Python e Mysql

E' possibile interagire direttamente con un database mysql (ma anche con altri) utilizzando moduli appositi in python

General Purpose Database Systems

IBM DB2
Firebird (and Interbase)
Informix
Ingres
MySQL
Oracle
PostgreSQL
SAP DB (also known as "MaxDB")
Microsoft SQL Server
Microsoft Access
Sybase

- Occorre installare mysql (su linux) per avere il modulo MySQLdb

- sudo apt-get install python-mysqldb
- Ora per python2/3 si trovano altri moduli ma la sintassi e’ la stessa

- (dopo aver risolto mille mila conflitti!!)

- A questo punto si puo' importare il modulo
  da python e utilizzarlo per collegarsi ad
  un database esistente o per crearne uno nuovo.

- Ci si connette ad un database specifico
  attraverso il server mysql che deve essere attivo.

- Il modulo MySQLdb ha diversi metodi (funzioni/comandi)
  che permettono di operare col database

- CONNESSIONE => metodo connect con 4 parametri

- db = MySQLdb.connect("localhost","root","roberto","lezioni")

- il metodo connect utilizza 4 parametri:
- l'host --> di solito localhost
- username: l'utente che deve connettersi al database coi suoi privilegi e permessi
- passw:
- name: il nome del database
- restituisce un 'oggetto' -> db
- type(db)

Come si Opera?

- cursor = db.cursor() #si crea un cursore sui dati del database
- restituisce un cursore che scorre il db

- il cursore 'esegue' comandi MySQLdb

I piu’ utili sono

- cursor.execute()
- cursor.fetchone()
- cursor.fetchall()

- Tutti I moduli di interazione python mysql seguono la stessa logica

- esempi vari:

- cursor.execute('show tables')
- cursor.fetchone()
- cursor.fetchone()
- cursor.fetchone()
- cursor.fetchone()

- ancora

- cursor.execute('show tables')
- cursor.fetchall()

- fetchall() e fetchone() possono restituire variabili a cui appendere I dati della call

- cursor.execute('show databases')

- A = cursor.fetchall()
- type(A)

- restituisce tuple.....

- o tuple di tuple!!

Esecuzione di comandi sql

- cursor.execute('query')

  esegue quindi comandi sql sotto forma di query o istruzioni sql passate sotto formato di stringa
- Esempio

  sql = "SELECT * FROM auto" #stringa di query in sql

- cursor.execute(sql)

- results = cursor.fetchall()

- type(results)

Come usare i dati?

- I valori resituiti dalla call sono a disposizione nella variabile results come tuple, tuple di tuple, tuple di altri tipi e possono essere elaborati con comandi python.....
- results[1]
- results[1][2]
- results[1][2][2]
- results[-1]

In generale un ciclo completo e’:

- # Open database connection
- db = MySQLdb.connect("localhost","testuser","test123","TESTDB")
- # prepare a cursor object using cursor() method
- cursor = db.cursor()
- # eseguire SQL query usando il metodo/comando execute().
- cursor.execute("SELECT VERSION()")
- # Fetch a single row usando il metodo/comando fetchone().
- data = cursor.fetchone()
- print "Database version : %s " % data
- # disconnect from server
- db.close()

Creare tabelle da Python

```
# aprire/creare la connessione al DB
db = MySQLdb.connect(“localhost”,”user”,”password”,”DatabaseDaUsare”)

#preparare il cursore
cursor = db.cursor()

#usarlo. Es verificare che una tabella non esista gia’ prima di crearla
cursor.execute(“DROP TABLE IF EXISTS EPLOYEE”)

# creare la tabella voluta
# Create table as per requirement
sql = """CREATE TABLE EMPLOYEE (
         FIRST_NAME  CHAR(20) NOT NULL,
         LAST_NAME  CHAR(20),
         AGE INT,
         SEX CHAR(1),
         INCOME FLOAT )"""

cursor.execute(sql)

#disconnettersi
db.close()
```

?? abbiamo salvato?

- Occorre fare il ‘commit’ nel database!
- db.commit()
- Serve a ‘salvare’ e rendere operative le modifiche al database.
- Non e’ necesario usarlo in lettura ma se modfico dati on the fly e voglio rendere le modifiche definitive devo committare sul DB

Inserire in tabella

```
# Open database connection
db = MySQLdb.connect("localhost","testuser","test123","TESTDB" )
# prepare a cursor object using cursor() method
cursor = db.cursor()
# Prepare SQL query to INSERT a record into the database.
sql = """INSERT INTO EMPLOYEE(FIRST_NAME,
         LAST_NAME, AGE, SEX, INCOME)
         VALUES ('Mac', 'Mohan', 20, 'M', 2000)"""
   # Execute the SQL command
   cursor.execute(sql)
   # Commit your changes in the database
   db.commit()
# disconnect from server
db.close()
```

L’interazione col DB e’ error prone: usare try ed except col rollback!

- try:
- # Execute the SQL command
- cursor.execute(sql)
- # Commit your changes in the database
- db.commit()
- except:
- # Rollback in case there is any error
- db.rollback()
- #rollbacks db to pre-transaction state in case of some problems during sql command execution.
- # disconnect from server
- db.close()

Creare e riempire un database

- cursor.execute('create database newdatabase')
- cursor.execute('use newdtabase')
- sql1 = 'CREATE TABLE studenti (Matricola integer primary key, NomeCognome char(50) not null)'
- cursor.execute(sql1)
- cursor.execute('show tables')
- cursor.fetchall()
- sql2 = 'INSERT INTO studenti (Matricola,NomeCognome) VALUES (1001,"Gino Landi")'

Update

- UPDATE Operation significa modificare contenuti (record) gia’ esistenti
- # Prepare SQL query to UPDATE required records
- sql = "UPDATE EMPLOYEE SET AGE = AGE + 1
- WHERE SEX = '%c'" % ('M')
- try:
- # Execute the SQL command
- cursor.execute(sql)
- # Commit your changes in the database
- db.commit()
- except:
- # Rollback in case there is any error
- db.rollback()
-
- # disconnect from server
- db.close()

Delete

- DELETE operation is required when you want to delete some records from your database. Following is the procedure to delete all the records from EMPLOYEE where AGE is more than 20
- sql = "DELETE FROM EMPLOYEE WHERE AGE > '%d'" % (20)
- try:
- # Execute the SQL command
- cursor.execute(sql)
- # Commit your changes in the database
- db.commit()
- except:
- # Rollback in case there is any error
- db.rollback()
-
- # disconnect from server
- db.close()

COMMIT

- COMMIT Operation

- Commit e’ l’operazione che da il segnale verde al database per finalizzare le modifiche.
- Dopo tale segnale le modifiche diventano definitive.

- Si chiama il metodo commit():

  db.commit()

ROLLBACK

- ROLLBACK Operation

- Se non si e’ soddisfatti delle modifiche si puo’ prima fare il rollback, cioe’ riportare allo stato precedente .

- Si chiama il metodo rollback():

  db.rollback()

CLOSING

- Disconnecting Database
- Per disconnettersi al DB si usa:

  db.close()

- Se la connessione al DB dall'utente con db.close() le transazioni pendenti vengono automaticamente recuperate col rollback().

Esempi

- Db = MySQLdb.connect("localhost","root","roberto","lezioni")
- cursor = db.cursor()
- cursor.execute('show tables')

- cursor.fetchall()

- sql3 = 'select * from auto where Provincia = "MI" '
- cursor.execute(sql3)
- cursor.fetchall()

- sql4 = 'select * from auto where Provincia = "MI" and Cognome = "rossi" '
- cursor.execute(sql4)
- cursor.fetchall()
- cursor.fetchall()

- cursor.execute(sql4)

- >>> results4 = cursor.fetchall()

- for row in results4:
- ... prov = row[0]
- ... num = row[1]
- ... nome = row[2]
- ... cognome = row[3]
- ... print "%s %s %s %s" %(prov,num,nome,cognome)

Inserire con Parametri

- cursor.execute("CREATE TABLE prova (Nome char(50), Cognome char(50))")
-
- insert_stmt = ("INSERT INTO prova (Nome, Cognome) " "VALUES (%s, %s)")
- data = ("Pino" , "Doe")
- cursor.execute(insert_stmt, data)

SINTASSI metodi di CURSOR

- cursor.execute(operation, params=None, multi=False)
- iterator = cursor.execute(operation, params=None, multi=True)
- Specify variables using %s or %(name)s parameter style
- insert_stmt = ("INSERT INTO employees (emp_no, first_name, last_name, age) " "VALUES (%s, %s, %s, %s)")
- data = (2, 'Jane', 'Doe', 23)

- cursor.execute(insert_stmt, data)

Esempio di utilizzo di description

- A = cursor.description
- for i in A:
- ... print i[0]

Inserire dati al volo

- ss = 'go'
- while ss!='end':
- ... insert_stmt = ("INSERT INTO prova / (Nome,Cognome) " "VALUES (%s, %s)")
- ... a = raw_input("nome:")
- ... b = raw_input("Cognome:")
- ... data = (a , b)
- ... cursor.execute(insert_stmt, data)
- ... ss = raw_input("finito?: inserisci end")

Altre queries

- cursor.execute('select * from EMPLOYEE,studenti')

- cursor.execute('select * from EMPLOYEE,studenti WHERE EMPLOYEE.SEX = "F" ')

```
>>> db = MySQLdb.connect("localhost","root","roberto","lezioni" )
>>> cursor = db.cursor()
>>> cursor.execute('show tables')
3L
>>> cursor.fetchall()
(('auto',), ('multe',), ('vigili',))
>>> sql3 = 'select * from auto where Provincia = "MI"'
>>> cursor.execute(sql3)
2L
>>> cursor.fetchall()
(('MI', '223432', 'luigi', 'bianchi'), ('MI', '345432', 'luigi', 'rossi'))
>>> sql4 = 'select * from auto where Provincia = "MI" and Cognome = "rossi"'
>>> cursor.execute(sql4)
1L
>>> cursor.fetchall()
(('MI', '345432', 'luigi', 'rossi'),)
>>> cursor.fetchall()
()
>>> cursor.execute(sql4)
1L
>>> results4 = cursor.fetchall()


 for row in results4:
...     prov = row[0]
...     num = row[1]
...     nome = row[2]
...     cognome = row[3]
...     print "%s %s %s %s" %(prov,num,nome,cognome)
```