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 - seconda parte

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

Matteo Gaeta
Full Professor – Senior Member IEEE

Quando si effettua una scelta, si cambia il futuro.(Deepak Chopra)

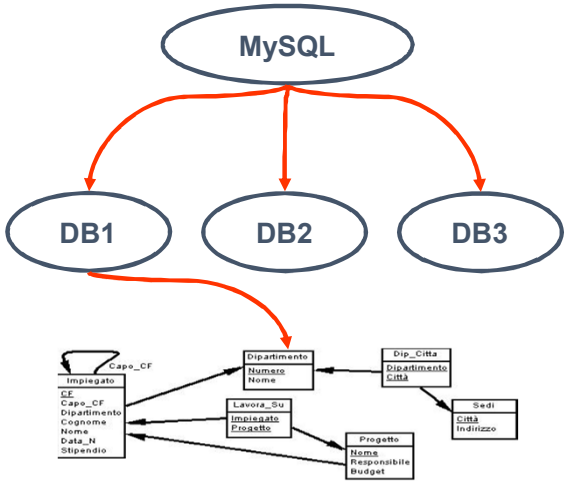
BASI DI DATI

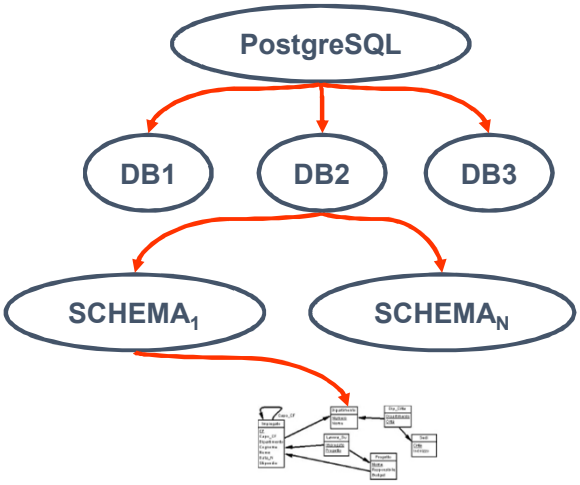
Concetti base – SQL

2

DATABASE VS SCHEMA

- Nelle prime versioni tutte le tabelle erano considerate parte del medesimo schema. Oggi, le tabelle sono spesso raggruppate in SCHEMI (insieme di oggetti





Nulla determina chi diventeremo come quelle cose che scegliamo di ignorare. (Sandor McNab)

BASI DI DATI

Concetti base – SQL

DATABASE VS SCHEMA

- Un cluster di database PostgreSQL contiene uno o più database denominati. Ruoli e alcuni altri tipi di oggetto sono condivisi nell'intero cluster. Gli utenti di un cluster non hanno necessariamente il privilegio di accedere a tutti i database nel cluster.
- Una connessione client al server può accedere solo ai dati in un unico database, quello specificato nella richiesta di connessione.
- Un database contiene uno o più schemi denominati, che a loro volta contengono tabelle. Gli schemi contengono anche altri tipi di oggetti con nome, inclusi tipi di dati, funzioni e operatori.
- Lo stesso nome dell'oggetto può essere utilizzato in diversi schemi senza conflitti; ad esempio, sia schema1 che myschema possono contenere tabelle denominate mytable.

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

BASI DI DATI

Concetti base – SQL

DATABASE VS SCHEMA

- A differenza dei database, gli schemi non sono rigidamente separati: un utente può accedere agli oggetti in uno qualsiasi degli schemi nel database a cui è connesso, se ha i privilegi per farlo. Ci sono diversi motivi per cui si potrebbe voler utilizzare gli schemi:
  - ✓ Consentire a molti utenti di utilizzare un database senza interferire tra loro.
  - ✓ Per organizzare gli oggetti del database in gruppi logici per renderli più gestibili.
  - ✓ Le applicazioni di terze parti possono essere inserite in schemi separati in modo che non entrino in conflitto con i nomi di altri oggetti.
- Gli schemi sono analoghi alle directory a livello di sistema operativo, tranne per il fatto che gli schemi non possono essere annidati.

BASI DI DATI

Concetti base – SQL

CREATE DATABASE

- **CREATE DATABASE,** crea un nuovo database (PostgreSQL).
  - ✓ Per creare un database, devi essere un superutente o avere lo speciale privilegio CREATEDB (Vedi CREATE USER)
- Per impostazione predefinita in PostgreSQL, il nuovo database verrà creato clonando il template di database di sistema standard1.
- Per un modello diverso può essere specificato scrivendo il nome TEMPLATE. In particolare, scrivendo **TEMPLATE** template0, puoi creare un database vergine contenente solo gli oggetti standard predefiniti dalla tua versione di PostgreSQL.

BASI DI DATI

Concetti base – SQL

CREATE DATABASE

- **CREATE DATABASE**

  CREATE DATABASE — create a new database
- **Synopsis**

```
CREATE DATABASE name
    [ [ WITH ] [ OWNER [=] user_name ]
           [ TEMPLATE [=] template ]
           [ ENCODING [=] encoding ]
           [ LC_COLLATE [=] lc_collate ]
           [ LC_CTYPE [=] lc_ctype ]
           [ TABLESPACE [=] tablespace_name ]
           [ ALLOW_CONNECTIONS [=] allowconn ]
           [ CONNECTION LIMIT [=] connlimit ]
           [ IS_TEMPLATE [=] istemplate ] ]
```

BASI DI DATI

Concetti base – SQL

PARAMETRI CREATE DATABASE

- ***name***, il nome di un database da creare.
- ***user_name***, il nome del ruolo dell'utente che sarà proprietario del nuovo database o DEFAULT per utilizzare il valore predefinito (ovvero, l'utente che esegue il comando). Per creare un database di proprietà di un altro ruolo, è necessario essere un membro diretto o indiretto di quel ruolo oppure essere un superutente.
- **template**, il nome del modello da cui creare il nuovo database o DEFAULT per utilizzare il modello predefinito (template1).

BASI DI DATI

Concetti base – SQL

PARAMETRI CREATE DATABASE

- ***encoding***, codifica del set di caratteri da utilizzare nel nuovo database. Specificare una costante di stringa (ad esempio, "SQL_ASCII") o un numero di codifica intero o DEFAULT per utilizzare la codifica predefinita (ovvero la codifica del database del modello). I set di caratteri supportati dal server PostgreSQL sono descritti nell'apposita Sezione della documentazione
- **lc_collate,** ordine di confronto (LC_COLLATE) da utilizzare nel nuovo database. Ciò influisce sull'ordinamento applicato alle stringhe, ad esempio nelle query con ORDER BY, nonché sull'ordine utilizzato negli indici sulle colonne di testo. L'impostazione predefinita è utilizzare l'ordine di confronto del template di riferimento del database.

BASI DI DATI

- **lc_ctype**, classificazione dei caratteri (LC_CTYPE) da utilizzare nel nuovo database. Ciò influisce sulla categorizzazione dei caratteri, ad es. Inferiore, superiore e cifra. L'impostazione predefinita è utilizzare la classificazione dei caratteri del database dei modelli. Vedi sotto per ulteriori restrizioni.

- **tablespace_name**, il nome del tablespace che verrà associato al nuovo database o DEFAULT per utilizzare il tablespace del database modello. Questo spazio tabella sarà lo spazio tabella predefinito utilizzato per gli oggetti creati in questo database. Vedi CREATE TABLESPACE per ulteriori informazioni.

- **allowconn**, se falso, nessuno può connettersi a questo database. L'impostazione predefinita è true, consentendo le connessioni (tranne quando limitate da altri meccanismi, come GRANT / REVOKE CONNECT).

BASI DI DATI

- **connlimit**, quante connessioni simultanee possono essere effettuate a questo database. -1 (l'impostazione predefinita) significa nessun limite.
- **istemplate**, se true, questo database può essere clonato da qualsiasi utente con privilegi CREATEDB; se false (impostazione predefinita), solo i superutenti o il proprietario del database possono clonarlo.
- L'istruzione **CREATE DATABASE** non può essere eseguito all'interno di un blocco di transazione. Utilizzare **DROP DATABASE** per rimuovere un database.
- Gli errori del tipo "***impossibile inizializzare la directory del database***" sono molto probabilmente correlati a autorizzazioni insufficienti sulla directory dei dati, un disco pieno o altri problemi del file system.
- Il **programma createdb** è un programma, fornito per comodità. I parametri di configurazione a livello di database (impostati tramite **ALTER DATABASE**) e le autorizzazioni a livello di database (impostati tramite **GRANT**) non vengono copiati dal template del database.

BASI DI DATI

Concetti base – SQL

CREATE SCHEMA

- Per creare uno schema, utilizzare il comando **CREATE SCHEMA**. Assegna allo schema un nome a tua scelta. Per esempio: ***CREATE SCHEMA myschema;***
- Per creare o accedere a oggetti in uno schema, scrivi un nome completo composto dal nome dello schema e dal nome della tabella separati da un punto: **schema.table**
- Tale modalità funziona ovunque sia previsto il nome di una tabella (le stesse idee si applicano ad altri tipi di oggetti come tipi, funzioni, ecc.), inclusi i comandi di modifica della tabella e i comandi di accesso ai dati (o altro)
- In realtà, la sintassi ancora più generale è: **database.schema.table** ciò è richiesto solo per la conformità **pro forma** con lo standard SQL. Se scrivi un nome di database, deve essere lo stesso del database a cui sei connesso.

BASI DI DATI

Concetti base – SQL

CREATE SCHEMA

- Quindi, per creare una tabella nel nuovo schema, usa:
  CREATE TABLE myschema.mytable (...);
- Per eliminare uno schema se è vuoto (tutti gli oggetti in esso sono stati eliminati), utilizzare:
  DROP SCHEMA myschema;
- Per eliminare uno schema che includa tutti gli oggetti contenuti, utilizzare:
  DROP SCHEMA myschema CASCADE;
- Spesso vorrai creare uno schema di proprietà di qualcun altro (poiché questo è uno dei modi per limitare le attività dei tuoi utenti a spazi ben definiti). La sintassi è: **CREATE SCHEMA schema_name AUTORIZZAZIONE nome_utente;**

BASI DI DATI

Concetti base – SQL

CREATE SCHEMA

- È possibile omettere il nome dello schema, nel qual caso il nome dello schema sarà lo stesso del nome utente.
- I nomi degli schemi che iniziano con pg_ sono riservati per scopi di sistema e non possono essere creati dagli utenti.
- **Lo SCHEMA PUBBLIC in PostgreSQL,** spesso creiamo tabelle senza specificare alcun nome di schema. Per impostazione predefinita tali tabelle (e altri oggetti) vengono automaticamente inseriti in uno schema denominato "public".
- In PostgreSQL ogni nuovo database contiene un tale schema, pertanto ***CREATE TABLE prodotti (...);*** e: **CREATE TABLE public.products (...);** sono equivalenti:

BASI DI DATI

Concetti base – SQL

DATABASE VS SCHEMA

- Nello standard SQL, la nozione di oggetti nello stesso schema di proprietà di utenti diversi non esiste.
- Inoltre, alcune implementazioni non consentono di creare schemi che hanno un nome diverso dal loro proprietario.
- In effetti, i concetti di schema e utente sono quasi equivalenti in un sistema di database che implementa solo il supporto dello schema di base specificato nello standard.
- Pertanto, molti utenti considerano i nomi qualificati costituiti realmente da nome_utente.nome_tabella; questo è il modo in cui PostgreSQL si comporterà efficacemente se crei uno schema utente per ogni utente.

BASI DI DATI

Concetti base – SQL

DATABASE VS SCHEMA

- Precisiamo che non esiste il concetto di uno schema pubblic nello standard SQL; per la massima conformità allo standard, non utilizzare lo schema pubblic.
- SQL Standard, consente la definizione di uno schema di base di dati come collezione di oggetti (tabelle, domini, viste, ecc.), utilizzando la seguente sintassi: ***Create Schema [NomeSchema] [[authorization] Autorizzazione] {DefElementoSchema},*** dove Autorizzazione, rappresenta il nome dell'utente proprietario dello schema; se il termine è omesso si assume come nome dello schema quello del proprietario.
- La descrizione di tutti i componenti dello schema puo avvenire in differita dalla creazione dello schema, senza alcun problema.

BASI DI DATI

Concetti base – SQL

DOMINI ELEMENTARI

- SQL rende disponibili alcune famiglie di domini elementari, a partire dai quali è possibile definire i domini da associare agli attributi dello schema; ad es. l'attributo di una relazione

Tipo	Domini SQL	Valori di esempio
Stringa	CHAR(n) VARCHAR(n)	'RSSMRA23A92H501L' 'Gianfilippo'
Intero	INT o INTEGER SMALLINT BIGINT	10000 50 300000000
Virgola mobile	REAL DOUBLE [PRECISION]	14.9702 14.970287591374
Tempo	TIME DATE TIMESTAMP	'06:30:00' '2015-01-30' '2015-01-03 06:30:00'
Booleano	BOOLEAN	TRUE

BASI DI DATI

Concetti base – SQL

- **Domini elementari (predefiniti) e Domini definiti dall'utente (semplici, ma riusabili)**
- ✓ **Carattere**
  - **singoli caratteri o stringhe di lunghezza fissa** - **char(n)**, **character(n)**, dove n è il numero massimo di caratteri che desideriamo memorizzare, ovviamente la lunghezza del campo definito con dominio char(n) è precisamente di dimensione n bytes indipendentemente dal valore inserito;
  - **stringhe di lunghezza variabile** – **varchar(n)**, dove la lunghezza del campo definito assume la dimensione del dato inserito + 1 byte di prefisso;
- ✓ **Numerici**
  - **valori numerici esatti – int** (o **smallint**, **bigint**) [**Unsigned**], in pratica interi di lunghezza fissa;
  - **valori numerici esatti con eventuale parte frazionaria - numeric**, **numeric(p), numeric (p,s),** con p max numero cifre rappresentabili e s la scala ovvero il numero di cifre dopo la virgola con s<=p. es numeric(3,1) consente di rappresentare da -99.9 a +99.9 mentre numeric(3,2) da -9.99 a +9.99;

BASI DI DATI

Concetti base – SQL

- **Domini elementari (predefiniti) e Domini definiti dall'utente (semplici, ma riusabili)**

- ✓ **Bit, bit(n):** sequenza fissa di n bit (valori appartenenti all'insieme {0,1}, es. <(00),(01),(10),(11,)>

- ✓ **Data:** ammette i campi in formato 'aaaa-mm-gg'
- ✓ **Time:** contiene un valore di tempo nel formato 'hh:mm:ss'
- ✓ **Timestamp:** contiene data e ora nel formato 'AAAAMMGGhhmmss'

- ✓ **Introdotti in SQL:1999:**
  - **BOOLEAN:** utilizzato per rappresentare i valori booleani true e false
  - **BLOB, CLOB** (**B**inary/**C**haracter **L**arge **O**bject): per grandi immagini e testi

- **Istruzione *CREATE DOMAIN***: definisce un dominio (semplice), utilizzabile in definizioni di relazioni, anche con vincoli e valori di default

BASI DI DATI

Concetti base – SQL

TABLE ⇔ Tabella ⇔ Relazione

ROW ⇔ Riga ⇔ Tupla

COLUMN ⇔ Colonna ⇔ Attributo

- **SQL come DDL ha tre comandi fondamentali, che agiscono a livello di struttura dei dati e non di valori dei dati:**

Create

Database, Schema, Table, Domain, Constraint, ….

Alter

Database, Schema, Table, Domain, Constraint, ….

Drop

Database, Schema, Table, Domain, Constraint, ….

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