

Linguaggio SQL

- DML: operatori insiemistici
- DML: query nidificate semplici

Database di esempio

S

<u>SNum</u>	SName	Status	City
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

SP

<u>SNum</u>	<u>PNum</u>	QTY
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	300
S4	P5	400

P

<u>PNum</u>	PName	Color	Weight	City
P1	Nut	Red	12	London
P2	Bolt	Green	17	Paris
P3	Screw	Blue	17	Rome
P4	Screw	Red	14	London
P5	Cam	Blue	12	Paris
P6	Cog	Red	19	London

Operatori insiemistici

- SQL mette a disposizione gli operatori insiemistici union, intersect, except corrispondenti agli operatori di unione, intersezione e differenza dell'algebra relazionale
- In Oracle l'operatore except viene chiamato minus
- Si possono applicare esclusivamente a tabelle definite sullo stesso insieme di attributi
- Di solito gli operatori insiemistici rimuovono dal risultato i duplicati a meno di chiedere esplicitamente di mantenerli aggiungendo la keyword all

Unione insiemistica

- Algebra relazionale:
Date due relazioni A e B , $A \cup B$ è l'insieme delle tuple x tali che x appartiene ad A o appartiene a B (o appartiene a entrambi)
- SQL:

```
select EspressioneListaAttributi1
from ListaTabelle1
where Condizioni1
union [all]
select EspressioneListaAttributi2
from ListaTabelle2
where Condizioni2
```
- *EspressioneListaAttributi1* esprime attributi dello stesso tipo di *EspressioneListaAttributi2*

Unione insiemistica

- Esempio:

```
select PNum
from P
where Weight>16
union
select PNum
from SP
where SNum='S2';
```

<u>PNum</u>
P1
P2
P3
P6

- Ricava i codici dei prodotti che pesano più di 16 kg e/o sono forniti dal fornitore S2

Unione insiemistica

- Esempio:

```
select PNum
from P
where Weight>16
      union all
select PNum
from SP
where SNum='S2';
```

<u>PNum</u>
P2
P3
P6
P1
P2

- Ricava i codici dei prodotti che pesano più di 16 kg e/o sono forniti dal fornitore S2 **mantenendo i duplicati**

Intersezione insiemistica

- Algebra relazionale:
Date due relazioni A e B , $A \cap B$ è l'insieme delle tuple x tali che x appartiene ad A e appartiene anche a B
- SQL:

```
select EspressioneListaAttributi1
from ListaTabelle1
where Condizioni1
      intersect [all]
select EspressioneListaAttributi2
from ListaTabelle2
where Condizioni2;
```
- *EspressioneListaAttributi1* esprime attributi dello stesso tipo di *EspressioneListaAttributi2*

Intersezione insiemistica

- Esempio:

```
select PNum from P  
where Weight>16
```

intersect

```
select PNum from SP  
where SNum='S2';
```

<u>PNum</u>
P2

- Ricava i codici dei prodotti che pesano più di 16 kg e che sono forniti dal fornitore S2
- Oracle non supporta l'operatore intersect all; PostgreSQL sì

Sottrazione insiemistica

- Date due relazioni A e B , **A except B** è l'insieme delle tuple x tali che x appartiene ad A **ma non** appartiene a B
- Sintassi

```
select EspressioneListaAttributi1
from ListaTabelle1
where Condizioni1
      except [all]
select EspressioneListaAttributi2
from ListaTabelle2
where Condizioni2;
```
- *EspressioneListaAttributi1* esprime attributi dello stesso tipo di *EspressioneListaAttributi2*

Sottrazione insiemistica

- Esempio:

```
select PNum from P  
where Weight>16
```

except

```
select PNum from SP  
where SNum='S2';
```

<u>PNum</u>
P3
P6

- Ricava i codici dei prodotti che pesano più di 16 kg ma che non sono forniti dal fornitore S2
- Oracle non supporta l'operatore minus all;
PostgreSQL supporta *except all*

Esempi (con soluzioni alternative)

- Trovare i codici dei fornitori che hanno fornito almeno un prodotto di colore rosso o verde

- Soluzione 1 (con operatore insiemistico):

```
select SP.SNum
from SP join P on SP.PNum=P.PNum
where Color='Red'
```

union

```
select SP.SNum
from SP join P on SP.PNum=P.PNum
where Color='Green';
```

- Soluzione 2 (senza operatore insiemistico):

```
select distinct SP.SNum
from SP join P on SP.PNum=P.PNum
where Color='Red' or Color='Green';
```

Esempi (con soluzioni alternative)

- Trovare i codici dei fornitori che hanno fornito almeno un prodotto di colore rosso **e** uno di colore verde
- Soluzione 1 (con operatore insiemistico):

```
select SP.SNum
from SP join P on SP.PNum=P.PNum
where Color='Red'
```

intersect

```
select SP.SNum
from SP join P on SP.PNum=P.PNum
where Color='Green';
```

- Soluzione 2 (senza operatore insiemistico):

```
select distinct SP1.SNum
from SP SP1 join P P1 on SP1.PNum=P1.PNum, SP SP2 join P P2 on
SP2.PNum=P2.PNum
where SP1.SNum=SP2.SNum AND P1.Color='Red' AND P2.Color='Green';
```

Esempi (con soluzioni alternative)

- Trovare i codici dei fornitori che hanno fornito almeno un prodotto di colore rosso **ma non** uno di colore verde

- Soluzione 1 (con operatore insiemistico):

```
select SP.SNum
from SP join P on SP.PNum=P.PNum
where Color='Red'
```

except

```
select SP.SNum
from SP join P on SP.PNum=P.PNum
where Color='Green';
```

- Soluzione 2 (senza operatore insiemistico):

?

– **DEVO USARE UN NUOVO COSTRUTTO: le query nidificate**

Interrogazioni nidificate

- È possibile annidare **sottoquery** in una query
- Le sottoquery sono utili per risolvere sottoproblemi
- Per es. ricavare i codici dei fornitori il cui status è inferiore alla media
- Sottoproblema: qual è la media degli status?
- Usiamo una (sotto)query per ottenere la media
select avg(Status) from S
e la annidiamo nell'altra query
select SNum
from S
where Status < (select avg(Status) from S);

Interrogazioni nidificate

- Esistono due tipi di sottoquery:
 - **semplici (o stratificate)**: è possibile valutare prima l'interrogazione più interna (una volta per tutte), poi, sulla base del suo risultato, valutare l'interrogazione più esterna
 - **correlate (o incrociate)**: l'interrogazione più interna fa riferimento a una delle tabelle appartenenti all'interrogazione più esterna. Per ciascuna riga candidata alla selezione nell'interrogazione più esterna, è necessario valutare nuovamente la sottoquery

Interrogazioni nidificate

- Le sottoquery annidate possono essere usate:
 - nella clausola *where* (utilizzando test di selezione sofisticati come *all/any, in, exists*)
 - nella clausola *from* (permettono di valutare query su tabelle derivate)
 - nella clausola *select* (standard SQL:2003, non supportata da tutti i DBMS)

Sottointerrogazioni semplici

- Iniziamo a vedere le sottointerrogazioni semplici specificate nella clausola *where*
- Se c'è la certezza che la query restituisca **una sola riga con un solo valore**, si possono utilizzare gli usuali operatori di confronto (**=, <>, <, >, <=, >=**)

Sottointerrogazioni semplici

- Esempio: trovare il codice dei fornitori che operano nella stessa città di S1

```
select SNum  
from S  
where City =  
( select City from S where SNum='S1' );
```

Sottointerrogazioni semplici con quantificatori

- Se la sottointerrogazione restituisce più di un valore, per effettuare il confronto è necessario utilizzare uno dei quantificatori seguenti:
 - **any**: il predicato deve essere soddisfatto da *almeno una riga* restituita dalla sottointerrogazione
 - **all**: il predicato deve essere soddisfatto da *tutte le righe* restituite dalla sottointerrogazione
- Sintassi
select ... from ...
where *EspressioneAttr* **operatoreDiConfronto** **any/all**
(select ... from ... where ...);

Sottointerrogazioni semplici con quantificatori

- Esempio: ricavare i codici dei fornitori che *non* hanno il valore di status più grande
- Scrivere
select SNum from S
where Status < max(Status);
sarebbe **errato** perché non è permesso usare operatori aggregati nella clausola where (la clausola where agisce a livello di riga)
- Possiamo usare il quantificatore any e una sottoquery

Sottointerrogazioni semplici con quantificatori

- Esempio: ricavare i codici dei fornitori che *non* hanno il valore di status più grande

```
select SNum from S
where Status < any
      ( select Status
        from S );
```

SNum
S1
S2
S4

- Solo i fornitori S3 e S5 con stato massimo (30) non vengono selezionati perché il loro status non è minore di nessuno status

Sottointerrogazioni semplici con quantificatori

- Esempio: ricavare i codici dei fornitori che hanno dato la fornitura con la più grande quantità di parti
- Scrivere qualcosa del tipo
select SNum, max(Qty) from SP;
oppure
select SNum from SP where Qty=max(Qty);
sarebbe **errato** per motivi già discussi
- Possiamo scrivere
select SNum from SP where Qty=(select max(Qty) from SP);
oppure usare il quantificatore all e una sottoquery

Sottointerrogazioni semplici con quantificatori

```
select SNum from SP
where Qty >= all
      ( select Qty
        from SP );
```

SNum
S1
S2
S4

- Vengono selezionati i fornitori S1, S2 e S4 perché la quantità fornita (400) è maggiore o al più uguale a qualsiasi valore di Qty

Sottointerrogazioni semplici con quantificatori

- Cosa succede a un'interrogazione di questo tipo se l'attributo assume (o può assumere) valori nulli?
- Per esempio Status è nullable, quindi la tabella S potrebbe assumere i seguenti valori:

<u>SNum</u>	SName	Status	City
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens
S6	Pippo	(null)	Rome

- Cosa restituisce la seguente query?
select SNum from S
where Status >= **all**
 (select Status
 from S);

Sottointerrogazioni semplici con quantificatori

- `select SNum from S`
`where Status >= all`
`(select Status`
`from S);`

SNum
- Il predicato della clausola `where` non è vero nel caso di S6 (*Status* >= *null* è unknown), quindi il risultato della query è vuoto

Sottointerrogazioni semplici con quantificatori

- Cosa restituisce la seguente query?
select SNum from S
where Status = **all**
 (select Status
 from S);
- Per questi motivi l'operatore di uguaglianza non viene (quasi) mai usato con il quantificatore all

Sottointerrogazioni semplici con quantificatori

- Cosa restituisce la seguente query?
select SNum from S
where Status = **all**
 (select Status
 from S);
- Per questi motivi l'operatore di uguaglianza non viene (quasi) mai usato con il quantificatore all
- Nulla! Uno Status che è uguale a tutti non esiste.

Utilizzo di IN e NOT IN

- Per verificare che un valore sia presente o meno in un insieme possiamo utilizzare le condizioni in e not in
- Esempio: elencare i fornitori che hanno fornito parti rosse

```
select distinct SNum from SP
where PNum in
    ( select PNum from P
      where Color='Red' );
```

SNum
S1
S4

Utilizzo di IN e NOT IN

- Esempio: elencare i fornitori che **non** hanno fornito anche parti blu (elencare i fornitori che forniscono solo parti non blu o nessuna parte)

```
select distinct SNum from SP
where PNum not in
      ( select PNum from P
        where Color='Blue' );
```

SNum
S1
S2
S3
S4

- È corretta?

Utilizzo di IN e NOT IN

- Esempio: elencare i fornitori che **non** hanno fornito anche parti blu (elencare i fornitori che forniscono solo parti non blu o nessuna parte)

```
select distinct SNum from SP
where PNum not in
      ( select PNum from P
        where Color='Blue' );
```

SNum
S1
S2
S3
S4

- No, abbiamo solo ricavato i fornitori che forniscono (anche) parti non blu** (considerate S1 e S2 hanno fornito parti blu, rosse e verdi, S3 parti verdi, S4 parti verdi e rosse e S5, non fornisce parti)

Database di esempio

S

<u>SNum</u>	SName	Status	City
S1	Smith	20	London
S2	Jones	10	Paris
S3	Blake	30	Paris
S4	Clark	20	London
S5	Adams	30	Athens

P

<u>PNum</u>	PName	Color	Weight	City
P1	Nut	Red	12	London
P2	Bolt	Green	17	Paris
P3	Screw	Blue	17	Rome
P4	Screw	Red	14	London
P5	Cam	Blue	12	Paris
P6	Cog	Red	19	London

SP

<u>SNum</u>	<u>PNum</u>	QTY
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	300
S4	P5	400

Utilizzo di IN e NOT IN

- Esempio: Esempio: elencare i fornitori che **non** hanno fornito anche parti blu (elencare i fornitori che forniscono solo parti non blu o nessuna parte)

```
select distinct SNum from S
where SNum not in
    ( select SNum from SP
      where PNum in
        ( select PNum from P
          where Color='Blue' ) );
```

SNum
S2
S3
S5

- **Ora è corretta**

Utilizzo di IN e NOT IN

- Esempio: elencare per ogni fornitore (con i suoi dettagli) la quantità massima fornita

```
select S.*, Qty
from S join SP on S.SNum=SP.SNum
where (SP.SNum, Qty) in
      (select SNum, max(Qty)
       from SP
       group by SNum);
```

SNum	SName	Status	City	Qty
S1	Smith	20	London	400
S2	Jones	10	Paris	400
S3	Blake	30	Paris	200
S4	Clark	20	London	400

- Notare che si considera l'appartenenza di una coppia di valori
- Manca S5. Come si può modificare la query per avere veramente ogni fornitore?

Utilizzo di IN e NOT IN

- Esempio: elencare per ogni fornitore la quantità massima fornita

```
select S.*, coalesce(Qty,0)
from S left join SP on S.SNum=SP.Snum
where SP.SNum is null or (SP.SNum, Qty) in
  ( select SNum, max(Qty)
    from SP
   group by SNum);
```

SNum	SName	Status	City	Qty
S1	Smith	20	London	400
S2	Jones	10	Paris	400
S3	Blake	30	Paris	200
S4	Clark	20	London	400
S5	Adams	30	Athens	0

Sottoquery nella clausola *from*

- Esempio: elencare il nome del fornitore e la quantità massima da lui fornita

```
select SName, MaxQty
from S left join (
    select SNum, max(Qty) as MaxQty
    from SP
    group by SNum) SMax
on S.SNum=SMax.SNum;
```

SName	MaxQty
Smith	400
Jones	400
Blake	200
Clark	400
Adams	(null)

- La sottoquery produce una tabella che viene usata dalla query più esterna
- Una sottoquery nella clausola *from* non può essere correlata

Sottoquery nella clausola *select*

- Esempio: elencare i fornitori e il numero totale di parti fornite

```
select SNum, (select sum(Qty) from SP  
              where SP.SNum=S.SNum) TotQty  
from S;
```

SNum	TotQty
S1	1350
S2	700
S3	200
S4	900
S5	(null)

- Una sottoquery nella clausola *select* deve restituire esattamente una riga con un valore
- Quasi sempre sarà una sottoquery correlata

Esercizio 5.1

- Elencare i fornitori che forniscono parti disponibili a Londra (sia con costruito in/not in che con costruito any/all)

Esercizio 5.2

- Elencare le città in cui non vi sono fornitori con status minore della media (sia con costruito in/not in che con costruito any/all)