Usa redirect per reinderizzare alla pagina principale in caso di form errato

**CONNESSIONI DATABASE**

Il database serve a rendere le informazioni persistenti e le loro modifiche persistenti, oltre che gestire grandi quantità di dati senza memorizzarli in memoria. Tramite python vogliamo accedere al database, usando mysql o mariadb che in realtà sono praticamente uguali, quello usato sarà sqlLite che non è un server ma solo una libreria, quindi un file. Abbiamo un database su file, quindi no nessun software, basta la libreria e per passare il database basta passare il file. Python aggiunge un livello tra server e librerie, tramite un API PEP: Python enanched proposal che sono una serie di documenti che standardizzano come python si interfacci con qualcosa (es database), l’unica cosa che cambia è che col file bisogna aprire il file, col server bisogna stabilire una connessione al server. Si basa su 4 concetti principali

1. Accesso al database tramite il metodo connect e un oggetto chiamato connection (apertura connessione)
2. Va creato un Cursore che rappresenta l’oggetto usato per le interrogazioni al database, viene creato dalla connessione stessa una volta effettuata
3. Un cursore può usare il metodo execute per effettuare la query fornendo opportuni parametri
4. Un cursore prende i risultati della query

**ESEMPIO**

1. Sql = ‘……..query….’
2. Conn = mysql.connector.connect(user=’root’, password=’’, host=localhost, database=’…’) %dipende dal database
3. Cursor = conn.cursor()
4. Cursor.execute(sql)
5. Translations = cursor.fetchall() %viene usato perchè non sappiamo quanti dati avremo, quindi li prendiamo tutti
6. Cursor.close() %le connessioni vanno chiuse prima di terminare la funzione con il ritorno della variabile
7. Conn.close()
8. Return Translation

La query è definita come stringa, se voglio usare parametri in mariadb, mysql uso %s, mentre su slqlite uso ?

**NON CONCATENARE LE STRINGHE!!!!**

**SI USANO I PLACE HOLDER PER AUMENTARE LA SICUREZZA CONTROL LA SEQUEL INJECTION**

Sempre meglio usare i placeholder i cui valori non saranno inseriti nella stringa ma passati nella execute, che verificherà il corretto formato dei parametri

Ex WHERE name=? Oppure WHERE name=%s

– conn = mysql.connector.connect(user='root', password='' , host='localhost', database='funnyecho')

– conn sqlite3.connect('example.db') %connessione con sqlite

– cursor.execute(sql, (txtbefore, txtafter) ) passaggio dei parametri come tupla durante la query

– cursor.execute(sql, (txtbefore,) ) passaggio di parametro singolo, va messa comunque la virgola altrimenti no tupla

Cursor.fetchall() prende tutti i risultati indipendentemente da quanti siano

Cursor.fetchone() prende il singolo risultato, ogni volta prendendo il prossimo

Vengono restituite tuple o array di tuple, se voglio un dizionario devo fare qualche passaggio in più

Per la modifica al database va fatto un conn.commit() che salva tutte le modifiche fatte al database, viene fatta sulla connessione, non sul cursore, infatti può prendere le modifiche di tutti i cursori, senza commit non salvo i dati e va fatto sempre prima della chiusura della connessione

Ricordati sempre di chiudere le connessioni, sempre assieme all’apertura

Una volta chiusa la connessione io posso comunque riaprirne un'altra, posso anche creare più cursori per la stessa connessione

Il modulo sqlite è incluso in python, basta importare la connessione