Laboratorio di Basi di Dati e Web

Docente: Alberto Belussi

Lezione 1

SQL

- Structured Query Language
- SQL è stato definito nel 1973 ed è oggi il linguaggio più diffuso per i DBMS relazionali
- Il linguaggio SQL è composto di diverse parti:
  - Definizione delle strutture dati e dei vincoli di integrità
  - Linguaggio per modificare dati (inserimento, aggiornamento e cancellazione)
  - Linguaggio per interrogare la base di dati (Query language)

Definizione Dati in SQL

- Istruzione CREATE TABLE:
  - Definisce lo schema di una 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

  numeric [(Precisione [, Scala])]

  decimal [(Precisione [, Scala])]

  - INTEGER
  - SMALLINT

  Se non interessa avere una rappresentazione precisa della parte frazionaria
- Esempi:

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

Dominio
TIPI NUMERICI APPROSSIMATI

- Permette di rappresentare valori numerici approssimati mediante l'approccio in virgola mobile.
- SQL mette a disposizione 3 diversi tipi numerici approssimati:
  - REAL (in postgres: "range of at least 1E-37 to 1E+37 with a precision of at least 6 decimal digits")
  - DOUBLE PRECISION (in postgres: "range of around 1E-307 to 1E+308 with a precision of at least 15 digits")
  - SQL-standard notazione: float [(Precisione)] (Precisione = cifre mantissa)

Domini per il TEMPO

- Permette di rappresentare istanti di tempo.
  - DATE: (year, month, day)
  - TIME: (hour, minute, second)
  - TIMESTAMP: date ∪ time

time [(Precisione)][with time zone]

timestamp [(Precisione)][with time zone]

Precisione = numero di cifre decimali usate per rappresentare le frazioni di secondo

with time zone = se specificato risultano disponibili due campi in più: timezone_hour e timezone_minute che rappresentano la differenza con l'ora di Greenwich.

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: sono proprietà che devono essere soddisfatte da ogni istanza della base di dati.

Vincoli di integrità intrarelazionali: riguardano proprietà che si riferiscono a singole relazioni della base di dati:

- NOT NULL: attributo non nullo
- UNIQUE: definisce chiavi
- PRIMARY KEY: definisce la chiave primaria (una sola, implica NOT NULL)
- CHECK(espr): vincolo generico

NOT NULL

- Implica che il valore nullo non sia 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
  - un insieme di attributi

```
Matricola     CHAR(6) UNIQUE

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 definizione separata a livello di tabella (necessario quanto la chiave primaria è composta da più 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);

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

PostgreSQL

- PostgreSQL è un DBMS relazionale ad oggetti
- Software multipiattaforma di pubblico dominio
- L'interazione tra un utente (programmatore della base di dati, o utente finale) e la base di dati considerata avviene secondo il modello client-server

PostgreSQL

- Per ogni connessione stabilita vengono coinvolti tre processi UNIX:
  - il postmaster: un processo deamon con funzione di supervisione (gestisce le basi di dati presenti sul server);
  - l'applicazione frontend dell'utente (**psql**): ogni utente che si connette lancia questo programma;
  - un backend database server (per ogni connessione).

Uso locale di PostgreSQL

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

    dblabXX è la base di dati dell'utente *userlabXX*
- Come ci si connette?
  - **export PGUSER=userlabXX**
  - **psql** –h <nome server> –d <nome database>

    psql –h sqlserver –d dblabXX

Uso locale di PostgreSQL

- Il nome della tabella è univoco in una base di dati. Quindi non possono essere create due tabelle con lo stesso nome
- Terminare ogni comando SQL con il carattere ;

Lavorare in psql

- **psql –h sqlserver –d dblabXX**
- 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=>

Creazione Tabelle e inserimento dati

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

Il comando viene eseguito. Appare messaggio CREATE e poi il prompt

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

Appare un messaggio del tipo INSERT 0 1 e poi il prompt

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