Select Avanzate

- I risultati di una qualunque query si possono:
  - Limitare in numero (LIMIT)
  - Ordinare (ORDER BY)
  - Eliminare i duplicati (DISTINCT)
  - Raggruppare (GROUP BY)
  - Manipolare (Operatori e Funzioni)

Limitare i Risultati

- **Select *target* FROM *Table_list* [...] LIMIT [*offset*,] *row_count***
- ***Offset*** è la posizione da cui partire (se non presente = 0)
- ***Row_count*** è il numero di righe da visualizzare
- Es.1: Trovare le prime tre righe della tabella PERSONE.
  - SELECT * FROM PERSONE LIMIT 3
- Trovare le prime tre righe della tabella PERSONE a partire dalla quarta.
  - SELECT * FROM PERSONE LIMIT 3,3

Ordinare – Order By

- **SELECT *Target_list* FROM *Table_list* [...] ORDER BY *Campo,Campo2,...* [ASC | DESC]**
- Es. Trovare tutti i figli che guadagnano più di 20 ordinati per Reddito in ordine crescente.
  - SELECT Nome ,Reddito FROM PERSONE WHERE Reddito>20 ORDER BY Reddito [ASC]
- Es. Trovare tutti i figli che guadagnano più di 20 ordinati per Reddito in ordine descrescente.
  - SELECT Nome ,Reddito FROM PERSONE WHERE Reddito>20 ORDER BY Reddito DESC

Eliminare i duplicati - DISTINCT

- SELECT *DISTINCT* target_list FROM Tab_list ...
- Es. Trovare i redditi di tutte le persone.
  - SELECT DISTINCT Reddito FROM PERSONE P
- Funziona su tutta la riga.
- Trovare Età e Reddito di tutte le persone.
  - SELECT DISTINCT Reddito,Eta FROM PERSONE P

Raggruppare – Group By

- SELECT T.Campo1,T.Campo2 FROM Tab1 T WHERE Condizione GROUP BY T.Campo1
- Per ogni valore di categoria viene visualizzata una e una sola riga corrispondente (la prima).

Principali Funzioni Aggregate

- COUNT (* o Campo)
  - Ritorna il numero totale dei risultati
- SUM(Campo)
  - Ritorna la somma totale dei dati contenuti nel campo richiesto
- AVG(Campo)
  - Ritorna la media dei dati contenuti nel campo richiesto
- MAX(Campo),MIN(Campo)
  - Ritorna il valore Massimo/Minimo dei dati contenuti nel campo richiesto

GROUP BY - Esempio

- Trovare per ogni persona il numero dei suoi figli
- SELECT Genitore AS Persona ,Count(Figlio) as NUMFIGLI FROM GENITORI G GROUP BY Genitore

Condizioni sui Gruppi - HAVING

- SELECT Campo1,Campo2... FROM t1 GROUP BY Campo1 ***HAVING*** Condizioni_su_Campo1
- Le condizioni che si possono specificare con la clausola WHERE sono anche molto complesse, ma non possono riferirsi ai risultati delle funzioni aggregate.
- Notare che con HAVING si possono specificare solo condizioni che coinvolgono i campi presenti nella clausola GROUP BY o il risultato di funzioni aggregate.
- Condizioni di altro tipo DEVONO essere specificate nella clausola WHERE.

HAVING - Esempio

- Trovare la media dei redditi:
  - SELECT AVG(Reddito) FROM PERSONE P Group BY Reddito HAVING AVG(Reddito)>20
- Trovare il reddito delle persone che sono genitori di almeno 2 figli.
  - SELECT nome, reddito FROM Persone, Genitori WHERE nome = genitore GROUP BY nome, reddito HAVING COUNT(figlio) >= 2;

Gli Operatori e le Funzioni

- Sono funzioni predefinite che possono essere richiamate all'interno di una interrogazione secondo determinate regole.
- Es. SELECT CONCAT('hello',' ','world')
- Per un riferimento agli operatori si veda il manuale MySQL o il Query Browser in basso a destra.

SELECT - Ricapitoliamo

- SELECT [DISTINCT] target_list
  FROM table_list
  [WHERE cond]
  [GROUP BY campo][HAVING cond]
  [ORDER BY campo [ASC|DESC]]
  [LIMIT offset, rowcount]

Comandi - Join

- Serve a selezionare dati da due o più tabelle.
- E' piu' efficiente di una catena di SELECT.
- Infatti:
  - Senza Join - I risultati della prima query impongono le condizioni sulle seguenti.
  - Con Join – Le condizioni sono applicate direttamente sul prodotto cartesiano.

Join – Theta vs ANSI

- Due stili un significato: Theta e ANSI
- Theta:
  - SELECT * FROM Tab1,Tab2 WHERE cond
- ANSI
  - SELECT * FROM Tab1 JOIN Tab2 ...

Cross Join

- Prodotto Cartesiano tra due o più Tabelle
- Theta Style:
  - SELECT target_list FROM Table1, Table2 ..
- ANSI Style:
  - SELECT target_list FROM Table1
    [*CROSS*] *JOIN* Table2
- NB. *Scambiando i nomi delle tabelle il risultato non cambia, può cambiare solo la disposizione.*

Cross Join - Esempio

- Theta Style:
  - SELECT * FROM PERSONE, GENITORI
- ANSI Style:
  - SELECT * FROM PERSONE *JOIN* GENITORI

Cross Join

Nome	Reddito	Eta	Sesso
Aldo	25	15	M
Amelia	79	28	F
Andrea	27	21	M
Anna	50	29	F
AnnaMaria	41	30	F
AntonGiulio	44	40	M
Beatrice	79	30	F
Ezechiele	11	10	M
Filippo	26	30	M
Franco	60	20	M
Giampa		41	F
Leonardo	79	30	M
Luigi	50	40	M
Luisa	75	87	F
Maria	55	42	F
Michelangelo	79	30	M
Olga	30	41	F
Sergio	85	35	M

Cross Join

Figlio	Genitore
Aldo	Franco
Aldo	Maria
Andrea	Franco
Andrea	Maria
AnnaMaria	Amelia
AnnaMaria	Michelangelo
AntonGiulio	Beatrice
AntonGiulio	Leonardo
Ezechiele	AnnaMaria
Ezechiele	AntonGiulio
Filippo	Anna
Filippo	Luigi
Franco	Sergio
Luigi	Luisa
Maria	Luisa
Olga	Anna
Olga	Luigi

Inner Join

- Prodotto Cartesiano tra due o più Tabelle **ma** filtrato dalle condizioni richieste sui campi.
- Theta Style:
  - SELECT target_list FROM ***Table1,Table2*** WHERE ***Table1.Campo_n = Table2.Campo_m***
- ANSI Style:
  - SELECT target_list FROM Table1 ***JOIN*** Table2 ***ON*** Table1.Campo_n=Table2.Campo_m

Inner Join - Esempio

- Theta Style:
  - SELECT P.Nome,P.Reddito FROM PERSONE P , GENITORI G WHERE P.Nome=G.Genitore
- ANSI Style:
  - SELECT P.Nome,P.Reddito FROM PERSONE P ***INNER JOIN*** GENITORI G ***ON*** P.Nome=G.Genitore

Inner Join

Nome	Reddito	Eta	Sesso
Aldo	25	15	M
Amelia	79	28	F
Andrea	27	21	M
Anna	50	29	F
AnnaMaria	41	30	F
AntonGiulio	44	40	M
Beatrice	79	30	F
Ezechiele	11	10	M
Filippo	26	30	M
Franco	60	20	M
Giampa		41	F
Leonardo	79	30	M
Luigi	50	40	M
Luisa	75	87	F
Maria	55	42	F
Michelangelo	79	30	M
Olga	30	41	F
Sergio	85	35	M

Inner Join

X

Figlio	Genitore
Aldo	Franco
Aldo	Maria
Andrea	Franco
Andrea	Maria
AnnaMaria	Amelia
AnnaMaria	Michelangelo
AntonGiulio	Beatrice
AntonGiulio	Leonardo
Ezechiele	AnnaMaria
Ezechiele	AntonGiulio
Filippo	Anna
Filippo	Luigi
Franco	Sergio
Luigi	Luisa
Maria	Luisa
Olga	Anna
Olga	Luigi

Inner Join – Dialetto MySQL: USING

- SELECT ... FROM T1 JOIN T2 USING (campo1, campo2,...)
- I nomi dei campi devono essere in comune tra le tabelle.
- Se non sono in comune non ci può essere un risultato.

Inner Join - MySQL USING

- Es. Trovare il reddito di tutte le persone che sono genitori. (Alteriamo il nome di un Campo)
- Rinominiamo con Query Browser il Campo Figlio della Tabella GENITORI.
- SELECT P.Nome,P.Reddito FROM PERSONE P JOIN GENITORI G ***USING*** *(Nome)*

Inner Join e Condizioni

- Mysql, senza la Clausola ON o USING, tratta la query come una CROSS.
  - SELECT COUNT(*) FROM PERSONE P ***JOIN*** GENITORI G
  - SELECT COUNT(*) FROM PERSONE P ***CROSS JOIN*** GENITORI G
  - SELECT COUNT(*) FROM PERSONE P ***INNER JOIN*** GENITORI G
- Sono identiche.

Outer Join – Left e Right

- Ritorna *tutti* i record in una tabella (la prima se il JOIN è LEFT la seconda se RIGHT), anche quelli che non soddisfano le clausole WHERE.

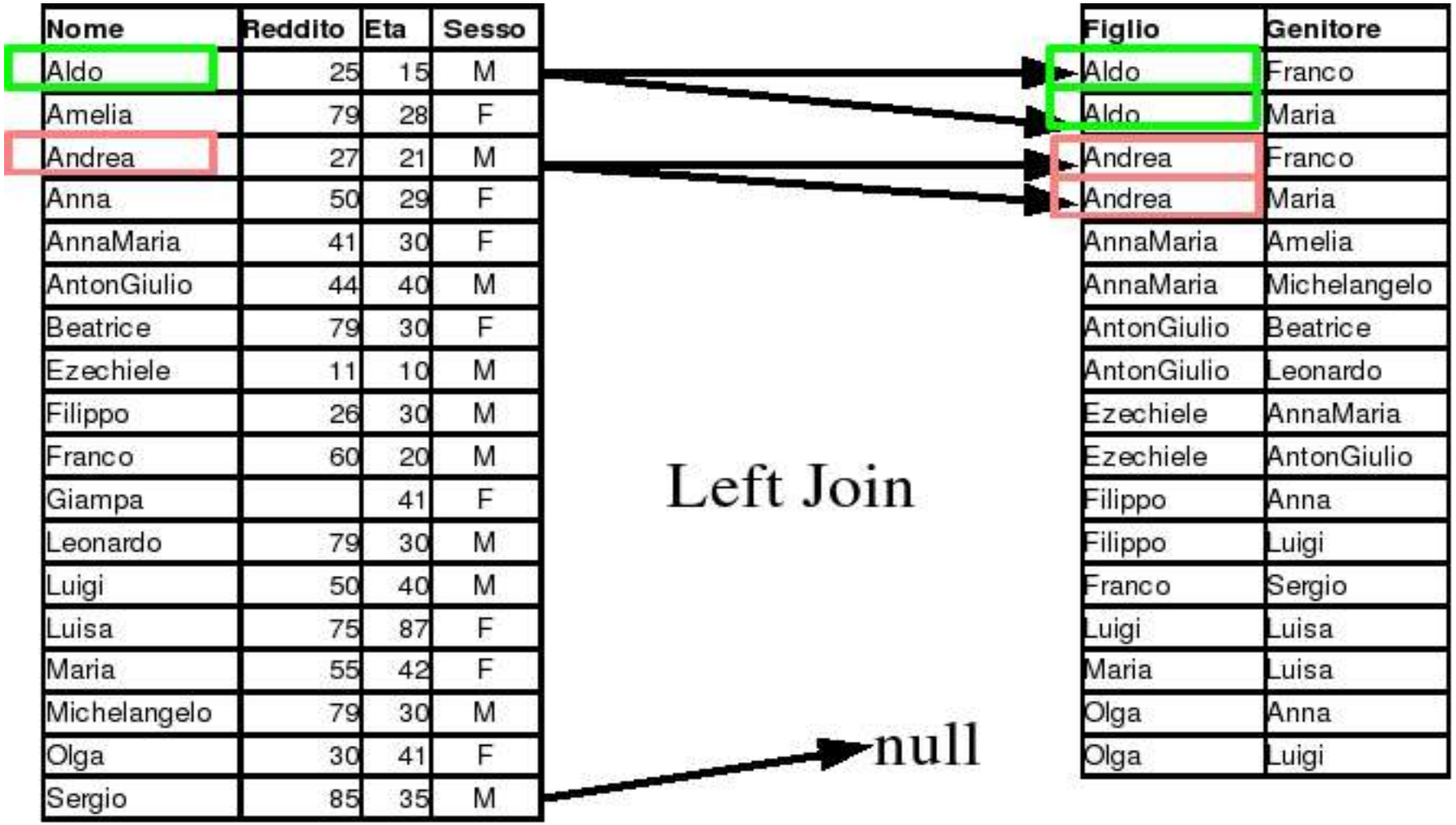
Left (Outer) Join

- MySQL supporta solo il costrutto ANSI Style
- SELECT Target_List FROM T1 *LEFT [OUTER] JOIN* T2 ON Condizioni
- Oracle invece supporta anche il costrutto Theta Style.

Left Join - Esempio

- Trovare tutte le persone e metterle in relazione con i loro genitori.
  - SELECT Nome, Reddito, Genitore FROM PERSONE LEFT OUTER JOIN GENITORI ON Nome=Figlio ORDER BY Genitore
- Miglioriamo la lettura dei risultati:
  - SELECT Nome, Reddito, IFNULL(Genitore,'[Figlio di NN]') as Genitore FROM PERSONE LEFT OUTER JOIN GENITORI ON Nome=Figlio ORDER BY Genitore

Left (Outer) Join



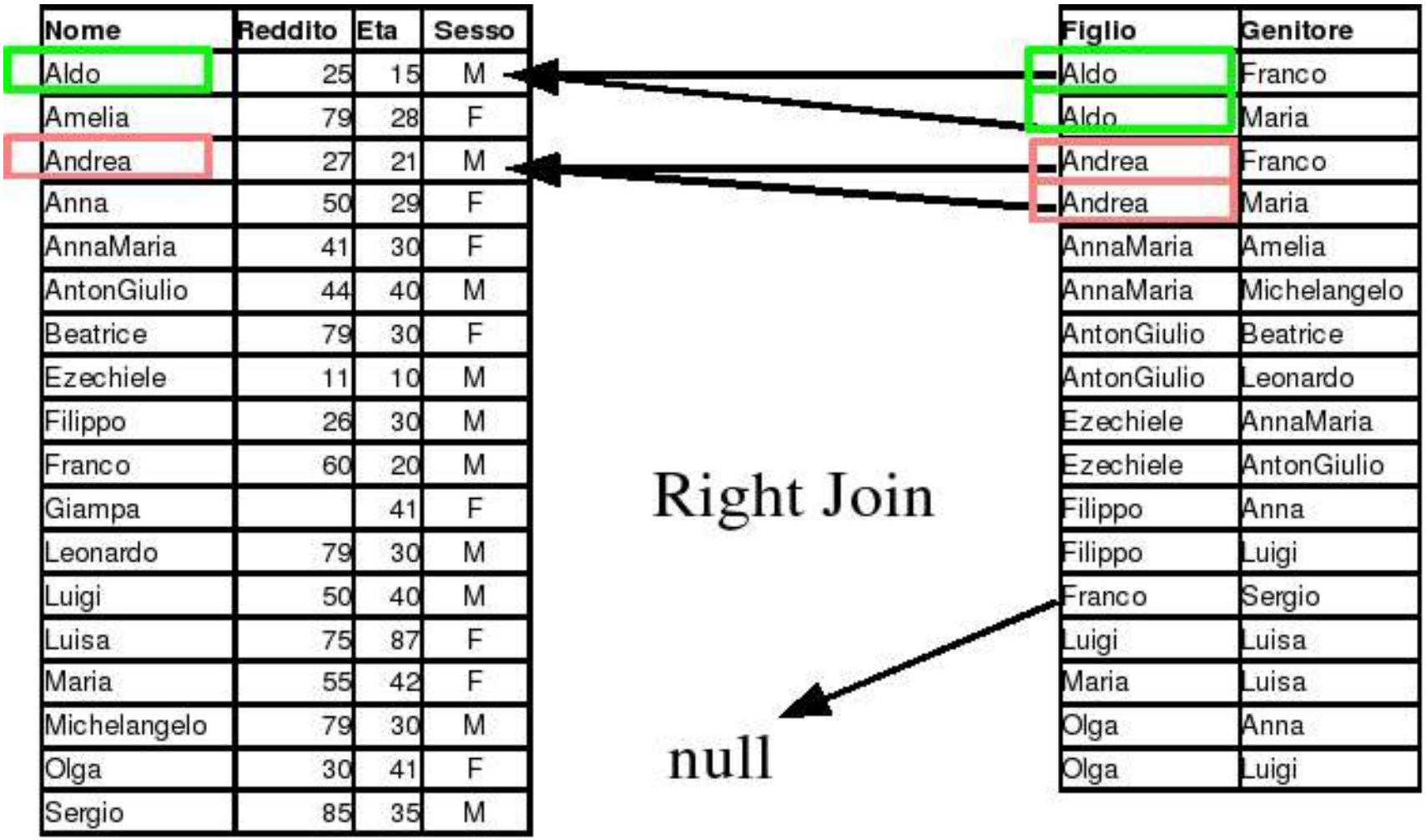
Right (Outer) Join

- MySQL supporta solo il costrutto ANSI Style
- SELECT Target_List FROM T1 *RIGHT [OUTER] JOIN* T2 ON Condizioni
- Oracle invece supporta anche il costrutto Theta Style.

Right Join - Esempio

- SELECT Nome, Reddito, Genitore FROM PERSONE P RIGHT OUTER JOIN GENITORI G ON (Nome=Figlio and Figlio like 'A%') ORDER BY Genitore
- Trovare tutte le persone e metterle in relazione con i loro genitori.
  - SELECT Nome, Reddito, Genitore FROM GENITORI RIGHT OUTER JOIN PERSONE ON Nome=Figlio ORDER BY Genitore
- Che differenza c'è?

Right Join


Nome
Reddito
Eta
Sesso
Aldo 25 15 M
Amelia 79 28 F
Andrea 27 21 M
Anna 50 29 F
AnnaMaria 41 30 F
AntonGiulio 44 40 M
Beatrice 79 30 F
Ezechiele 11 10 M
Filippo 26 30 M
Franco 60 20 M
Giampa 41 F
Leonardo 79 30 M
Luigi 50 40 M
Luisa 75 87 F
Maria 55 42 F
Michelangelo 79 30 M
Olga 30 41 F
Sergio 85 35 M
Figlio
Genitore
Aldo Franco
Aldo Maria
Andrea Franco
Andrea Maria
AnnaMaria Amelia
AnnaMaria Michelangelo
AntonGiulio Beatrice
AntonGiulio Leonardo
Ezechiele AnnaMaria
Ezechiele AntonGiulio
Filippo Anna
Filippo Luigi
Franco Sergio
Luigi Luisa
Maria Luisa
Olga Anna
Olga Luigi
Right Join
null

Left o Right Join?

- Scambio di tabelle
- Cosa usare?
  - Nella Maggioranza dei casi non importa.
  - Per ragioni di consistenza e' consigliabile usare solo una delle due.
  - Per ragioni di praticita' la maggior parte delle persone considerano la Left Join piu' semplice da 'visualizzare' nella crazione di queries.

Natural Join

- E' un modo sintetico per fare una (qualunque) JOIN su tutti i campi in comune.

Full (Outer) Join

- Non Supportata da MySQL.
- Vedremo come simulare una Full Join con il costrutto:

  SELECT... UNION SELECT ...