Introduzione ai DB

Python

Un Database, o base di dati, è un insieme organizzato di dati strutturati e correlati tra loro, progettato per essere facilmente accessibile, gestito e aggiornato.

I database vengono utilizzati per memorizzare informazioni in modo persistente, consentendo operazioni di inserimento, modifica, cancellazione e interrogazione dei dati.

Questi sistemi sono fondamentali in moltissimi ambiti, dal software gestionale ai siti web, passando per applicazioni mobile e sistemi industriali.

L’accesso e la gestione dei database sono solitamente affidati a un DBMS (Database Management System), ovvero un software che si occupa dell'interazione tra l'utente e il database.

Database relazionali (RDBMS)

- Basati su tabelle (righe e colonne), utilizzano il linguaggio SQL. Ogni tabella ha relazioni con altre tramite chiavi. Esempi: MySQL, PostgreSQL, Oracle.

Database non relazionali (NoSQL)

- Progettati per grandi quantità di dati non strutturati o semi-strutturati. Non usano lo schema rigido delle tabelle. Tipologie comuni:
- Documentali: archiviano dati in formato JSON o BSON (es. MongoDB)
- Key-Value: semplici coppie chiave-valore (es. Redis)
- Colonnari: ottimizzati per letture su grandi volumi (es. Cassandra)
- A grafo: rappresentano dati e relazioni con nodi e archi (es. Neo4j)

Database in memoria

- Conservano i dati nella RAM per garantire altissime prestazioni (es. Redis, Memcached). Ideali per cache e dati temporanei.

Database distribuiti

- I dati sono distribuiti su più nodi di rete. Garantisce scalabilità orizzontale e tolleranza ai guasti (es. Apache Cassandra, Google Bigtable).

Database a oggetti

- Conservano i dati sotto forma di oggetti, come nei linguaggi OOP, permettendo una stretta integrazione tra codice e dati (es. db4o, ObjectDB).

I database relazionali si basano sul modello relazionale introdotto da Edgar F. Codd negli anni ’70. I dati sono organizzati in tabelle (dette anche relazioni), dove ogni riga rappresenta un record (o tupla) e ogni colonna un attributo.

Le tabelle sono collegate tra loro tramite chiavi primarie (primary key) e chiavi esterne (foreign key), che permettono di stabilire relazioni logiche tra i dati.

Il linguaggio utilizzato per interagire con questi database è l’SQL (Structured Query Language), che consente di definire la struttura dei dati (DDL), interrogarli (DQL), modificarli (DML) e gestirne i permessi (DCL).

Il modello garantisce la consistenza e l’integrità dei dati attraverso vincoli come UNIQUE, NOT NULL, CHECK, oltre alla gestione delle transazioni tramite il principio ACID (Atomicità, Consistenza, Isolamento, Durabilità).

Un database relazionale presenta alcune caratteristiche fondamentali che ne definiscono l'affidabilità e la struttura logica.

In primo luogo, la normalizzazione consente di ridurre la ridondanza e migliorare la coerenza dei dati suddividendoli in più tabelle collegate. I vincoli di integrità garantiscono che i dati inseriti rispettino regole precise, evitando anomalie.

L'astrazione dei dati permette di separare la struttura logica da quella fisica, rendendo il database più flessibile e manutenibile. L'utilizzo del linguaggio SQL standardizzato consente portabilità tra diversi sistemi RDBMS.

Inoltre, il supporto alle transazioni ACID garantisce affidabilità nelle operazioni, anche in ambienti concorrenti o in presenza di errori. Infine, i database relazionali sono altamente scalabili verticalmente, cioè possono essere potenziati aumentando le risorse hardware del server.

SQL (Structured Query Language) è il linguaggio standard utilizzato per interagire con i database relazionali.

È progettato per eseguire operazioni di creazione, lettura, aggiornamento e cancellazione dei dati (CRUD), oltre a definire la struttura delle tabelle e gestire i permessi degli utenti.

SQL consente di scrivere query per filtrare, ordinare, raggruppare e correlare i dati tra più tabelle. È un linguaggio dichiarativo, il che significa che si specifica cosa ottenere, non come ottenerlo.

È supportato da tutti i principali RDBMS (Database relazionali), come Oracle, PostgreSQL, SQL Server e MySQL.

MySQL è uno dei sistemi di gestione di database relazionali (RDBMS) più popolari al mondo, basato su SQL.

È open-source e sviluppato inizialmente da MySQL AB, poi acquisito da Sun Microsystems e infine da Oracle Corporation.

MySQL è largamente usato in applicazioni web, specialmente in abbinamento con PHP e Apache nei cosiddetti stack LAMP/WAMP.

È noto per la sua velocità, affidabilità e facilità d’uso, ed è spesso scelto per progetti piccoli e medi, ma anche per grandi sistemi grazie alla sua scalabilità. Supporta funzionalità avanzate come le transazioni, le stored procedure, la replica e la gestione sicura degli utenti.

In Python, sqlite3 è un modulo integrato nella libreria standard che permette di interfacciarsi con SQLite, un sistema di database relazionale leggero e senza bisogno di installazione o configurazione esterna.

SQLite memorizza i dati in un singolo file .db e supporta tutte le funzionalità base dell'SQL.

È ideale per applicazioni locali, test, piccoli progetti o prototipi.

Il modulo sqlite3 consente di eseguire comandi SQL direttamente da Python, creando tabelle, inserendo dati, eseguendo query e altro ancora.

Spiegazione

1. connect() apre (o crea) un database SQLite.
2. Il cursore (cursor()) serve per eseguire i comandi SQL.
3. Viene creata una tabella studenti con tre colonne.
4. Viene inserito un dato in modo sicuro (parametrico).
5. Si esegue una query di selezione e si stampano i risultati.
6. Infine, si chiude la connessione.

```
import sqlite3

# Connessione a un database (o creazione se non esiste)
conn = sqlite3.connect('esempio.db')

# Creazione di un cursore per eseguire comandi SQL
cur = conn.cursor()

# Creazione di una tabella
cur.execute('''
    CREATE TABLE IF NOT EXISTS studenti (
        id INTEGER PRIMARY KEY,
        nome TEXT,
        voto INTEGER
    )
''')

# Inserimento di un record
cur.execute("INSERT INTO studenti (nome, voto) VALUES (?, ?)", ("Marco", 28))

# Salvataggio delle modifiche
conn.commit()

# Esecuzione di una query
cur.execute("SELECT * FROM studenti")
risultati = cur.fetchall()

# Stampa dei risultati
for riga in risultati:
    print(riga)

# Chiusura della connessione
conn.close()
```

Il termine CRUD è un acronimo che rappresenta le quattro operazioni fondamentali che possono essere eseguite su un insieme di dati in un sistema informatico: Create, Read, Update e Delete.

Queste operazioni sono alla base di qualsiasi interazione con un database o una risorsa dati persistente.

"Create" serve a inserire nuovi dati, "Read" a leggere o recuperare informazioni esistenti, "Update" a modificarle, e "Delete" a eliminarle.

Il modello CRUD è un concetto centrale nello sviluppo software, soprattutto in applicazioni web e sistemi gestionali, ed è spesso implementato attraverso interfacce utente (GUI), API RESTful o comandi SQL nei database relazionali.

Progettare un'applicazione CRUD significa costruire una struttura che consenta all'utente di gestire un insieme di dati in modo completo e coerente, spesso con l'ausilio di un backend (come Python, Java o PHP) collegato a un database.


Buon MasterD a tutti