INTRODUZIONE A SQL

1. COS'È SQL

- DEFINIZIONE DI SQL
  - SQL È UN LINGUAGGIO DI PROGRAMMAZIONE UTILIZZATO PER GESTIRE E MANIPOLARE I DATABASE RELAZIONALI
- STANDARDIZZAZIONE DI SQL
  - ANSI
    - SQL È STANDARDIZZATO DALL'ANSI
  - UTILIZZO DI SQL
    - SQL VIENE UTILIZZATO PER ESEGUIRE OPERAZIONI DI QUERY, INSERIMENTO, AGGIORNAMENTO E CANCELLAZIONE DI DATI ALL'INTERNO DI UN DATABASE
- UTILITÀ DI SQL
  - SQL VIENE UTILIZZATO PER CREARE E MODIFICARE STRUTTURE DI DATABASE, GESTIRE I DATI, CONTROLLARE L'ACCESSO AI DATI E MANTENERE L'INTEGRITÀ DEI DATI

2. A COSA SERVE SQL

- CREAZIONE E MODIFICA DI STRUTTURE DI DATABASE
  - SQL VIENE UTILIZZATO PER CREARE E MODIFICARE TABELLE, VISTE E INDICI ALL'INTERNO DI UN DATABASE
- GESTIONE DEI DATI
  - SQL PERMETTE DI INSERIRE, AGGIORNARE, ELIMINARE E RECUPERARE DATI MEMORIZZATI NEI DATABASE
- CONTROLLO DELL'ACCESSO AI DATI
  - SQL GESTISCE I PERMESSI E I RUOLI DEGLI UTENTI PER CONTROLLARE L'ACCESSO AI DATI
- MANTENIMENTO DELL'INTEGRITÀ DEI DATI
  - SQL GARANTISCE CHE I DATI SIANO CONSISTENTI E VALIDI PER MANTENERE L'INTEGRITÀ DEL DATABASE

INTRODUZIONE A SQL

3. TIPI DI DATI COMUNI IN SQL

NUMERI INTERI

SQL SUPPORTA I TIPI DI DATI INTEGER, INT, SMALLINT E BIGINT PER RAPPRESENTARE NUMERI INTERI

NUMERI IN VIRGOLA MOBILE

I TIPI DI DATI FLOAT, REAL E DOUBLE VENGONO UTILIZZATI PER RAPPRESENTARE NUMERI IN VIRGOLA MOBILE CON DIVERSE PRECISIONI

NUMERI CON PRECISIONE ESATTA

SQL UTILIZZA I TIPI DI DATI NUMERIC E DECIMAL PER RAPPRESENTARE NUMERI CON PRECISIONE ESATTA, SPECIFICANDO LA PRECISIONE TOTALE E LA SCALA

STRINGHE DI CARATTERI

I TIPI DI DATI CHAR, VARCHAR E TEXT VENGONO UTILIZZATI PER RAPPRESENTARE STRINGHE DI CARATTERI DI LUNGHEZZA FISSA O VARIABILE

TIPI DI DATI TEMPORALI

SQL SUPPORTA I TIPI DI DATI DATE, TIME E TIMESTAMP PER RAPPRESENTARE DATE E ORARI

VALORI BOOLEANI

IL TIPO DI DATO BOOLEAN VIENE UTILIZZATO PER RAPPRESENTARE VALORI BOOLEANI, OVVERO VERO O FALSO

DATI BINARI

I TIPI DI DATI BLOB E BYTEA VENGONO UTILIZZATI PER RAPPRESENTARE DATI BINARI, COME IMMAGINI O FILE

INTRODUZIONE A SQL

4. PRINCIPALI COMANDI SQL E SINTASSI

- **CREAZIONE DI UNA TABELLA**
  - IL COMANDO CREATE TABLE VIENE UTILIZZATO PER CREARE UNA NUOVA TABELLA NEL DATABASE
- **INSERIMENTO DI DATI**
  - IL COMANDO INSERT INTO VIENE UTILIZZATO PER INSERIRE NUOVI DATI IN UNA TABELLA ESISTENTE
- **LETTURA DI DATI**
  - IL COMANDO SELECT VIENE UTILIZZATO PER SELEZIONARE DATI DA UNA O PIÙ TABELLE
- **AGGIORNAMENTO DI DATI**
  - IL COMANDO UPDATE VIENE UTILIZZATO PER MODIFICARE I DATI ESISTENTI IN UNA TABELLA
- **CANCELLAZIONE DI DATI**
  - IL COMANDO DELETE VIENE UTILIZZATO PER ELIMINARE DATI DA UNA TABELLA
- **AGGIUNTA DI UNA COLONNA**
  - IL COMANDO ALTER TABLE VIENE UTILIZZATO PER AGGIUNGERE UNA NUOVA COLONNA A UNA TABELLA ESISTENTE
- **CREAZIONE DI UNA VISTA**
  - IL COMANDO CREATE VIEW VIENE UTILIZZATO PER CREARE UNA VISTA, OVVERO UNA RAPPRESENTAZIONE VIRTUALE DEI DATI DI UNA O PIÙ TABELLE
- **UNIONI**
  - LE UNIONI VENGONO UTILIZZATE PER COMBINARE I DATI DI PIÙ TABELLE IN UNA SINGOLA QUERY, UTILIZZANDO I COMANDI INNER JOIN E LEFT JOIN

INTRODUZIONE A SQL

5. PROGRAMMI E INTERFACCE GRAFICHE PER GESTIRE DATABASE SQL

MYSQL

MYSQL È UN POPOLARE SISTEMA DI GESTIONE DI DATABASE RELAZIONALI OPEN-SOURCE, UTILIZZATO PER APPLICAZIONI WEB

POSTGRESQL

POSTGRESQL È UN SISTEMA DI GESTIONE DI DATABASE RELAZIONALI AVANZATO E OPEN-SOURCE, UTILIZZATO PER APPLICAZIONI COMPLESSE E CRITICHE

SQLITE

SQLITE È UN DATABASE RELAZIONALE LEGGERO E INCORPORATO, UTILIZZATO PER APPLICAZIONI MOBILI E DESKTOP

MICROSOFT SQL SERVER

MICROSOFT SQL SERVER È UN SISTEMA DI GESTIONE DI DATABASE RELAZIONALI SVILUPPATO DA MICROSOFT, UTILIZZATO PRINCIPALMENTE IN AMBIENTI AZIENDALI

ORACLE DATABASE

ORACLE DATABASE È UN POTENTE SISTEMA DI GESTIONE DI DATABASE RELAZIONALI UTILIZZATO IN AMBIENTI AZIENDALI PER APPLICAZIONI CRITICHE

MARIADB

MARIADB È UN FORK DI MYSQL, COMPATIBILE CON ESSO MA CON ULTERIORI FUNZIONALITÀ E MIGLIORAMENTI

INTERFACCE GRAFICHE PER GESTIRE DATABASE SQL

ESISTONO DIVERSE INTERFACCE GRAFICHE, COME PHPMYADMIN, PGADMIN, DBEAVER, HEIDISQL, SQL SERVER MANAGEMENT STUDIO E ORACLE SQL DEVELOPER, CHE CONSENTONO DI GESTIRE DATABASE SQL IN MODO PIÙ INTUITIVO E VISUALE

INTRODUZIONE A SQL

6. ESEMPIO DI UTILIZZO DI SQL

CREAZIONE DI UNA TABELLA

UTILIZZANDO IL COMANDO CREATE TABLE, È POSSIBILE CREARE UNA NUOVA TABELLA NEL DATABASE

INSERIMENTO DI DATI

CON IL COMANDO INSERT INTO, È POSSIBILE INSERIRE NUOVI DATI IN UNA TABELLA ESISTENTE

LETTURA DI DATI

UTILIZZANDO IL COMANDO SELECT, È POSSIBILE SELEZIONARE DATI DA UNA O PIÙ TABELLE

AGGIORNAMENTO DI DATI

CON IL COMANDO UPDATE, È POSSIBILE MODIFICARE I DATI ESISTENTI IN UNA TABELLA

CANCELLAZIONE DI DATI

UTILIZZANDO IL COMANDO DELETE, È POSSIBILE ELIMINARE DATI DA UNA TABELLA

ESEMPIO COMPLETO

DI SEGUITO È RIPORTATO UN ESEMPIO DI UTILIZZO DI SQL PER CREARE UNA TABELLA, INSERIRE DATI, SELEZIONARE, AGGIORNARE E CANCELLARE DATI

VINCOLI

1. **NOT NULL**
   - Impedisce che una colonna possa contenere valori nulli. Garantisce che ogni riga della tabella abbia un valore in questa colonna.
2. **PRIMARY KEY**
   - Identifica in modo univoco ogni riga di una tabella. Deve contenere valori unici e non nulli. Può essere applicato a una singola colonna o a una combinazione di colonne.
3. **UNIQUE**
   - Garantisce che tutti i valori in una colonna (o in un gruppo di colonne) siano unici. Simile a `PRIMARY KEY`, ma una tabella può avere più vincoli `UNIQUE`.
4. **FOREIGN KEY**
   - Assicura l'integrità referenziale tra tabelle. Una colonna con questo vincolo deve contenere valori che esistono come `PRIMARY KEY` in un'altra tabella.
5. **CHECK**
   - Impone una condizione sui valori che possono essere inseriti in una colonna. Se la condizione risulta falsa, l'inserimento o l'aggiornamento viene rifiutato.
6. **DEFAULT**
   - Assegna un valore predefinito a una colonna se non viene specificato alcun valore durante l'inserimento di una nuova riga.

ESEMPIO DI CODICE

```sql
-- Creazione della tabella
CREATE TABLE users (
    id SERIAL PRIMARY KEY,
    username VARCHAR(50) NOT NULL,
    email VARCHAR(100) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Inserimento di dati
INSERT INTO users (username, email) VALUES ('john_doe', 'john@example.com');

-- Selezione di dati
SELECT * FROM users;

-- Aggiornamento di dati
UPDATE users SET email = 'john_doe_updated@example.com' WHERE username = 'john_doe';

-- Cancellazione di dati
DELETE FROM users WHERE username = 'john_doe';
```