Selezione, proiezione e join

- I padri di persone che guadagnano più di 20

$$\text{PROJ}_{\text{Padre}}(\text{paternita} \text{ JOIN }_{\text{Figlio} = \text{Nome}} \text{SEL}_{\text{Reddito}>20}(\text{persone}))$$

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

Un commento

- In algebra relazionale

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

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

Algebra e SQL

- In algebra possiamo scrivere un'intrerrogazione in più modi e ci sono differenze nell'efficienza
  - L'algebra è procedurale
- In SQL, possiamo dire che è il sistema che si preoccupa dell'efficienza
  - SQL è, almeno in parte, "dichiarativo"

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

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

```
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

- Nella clausola FROM:
  - equijoin
    - R1 JOIN R2 ON R1.A = R2.B
  - Equijoin su attributi con lo stesso nome
    - R1 JOIN R2 USING (A)

Join esplicito

- Padre e madre di ogni persona

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

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

select madre, figlio, padre
from maternita join paternita using(figlio)
```

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

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

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

```
select f.nome, f.reddito, p.reddito as RedditoPadre
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 as RedditoPadre
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 figlio, padre, madre
from paternita left join maternita using(figlio)

select paternita.figlio, padre, madre
from paternita full join maternita using(figlio)
```

Note:

- left outer, full outer, right outer equivalenti a left, full, right
- sqliteonline non supporta full e right;

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
```

Attenzione al tipo – guardatelo da soli (ma non è importante ai fini dell'esame)

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

Persone

Nome	Età	Reddito
Andrea	27	21
…	…	…
Luisa	75	87
Nicola	43	NULL

- Le persone il cui reddito è o potrebbe essere maggiore di 40

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

Gestione dei valori nulli

Persone

Nome	Età	Reddito
Andrea	27	21
…	…	…
Luisa	75	87
Nicola	43	NULL

- Le persone il cui reddito è o potrebbe essere maggiore di 40

```
SELECT * FROM Persone
WHERE Reddito > 40 OR Reddito IS null
```

Unione

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

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

Operazione non commutativa (in molti sistemi)

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

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

	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!

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

Notazione posizionale, 2

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

NO!
Funziona, ma produce un risultato indesiderabile

```
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
```

Un'altra anomalia degli operatori inseimistici

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

```
select padre
from paternita
union
select padre
from paternita
```

Un'altra anomalia degli operatori inseimistici

```
select padre
from paternita
union all
select madre
from maternita
```

Operatori aggregati: COUNT

- Il numero di figli di Franco

γ count(*) $\rightarrow$ NumFigliDiFranco (σ Padre = 'Franco' (Paternita))

```
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

$$\gamma\ \text{avg(Reddito)} \rightarrow \text{RedditoMedioFigliDiFranco}$$
$$(\sigma\ \text{Padre = 'Franco'}\ (\text{Paternita}) \bowtie \text{Figlio=Nome Persone})$$

```
select avg(reddito) redditoMedioFigliDiFranco
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 raggruppamenti

- Il numero di figli di ciascun padre

  γ Padre; count(*) → NumFigli (Paternita)

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

- Gli attributi nella target list (Padre) debbono comparire nella GROUP BY
- Purtroppo in alcuni sistemi (come sqliteonlite) questo non accade

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
```

Un errore "classico"

- La persona con il reddito massimo

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

- NO!! Cerchiamo di mettere insieme una ennupla con una aggregazione
- "Le persone con reddito superiore alla media"

Purtroppo

- In alcuni sistemi (es. Sqliteonline) funziona

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

- Ma concettualmente è scorretto: cerca di mettere insieme una ennupla con una aggregazione
- Vediamo una cosa simile:
  - "Le persone con reddito superiore alla media"

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
```