SQL
Esempi

Esercitazioni pratiche

- Per SQL è possibile (e fondamentale) svolgere esercitazioni pratiche
- Verranno anche richieste copme condizione per svolgere le prove parziali
- Soprattutto sono utilissime
- Si può utilizzare qualunque DBMS
  - IBM DB2, Microsoft SQL Server, Oracle, PostgresSQL, ...
- A lezione utilizziamo PostgresSQL

CREATE TABLE, esempi

```
CREATE TABLE corsi(
    codice numeric NOT NULL PRIMARY KEY,
    titolo character(20) NOT NULL,
    cfu numeric NOT NULL)

CREATE TABLE esami(
    corso numeric REFERENCES corsi (codice),
    studente numeric REFERENCES studenti (matricola),
    data date NOT NULL,
   voto numeric NOT NULL,
   PRIMARY KEY (corso, studente))
```

La chiave primaria viene definita come NOT NULL anche se non lo specifichiamo (in Postgres)

DDL, in pratica

- In molti sistemi si utilizzano strumenti diversi dal codice SQL per definire lo schema della base di dati
- Vediamo

SQL, operazioni sui dati

- interrogazione:
  - SELECT
- modifica:
  - INSERT, DELETE, UPDATE

Inserimento

(necessario per gli esercizi)

INSERT INTO Tabella [(Attributi)]
VALUES(Valori)

oppure

INSERT INTO Tabella [(Attributi)]
SELECT ...
(vedremo più avanti)

```
INSERT INTO Persone VALUES ('Mario',25,52)

INSERT INTO Persone(Nome, Reddito, Eta)
      VALUES('Pino',52,23)

INSERT INTO Persone(Nome, Reddito)
      VALUES('Lino',55)
```

Maternità

Madre	Figlio
Luisa	Maria
Luisa	Luigi
Anna	Olga
Anna	Filippo
Maria	Andrea
Maria	Aldo

Paternità

Padre	Figlio
Sergio	Franco
Luigi	Olga
Luigi	Filippo
Franco	Andrea
Franco	Aldo

Persone

Nome	Età	Reddito
Andrea	27	21
Aldo	25	15
Maria	55	42
Anna	50	35
Filippo	26	30
Luigi	50	40
Franco	60	20
Olga	30	41
Sergio	85	35
Luisa	75	87

Esercizi

- Definire la base di dati per gli esercizi
  - installare un sistema
  - creare lo schema
  - creare le relazioni (CREATE TABLE)
  - inserire i dati

  - eseguire le interrogazioni
    - suggerimento, usare schemi diversi

    set search_path to <*nome schema*>

```
create table persone (
  nome char (10) not null primary key,
  eta numeric not null,
  reddito numeric not null);
create table paternita (
  padre char (10) not null ,
  figlio char (10) not null primary key);
...
insert into Persone values('Andrea',27,21);
...
insert into Paternita values('Sergio','Franco');
...
```

Istruzione SELECT (versione base)

SELECT ListaAttributi
FROM ListaTabelle
[WHERE Condizione]

- "target list"
- clausola FROM
- clausola WHERE

Intuitivamente

SELECT ListaAttributi
FROM ListaTabelle
[WHERE Condizione]

- Prodotto cartesiano di ListaTabelle
- Selezione su Condizione
- Proiezione su ListaAttributi

Selezione e proiezione

- Nome e reddito delle persone con meno di trenta anni

$$PROJ_{Nome,\ Reddito}(SEL_{Eta<30}(Persone))$$

```
select nome, reddito
from persone
where eta < 30
```

Selezione, senza proiezione

- Nome, età e reddito delle persone con meno di trenta anni

$$SEL_{Eta<30}(Persone)$$

```
select *
from persone
where eta < 30
```

Proiezione, senza selezione

- Nome e reddito di tutte le persone

$$PROJ_{Nome,\ Reddito}(Persone)$$

```
select nome, reddito
from persone
```

Proiezione, con ridenominazione

- Nome e reddito di tutte le persone

$$REN_{Anni \leftarrow Eta}(PROJ_{Nome,\ Eta}(Persone))$$

```
select nome, eta as anni
from persone
```

Proiezione, attenzione

```
select
    madre
from maternita
```

```
select distinct
    madre
from maternita
```

Condizione complessa

```
select *
from persone
where reddito > 25
      and (eta < 30 or eta > 60)
```

Maternità

Madre	Figlio
Luisa	Maria
Luisa	Luigi
Anna	Olga
Anna	Filippo
Maria	Andrea
Maria	Aldo

Paternità

Padre	Figlio
Sergio	Franco
Luigi	Olga
Luigi	Filippo
Franco	Andrea
Franco	Aldo

Persone

Nome	Età	Reddito
Andrea	27	21
Aldo	25	15
Maria	55	42
Anna	50	35
Filippo	26	30
Luigi	50	40
Franco	60	20
Olga	30	41
Sergio	85	35
Luisa	75	87

Selezione, proiezione e join

- I padri di persone che guadagnano più di 20

$$PROJ_{Padre}(paternita \; JOIN_{Figlio = Nome} \; SEL_{Reddito>20}(persone))$$

```
select distinct padre
from persone, paternita
where figlio = nome and reddito > 20
```

- Le persone che guadagnano più dei rispettivi padri; mostrare nome, reddito e reddito del padre

$$\text{PROJ}_{\text{Nome, Reddito, RP}}\,(\text{SEL}_{\text{Reddito>RP}}\,(\text{REN}_{\text{NP,EP,RP} \leftarrow \text{Nome,Eta,Reddito}}(\text{persone})\ \text{JOIN}_{\text{NP=Padre}}\ (\text{paternita JOIN}_{\text{Figlio =Nome}}\ \text{persone})))$$

```
select f.nome, f.reddito, p.reddito
from persone p, paternita, persone f
where p.nome = padre and
        figlio = f.nome and
        f.reddito > p.reddito
```

SELECT, con ridenominazione del risultato

```
select figlio, f.reddito as reddito,
            p.reddito as redditoPadre
from persone p, paternita, persone f
where p.nome = padre and figlio = f.nome
            and f.reddito > p.reddito
```

Join esplicito

- Padre e madre di ogni persona

```
select paternita.figlio,padre, madre
from maternita, paternita
where paternita.figlio = maternita.figlio

select madre, paternita.figlio, padre
from maternita join paternita on
                paternita.figlio = maternita.figlio
```

- Le persone che guadagnano più dei rispettivi padri; mostrare nome, reddito e reddito del padre

```
select f.nome, f.reddito, p.reddito
from persone p, paternita, persone f
where p.nome = padre and
        figlio = f.nome and
        f.reddito > p.reddito

select f.nome, f.reddito, p.reddito
from (persone p join paternita on p.nome = padre)
        join persone f on figlio = f.nome
where f.reddito > p.reddito
```

Join esterno: "outer join"

- Padre e, se nota, madre di ogni persona

```
select paternita.figlio, padre, madre
from paternita left join maternita
        on paternita.figlio = maternita.figlio
```

```
select paternita.figlio, padre, madre
from paternita left outer join maternita
        on paternita.figlio = maternita.figlio
```

- outer e' opzionale

Ordinamento del risultato

- Nome e reddito delle persone con meno di trenta anni in ordine alfabetico

```
select nome, reddito
from persone
where eta < 30
order by nome
```

Espressioni nella target list

```
select Nome, Reddito/12 as redditoMensile
from Persone
```

Condizione “LIKE”

- Le persone che hanno un nome che inizia per 'A' e ha una 'd' come terza lettera

```
select *
from persone
where nome like 'A_d%'
```

Gestione dei valori nulli

Impiegati

Matricola	Cognome	Filiale	Età
5998	Neri	Milano	45
9553	Bruni	Milano	NULL

- Gli impiegati la cui età è o potrebbe essere maggiore di 40

SEL (Età > 40) OR (Età IS NULL) (Impiegati)

Unione

```
select A,  B
from R
union
select A , B
from S
```

```
select A,  B
from R
union all
select A , B
from S
```

Notazione posizionale!

```
select padre, figlio
from paternita
union
select madre, figlio
from maternita
```

	Figlio
Sergio	Franco
Luigi	Olga
Luigi	Filippo
Franco	Andrea
Franco	Aldo
Luisa	Maria
Luisa	Luigi
Anna	Olga
Anna	Filippo
Maria	Andrea
Maria	Aldo

Padre	Figlio
Sergio	Franco
Luigi	Olga
Luigi	Filippo
Franco	Andrea
Franco	Aldo
Luisa	Maria
Luisa	Luigi
Anna	Olga
Anna	Filippo
Maria	Andrea
Maria	Aldo

Notazione posizionale, 2

```
select padre, figlio
from paternita
union
select figlio, madre
from maternita
```

NO!

```
select padre, figlio
from paternita
union
select madre, figlio
from maternita
```

OK

Notazione posizionale, 3

- Anche con le ridenominazioni non cambia niente:

```
select padre as genitore, figlio
from paternita
union
select figlio, madre as genitore
from maternita
```

- Corretta:

```
select padre as genitore, figlio
from paternita
union
select madre as genitore, figlio
from maternita
```

Differenza

```
select Nome
from Impiegato
except
select Cognome as Nome
from Impiegato
```

Intersezione

```
select Nome
from Impiegato
intersect
select Cognome as Nome
from Impiegato
```

Operatori aggregati: COUNT

- Il numero di figli di Franco

```
select count(*) as NumFigliDiFranco
from Paternita
where Padre = 'Franco'
```

COUNT DISTINCT

```
select count(*) from persone

select count(reddito) from persone

select count(distinct reddito) from persone
```

Altri operatori aggregati

- SUM, AVG, MAX, MIN

- Media dei redditi dei figli di Franco

```
select avg(reddito)
from persone join paternita on nome=figlio
where padre='Franco'
```

Operatori aggregati e valori nulli

```
select avg(reddito) as redditomedio
from persone
```

Operatori aggregati e target list

- un'interrogazione scorretta:

```
select nome, max(reddito)
from persone
```

- di chi sarebbe il nome? La target list deve essere omogenea

```
select min(eta), avg(reddito)
from persone
```

Operatori aggregati e raggruppamenti

- Il numero di figli di ciascun padre

```
select Padre, count(*)  AS NumFigli
from paternita
group by Padre
```

Condizioni sui gruppi

- I padri i cui figli hanno un reddito medio maggiore di 25; mostrare padre e reddito medio dei figli

```
select padre, avg(f.reddito)
from persone f join paternita on figlio = nome
group by padre
having avg(f.reddito) > 25
```

Operatori aggregati e target list

- un'interrogazione scorretta:

```
select nome, max(reddito)
from persone
```

- di chi sarebbe il nome? La target list deve essere omogenea

```
select min(eta), avg(reddito)
from persone
```

Viste

CREATE VIEW AS
  SELECT …

anche (non in tutti i sistemi)

CREATE VIEW AS
  SELECT …
  UNION
  SELECT …

Vedere esempi svolti il 1/11/2016

Interrogazioni nidificate (nested query o subquery)

- Varie forme di nidificazione
  - nella WHERE
  - nella FROM
  - nella SELECT

- Coerente con i tipi
  - anche Booleano (EXISTS)

Nella WHERE

- sulla base di dati (usata negli esercizi del 24/10/2016)

$$Studenti(\underline{Matricola},\ Cognome,\ Nome)$$
$$Esami(\underline{Studente},\underline{Materia},Voto,Data)$$

- L'esame con il voto più basso in assoluto

```
select e.*
from esami e
where e.voto = (select min(voto)
                from esami)
```

Correlated subquery

- Per ogni materia, lo studente che ha preso il voto più basso

```
select e.*
from esami e
where e.voto = (select min(voto)
                from esami
                where materia = e.materia)
```

- L'interrogazione interna viene eseguita una volta per ciascuna ennupla della FROM esterna

Nella FROM

- Per ogni materia, l'esame con il voto più basso

```
select e.*
from esami e join
        (select materia, min(voto) as  votomin
            from esami
            group by materia) m
        on e.materia = m.materia
where e.voto = m.votomin
```

Simile, senza JOIN esplicito

- Per ogni materia, l'esame con il voto più basso

```
select e.*
from esami e join
        (select materia, min(voto) as  votomin
            from esami
            group by materia) m
        on e.materia = m.materia
where e.voto = m.votomin
```

Nella FROM

- Per ogni materia, lo studente con il voto più basso

```
select e.*, s.cognome, s.nome
from (esami e join
        (select materia, min(voto) as  votomin
        from esami
         group by materia) m
        on e.materia = m.materia)
        join studenti s
           on e.studente =  s.matricola
where e.voto = m.votomin
```

Ancora nella FROM

- Tutti gli esami, con il voto più basso nella stessa materia

```
select e.*, m.votomin
from esami e join
                (select materia, min(voto) as votomin
                  from esami group by materia) m
 on e.materia = m.materia
```

- Più complessa (per curiosità, non la scriverei mai così):
  - La materia con il voto medio più alto

```
select materia, votoMedio as votoMedioMax
from (select materia, avg(voto) as votoMedio
        from esami
        group by materia) as votiMedi,
       (select max(votoMedio) as mediaMax
        from (select materia, avg(voto) as votoMedio
                from esami
                group by materia) as votiMedi) as votoMedioMax
where votoMedio = mediaMax
```

Altre nidificazioni nella FROM

- nome e reddito del padre di Franco

```
select Nome, Reddito
from Persone, Paternita
where Nome = Padre and Figlio = 'Franco'

select Nome, Reddito
from Persone
where Nome =      (    select Padre
                       from Paternita
                       where Figlio = 'Franco')
```

- Nome e reddito dei padri di persone che guadagnano più di 20

```
select distinct P.Nome, P.Reddito
from Persone P, Paternita, Persone F
where P.Nome = Padre and Figlio = F.Nome
      and F.Reddito > 20

select Nome, Reddito
from Persone
where Nome in (select Padre
               from Paternita
               where Figlio = any (select Nome
                                   from Persone
                                   where Reddito > 20))
```

notare la distinct

- In questo caso la nidificazione non aiuta molto

- Nome e reddito dei padri di persone che guadagnano più di 20

```
select distinct P.Nome, P.Reddito
from Persone P, Paternita, Persone F
where P.Nome = Padre and Figlio = F.Nome
      and F.Reddito > 20
```

```
select Nome, Reddito
from Persone
where Nome in (select Padre
               from Paternita, Persone
               where Figlio = Nome
               and Reddito > 20)
```

- Nome e reddito dei padri di persone che guadagnano più di 20, con indicazione del reddito del figlio

```
select distinct P.Nome, P.Reddito, F.Reddito
from Persone P, Paternita, Persone F
where P.Nome = Padre and Figlio = F.Nome
      and F.Reddito > 20
```

```
select Nome, Reddito, ????
from Persone
where Nome in (select Padre
               from Paternita
               where Figlio = any (select Nome
                                   from Persone
                                   where Reddito > 20))
```

EXISTS

- Quantificatore esistenziale
- Correlazione fra la sottointerrogazione e le variabili nel resto

- Le persone che hanno almeno un figlio

```
select *
from Persone
where exists (      select *
                    from Paternita
                    where Padre = Nome) or
          exists (  select *
                    from Maternita
                    where Madre = Nome)
```

- I padri i cui figli guadagnano tutti più di 20

```
select distinct Padre
from Paternita Z
where not exists (
                    select *
                    from Paternita W, Persone
                    where W.Padre = Z.Padre
                          and W.Figlio = Nome
                          and Reddito <= 20)
```

- I padri i cui figli guadagnano tutti più di 20

```
select distinct Padre
from Paternita
where not exists (
                    select *
                    from Persone
                    where Figlio = Nome
                                and Reddito <= 20)
```

NO!!! provare ad eseguire per vedere la differenza

- I padri i cui figli guadagnano tutti più di 20

```
select distinct Padre
from Paternita
where not exists (
                    select *
                    from Persone
                    where Reddito <= 20)
```

NO!!! provare anche questa

Nella SELECT

- Calcolo di valori con la nidifcazione
- Per ogni esame tutti i dati e il voto medio in quell'esame (correlazione)

```
select e.*, (select avg(voto)
             from esami
             where materia = e.materia) as votoMedio
from esami e
```

Disgiunzione e unione (ma non sempre)

```
select * from Persone where Reddito > 30
        union
select F.*
from Persone F, Paternita, Persone P
where F.Nome = Figlio and Padre = P.Nome
        and P.Reddito > 30

select *
from Persone F
where Reddito > 30 or
        exists (select *
                from Paternita, Persone P
                where F.Nome = Figlio and Padre = P.Nome
                        and P.Reddito > 30)
```

Differenza e nidificazione

```
select Nome from Impiegato
except
select Cognome as Nome from Impiegato

select Nome
from Impiegato I
where not exists (select *
                  from Impiegato
                  where Cognome = I.Nome)
```

Operazioni di aggiornamento

```
INSERT INTO Persone VALUES ('Mario',25,52)

INSERT INTO Persone(Nome, Reddito, Eta)
       VALUES('Pino',52,23)

INSERT INTO Persone(Nome, Reddito)
       VALUES('Lino',55)

INSERT INTO Persone ( Nome )
SELECT Padre
FROM Paternita
WHERE Padre NOT IN (SELECT Nome
                    FROM Persone)
```

Eliminazione di ennuple

DELETE FROM Tabella
[WHERE Condizione]

```
DELETE FROM Persone
WHERE Eta < 35

DELETE FROM Paternita
WHERE Figlio NOT in (      SELECT Nome
                           FROM Persone)

DELETE FROM Paternita
```

Modifica di ennuple

```
UPDATE NomeTabella
SET  Attributo = < Espressione |
                   SELECT … |
                   NULL |
                   DEFAULT >
        [ WHERE Condizione ]
```

```
UPDATE Persone SET Reddito = 45
  WHERE Nome = 'Piero'

UPDATE Persone
  SET Reddito = Reddito * 1.1
  WHERE Eta < 30
```