Linguaggio SQL

Linguaggio SQL: fondamenti

- Linguaggio SQL
- Istruzione del linguaggio
- Notazione e base di dati di esempio
- Istruzione SELECT
- Funzioni aggregate
- Operatore GROUP BY

Il linguaggio SQL

- Linguaggio per gestire le basi di dati relazionali
  - Structured Query Language
- SQL include istruzioni per
  - definire lo schema di una base di dati relazionale
  - leggere e scrivere i dati
  - definire lo schema di tabelle derivate
  - definire i privilegi di accesso degli utenti
  - gestire le transazioni
- Il linguaggio è utilizzabile in modalità
  - interattiva
  - compilata
    - un linguaggio ospite (host) contiene le istruzioni SQL
    - le istruzioni SQL si distinguono dalle istruzioni del linguaggio ospite per mezzo di opportuni artifici sintattici

Il linguaggio SQL

- Il linguaggio SQL è un linguaggio a livello di set
  - gli operatori operano su relazioni
  - il risultato è sempre una relazione
- Il linguaggio SQL è *dichiarativo*
  - descrive *cosa fare* e non come fare
  - si pone ad un livello di astrazione superiore rispetto ai linguaggi di programmazione tradizionali

Istruzioni del linguaggio SQL

Linguaggio SQL

Il linguaggio SQL

- Può essere diviso in
  - DML (Data Manipulation Language)
    - linguaggio di manipolazione dei dati
  - DDL (Data Definition Language)
    - linguaggio di definizione della struttura della base di dati

Data Manipulation Language

- Interrogazione di una base dati per estrarre i dati di interesse
  - SELECT
- Modifica dell'istanza di una base dati
  - INSERT: inserimento di nuove informazioni in una tabella
  - UPDATE: aggiornamento di dati presenti nella base dati
  - DELETE: cancellazione di dati obsoleti

Data Definition Language

- Definizione dello schema di una base di dati
  - creazione, modifica e cancellazione di tabelle: CREATE, ALTER, DROP TABLE
- Definizione di tabelle derivate
  - creazione, modifica e cancellazione di tabelle il cui contenuto è ottenuto da altre tabelle della base dati: CREATE, ALTER, DROP VIEW
- Definizione di strutture dati accessorie per recuperare efficientemente i dati
  - creazione e cancellazione di indici: CREATE, DROP INDEX
- Definizione dei privilegi di accesso degli utenti
  - concessione e revoca di privilegi sulle risorse: GRANT, REVOKE
- Definizione di transazioni
  - terminazione di una transazione: COMMIT, ROLLBACK

Notazione e base di dati di esempio

Linguaggio SQL

Sintassi

- Notazione
  - parole chiave del linguaggio
    - caratteri maiuscoli e colore rosso scuro
  - termini variabili
    - corsivo
- Grammatica
  - parentesi angolari < >
    - isolano un termine della sintassi
  - parentesi quadre [ ]
    - indicano che il termine all'interno è opzionale
  - parentesi graffe { }
    - indicano che il termine racchiuso può non comparire o essere ripetuto un numero arbitrario di volte
  - barra verticale |
    - indica che deve essere scelto uno tra i termini separati dalle barre

Base dati di esempio: Forniture-Prodotti

P

CodP	NomeP	Colore	Taglia	Magazzino
P1	Maglia	Rosso	40	Torino
P2	Jeans	Verde	48	Milano
P3	Camicia	Blu	48	Roma
P4	Camicia	Blu	44	Torino
P5	Gonna	Blu	40	Milano

Chiave esterna

Chiave esterna

FP

CodF	CodP	Qta
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F1	P5	100
F2	P1	300
F2	P2	400
F3	P2	200
F4	P3	200

F

CodF	NomeF	NSoci	Sede
F1	Andrea	2	Torino
F2	Luca	1	Milano
F3	Antonio	3	Milano
F4	Gabriele	2	Torino
F5	Matteo	3	Venezia

Base dati di esempio: Forniture-Prodotti

- Base dati forniture prodotti
  - tabella P: descrive i prodotti disponibili
    - chiave primaria: CodP
  - tabella F: descrive i fornitori
    - chiave primaria: CodF
  - tabella FP: descrive le forniture, mettendo in relazione i prodotti con i fornitori che li forniscono
    - chiave primaria: (CodF, CodP)
    - CodF: chiave esterna. CodF (FP) REFERENCES CodF(F)
    - CodP: Chiave esterna. CodP (FP) REFERENCES CodP(P)

Istruzione SELECT

Linguaggio SQL

SELECT

SELECT [DISTINCT] *ElencoAttributiDaVisualizzare*

FROM *ElencoTabelleDaUtilizzare*

[WHERE *CondizioniDiTupla*]

[GROUP BY *ElencoAttributiDiRaggruppamento*]

[HAVING *CondizioniSuAggregati*]

[ORDER BY *ElencoAttributiDiOrdinamento*]

Istruzione SELECT (n.1)

- Trovare il codice e il numero di soci dei fornitori di Milano

```
SELECT CodF, NSoci
FROM   F
WHERE Sede='Milano';
```

$$R = \pi_{CodF,\ NSoci}\left(\sigma_{Sede='Milano'}(F)\right)$$

F

CodF	NomeF	NSoci	Sede
F1	Andrea	2	Torino
F2	Luca	1	Milano
F3	Antonio	3	Milano
F4	Gabriele	2	Torino
F5	Matteo	3	Venezia

R

CodF	NSoci
F2	1
F3	3

Istruzione SELECT (n.2)

- Trovare il codice di tutti i prodotti

```
SELECT CodP
FROM P;
```

$$R = \pi_{CodP}(P)$$

P

CodP	NomeP	Colore	Taglia	Magazzino
P1	Maglia	Rosso	40	Torino
P2	Jeans	Verde	48	Milano
P3	Camicia	Blu	48	Roma
P4	Camicia	Blu	44	Torino
P5	Gonna	Blu	40	Milano
P6	Bermuda	Rosso	42	Torino

R

CodP
P1
P2
P3
P4
P5
P6

Istruzione SELECT (n.3)

- Trovare il codice di tutti i prodotti

FP

CodF	CodP	Qta
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F1	P5	100
F1	P6	100
F2	P1	300
F2	P2	400
F3	P2	200
F4	P3	200
F4	P4	300
F4	P5	400

```
SELECT CodP
FROM FP;
```

R

CodP
P1
P2
P3
P4
P5
P6
P1
P2
P2
P3
P4
P5

Istruzione SELECT (n.3)

- Trovare il codice dei prodotti forniti da almeno un fornitore

SELECT CodP
FROM FP;

$$\neq$$

$$R = \pi_{CodP}(FP)$$

- Non effettua la rimozione dei duplicati

Eliminazione dei duplicati: DISTINCT

- Parola chiave DISTINCT permette l'eliminazione dei duplicati

- Trovare il codice dei prodotti *diversi* forniti da almeno un fornitore

```
SELECT DISTINCT CodP
FROM FP;
```

FP

CodF	CodP	Qta
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F1	P5	100
F1	P6	100
F2	P1	300
F2	P2	400
F3	P2	200
F4	P3	200
F4	P4	300
F4	P5	400

R

CodP
P1
P2
P3
P4
P5
P6

Selezione di tutte le informazioni

- Trovare *tutte* le informazioni sui prodotti

```
SELECT CodP, NomeP, Colore, Taglia, Magazzino
FROM P;
```

oppure

```
SELECT *
FROM P;
```

R

CodP	NomeP	Colore	Taglia	Magazzino
P1	Maglia	Rosso	40	Torino
P2	Jeans	Verde	48	Milano
P3	Camicia	Blu	48	Roma
P4	Camicia	Blu	44	Torino
P5	Gonna	Blu	40	Milano
P6	Bermuda	Rosso	42	Torino

Selezione con espressione

- Trovare il codice dei prodotti e la taglia espressa con la misura americana

```
SELECT CodP, Taglia-14 [AS TagliaUSA]
FROM P;
```

P

CodP	NomeP	Colore	Taglia	Magazzino
P1	Maglia	Rosso	40	Torino
P2	Jeans	Verde	48	Milano
P3	Camicia	Blu	48	Roma
P4	Camicia	Blu	44	Torino
P5	Gonna	Blu	40	Milano
P6	Bermuda	Rosso	42	Torino

R

CodP	TagliaUSA
P1	26
P2	34
P3	34
P4	30
P5	26
P6	38

- Definizione di una nuova colonna *temporanea* per l'espressione calcolata
  - il nome della colonna temporanea può essere definito con la parola chiave AS

Clausola WHERE

- Permette di esprimere condizioni di selezione espresse singolarmente ad ogni tupla
- Espressione booleana di predicati
- Predicati semplici
  - espressioni di confronto tra attributi e costanti
  - ricerca testuale
  - valori NULL

Clausola WHERE (n.1)

- Trovare il codice dei fornitori di Milano

```
SELECT CodF
FROM F
WHERE Sede='Milano';
```

F

CodF	NomeF	NSoci	Sede
F1	Andrea	2	Torino
F2	Luca	1	Milano
F3	Antonio	3	Milano
F4	Gabriele	2	Torino
F5	Matteo	3	Venezia

R

CodF
F2
F3

Clausola WHERE (n.2)

- Trovare il codice e il numero di soci dei fornitori che non hanno sede a Milano

```
SELECT CodF, NSoci
FROM F
WHERE Sede<>'Milano';
```

F

CodF	NomeF	NSoci	Sede
F1	Andrea	2	Torino
F2	Luca	1	Milano
F3	Antonio	3	Milano
F4	Gabriele	2	Torino
F5	Matteo	3	Venezia

R

CodF	NSoci
F1	2
F4	2
F5	3

Espressioni booleane (n.1)

- Trovare il codice dei fornitori di Milano con più di 2 soci

```
SELECT CodF
FROM F
WHERE Sede='Milano' AND NSoci>2;
```

F

CodF	NomeF	NSoci	Sede
F1	Andrea	2	Torino
F2	Luca	1	Milano
F3	Antonio	3	Milano
F4	Gabriele	2	Torino
F5	Matteo	3	Venezia

R

CodF
F3

Espressioni booleane (n.2)

- Trovare il codice e il numero di soci dei fornitori di Milano e di Torino

```
SELECT CodF, NSoci
FROM F
WHERE Sede='Milano' OR Sede='Torino';
```

F

CodF	NomeF	NSoci	Sede
F1	Andrea	2	Torino
F2	Luca	1	Milano
F3	Antonio	3	Milano
F4	Gabriele	2	Torino
F5	Matteo	3	Venezia

R

CodF	NSoci
F1	2
F2	1
F3	3
F4	2

Espressioni booleane (n.3)

- Trovare il codice e il numero di soci dei fornitori che hanno sede a Milano e a Torino
  - la richiesta non può essere soddisfatta
    - ogni fornitore ha una sola sede

F

CodF	NomeF	NSoci	Sede
F1	Andrea	2	Torino
F2	Luca	1	Milano
F3	Antonio	3	Milano
F4	Gabriele	2	Torino
F5	Matteo	3	Venezia

Ricerca testuale

- Operatore LIKE

  NomeAttributo LIKE *StringaDiCaratteri*

  - il carattere _ rappresenta un singolo carattere qualsiasi (obbligatoriamente presente)
  - il carattere % rappresenta una sequenza qualsiasi di n caratteri (anche vuota)

Ricerca testuale (n.1)

- Trovare il codice e il nome dei prodotti il cui nome inizia con la lettera C

```
SELECT CodP, NomeP
FROM P
WHERE NomeP LIKE 'C%';
```

P

CodP	NomeP	Colore	Taglia	Magazzino
P1	Maglia	Rosso	40	Torino
P2	Jeans	Verde	48	Milano
P3	**C**amicia	Blu	48	Roma
P4	**C**amicia	Blu	44	Torino
P5	Gonna	Blu	40	Milano
P6	Bermuda	Rosso	42	Torino

R

CodP	NomeP
P3	Camicia
P4	Camicia

Ricerca testuale (n.2)

- L'attributo Indirizzo contiene la stringa 'Torino'

Indirizzo LIKE '%Torino%'

- Il codice fornitore è pari a 2 e
  - è preceduto da un carattere ignoto
  - è costituito esattamente da 2 caratteri

CodF LIKE '_2'

- L'attributo magazzino non contiene una 'e' in seconda posizione

Magazzino NOT LIKE '_e%'

Ricerca di valori NULL

- Operatore speciale IS

  NomeAttributo IS [NOT] NULL

- In presenza di valori NULL qualsiasi predicato di confronto è falso

Gestione di valori NULL

- Trovare il codice e il nome dei prodotti con taglia maggiore di 44

```
SELECT CodP, NomeP
FROM P
WHERE Taglia>44;
```

P

CodP	NomeP	Colore	Taglia	Magazzino
P1	Maglia	Rosso	40	Torino
P2	Jeans	Verde	48	Milano
P3	Camicia	Blu	48	Roma
P4	Camicia	Blu	44	Torino
P5	Gonna	Blu	NULL	Milano
P6	Bermuda	Rosso	42	Torino

R

CodP	NomeP
P2	Jeans
P3	Camicia

- Le tuple per cui la taglia è NULL non sono selezionate: il predicato Taglia>44 è falso
- In presenza di valori NULL qualsiasi predicato di confronto è falso

Ricerca di valori NULL (n.1)

- Trovare il codice e il nome dei prodotti per cui la taglia non è indicata

```
SELECT CodP, NomeP
FROM P
WHERE Taglia IS NULL;
```

P

CodP	NomeP	Colore	Taglia	Magazzino
P1	Maglia	Rosso	40	Torino
P2	Jeans	Verde	48	Milano
P3	Camicia	Blu	48	Roma
P4	Camicia	Blu	44	Torino
P5	Gonna	Blu	NULL	Milano
P6	Bermuda	Rosso	42	Torino

R

CodP	NomeP
P5	Gonna

Ricerca di valori NULL (n.2)

- Trovare il codice e il nome dei prodotti con la taglia maggiore di 44 o che potrebbero avere taglia maggiore di 44

```
SELECT CodP, NomeP
FROM P
WHERE Taglia>44 OR Taglia IS NULL;
```

P

CodP	NomeP	Colore	Taglia	Magazzino
P1	Maglia	Rosso	40	Torino
P2	Jeans	Verde	48	Milano
P3	Camicia	Blu	48	Roma
P4	Camicia	Blu	44	Torino
P5	Gonna	Blu	NULL	Milano
P6	Bermuda	Rosso	42	Torino

R

CodP	NomeP
P2	Jeans
P3	Camicia
P5	Gonna

Ordinamento risultato

- Clausola ORDER BY

  ORDER BY *NomeAttributo* [ASC | DESC]
  {, *NomeAttributo* [ASC | DESC] }

- L'ordinamento può essere crescente (ASC) o decrescente (DESC)
  - L'ordinamento implicito è crescente (ASC)
- Gli attributi di ordinamento devono comparire nella clausola SELECT
  - anche implicitamente (come SELECT *)

Ordinamento del risultato (n.1)

- Trovare il codice dei prodotti e la loro taglia ordinando il risultato in ordine decrescente di taglia

```
SELECT CodP, Taglia
FROM P
ORDER BY Taglia DESC;
```

P

CodP	NomeP	Colore	Taglia	Magazzino
P1	Maglia	Rosso	40	Torino
P2	Jeans	Verde	48	Milano
P3	Camicia	Blu	48	Roma
P4	Camicia	Blu	44	Torino
P5	Gonna	Blu	40	Milano
P6	Bermuda	Rosso	42	Torino

R

CodP	Taglia
P2	48
P3	48
P4	44
P6	42
P1	40
P5	40

Ordinamento del risultato (n.2)

- Trovare tutte le informazioni sui prodotti ordinando il risultato in ordine crescente di nome e decrescente di taglia

```
SELECT CodP, NomeP, Colore, Taglia, Magazzino
FROM P
ORDER BY NomeP, Taglia DESC;
```

```
SELECT *
FROM P
ORDER BY NomeP, Taglia DESC;
```

R

CodP	NomeP	Colore	Taglia	Magazzino
P6	Bermuda	Rosso	42	Torino
P3	Camicia	Blu	48	Roma
P4	Camicia	Blu	44	Torino
P5	Gonna	Blu	40	Milano
P2	Jeans	Verde	48	Milano
P1	Maglia	Rosso	40	Torino

Ordinamento del risultato (n.3)

- Trovare il codice dei prodotti e la taglia espressa come taglia americana, ordinando il risultato in ordine crescente di taglia

```
SELECT CodP, Taglia-14 AS TagliaUSA
FROM P
ORDER BY TagliaUSA;
```

P

CodP	NomeP	Colore	Taglia	Magazzino
P1	Maglia	Rosso	40	Torino
P2	Jeans	Verde	48	Milano
P3	Camicia	Blu	48	Roma
P4	Camicia	Blu	44	Torino
P5	Gonna	Blu	40	Milano
P6	Bermuda	Rosso	42	Torino

R

CodP	TagliaUSA
P5	26
P1	28
P6	28
P4	30
P2	34
P3	34

Join

- Definito mediante le clausole FROM e WHERE
- Il risultato e l'efficienza dell'interrogazione
  - sono indipendenti dall'ordine delle tabelle nella clausola FROM
  - sono indipendenti dall'ordine dei predicati nella clausola WHERE
  - l'ordine di esecuzione ottimale è selezionato dal DBMS (modulo ottimizzatore)
- Clausola FROM con N tabelle
  - almeno N-1 condizioni di join nella clausola WHERE

Join (n.1)

- Trovare il nome dei fornitori che forniscono il prodotto P2

CodF	NomeF	NSoci	Sede
F1	Andrea	2	Torino
F2	Luca	1	Milano
F3	Antonio	3	Milano
F4	Gabriele	2	Torino
F5	Matteo	3	Venezia

CodF	CodP	Qta
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F1	P5	100
F2	P1	300
F2	P2	400
F3	P2	200
F4	P3	200

Prodotto cartesiano

- Trovare il nome dei fornitori che forniscono il prodotto P2

```
SELECT NomeF
FROM F, FP ;
```

Prodotto cartesiano

F.CodF	F.NomeF	F.NSoci	F.Sede	FP.CodF	FP.CodP	FP.Qta
F1	Andrea	2	Torino	F1	P1	300
F1	Andrea	2	Torino	F1	P2	200
F1	Andrea	2	Torino	F1	P3	400
F1	Andrea	2	Torino	F1	P4	200
F1	Andrea	2	Torino	F1	P5	100
F1	Andrea	2	Torino	F1	P6	100
F1	Andrea	2	Torino	F2	P1	300
…	…	…	…	…	…	…
F2	Luca	1	Milano	F1	P1	300
…	…	…	…	…	…	…
F2	Luca	1	Milano	F2	P1	300
…	…	…	…	…	…	…

Join (n.1)

=

F.CodF	F.NomeF	F.NSoci	F.Sede	FP.CodF	FP.CodP	FP.Qta
F1	*Andrea*	*2*	*Torino*	*F1*	*P1*	*300*
F1	*Andrea*	*2*	*Torino*	*F1*	*P2*	*200*
F1	*Andrea*	*2*	*Torino*	*F1*	*P3*	*400*
F1	*Andrea*	*2*	*Torino*	*F1*	*P4*	*200*
F1	*Andrea*	*2*	*Torino*	*F1*	*P5*	*100*
F1	*Andrea*	*2*	*Torino*	*F1*	*P6*	*100*
F1	Andrea	2	Torino	F2	P1	300
…	…	…	…	…	…	…
F2	Luca	1	Milano	F1	P1	300
…	…	…	…	…	…	…
F2	*Luca*	*1*	*Milano*	*F2*	*P1*	*300*
…	…	…	…	…	…	…

Join (n.1)

=

F.CodF	F.NomeF	F.NSoci	F.Sede	FP.CodF	FP.CodP	FP.Qta
F1	Andrea	2	Torino	F1	P1	300
F1	Andrea	2	Torino	F1	P2	200
F1	Andrea	2	Torino	F1	P3	400
F1	Andrea	2	Torino	F1	P4	200
F1	Andrea	2	Torino	F1	P5	100
F1	Andrea	2	Torino	F1	P6	100
F2	Luca	1	Milano	F2	P1	300
F2	Luca	1	Milano	F2	P2	400
F3	Antonio	3	Milano	F3	P2	200
F4	Gabriele	2	Torino	F4	P3	200
F4	Gabriele	2	Torino	F4	P4	300
F4	Gabriele	2	Torino	F4	P5	400

Join (n.1)

- Trovare il nome dei fornitori che forniscono il prodotto P2

```
SELECT NomeF
FROM F, FP
WHERE F.CodF=FP.CodF ;
```

Condizione di join

NomeTabella.NomeAttributo

Join (n.1)

- Trovare il nome dei fornitori che forniscono il prodotto P2

```
SELECT NomeF
FROM F, FP
WHERE F.CodF=FP.CodF AND CodP='P2';
```

Condizione di join

NomeTabella.NomeAttributo

Join (n.1)

FP.CodP='P2'

=

F.CodF	F.NomeF	F.NSoci	F.Sede	FP.CodF	FP.CodP	FP.Qta
F1	Andrea	2	Torino	F1	P1	300
F1	Andrea	2	Torino	F1	P2	200
F1	Andrea	2	Torino	F1	P3	400
F1	Andrea	2	Torino	F1	P4	200
F1	Andrea	2	Torino	F1	P5	100
F1	Andrea	2	Torino	F1	P6	100
F2	Luca	1	Milano	F2	P1	300
F2	Luca	1	Milano	F2	P2	400
F3	Antonio	3	Milano	F3	P2	200
F4	Gabriele	2	Torino	F4	P3	200
F4	Gabriele	2	Torino	F4	P4	300
F4	Gabriele	2	Torino	F4	P5	400

Join (n.1)

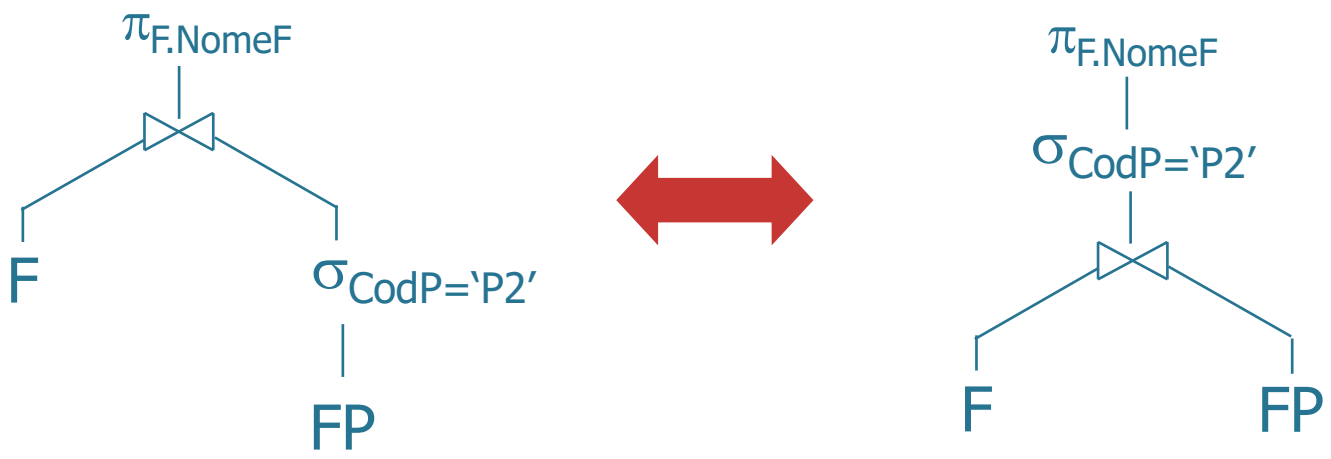
F.CodF	F.NomeF	F.NSoci	F.Sede	FP.CodF	FP.CodP	FP.Qta
F1	Andrea	2	Torino	F1	P2	200
F2	Luca	1	Milano	F2	P2	400
F3	Antonio	3	Milano	F3	P2	200

R

NomeF
Andrea
Luca
Antonio

Join (n.1)

- Trovare il nome dei fornitori che forniscono il prodotto P2
  - in algebra relazionale



Join (n.1)

- Trovare il nome dei fornitori che forniscono il prodotto P2
  - in algebra relazionale

```
SELECT NomeF
FROM F, FP
WHERE F.CodF=FP.CodF
        AND CodP='P2';
```

⟷

```
SELECT NomeF
FROM FP,F
WHERE CodP='P2' AND
        F.CodF=FP.CodF;
```

- Il risultato e l'efficienza sono indipendenti
  - dall'ordine dei predicati nella clausola WHERE
  - dall'ordine delle tabelle nella clausola FROM

Dichiaratività del linguaggio SQL

- In algebra relazionale (linguaggio procedurale) si definisce l'ordine in cui sono applicati gli operatori
- In SQL (linguaggio dichiarativi) l'ordine migliore è scelto dall'ottimizzatore indipendentemente
    - dall'ordine delle condizioni nella clausola WHERE
    - dall'ordine delle tabelle nella clausola FROM

Join (n.2)

- Trovare il nome dei fornitori che forniscono almeno un prodotto rosso

```
SELECT NomeF
FROM F, FP, P
WHERE  F.CodF=FP.CodF AND P.CodP=FP.CodP
          AND Colore='Rosso';
```

- Clausola FROM con N tabelle
  - almeno N-1 condizioni di join nella clausola WHERE

Join (n.2)

- Trovare le coppie di codici dei fornitori tali che entrambi i fornitori abbiano sede nella stessa città

```
SELECT FX.CodF, FY.CodF
FROM F AS FX, F AS FY
WHERE  FX.Sede=FY.Sede;
```

F AS FX

CodF	NomeF	NSoci	Sede
F1	Andrea	2	Torino
F2	Luca	1	Milano
F3	Antonio	3	Milano
F4	Gabriele	2	Torino
F5	Matteo	3	Venezia

F AS FY

CodF	NomeF	NSoci	Sede
F1	Andrea	2	Torino
F2	Luca	1	Milano
F3	Antonio	3	Milano
F4	Gabriele	2	Torino
F5	Matteo	3	Venezia

Join (n.2)

- Trovare le coppie di codici dei fornitori tali che entrambi i fornitori abbiano sede nella stessa città

```
SELECT FX.CodF, FY.CodF
FROM F AS FX, F AS FY
WHERE  FX.Sede=FY.Sede;
```

R

FX.CodF	FY.CodF
F1	F1
F1	F4
F2	F2
F2	F3
F3	F2
F3	F3
F4	F1
F4	F4
F5	F5

- Sono presenti
  - coppie di valori uguali
  - permutazioni della stessa coppia di valori

Join (n.2)

- Trovare le coppie di codici dei fornitori tali che entrambi i fornitori abbiano sede nella stessa città

```
SELECT FX.CodF, FY.CodF
FROM F AS FX, F AS FY
WHERE  FX.Sede=FY.Sede  AND
       FX.CodF <> FY.CodF;
```

R

FX.CodF	FY.CodF
~~F1~~	~~F1~~
F1	F4
~~F2~~	~~F2~~
F2	F3
F3	F2
~~F3~~	~~F3~~
F4	F1
~~F4~~	~~F4~~
~~F5~~	~~F5~~

- Elimina le coppie di valori uguali

Join (n.2)

- Trovare le coppie di codici dei fornitori tali che entrambi i fornitori abbiano sede nella stessa città

```
SELECT FX.CodF, FY.CodF
FROM F AS FX, F AS FY
WHERE  FX.Sede=FY.Sede  AND
       FX.CodF < FY.CodF;
```

R

FX.CodF	FY.CodF
~~F1~~	~~F1~~
F1	F4
~~F2~~	~~F2~~
F2	F3
~~F3~~	~~F2~~
~~F3~~	~~F3~~
~~F4~~	~~F1~~
~~F4~~	~~F4~~
~~F5~~	~~F5~~

R

FX.CodF	FY.CodF
F1	F4
F2	F3

- Elimina le permutazioni della stessa coppia di valori

Join: sintassi alternativa

- Permette di specificare diversi tipi di join
  - outer join
- Permette di distinguere
  - condizioni di join
  - condizioni di selezione sulle tuple

SELECT [DISTINCT] *Attributi*
FROM *Tabella* *TipoJoin* JOIN *Tabella* ON *CondizioneDiJoin*
[WHERE *CondizioniDiTupla*];

TipoJoin = < INNER | [FULL | LEFT | RIGHT] OUTER >

INNER join

- Trovare il nome dei fornitori che forniscono almeno un prodotto rosso

```
SELECT NomeF
FROM P INNER JOIN FP ON P.CodP=FP.CodP
        INNER JOIN F ON F.CodF=FP.CodF
WHERE P.Colore='Rosso';
```

OUTER join

- Trovare il codice e il nome dei fornitori insieme al codice dei relativi prodotti forniti, visualizzando anche i fornitori che non hanno forniture

```
SELECT F.CodF, NomeF, CodP
FROM F LEFT OUTER JOIN FP ON
        F.CodF=FP.CodF;
```

F.CodF	F.NomeF	FP.CodP
F1	Andrea	P1
F1	Andrea	P2
F1	Andrea	P3
F1	Andrea	P4
F1	Andrea	P5
F1	Andrea	P6
F2	Luca	P1
F2	Luca	P2
F3	Antonio	P2
F4	Gabriele	P3
F4	Gabriele	P4
F4	Gabriele	P5
F5	*Matteo*	*NULL*

Funzioni aggregate

Introduzione a SQL

Funzione aggregata

- Opera su un insieme di valori
- Produce come risultato un unico valore (aggregato)
- E' indicata nella clausola SELECT
  - non si possono indicare anche attributi non aggregati
  - possono essere richieste più funzioni aggregate contemporaneamente
- Le funzioni aggregate sono valutate solo dopo l'applicazione di tutti i predicati nella clausola WHERE

Funzioni aggregate

- COUNT: conteggio degli elementi in un attributo
- SUM: somma dei valori di un attributo
- AVG: media dei valori di un attributo
- MAX: massimo valore di un attributo
- MIN: minimo valore di un attributo

COUNT

- Conteggio del numero di elementi di un insieme
  - righe di una tabella
  - valori (eventualmente distinti) di uno o più attributi

COUNT (<*| [DISTINCT | ALL] ListaAttributi >)}

- Se l'argomento della funzione è preceduto da DISTINCT, conta il numero di valori distinti dell'argomento

Funzione COUNT (n.1)

- Trovare il numero di fornitori

```
SELECT COUNT(*)
FROM F;
```

F

CodF	NomeF	NSoci	Sede
F1	Andrea	2	Torino
F2	Luca	1	Milano
F3	Antonio	3	Milano
F4	Gabriele	2	Torino
F5	Matteo	3	Venezia

R

5

Funzione COUNT (n.2)

- Trovare il numero di fornitori che hanno almeno una fornitura

FP

CodF	CodP	Qta
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F1	P5	100
F1	P6	100
F2	P1	300
F2	P2	400
F3	P2	200
F4	P3	200
F4	P4	300
F4	P5	400

```
SELECT COUNT(*)
FROM FP;
```

R

12

Conta il numero di forniture, non di fornitori

Funzione COUNT (n.2)

- Trovare il numero di fornitori che hanno almeno una fornitura

FP

CodF	CodP	Qta
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F1	P5	100
F1	P6	100
F2	P1	300
F2	P2	400
F3	P2	200
F4	P3	200
F4	P4	300
F4	P5	400

```
SELECT COUNT(CodF)
FROM FP;
```

R

12

Conta il numero di forniture, non di fornitori

Funzione COUNT (n.2)

- Trovare il numero di fornitori che hanno almeno una fornitura

FP

CodF	CodP	Qta
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F1	P5	100
F1	P6	100
F2	P1	300
F2	P2	400
F3	P2	200
F4	P3	200
F4	P4	300
F4	P5	400

```
SELECT COUNT(DISTINCT CodF)
FROM FP;
```

R

4

Conta il numero di fornitori diversi

Funzioni aggregate e WHERE

- Le funzioni aggregate sono valutate solo dopo l’applicazione di tutti i predicati nella clausola WHERE

Funzioni aggregate e WHERE

- Trovare il numero di fornitori che forniscono il prodotto P2

FP

CodF	CodP	Qta
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F1	P5	100
F1	P6	100
F2	P1	300
F2	P2	400
F3	P2	200
F4	P3	200
F4	P4	300
F4	P5	400

```
SELECT COUNT(*)
FROM FP
WHERE CodP='P2';
```

CodF	CodP	Qta
F1	P2	200
F2	P2	400
F3	P2	200

R

3

- Le funzioni aggregate sono valutate solo dopo l'applicazione di tutti i predicati nella clausola WHERE

SUM, MAX, MIN, AVG

- SUM, MAX, MIN e AVG
  - ammettono come argomento un attributo o un’espressione
- SUM e AVG
  - ammettono come argomento solo attributi di tipo numerico o intervallo di tempo
- MAX e MIN
  - richiedono che l’espressione sia ordinabile
  - possono essere applicate anche su stringhe di caratteri e istanti di tempo

Esempio: SUM

- Trovare la quantità totale di pezzi forniti per il prodotto P2

FP

CodF	CodP	Qta
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F1	P5	100
F1	P6	100
F2	P1	300
F2	P2	400
F3	P2	200
F4	P3	200
F4	P4	300
F4	P5	400

```
SELECT SUM(Qta)
FROM FP
WHERE CodP='P2';
```

CodF	CodP	Qta
F1	P2	200
F2	P2	400
F3	P2	200

R

800

Operatore GROUP BY

Introduzione a SQL

GROUP BY

- Clausola di raggruppamento

  GROUP BY ElencoAttributiDiRaggruppamento

  - L'ordine degli attributi di raggruppamento è ininfluente
- Nella clausola SELECT possono comparire *solo*
  - attributi presenti nella clausola GROUP BY
  - funzioni aggregate
- Gli attributi univocamente determinati da attributi già presenti nella clausola GROUP BY possono essere aggiunti senza alterare il risultato

Raggruppamento

- *Per ogni prodotto*, trovare la quantità totale di pezzi forniti

FP

CodF	CodP	Qta
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F1	P5	100
F1	P6	100
F2	P1	300
F2	P2	400
F3	P2	200
F4	P3	200
F4	P4	300
F4	P5	400

FP

CodF	CodP	Qta
F1	P1	300
F2	P1	300
F1	P2	200
F2	P2	400
F3	P2	200
F1	P3	400
F4	P3	200
F1	P4	200
F4	P4	300
F1	P5	100
F4	P5	400
F1	P6	100

R

CodP	
P1	600
P2	800
P3	600
P4	500
P5	500
P6	100

Raggruppamento

- *Per ogni prodotto*, trovare la quantità totale di pezzi forniti

FP

CodF	CodP	Qta
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F1	P5	100
F1	P6	100
F2	P1	300
F2	P2	400
F3	P2	200
F4	P3	200
F4	P4	300
F4	P5	400

FP

CodF	CodP	Qta
F1	P1	300
F2	P1	300
F1	P2	200
F2	P2	400
F3	P2	200
F1	P3	400
F4	P3	200
F1	P4	200
F4	P4	300
F1	P5	100
F4	P5	400
F1	P6	100

R

CodP	
P1	600
P2	800
P3	600
P4	500
P5	500
P6	100

```
SELECT CodP, SUM(Qta)
FROM FP
GROUP BY CodP;
```

GROUP BY e WHERE

- Per ogni prodotto, trovare la quantità totale di pezzi forniti da fornitori con sede a Milano

F

CodF	NomeF	NSoci	Sede
F1	Andrea	2	Torino
F2	Luca	1	Milano
F3	Antonio	3	Milano
F4	Gabriele	2	Torino
F5	Matteo	3	Venezia

FP

CodF	CodP	Qta
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F1	P5	100
F1	P6	100
F2	P1	300
F2	P2	400
F3	P2	200
F4	P3	200
F4	P4	300
F4	P5	400

GROUP BY e WHERE

- Per ogni prodotto, trovare la quantità totale di pezzi forniti da fornitori con sede a Milano

```
SELECT   ...
FROM FP, F
WHERE FP.CodF=F.CodF AND Sede='Milano'
...
```

GROUP BY e WHERE

- Per ogni prodotto, trovare la quantità totale di pezzi forniti da fornitori con sede a Milano

F.CodF	F.NomeF	F.NSoci	F.Sede	FP.CodF	FP.CodP	FP.Qta
F1	Andrea	2	Torino	F1	P1	300
F1	Andrea	2	Torino	F1	P2	200
F1	Andrea	2	Torino	F1	P3	400
F1	Andrea	2	Torino	F1	P4	200
F1	Andrea	2	Torino	F1	P5	100
F1	Andrea	2	Torino	F1	P6	100
F2	*Luca*	*1*	*Milano*	*F2*	*P1*	*300*
F2	*Luca*	*1*	*Milano*	*F2*	*P2*	*400*
F3	*Antonio*	*3*	*Milano*	*F3*	*P2*	*200*
F4	Gabriele	2	Torino	F4	P3	200
F4	Gabriele	2	Torino	F4	P4	300
F4	Gabriele	2	Torino	F4	P5	400

GROUP BY e WHERE

- Per ogni prodotto, trovare la quantità totale di pezzi forniti da fornitori con sede a Milano

```
SELECT CodP, SUM(Qta)
FROM FP, F
WHERE FP.CodF=F.CodF AND Sede='Milano'
GROUP BY CodP;
```

- I prodotti senza forniture non sono inclusi nel risultato

GROUP BY e WHERE

- Per ogni prodotto, trovare la quantità totale di pezzi forniti da fornitori con sede a Milano

FP.CodP	FP.Qta
P1	300
P2	400
P2	200

R

FP.CodP	
P1	300
P2	600

GROUP BY e SELECT

- Per ogni prodotto, trovare il codice, *il nome* e la quantità totale fornita

```
SELECT P.CodP, NomeP, SUM(Qta)
FROM P, FP
WHERE P.CodP=FP.CodP
GROUP BY P.CodP, NomeP
```

- Gli attributi univocamente determinati da attributi già presenti nella clausola GROUP BY possono essere aggiunti *senza alterare il risultato*

Condizione di selezione sui gruppi: HAVING

- Non è possibile utilizzare la clausola WHERE per definire condizioni di selezione sui gruppi

- Condizione di selezione sui gruppi espressa in clausola HAVING:

  HAVING Condizioni di gruppo

  - permette di specificare condizioni *solo* su funzioni aggregate

Condizione di selezione sui gruppi (n.1)

- Trovare la quantità totale di pezzi forniti per i prodotti per cui sono forniti *in totale* almeno 600 pezzi

FP

CodF	CodP	Qta
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F1	P5	100
F1	P6	100
F2	P1	300
F2	P2	400
F3	P2	200
F4	P3	200
F4	P4	300
F4	P5	400

FP

CodF	CodP	Qta
F1	P1	300
F2	P1	300
F1	P2	200
F2	P2	400
F3	P2	200
F1	P3	400
F4	P3	200
F1	P4	200
F4	P4	300
F1	P5	100
F4	P5	400
F1	P6	100

R

CodP	
P1	600
P2	800
P3	600

Condizione di selezione sui gruppi (n.1)

- Trovare la quantità totale di pezzi forniti per i prodotti per cui sono forniti *in totale* almeno 600 pezzi

```
SELECT CodP, SUM(Qta)
FROM FP
GROUP BY CodP
HAVING SUM(Qta)>=600;
```

- La clausola HAVING permette di specificare condizioni su funzioni aggregate

Condizione di selezione sui gruppi (n.2)

- Trovare il codice dei prodotti rossi forniti da più di un fornitore

P

CodP	NomeP	Colore	Taglia	Magazzino
P1	Maglia	Rosso	40	Torino
P2	Jeans	Verde	48	Milano
P3	Camicia	Blu	48	Roma
P4	Camicia	Blu	44	Torino
P5	Gonna	Blu	40	Milano
P6	Bermuda	Rosso	42	Torino

FP

CodF	CodP	Qta
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F1	P5	100
F1	P6	100
F2	P1	300
F2	P2	400
F3	P2	200
F4	P3	200
F4	P4	300
F4	P5	400

Condizione di selezione sui gruppi (n.2)

- Trovare il codice dei prodotti rossi forniti da più di un fornitore

```
SELECT FP.CodP
FROM FP, P
WHERE FP.CodP=P.CodP AND Colore='Rosso'
GROUP BY FP.CodP
HAVING COUNT(*)>1;
```

Condizione di selezione sui gruppi (n.2)

- Trovare il codice dei prodotti rossi forniti da più di un fornitore

F.CodF	F.CodP	F.Qta	P.CodP	P.NomeP	P.Colore	P.Taglia	P.Magazzino
F1	P1	300	P1	Maglia	Rosso	40	Torino
F2	P1	300	P1	Maglia	Rosso	40	Torino
F1	P6	100	P6	Bermuda	Rosso	42	Torino

R

CodP
P1