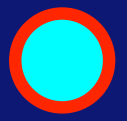
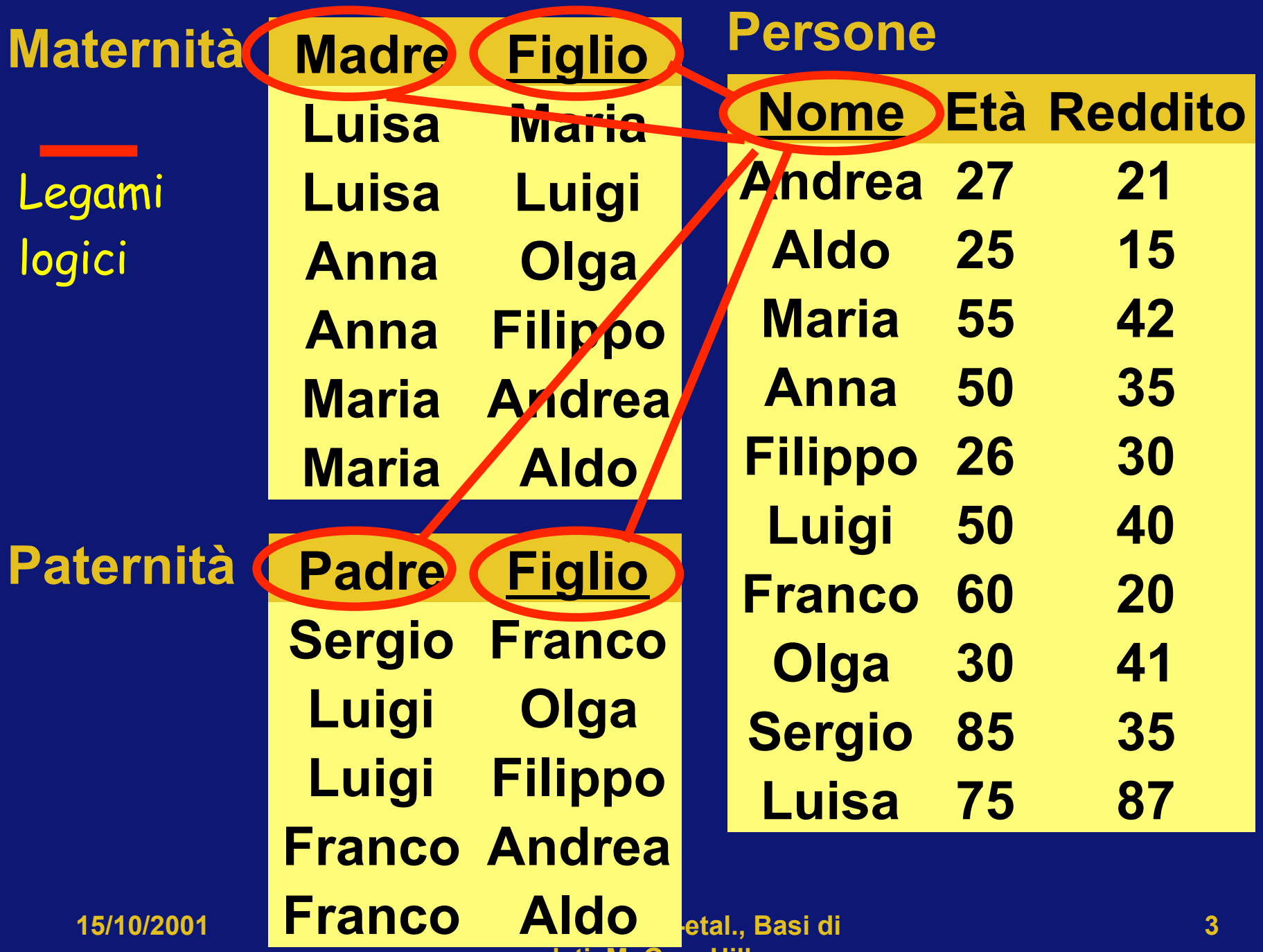
Universita' di Milano Bicocca
Corso di Basi di dati 1 in eLearning
C. Batini
7. SQL DML
7.5 Join

# Nota bene

Anche qui e nel seguito il simbolo 

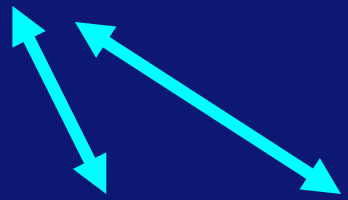
Indica che la trasparenza dovra' essere approfondita a cura dello studente. Se lo studente trova difficolta' puo' chiedere spiegazioni


Maternità
Madre
Figlio
Luisa
Maria
Luisa
Luigi
Anna
Olga
Anna
Filippo
Maria
Andrea
Maria
Aldo
Legami logici
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

# Legami logici nello schema esempio

Maternita' (Madre, Figlio)

Persone (Nome, Citta', Reddito)



Paternita' (Padre, Figlio)

# Join Esplicito

# Join implicito

- Interrogazione: Ogni persona, con madre e padre
- Espressione della interrogazione con Join implicito

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

L'esecuzione della operazione costruisce una relazione in cui, per ogni figlio, padre e madre vengono collegati.

# Join esplicito

- Interrogazione: Ogni persona, con madre e padre
- Espressione della interrogazione con Join esplicito

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

Con questa espressione il JOIN e' esplicitamente dichiarato nella interrogazione, insieme agli attributi coinvolti, e la condizione applicata.

# SELECT con join esplicito, sintassi

SELECT …
FROM Tabella { … JOIN Tabella ON CondizioneDiJoin },
…
[ WHERE AltraCondizione ]

Quindi la condizione di join non compare nella clausola WHERE ed e' spostata nella FROM.

Nella WHERE restano le (eventuali) altre condizioni

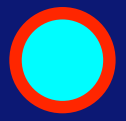
Si ottiene una espressione della interrogazione piu' simile a quella dell'algebra relazionale, e piu' comprensibile.

- 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
```

# Ulteriore estensione: join naturale (meno diffuso)



Algebra – Equijoin

$$PROJ_{Figlio,Padre,Madre}(paternita\ JOIN_{Figlio = Nome}\ REN_{Nome \leftarrow Figlio}(maternita))$$

Algebra – Join naturale
paternita JOIN maternita

SQL – Join esplicito
```
select madre, paternita.figlio, padre
from maternita join paternita on
paternita.figlio = maternita.figlio
```

SQL – Join naturale
```
Select madre, paternita.figlio, padre
from maternita natural join paternita
```

# Join interni ed esterni

- Come nell'algebra:
  - Join interno (default) – non vengono considerate le righe che non si accoppiano
  - E' il theta join dell'algebra
- Join esterno tre modalita':
  - 1. Right outer/  2. Left outer
  - Join interno esteso con le tuple della tabella destra/ sinistra
  - 3. Full outer
  - Join interno esteso con tutte le tuple

# Ricordiamo nell'algebra un esempio di Join esterno sinistro

**Impiegati**

| Impiegato | Reparto |
|---|---|
| Rossi | A |
| Neri | B |
| Bianchi | B |

**Reparti**

| Reparto | Capo |
|---|---|
| B | Mori |
| C | Bruni |

**Impiegati $JOIN_{LEFT}$ Reparti**

| Impiegato | Reparto | Capo |
|---|---|---|
| Neri | B | Mori |
| Bianchi | B | Mori |
| Rossi | A | NULL |

# Join esterno: "outer join"

- Ogni persona con padre (e, se nota, madre)

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

# Outer jo

Persona, padre, (e se nota, madre)

```
select paternita.figlio, pad   , madre
from maternita left out
  on maternita.figlio = pa
```

Persona, madre, (e se noto, padre)

```
select paternita.figlio, pad   , madre
from maternita right outer join paternita
  on maternita.fi
```

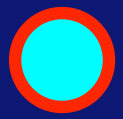
Persona, madre, padre

```
select paternita.figlio,    re, madre
from maternita full outer join paternita
  on maternita.figlio = paternita.figlio
```

# Ordinamento del risultato

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

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



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

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

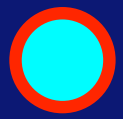
**Persone**

| Nome | Reddito |
|---|---|
| Andrea | 21 |
| Aldo | 15 |
| Filippo | 30 |

**Persone**

| Nome | Reddito |
|---|---|
| Aldo | 15 |
| Andrea | 21 |
| Filippo | 30 |

# Ordinamento su piu' attributi: sintassi e semantica



- Va riportato l'elenco degli attributi *A1, A2, .. An* (Es. Cognome, Nome, Citta')
- Viene fatto sul primo, poi sul secondo, poi su *An*
- Per default l'ordinamento e' ascendente. se lo si vuole discendente occorre usare il qualificatore desc

# Esercizio 7.5.1

Si consideri nuovamente lo schema di base dati:

STUDENTI (Matricola, Cognome, Nome, DataNascita)
ESAMI (Studente, Voto, Corso)
CORSI (Codice, Titolo, Docente)

Scrivere in SQL le seguenti interrogazioni:

1. Trova tutti gli studenti con gli esame superati e i voti.
2. Trova gli studenti che hanno sostenuto esami di corsi con docente "Rossi" e con voto maggiore di 27
3. Trova gli studenti che hanno sostenuto l'esame del corso di Analisi

# Esercizio 7.5.2

- Eseguire l'esercizio 4.7 del testo di riferimento, domande 1-6

# Limiti dei precedenti operatori

Con i precedenti operatori non possiamo esprimere interrogazioni del tipo:

- Trova lo stipendio medio
- Trova l'eta' massima

che calcolano operatori di aggregazione su un insieme di tuple.

Nella prossima lezione tratteremo gli operatori aggregati.

# Concetti introdotti

- Join esplicito
- Join naturale
- Join esterno sinistro
- Join esterno destro
- Full outer join