Laboratorio di Basi di Dati
Per Bioinformatica

Laurea in Bioinformatica - A.A. 2009/10

Docente: Carlo Combi
Email: carlo.combi@univr.it

Lezione 1

SQL

- Structured Query Language
- SQL è stato definito nel 1973 ed è oggi il linguaggio più diffuso per i DBMS relazionali
- È un linguaggio con varie funzionalità:
  - Definizione delle strutture dati e dei vincoli di integrità (Data Definition Language DDL)
  - Linguaggio per modificare dati (Data Manipulation Language DML)
  - Linguaggio per interrogare la base di dati (Query Language)

Definizione Dati in SQL

- Istruzione CREATE TABLE:
  - Definisce uno schema di relazione (o tabella) e ne crea un'istanza vuota
  - Specifica attributi, domini, vincoli

CREATE TABLE NomeTabella
(Attributo Tipo [Valore Default][Vincolo Attributo]
{, Attributo Tipo [Valore Default][Vincolo Attributo]}
{, Vincolo Tabella});

Domini

- Domini elementari (predefiniti):
  - Carattere: singoli caratteri o stringhe anche di lunghezza variabile
  - Bit: singoli booleani (flag) o stringhe
  - Numerici, esatti e approssimati
  - Data, ora, intervalli di tempo
- Domini definiti dall'utente

Dominio CARATTERE

- Permette di rappresentare singoli caratteri oppure stringhe.
- La lunghezza delle stringhe può essere fissa o variabile.

character [varying][(Lunghezza)]

Forme abbreviate:

character ⇨ CHAR

character varying (20) ⇨ VARCHAR(20)

Dominio BIT

- Tipicamente usato per rappresentare attributi, detti FLAG, che specificano se l'oggetto rappresentato da una tupla possiede o meno quella proprietà.
- Si può anche definire un dominio "stringa di bit".

bit [varying][(Lunghezza)]

Dominio TIPI NUMERICI ESATTI

- Permette di rappresentare valori interi o valori decimali in virgola fissa.
- SQL mette a disposizione 4 diversi tipi:
  - NUMERIC
  - DECIMAL

  Numeri in base decimale
  - INTEGER
  - SMALLINT

  Se non interessa avere una rappresentazione della parte frazionaria

numeric [(Precisione [, Scala])]
decimal [(Precisione [, Scala])]

numeric(4,2) → 4 cifre significative,
2 cifre dopo la virgola

Dominio
TIPI NUMERICI APPROSSIMATI

- Permette di rappresentare valori numerici approssimati mediante una rappresentazione in virgola mobile.
- SQL mette a disposizione 3 diversi tipi:
  - FLOAT
  - REAL
  - DOUBLE PRECISION
- SQL-standard notazione:
  - float [(Precisione)] (Precisione = cifre mantissa)

Domini per il TEMPO

- Permettono di rappresentare istanti di tempo in tre diverse forme:
  - DATE (year, month, day)
  - TIME (hour, minute, second)
  - TIMESTAMP (tutti i campi da year a second)

time [(Precisione)][with time zone]

timestamp [(Precisione)][with time zone]

CREATE TABLE: Esempio

```
CREATE TABLE Impiegato
(    Matricola    CHAR(6),
     Nome         VARCHAR(20),
     Cognome      VARCHAR(20),
     Qualifica    VARCHAR(20),
     Stipendio    FLOAT       );
```

Vincoli intrarelazionali

- Vincoli di integrità: proprietà che devono essere soddisfatte da ogni istanza della base di dati
- Vincoli di integrità intrarelazionali: si riferiscono a singole relazioni della base di dati:
- NOT NULL: attributo non nullo
- UNIQUE: definisce (super)chiavi
- PRIMARY KEY: definisce la chiave primaria (una sola, implica NOT NULL)
- CHECK(condizione): vincolo generico

NOT NULL

- Il valore nullo non è ammesso come valore dell'attributo.
  - Il valore dell'attributo deve essere specificato in fase di inserimento.

Nome VARCHAR(20) NOT NULL

UNIQUE

- Impone che i valori di un attributo (o di un insieme di attributi) siano una superchiave, quindi tuple differenti della tabella non possono avere gli stessi valori.
- Si può definire su
  - un solo attributo

```
Matricola   CHAR(6) UNIQUE
```

  - un insieme di attributi

```
Nome        VARCHAR(20),
Cognome     VARCHAR(20),
UNIQUE(Nome,Cognome)
```

Su più attributi: attenzione!

```
Nome        VARCHAR(20) NOT NULL,
Cognome     VARCHAR(20) NOT NULL,
UNIQUE(Nome,Cognome)
```

Impone che non ci siano due righe che abbiano uguali sia il nome che il cognome

```
Nome        VARCHAR(20) NOT NULL UNIQUE,
Cognome     VARCHAR(20) NOT NULL UNIQUE,
```

Impone che non ci siano due righe che abbiano lo stesso nome o lo stesso cognome

PRIMARY KEY

- Specifica la chiave primaria della relazione
  - Si usa una sola volta per tabella
  - Implica una definizione di NOT NULL

- Due forme:
  - Nella definizione di un attributo, se forma da solo la chiave primaria

```
Matricola CHAR(6) PRIMARY KEY
```

  - Come elemento separato (necessario quando la chiave primaria e' composta da piu' attributi)

```
Nome        VARCHAR(20),
Cognome     VARCHAR(20),
PRIMARY KEY(Nome,Cognome)
```

CREATE TABLE: Esempio

```
CREATE TABLE Impiegato
   (    Matricola     CHAR(6)  PRIMARY KEY,
        Nome          VARCHAR(20) NOT NULL,
        Cognome       VARCHAR(20) NOT NULL,
        Qualifica     VARCHAR(20),
        Stipendio     FLOAT  DEFAULT 0.0,
        UNIQUE(Cognome, Nome)                 );
```

Il valore che deve assumere l'attributo quando viene inserita una riga nella tabella senza che sia specificato un valore per l'attributo stesso. Se non specificato, si assume come valore di default null

CREATE TABLE: esempio Check

```
CREATE TABLE Impiegato
(    Matricola    CHAR(6)  PRIMARY KEY,
     Nome         VARCHAR(20) NOT NULL,
     Cognome      VARCHAR(20) NOT NULL,
     Qualifica    VARCHAR(20),
     Stipendio    FLOAT  DEFAULT 100.0,
     UNIQUE(Cognome, Nome),
     CHECK (Stipendio>= 100)    );
```

INSERT

- Come popolare una tabella (inserimento righe):

INSERT INTO NomeTabella
[(ElencoAttributi)] VALUES (Elenco di Valori);

- Esempio:

```
INSERT INTO Impiegato(Matricola,Nome,Cognome)
VALUES ('A00001', 'Mario', 'Rossi');
```

SQL: Creazione Tabelle e inserimento dati

```
CREATE TABLE indirizzi (
        nome            varchar(20),
        cognome         varchar(20),
        indirizzo       varchar(50),
        email           varchar(30)   );
```

```
INSERT INTO indirizzi
VALUES ('Mario', 'Rossi', 'via Dante, 3', 'mario@rossi.com' );
```

PostgreSQL

PostgreSQL

- Sistema di gestione di basi di dati relazionali ad oggetti sviluppato in varie fasi fin dal 1977
- È largamente considerato il sistema di gestione di basi di dati *Open Source* più avanzato
- L'interazione tra un utente (programmatore della base di dati o utente finale) e la base di dati segue il modello client-server

PostgreSQL

Per ogni connessione stabilita vengono coinvolti tre processi UNIX:

- il postmaster: processo deamon con funzione di supervisione (gestisce le basi di dati presenti sul server);
- l'applicazione front-end dell'utente (**psql**): ogni utente che si connette lancia questo programma;
- un back-end database server (per ogni connessione)

Uso locale di PostgreSQL

- Il server **dbserver** è anche un server PostgreSQL
  - sono disponibili tante basi di dati quanti sono gli utenti. Ogni utente accede alla propria base di dati:

    dblabXXX e' la base di dati dell'utente *userlabXXX*
- Come ci si connette?
  - **export PGUSER = userlabXXX**
  - **psql** -h <nome server> -d <nome database>

    psql –h dbserver -d dblabXXX

Uso locale di PostgreSQL

- Il nome della tabella e' univoco in una base di dati. Quindi non possono essere create due tabelle con lo stesso nome

- Terminare ogni comando con il carattere `;'

Lavorare in *psql*

- *psql* è un client a riga di comando, distribuito insieme a PostgreSQL

- *psql*, pur essendo semplice, è molto potente e permette l'interazione diretta con il server PostgreSQL

Accesso all'archivio

- **`psql -h dbserver -d dblabXXX`**

- Noteremo, lavorando dalla linea di comando, un messaggio di benvenuto ed il cambiamento del prompt:

```
Welcome to the POSTGRESQL interactive sql monitor: ….
type \? for help on slash commands
type \q to quit
type \i FILENAME    execute commands from file
type \r reset (clear) the query buffer
type \g or terminate with semicolon to execute query
You are currently connected to the database: dblabXX
dblabXX=>
```

Introduzione alla sintassi di *psql*

- Subito dopo l'avvio di *psql* appare un breve sommario di quattro *comandi slash*

  - `\h` per l'aiuto su SQL
  - `\?` per l'aiuto sui comandi *psql*
  - `\g` per l'esecuzione delle query
  - `\q` per uscire da *psql* una volta terminato

Comandi slash

- **\a**
  attiva e disattiva la modalità di allineamento
- **\c[onnect] [nomedb| - [utente]]** si connette ad un nuovo database
- **\C <titolo>**
  titolo della tabella
- **\d <table>**
  descrive la tabella (o la vista)
- **\d{t|i|s|v}**
  elenca tabelle/indici/sequenze/viste
- **\d{p|S|l}**
  elenca permessi/tabelle di sistema/large object
- **\da**
  elenca gli aggregati

Comandi slash

- **`\dd [oggetto]`**
  elenca i commenti per tabella, tipo, funzione o operatore
- **`\df`**
  elenca le funzioni
- **`\do`**
  elenca gli operatori
- **`\dT`**
  elenca i tipi di dati
- **`\e [file]`**
  permette la modifica del buffer della query attuale o di [file] con un editor esterno
- **`\echo <testo>`**
  scrive il testo sullo stdout
- **`\f <sep>`**
  cambia il separatore dei campi

Comandi slash

- **\g [file]**
  invia la query al backend (e I risultati in [file] o |pipe)
- **\h [cmd]**
  aiuto sulla sintassi dei comandi sql; * per tutti i comandi
- **\H**
  attiva o disattiva la modalita' HTML (attualmente disinserita)
- **\i <file>**
  legge ed esegue la query da file
- **\l**
  elenca tutti i database
- **\o [file]**
  invia tutti I risultati della query nel [file] o |pipe
- **\p**
  mostra il contenuto del buffer della query attuale

Comandi slash

- **\pset <opt>**
  mostra l'output della tabella;
  <opt>={ format|border|expanded|fieldsep|null|recordsep|
  tuples_only|ttitle|tableattr|pager}
- **\q**
  esce da psql
- **\r**
  cancella il buffer della query
- **\s [file]**
  stampa lo storico dei comandi (history) e lo salva in [file]
- **\set <var> <valore>**
  imposta una variabile interna
- **\t**
  visualizza solo le righe
- **\T <tags>**
  imposta gli attributi dei tag HTML per le tabelle

Comandi slash

- **\unset <var>**

  elimina una variabile interna

- **\w <file>**

  scrive il buffer della query attuale su <file>

- **\x**

  attiva e disattiva l'ouptut esteso

- **\z**

  elenca i permessi di accesso alle tabelle

- **\! [cmd]**

  esce sulla shell o esegue un comando

Tipi di Dati principali in PostgreSQL

- **varchar(n):** stringa di lunghezza variabile minore o uguale a "n"
- **char:** carattere singolo
- **char(n):** stringa di lunghezza fissa di "n" caratteri
- **integer:** un intero di non più di nove cifre
- **float:** un numero in virgola mobile
- **real:** numero reale
- **date:** data
- **time:** l'orario
- **timestamp:** data + orario
- **interval:** intervallo di tempo
- **bytea:** stringa binaria di lunghezza variabile