)    student_ID (str)
tudent_tot_cred (int)
---------------------  ----------------------  --------------------------  ------------------------  ------------------
--------------------
               12121  Wu                      Finance                                      90000               23121
                 110
               22222  Einstein                Physics                                      95000               44553
                  56
               22222  Einstein                Physics                                      95000               45678
                  46
               22222  Einstein                Physics                                      95000               70557
                   0
               33456  Gold                    Physics                                      87000               44553
                  56
               33456  Gold                    Physics                                      87000               45678
```

(Κομμένη εικόνα)

Η miniDB προς το παρόν δεν υποστηρίζει επιλογή στηλών στα inner join. Σε περίπτωση που ζητηθεί κάτι διαφορετικό από “*”, εμφανίζεται σχετικό μήνυμα.

```
>>> db.execSQL('SELECT student_ID FROM instructor INNER JOIN student ON dept_name==dept_name WHERE instructor_salary>80000')
You cannot select specific columns with 'Inner Join' yet. Showing all columns...

## Select ops no. -> 156
# Left table size -> 12
# Right table size -> 13
```

SELECT-…-ORDER BY

```
>>> db.execSQL('SELECT * FROM instructor WHERE salary>80000 ORDER BY name')
SELECT: *
FROM: ['instructor']
INNER JOIN (TABLE 2): None
ON: None
WHERE: ['salary>80000']
ORDER BY: ['name']
LIMIT: None
ASC: True

## instructor ##
  ID (str) #PK#  name (str)    dept_name (str)      salary (int)
---------------  ------------  -----------------  --------------
          83821  Brandt        Comp. Sci.                  92000
          22222  Einstein      Physics                     95000
          33456  Gold          Physics                     87000
          12121  Wu            Finance                     90000
```

(Ίδιο αποτέλεσμα με “ORDER BY name ASC”)

SELECT-…-ORDER BY (DESC)

```
>>> db.execSQL('SELECT * FROM instructor WHERE salary>80000 ORDER BY name DESC')
SELECT: *
FROM: ['instructor']
INNER JOIN (TABLE 2): None
ON: None
WHERE: ['salary>80000']
ORDER BY: ['name']
LIMIT: None
ASC: False

## instructor ##
  ID (str) #PK#  name (str)    dept_name (str)      salary (int)
---------------  ------------  -----------------  --------------
         12121  Wu            Finance                     90000
         33456  Gold          Physics                     87000
         22222  Einstein      Physics                     95000
         83821  Brandt        Comp. Sci.                  92000
```

SELECT-…-LIMIT

```
>>> db.execSQL('SELECT * FROM instructor WHERE salary>80000 LIMIT 2')
SELECT: *
FROM: ['instructor']
INNER JOIN (TABLE 2): None
ON: None
WHERE: ['salary>80000']
ORDER BY: [None]
LIMIT: 2
ASC: True

## instructor ##
  ID (str) #PK#  name (str)    dept_name (str)      salary (int)
---------------  ------------  -----------------  --------------
         12121  Wu            Finance                     90000
         22222  Einstein      Physics                     95000
```

(Ορισμένοι SQL compilers υποστηρίζουν την εντολή SELECT TOP για αυτή τη λειτουργία. Το “LIMIT” χρησιμοποιείται από την MySQL και ταιριάζει περισσότερο με τον κώδικά μας, μιας και ξέρουμε ότι πάντα θα βρίσκεται στο τέλος του query.)

INSERT

```
>>> db.execSQL("insert into instructor values ['99999','Tesla','Engineering',69420]")
INSERT INTO: ['instructor']
VALUES: ['99999','Tesla','Engineering',69420]
>>> db.execSQL("select * from instructor")
SELECT: *
FROM: ['instructor']
INNER JOIN (TABLE 2): None
ON: None
WHERE: [None]
ORDER BY: [None]
LIMIT: None
ASC: True

## instructor ##
  ID (str) #PK#  name (str)    dept_name (str)      salary (int)
---------------  ------------  -----------------  --------------
          10101  Srinivasan    Comp. Sci.                  65000
          12121  Wu            Finance                     90000
          15151  Mozart        Music                       40000
          22222  Einstein      Physics                     95000
          32343  El Said       History                     60000
          33456  Gold          Physics                     87000
          45565  Katz          Comp. Sci.                  75000
          58583  Califieri     History                     62000
          76543  Singh         Finance                     80000
          76766  Crick         Biology                     72000
          83821  Brandt        Comp. Sci.                  92000
          98345  Kim           Elec. Eng.                  80000
          99999  Tesla         Engineering                 69420
```

INSERT + SELECT

```
>>> db.execSQL("INSERT INTO underpaid_instructor SELECT * FROM instructor WHERE salary<50000")
SAVE AS: ['underpaid_instructor']
SELECT QUERY: SELECT * FROM instructor WHERE salary<50000
>>>
>>> db.execSQL("SELECT * FROM underpaid_instructor")

## underpaid_instructor ##
  ID (str) #PK#  name (str)    dept_name (str)      salary (int)
---------------  ------------  -----------------  --------------
          15151  Mozart        Music                       40000
```

Αξιοποιεί τη λειτουργία save_as της miniDB. Πρόκειται για έναν συνδυασμό της INSERT και της SELECT στην οποία δημιουργείται νέος πίνακας με τα αποτελέσματα ενός SELECT query.

UPDATE

```
>>> db.execSQL("UPDATE instructor SET salary=0 WHERE name<K")
UPDATE: ['instructor']
SET: ['salary=0']
WHERE: ['name<K']
>>> db.execSQL("SELECT * FROM instructor")

## instructor ##
  ID (str) #PK#  name (str)    dept_name (str)      salary (int)
---------------  ------------  -----------------  --------------
          10101  Srinivasan    Comp. Sci.                  65000
          12121  Wu            Finance                     90000
          15151  Mozart        Music                       40000
          22222  Einstein      Physics                         0
          32343  El Said       History                         0
          33456  Gold          Physics                         0
          45565  Katz          Comp. Sci.                  75000
          58583  Califieri     History                         0
          76543  Singh         Finance                     80000
          76766  Crick         Biology                         0
          83821  Brandt        Comp. Sci.                      0
          98345  Kim           Elec. Eng.                  80000
```

DELETE

```
>>> db.execSQL("DELETE FROM instructor WHERE salary>60000")
DELETE FROM: ['instructor']
WHERE: ['salary>60000']
Deleted 10 rows
>>> db.execSQL("SELECT * FROM instructor")

## instructor ##
  ID (str) #PK#  name (str)    dept_name (str)      salary (int)
---------------  ------------  -----------------  --------------
          15151  Mozart        Music                       40000
          32343  El Said       History                     60000
```

CREATE TABLE

```
>>> db.execSQL("CREATE TABLE building(ID str, address str, size int)")
NAME: ['building']
COLUMN NAMES: ['ID', 'address', 'size']
COLUMN TYPES: [<class 'str'>, <class 'str'>, <class 'int'>]
New table "building"
>>> db.execSQL("INSERT INTO building values ['123','Karaoli kai Dimitriou 80',500]")
>>> db.execSQL("SELECT * FROM building")

## building ##
  ID (str)  address (str)              size (int)
----------  -----------------------  ------------
       123  Karaoli kai Dimitriou 80           500
```

DROP TABLE

```
>>> db.execSQL("DROP TABLE building")
Deleted 1 rows
Deleted 1 rows
Deleted 1 rows
```

CREATE INDEX (B-tree)

```
>>> db.execSQL("CREATE INDEX idx ON instructor")
CREATE INDEX: ['idx']
ON: ['instructor']
Creating Btree index.
```

CREATE DATABASE

```
C:\Users\Stergios\Desktop\miniDB>python -i sql_compiler.py
>>> execSQL('CREATE DATABASE mydatabase')
New table "meta_length"
New table "meta_locks"
New table "meta_insert_stack"
New table "meta_indexes"
>>> db.execSQL('CREATE TABLE myinstructor(ID str, name str, salary int)')
New table "myinstructor"
>>> db.execSQL('INSERT INTO myinstructor VALUES ["12345","Mozart",50000]')
>>> db.execSQL('SELECT * FROM myinstructor')

## myinstructor ##
  ID (str)  name (str)      salary (int)
----------  ------------  --------------
     12345  Mozart                50000
```