UNIVERSITÀ DEGLI STUDI DI SALERNO

Atzeni, Ceri, Fraternali, Paraboschi, Torlone
Basi di dati *Quinta edizione*
McGraw-Hill Education, 2018

1

BASI DI DATI

Concetti Base SQL-DDL - prima parte

DIEM
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE ED ELETTRICA E MATEMATICA APPLICATA

Matteo Gaeta
Full Professor – Senior Member IEEE

Uno sforzo continuo - non la forza o l'intelligenza - è la chiave che sprigiona il nostro potenziale.
(Sir Winston Churchill)

BASI DI DATI

Concetti base – SQL

2

CENNI STORICI

- SQL (si pronuncia "ess-chiu-el" e non "siquel" come si sente spesso) inizia nel 1974 con la definizione da parte di D. Chamberlin e altri presso i laboratori di R&S dell'IBM di un linguaggio chiamato SEQUEL (Structured English Query Language) per la specificazione delle caratteristiche dei DB relational model-based.
- Tra il 1976 ed il 1977 fu effettuata una revisione del linguaggio (SEQUEL/2), che in seguito cambio' nome per motivi legali, diventando SQL.
- A partire dal 1981 IBM comincio' a rilasciare i suoi prodotti relazionali e nel 1983 comincio' a vendere DB2.
- Nel corso degli anni ottanta numerose compagnie (ad esempio Oracle e Sybase, solo per citarne alcuni) commercializzarono prodotti basati su SQL, che divenne lo standard industriale di fatto per quanto riguarda i database relazionali.

Tutti prendono decisioni costantemente.
Anche non scegliere è una scelta
(Søren Kierkegaard)

BASI DI DATI

Concetti base – SQL

CENNI STORICI

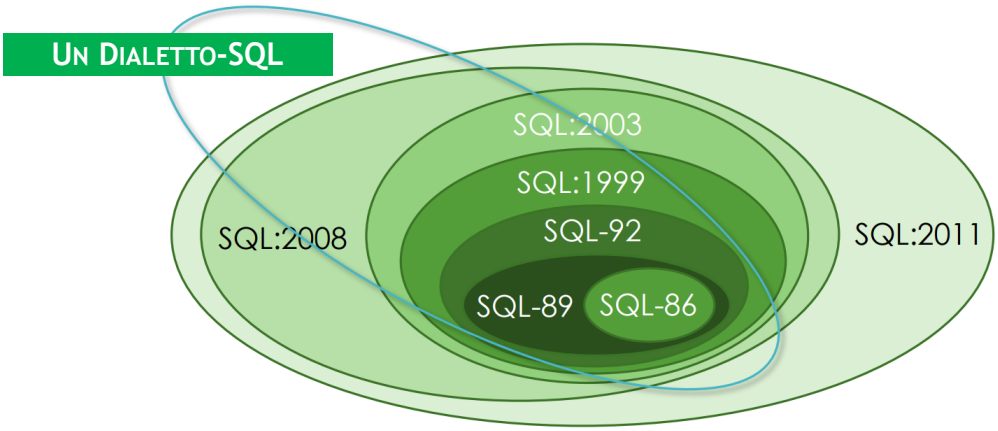
- Nel 1986 l'ANSI adotto' SQL (ovvero il dialetto SQL di IBM) come standard per i linguaggi relazionali e nel 1987 esso divenne lo standard ISO di SQL, detto SQL/86.
- Negli anni successivi esso ha subito varie revisioni che hanno portato prima alla versione SQL/89 e successivamente alla attuale SQL/92.
- Il fatto di avere uno standard definito per un linguaggio per database relazionali, aprì potenzialmente la strada alla intercomunicabilita' fra tutti i prodotti che si basano su di esso. Purtroppo nella realtà le cose sono andate diversamente.
- SQL Base (1986 e 1999) – Costrutti base e Integrità Refenziale; SQL2 (1992) detto anche SQL-92 – Modello Relazionale, Vari livelli nuovi e 3 livelli: Entry, Intermediate e Full; SQL3 (dal 1999 al 2016) – Modello Relazionale ad Oggetti; Trigger; Funzioni Esterne; Nuove Parti SQL-JRT, SQL-XML; estensioni XML; Gestione Dati Temporali e del formato JSON, ecc.

Domani sarò ciò che oggi ho scelto di essere.
(James Joyce)

BASI DI DATI

Concetti base – SQL

I DIALETTI SQL



- Ogni produttore adotta ed implementa nel proprio database solo il cuore del linguaggio SQL (il cosiddetto Entry level o al massimo l'Intermediate level), estendendolo in maniera proprietaria a seconda della propria visione del mondo dei database. Quindi vi sono diverse versioni, vediamo gli aspetti essenziali.

BASI DI DATI

Concetti base – SQL

LINGUAGGIO DICHIARATIVO SQL: STRUTTURA

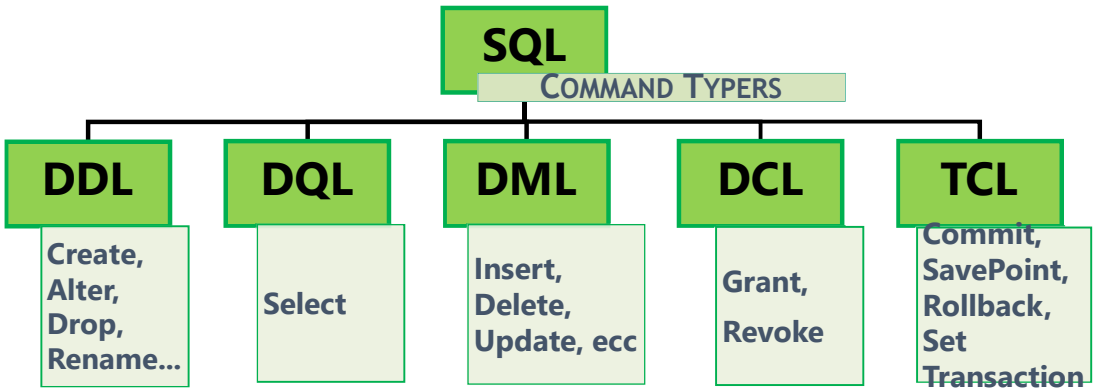
SQL non richiede la stesura di sequenze di operazioni (come ad es. i linguaggi imperativi), ma di specificare le proprietà logiche delle informazioni ricercate:

- **DDL (Data Definition Language)** – consente di creare, modificare o eliminare gli oggetti in un database. Esso permette di definire la struttura ma non di manipolare i dati contenuti nel DB. Ovviamente l'utente deve avere i permessi (assegnati tramite il DCL) - per agire sulla struttura del DB
- **DML (Data Manipulation Language)** – fornisce i comandi per inserire, modificare, eliminare o leggere i dati all'interno delle tabelle di un DB (SELECT, INSERT, UPDATE, DELETE ecc.). Ovviamente la struttura dei dati deve già essere stata definita tramite il DDL
- **DCL (Data Control Language)** – consente di fornire o revocare agli utenti i permessi necessari per poter utilizzare i comandi DML e DDL, oltre agli stessi comandi DCL
- **DQL (Data Query Language)** – permette di creare query sui DB e sui SI da parte degli utenti oltre a rendere possibile l'estrazione di informazioni dal DB interrogando la base dei dati interfacciandosi dunque con l'utente e le sue richieste di servizio

BASI DI DATI

Concetti base – SQL

LINGUAGGIO DICHIARATIVO SQL: STRUTTURA



BASI DI DATI

Concetti base – SQL

NOTAZIONE

- Notazione del Metalinguaggio utilizzato per specificare la sintassi dei comandi:
  - ✓ Le **parantesi quadre** **[ ]** indicano che il termine contenuto al suo interno è opzionale, cioè puo non comparire o comparire solo una volta.
  - ✓ Le **parentesi graffe** **{ }** indicano che il termine racchiuso tra parentesi può non comparire o essere ripetuto un numero arbitrario di volte
  - ✓ Le **barre verticali** **|** indicano che si è obbligati a scegliere uno e uno solo tra i termini separati dalle barre
  - ✓ Le **parentesi tonde** **( )** dovranno essere intese sempre come termini del linguaggio SQL e non come simboli del linguaggio SQL utilizzato per la definizione della grammatica

BASI DI DATI

Concetti base – SQL

SQL CASE-INSENSITIVE

- Ricordiamo che SQL è un linguaggio case-insensitive, ossia non distingue tra i caratteri maiuscoli e caratteri minuscoli, ad eccezione dell'interno delle stringhe.

- Ad esempio le due istruzioni:
  - ✓ Select * from QualcheTabella;
  - ✓ Select * from qualcheTABELLA;

  Sono equivalenti

- Tutti i comandi SQL terminano con un punto e virgola.

- Se si utilizza pgAdmin, il punto e virgola può essere omesso, in quanto il tool lo aggiunge automaticamente prima di inviare il comando al server

BASI DI DATI

Concetti base – SQL

La Terminazione delle Istruzioni

- Si osservi che tutto quelllo che «**psql**» non riesce ad interpretare come un suo comando interno viene trattenuto come una istruzione SQL.
- Le istruzioni possono richiedere piu righe, quindi è necessario informare «**psql**» della conclusione di queste righe permettendo all'ambiente di analizzarle e inviarle al server
- Le istruzioni possono terminare con un punto e virgola «**;**» oppure con il comando «**\g**»

BASI DI DATI

Concetti base – SQL

Token

- Un **token (o token lessicale)**, in informatica, è un blocco di testo categorizzato, normalmente costituito da caratteri indivisibili chiamati lessemi
- I token sono frequentemente **definiti come espressioni regolari**, che sono comprese da un analizzatore lessicale come Lex. L'analizzatore lessicale legge in un flusso di lessemi e li categorizza in token: se esso trova un token non valido, restituisce un errore
- L'operazione successiva alla tokenizzazione è il parsing
- Durante il parsing i dati interpretati possono essere caricati in strutture dati, per uso generico, interpretazione o compilazione

BASI DI DATI

Concetti base – SQL

LEXICAL STRUCTURE

- Molti manuali descrivono l'input SQL consistente in una sequenza di comandi
- Un comando è composto da una sequenza di token terminata da un punto e virgola (";"). La fine del flusso di input termina anche un comando
- Un token può essere una **parola chiave**, un **identificatore**, un **identificatore tra virgolette**, un **letterale** (o costante) o un **carattere speciale**
- I token sono normalmente separati da spazi (spazio, tabulazione, nuova riga), ciò non è vincolante a meno che non si determini qualche ambiguità

BASI DI DATI

Concetti base – SQL

LEXICAL STRUCTURE

- Ad esempio, il seguente è (sintatticamente) un input SQL valido, ovvero una sequenza di tre comandi, uno per riga (anche se ciò non è richiesto):

```
SELECT * FROM MY_TABLE;
UPDATE MY_TABLE SET A = 5;
INSERT INTO MY_TABLE VALUES (3, 'hi there');
```

- Questa è una sequenza di tre comandi, uno per riga (sebbene non richiesto); anche i comandi posso essere su una o più righe

BASI DI DATI

Concetti base – SQL

LEXICAL STRUCTURE

- Come in molti linguaggi possono essere presenti commenti nell'input SQL.
- La sintassi SQL non è molto coerente per quanto riguarda i token che identificano i comandi e quali sono operandi o parametri.
- I primi pochi token sono generalmente il nome del comando, normalmente parliamo di un comando "**SELECT**" – **SELEZIONA**, "**UPDATE**" – **AGGIORNA** e "**INSERT**" - **INSERISCI**
- Si osservi che il comando **UPDATE** richiede sempre che un token **SET** appaia in una certa posizione; anche la variazione di INSERT richiede dei VALORI per essere completa.

BASI DI DATI

Concetti base – SQL

PAROLE CHIAVE E IDENTIFICATORI

- I Token come SELECT, UPDATE o VALUES sono **parole chiave**, cioè parole che hanno un significato fisso nel linguaggio SQL.
- I token MY_TABLE e A **sono esempi di identificatori**. i nomi che identificano tabelle, colonne o altri oggetti di database, anche se sono identificatori vengono spesso chiamati semplicemente nomi.
- **Parole chiave e identificatori hanno la stessa struttura lessicale**, il che significa che non si può sapere se un token è un identificatore o una parola chiave senza conoscere la lingua. È possibile trovare un elenco completo delle parole chiave nell'Appendice C della documentazione Postgresql-NN.xx.

BASI DI DATI

Concetti base – SQL

PAROLE CHIAVE E IDENTIFICATORI

- Gli **identificatori SQL** e **le parole chiave** devono iniziare con una lettera (a-z, ma anche lettere con segni diacritici e lettere non latine) o un trattino basso (_)
- I caratteri successivi in un identificatore o una parola chiave in PostgreSQL possono essere lettere, trattini bassi, cifre (0-9) o segni di dollaro ($), **quest'ultimi non sono consentiti secondo lo standard SQL**, quindi il loro utilizzo potrebbe rendere le applicazioni meno portabili
- Il sistema PostgreSQL non utilizza più di «NAMEDATALEN» - 1 byte per un identificatore; è possibile scrivere nomi più lunghi nei comandi, ma verranno troncati.
- Per impostazione predefinita, NAMEDATALEN è 64, quindi la lunghezza massima degli identificatori è 63 byte. Tale limite se problematico, può essere aumentato modificando la costante NAMEDATALEN in src / include / pg_config_manual.h.

BASI DI DATI

Concetti base – SQL

COSTANTI (CONSTANTS)

- Esistono tre tipi di costanti tipizzate in modo implicito in PostgreSQL:
  - ✓ Stringhe (String Constants);
  - ✓ Stringhe di bit (Bit-string Constants);
  - ✓ Numeri (Numeric Constants).
- Le costanti possono anche essere specificate con tipi espliciti, che possono consentire una rappresentazione più accurata e una gestione più efficiente da parte del sistema.
- In particolare PostgreSQL prevede come String Constants, anche:
  - ✓ String Constants with C-style Escapes;
  - ✓ String Constants with Unicode Escapes;
  - ✓ Dollar-quoted String Constants.

BASI DI DATI

Concetti base – SQL

COSTANTI STRINGA DI BIT

- Le Bit-string Constants hanno immediatamente l'aspetto di costanti stringa regolari, ma con una B (maiuscola o minuscola) prima della virgoletta di apertura (senza spazi vuoti intermedi), ad esempio, B'1001'. Gli unici caratteri ammessi all'interno delle costanti stringa di bit sono 0 e 1

- Le Bit-string Constants, in alternativa, possono essere specificate in notazione esadecimale, utilizzando una X iniziale (maisucola o minuscola), ad esempio X'1FF'. Tale notazione è equivalente a una Bit-string constants con quattro cifre binarie per ogni cifra esadecimale.

- Entrambe le forme di costante stringa di bit possono essere posizionate su più righe allo stesso modo della stringa normale costanti.

- La quotazione in dollari non può essere utilizzata in una costante stringa di bit.

BASI DI DATI

Concetti base – SQL

COSTANTI NUMERICHE

- Le Numeric Constants sono accettate in queste forme generali:
  - ✓ **digits;**
  - ✓ **digits.[digits] [e[+ -]digits];**
  - ✓ **[digits].digits [e[+ -]digits];**
  - ✓ **digitse[+-]digits**
- dove digits sono una o più cifre decimali (da 0 a 9). Almeno una cifra deve essere prima o dopo punto decimale, se utilizzato.

- Almeno una cifra deve seguire l'indicatore di esponente (e), se presente.

- Non possono esserci spazi o altri caratteri incorporati nella costante.

- Nota che qualsiasi simbolo di plus iniziale o il segno meno non è considerato parte della costante; esso è considerato un operatore applicato alla costante.

BASI DI DATI

Concetti base – SQL

COSTANTI NUMERICHE

- Alcuni esempi di costanti numeriche valide: 42; 3.5; 4.; .001; 5e2; 1.925e-3
- Inizialmente le costanti che contengono punti decimali e/o esponenti si presume sempre che siano di tipo numeric, diversamente di tipo integer (se il valore rientra nel tipo integer - 32 bit) oppure di tipo bigint (se il valore rientra nel tipo bigint – 64 bit)
- Il tipo di dati inizialmente assegnato di una costante numerica è solo un punto di partenza per la risoluzione del tipo, successivamente nella maggior parte dei casi la costante verrà automaticamente forzata sul tipo più appropriato a seconda del contesto.
- Se necessario, è possibile forzare l'interpretazione di un valore numerico come uno specifico tipo di dati. Ad esempio, è possibile forzare un valore numerico da trattare come di tipo reale (float4)

BASI DI DATI

Concetti base – SQL

OPERATORI

- **Il nome di un operatore è una sequenza** di un massimo di NAMEDATALEN-1 (63 predefinito) caratteri dal file seguente elenco: **+ - * / <> = ~! @ #% ^ & | `?**
- Ci sono alcune limitazioni sui nomi degli operatori, tuttavia: **• - e /* non possono apparire da nessuna parte nel nome di un operatore**, poiché verranno presi come inizio di un commento.
- Un nome di operatore composto da più caratteri **non può terminare con + o -,** a meno che il nome non contenga anche almeno uno di questi caratteri:~!@#% ^&|`?
- Ad es., @ - è un nome operatore consentito, ma * - non lo è. Questo vincolo consente a PostgreSQL di analizzare le query conformi a SQL senza richiedere spazi tra i token.
- Quando si lavora con nomi di operatori non standard SQL, sarà generalmente necessario separare gli operatori adiacenti con spazi per evitare ambiguità.

BASI DI DATI

Concetti base – SQL

CARATTERI SPECIALI

- Alcuni caratteri che non sono alfanumerici hanno un significato speciale diverso dall'essere un operatore classico (+, - [, $, &, ecc.).
- I dettagli sull'utilizzo dei caratteri speciali possono essere trovati nella posizione in cui si l'elemento di sintassi in cui è utilizzato, nel seguito alcune informazioni essenziali:
  - ✓ **Un segno di dollaro ($)** seguito da cifre viene utilizzato per rappresentare un parametro posizionale nel corpo di una definizione di funzione o un'istruzione. In altri contesti il segno del dollaro può far parte di un file identificatore o una costante stringa quotata in dollari.
  - ✓ **Le parentesi (())** hanno il loro significato usuale per raggruppare le espressioni e imporre la precedenza. In alcuni casi sono richieste come parte della sintassi fissa di un particolare comando SQL.
  - ✓ **Le parentesi ([])** vengono utilizzate per selezionare gli elementi di un array.
  - ✓ **Le virgole (,)** sono usate in alcuni costrutti sintattici per separare gli elementi di un elenco.

BASI DI DATI

Concetti base – SQL

CARATTERI SPECIALI

- I dettagli sull'utilizzo dei caratteri speciali possono essere trovati nella posizione in cui si l'elemento di sintassi in cui è utilizzato, nel seguito alcune informazioni essenziali:
  - ✓ **Il punto e virgola (;)** termina un comando SQL. Non può apparire da nessuna parte all'interno di un comando, tranne all'interno di una stringa costante o identificatore tra virgolette.
  - ✓ **I due punti (:)** vengono utilizzati per selezionare le "sezioni" dagli array. In alcuni dialetti SQL (come Embedded SQL), i due punti vengono utilizzati per aggiungere un prefisso ai nomi delle variabili.
  - ✓ **L'asterisco (*)** è utilizzato in alcuni contesti per indicare tutti i campi di una riga di tabella o di un valore composto. Ha anche un significato speciale quando viene utilizzato come argomento di una funzione aggregata.
  - ✓ **Il punto (.)** Viene utilizzato nelle costanti numeriche e per separare i nomi di schema, tabella e colonna.

BASI DI DATI

Concetti base – SQL

COMMENTI

- Un commento è una sequenza di caratteri che inizia con doppi trattini e si estende fino alla fine di la linea, ad esempio:
  - - Questo è un commento SQL standard

- In alternativa, è possibile utilizzare commenti di blocco in stile C:
  /* commento anche su più righe */,
  dove il commento inizia con /* e si estende all'occorrenza corrispondente di */.

- Un commento viene rimosso dal flusso di input prima di un'ulteriore analisi della sintassi e viene effettivamente sostituito da spazi bianchi.

BASI DI DATI

Materiale utilizzato e bibliografia

- ***Le slide utilizzate dai docenti per le attività frontali sono in gran parte riconducibili e riprese dalle slide originali (con alcuni spunti parziali ripresi dai libri indicati) realizzate da:***
  - ✓ *autori del libro Basi di Dati (Atzeni e altri) testo di riferimento del corso Basi di Dati e sono reperibili su internet su molteplici link oltre che laddove indicato dagli stessi autori del libro;*
  - ✓ *Prof.ssa Tiziana Catarci e dal dott. Ing. Francesco Leotta – corso di Basi di Dati dell'Università degli Studi La Sapienza di Roma al seguente link ed altri: http://www.dis.uniroma1.it/~catarci/basidatGEST.html (molto Interessanti anche le lezioni su YouTube).*
  - ✓ *Proff. Luca Allulli e Umberto Nanni, Libro Fondamenti di basi di dati, editore HOEPLI (testo di facile lettura ed efficace).*
- ***Diverse slide su specifici argomenti utilizzate dai docenti per le attività frontali sono anche in parte riconducibili e riprese dalle slide originali – facilmente reperibili e accessibili su internet - realizzate da:***

Prof.ssa Roberta Aiello – corso Basi di Dati dell'Università di Salerno
Prof. Dario Maio - corso Basi di Dati dell'Università di Bologna al seguente link ed altri: http://bias.csr.unibo.it/maio
Prof. Marco Di Felice - corso Basi di Dati dell'Università di Bologna al seguente link ed altri: http://www.cs.unibo.it/difelice/dbsi/
Prof Marco Maggini e prof Franco Scarselli - corso Basi di Dati dell'Università di Siena ai seguenti link ed altri: http://staff.icar.cnr.it/pontieri/didattica/LabSI/lezioni/_preliminari-DB1%20(Maggini).pdf
Prof.ssa Raffaella Gentilini - corso Basi di Dati dell'Università di Perugia al seguente link ed altri: http://www.dmi.unipg.it/raffaella.gentilini/BD.htm
Prof. Enrico Giunchiglia - corso Basi di Dati dell'Università di Genova al seguente link ed altri: http://www.star.dist.unige.it/~enrico/BasiDiDati/
Prof. Maurizio Lenzerini - corso Basi di Dati dell'Università degli Studi La Sapienza di Roma al seguente link ed altri http://didatticainfo.altervista.org/Quinta/Database2.pdf

- The PostgreSQL Global Development Group - PostgreSQL nn.xx Documentation
- PostgreSQL (appendice - scaricabile dal sito del libro (area studenti) e www.postgresql.org