SQL

4 lezione

interrogazioni di tipo insiemistico

Francesca Gasparini

gasparini@disco.unimib.it

Un esempio completo

Impiegato

ID impiegato	Nome	Cognome	Dipart	Ufficio	Stipendio	premioprod	Mansione	Città	Idcapo

Dip

Nome	Indirizzo	Città

Query: Per ogni dipartimento di Milano in cui il numero di impiegati è almeno 2, si vuole conoscere il valor medio degli stipendi, ordinando il risultato per valori decrescenti di stipendio medio e quindi per dipart.

SELECT I.dipart, AVG(Stipendio) AS AvgStipendio
FROM Impiegato AS I, dip AS S
WHERE I.dipart = S.nome AND S.Città = 'Milano'
GROUP BY I.dipart
HAVING COUNT(*) >= 2
ORDER BY AVG(Stipendio) DESC , dipart

dipart	AvgStipendio
Direzione	76,5
Amministrazione	41,6666666666667

Un esempio completo

Impiegato

ID impiegato	Nome	Cognome	Dipart	Ufficio	Stipendio	premioprod	Mansione	Città	Idcapo

Dip

Nome	Indirizzo	Città

Query: Per ogni dipartimento di Milano in cui il numero di impiegati è almeno 2, si vuole conoscere il valor medio degli stipendi, ordinando il risultato per valori decrescenti di stipendio medio e quindi per dipart.

SELECT I.dipart, AVG(Stipendio) AS AvgStipendio
FROM Impiegato AS I, dip AS S
WHERE I.dipart = S.nome AND S.Città = 'Milano'
GROUP BY I.dipart
HAVING COUNT(*) >= 2
ORDER BY AVG(Stipendio) DESC , dipart

dipart	AvgStipendio
Direzione	76,5
Amministrazione	41,6666666666667

Un esempio completo

Impiegato

ID impiegato	Nome	Cognome	Dipart	Ufficio	Stipendio	premioprod	Mansione	Città	Idcapo

Dip

Nome	Indirizzo	Città

Query: Per ogni dipartimento di Milano in cui il numero di impiegati è almeno 2, si vuole conoscere il valor medio degli stipendi, ordinando il risultato per valori decrescenti di stipendio medio e quindi per dipart.

SELECT I.dipart, AVG(Stipendio) AS AvgStipendio
FROM Impiegato AS I, dip AS S
WHERE I.dipart = S.nome AND S.Città = 'Milano'
GROUP BY I.dipart
HAVING COUNT(*) >= 2
ORDER BY AVG(Stipendio) DESC , dipart

dipart	AvgStipendio
Direzione	76,5
Amministrazione	41,6666666666667

Un esempio completo

Impiegato

ID impiegato	Nome	Cognome	Dipart	Ufficio	Stipendio	premioprod	Mansione	Città	Idcapo

Dip

Nome	Indirizzo	Città

Query: Per ogni dipartimento di Milano in cui il numero di impiegati è almeno 2, si vuole conoscere il valor medio degli stipendi, ordinando il risultato per valori decrescenti di stipendio medio e quindi per dipart.

SELECT I.dipart, AVG(Stipendio) AS AvgStipendio
FROM Impiegato AS I, dip AS S
WHERE I.dipart = S.nome AND S.Città = 'Milano'
GROUP BY I.dipart
HAVING COUNT(*) >= 2
ORDER BY AVG(Stipendio) DESC , dipart

dipart	AvgStipendio
Direzione	76,5
Amministrazione	41,6666666666667

Un esempio completo

Impiegato

ID impiegato	Nome	Cognome	Dipart	Ufficio	Stipendio	premioprod	Mansione	Città	Idcapo

Dip

Nome	Indirizzo	Città

Query: Per ogni dipartimento di Milano in cui il numero di impiegati è almeno 2, si vuole conoscere il valor medio degli stipendi, ordinando il risultato per valori decrescenti di stipendio medio e quindi per dipart.

SELECT I.dipart, AVG(Stipendio) AS AvgStipendio
FROM Impiegato AS I, dip AS S
WHERE I.dipart = S.nome AND S.Città = 'Milano'
GROUP BY I.dipart
HAVING COUNT(*) >= 2
ORDER BY AVG(Stipendio) DESC , dipart

dipart	AvgStipendio
Direzione	76,5
Amministrazione	41,6666666666667

Un esempio completo

Impiegato

ID impiegato	Nome	Cognome	Dipart	Ufficio	Stipendio	premioprod	Mansione	Città	idcapo

Dip

Nome	Indirizzo	Città

Query: Per ogni dipartimento di Milano in cui il numero di impiegati è almeno 2, si vuole conoscere il valor medio degli stipendi, ordinando il risultato per valori decrescenti di stipendio medio e quindi per dipart.

```
SELECT I.dipart, AVG(Stipendio) AS AvgStipendio
FROM Impiegato AS I, dip AS S
WHERE I.dipart = S.nome AND S.Città = 'Milano'
GROUP BY I.dipart
HAVING COUNT(*) >= 2
ORDER BY AVG(Stipendio) DESC , dipart
```

dipart	AvgStipendio
Direzione	76,5
Amministrazione	41,6666666666667

Ordine di esecuzione query

•Esecuzione della query senza considerare GROUP BY e operatori aggregati.

•Raggruppamento delle righe del risultato come specificato da GROUP BY.

•Applicazione degli operatori di aggregazione sui gruppi di righe precedentemente costruiti.

•Selezione dei gruppi risultanti come specificato dalla clausola HAVING.

•Ordinamento secondo la clausola order by.

Esempio: gestione ordini

- Cliente(CODCLI, INDIRIZZO, P_IVA);
- Ordine(CODORD,CODCLI,DATA,IMPORTO)
- Dettaglio(CODORD,CODPROD,QTA)
- Prodotto(CODPROD,NOME, PREZZO)

Ordine

CODORD	CODCLI	DATA	IMPORTO
1	3	1-6-97	50.000.000
2	4	3-8-97	8.000.000
3	3	1-9-97	5.500.000
4	1	1-7-97	12.000.000
5	1	1-8-97	1.500.000
6	3	3-9-97	27.000.000

Query con raggruppamento

QUERY: Estrarre la somma degli importi degli ordini successivi al 10-6-97 per quei clienti che hanno emesso almeno 2 ordini

Ordine

CODORD	CODCLI	DATA	IMPORTO
1	3	1-6-97	50.000.000
2	4	3-8-97	8.000.000
3	3	1-9-97	5.500.000
4	1	1-7-97	12.000.000
5	1	1-8-97	1.500.000
6	3	3-9-97	27.000.000

Query con raggruppamento

QUERY: Estrarre la somma degli importi degli ordini successivi al 10-6-97 per quei clienti che hanno emesso almeno 2 ordini

```
SELECT CodCli, SUM(Importo)
FROM Ordine
WHERE Data > 10-6-97
GROUP BY CodCli
HAVING COUNT(*) >= 2
```

Esecuzione query passo passo

- **Passo 1:** Valutazione where

CodOrd	CodCli	Data	Importo
2	4	3-8-97	8.000.000
3	3	1-9-97	5.500.000
4	1	1-7-97	12.000.000
5	1	1-8-97	1.500.000
6	3	3-9-97	27.000.000

- **Passo 2:** Raggruppamento - si valuta la clausola **group by**

CodOrd	CodCli	Data	Importo
4	1	1-7-97	12.000.000
5	1	1-8-97	1.500.000
3	3	1-9-97	1.500.000
6	3	3-9-97	5.500.000
2	4	3-8-97	8.000.000

Esecuzione query passo passo

- **Passo 3 -** Calcolo degli aggregati: si calcolano **sum(Importo)** e **count(Importo)** per ciascun gruppo

CodCli	sum (Importo)	count(Importo)
1	13.500.000	2
3	32.500.000	2
4	5.000.000	1

- **Passo 4 -** Estrazione dei gruppi: si valuta il predicato **count(Importo) >= 2**

CodCli	sum (Importo)	count (Importo)
1	13.500.000	2
3	32.500.000	2
4	5.000.000	1

- **Passo 5 –** Produzione del risultato:

CodCli	sum (Importo)
1	13.500.000
3	32.500.000

group by: nota

GROUP BY attributo1, attributo2

è identico a

GROUP BY attributo2, attributo1

having: osservazioni

- La sintassi permette di definire la clausola having anche senza la clausola group by.
- In questo caso l'intero insieme di righe è trattato come un unico raggruppamento.

- quali predicati vanno usati per la clausola where e quali per la clausola having?
- Per la clausola having devono essere usati solo predicati in cui compaiono operatori aggregati.

Operatori insiemistici

Permettono di costruire query binarie concatenando due query SQL

- **union**
- **intersect**
- **except (minus - differenza)**

Sintassi:
SelectSQL { < union | intersect | except > [all]
SelectSQL }

i duplicati vengono eliminati (a meno che si usi all)

Unione

- La select da sola non permette di fare unioni; serve un costrutto esplicito:

```
select  …
union  [all]
select ...
```

- i duplicati vengono eliminati (a meno che si usi all) diversamente dalle query viste precedentemente

Unione

una interrogazione o sottointerrogazione può essere costituita da due o più interrogazioni connesse dall'operatore UNION

l'operatore UNION restituisce tutte le tuple **distinte** restituite da almeno una delle sottointerrogazioni a cui è applicata

l'operatore **UNION** elimina i duplicati dal risultato

l'operatore **UNION ALL** non elimina i duplicati

Unione

•l'operatore UNION impone alcune restrizioni sulle interrogazioni su cui opera

•le interrogazioni devono restituire lo **stesso numero di colonne**, e le colonne corrispondenti devono avere lo stesso dominio (non è richiesto che abbiano la stessa lunghezza) o domini compatibili

•la corrispondenza si basa sulla **posizione delle colonne**, indipendentemente dal loro nome

•se si usa una clausola di **ORDER BY** questa deve essere usata **una sola volta alla fine dell'interrogazione** e *non alla fine di ogni* SELECT

Unione

R

A	B
1	a
1	a
2	a
2	b
2	c
3	b

S

C	B
1	a
1	b
2	a
2	c
3	c
4	d

•le interrogazioni devono restituire lo **stesso numero di colonne**, e le colonne corrispondenti devono avere lo stesso dominio (*non è richiesto che abbiano la stessa lunghezza*) o domini compatibili

```
SELECT A
FROM R
UNION
SELECT C
FROM S
```

1
2
3
4

```
SELECT A
FROM R
UNION
SELECT C AS A
FROM S
```

A
1
2
3
4

Unione

R

A	B
1	a
1	a
2	a
2	b
2	c
3	b

S

C	B
1	a
1	b
2	a
2	c
3	c
4	d

```
SELECT A,B
FROM R
UNION
SELECT B,C AS A
FROM S
```

Non corretta!

```
SELECT B
FROM R
UNION ALL
SELECT B
FROM S
```

B
a
a
a
b
c
b
a
b
a
c
c
d

•la corrispondenza si basa sulla **posizione delle colonne**, indipendentemente dal loro nome

UNION

- Estrarre i codici degli ordini i cui importi superano 500 euro oppure in cui qualche prodotto è presente con quantità superiore a 1000

```
SELECT CodOrd
FROM Ordine
WHERE Importo > 500
      UNION
SELECT CodOrd
FROM Dettaglio
WHERE Qta > 1000
```

Cliente(CODCLI, INDIRIZZO, P_IVA);
Ordine(CODORD,CODCLI,DATA,IMPORTO)
Dettaglio(CODORD,CODPROD,QTA)
Prodotto(CODPROD,NOME, PREZZO)

Notazione posizionale

```
SELECT padre
FROM paternita
   UNION
SELECT madre
FROM maternita
```

padre	figlio
sergio	franco
luigi	olga
luigi	filippo
franco	andrea
franco	aldo

madre	figlio
luisa	maria
luisa	luigi
anna	olga
anna	filippo
maria	andrea
maria	aldo

padre
anna
franco
luigi
luisa
maria
sergio

Se gli attributi hanno un nome diverso il risultato normalmente non usa alcun nome o usa il nome del primo operando

Notazione posizionale, 2

```
SELECT padre, figlio
FROM paternita
   UNION
SELECT figlio, madre
FROM maternita
```

padre	figlio
aldo	maria
andrea	maria
filippo	anna
franco	aldo
franco	andrea
luigi	filippo
luigi	luisa
luigi	olga
maria	luisa
olga	anna
sergio	franco

padre	figlio
sergio	franco
luigi	olga
luigi	filippo
franco	andrea
franco	aldo

madre	figlio
luisa	maria
luisa	luigi
anna	olga
anna	filippo
maria	andrea
maria	aldo

```
SELECT padre, figlio
FROM paternita
   UNION
SELECT madre, figlio
FROM maternita
```

padre	figlio
anna	filippo
anna	olga
franco	aldo
franco	andrea
luigi	filippo
luigi	olga
luisa	luigi
luisa	maria
maria	aldo
maria	andrea
sergio	franco

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
```

Uso della parola chiave all

• Estrarre i padri di persone con nome “Giorgio” o “Giovanni”, presentando due volte i padri che hanno due figli con ciascuno dei nomi

```
select Padre
from Paternita
where Figlio = 'Giorgio'
      union all
select Padre
from Paternita
where Figlio = 'Giovanni'
```

Intersezione e differenza

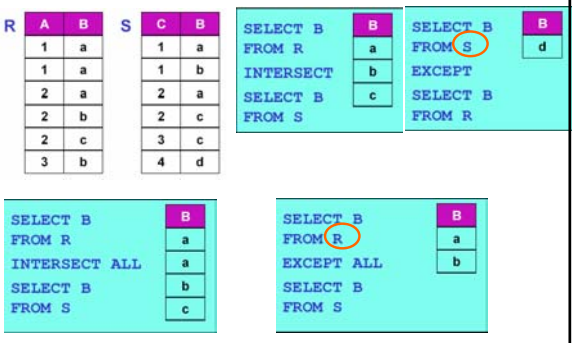
gli operatori INTERSECT ed EXCEPT (o MINUS) eseguono l'intersezione e la differenza

per questi operatori valgono le stesse condizioni di applicabilità viste per l‘operatore UNION

le interrogazioni devono restituire lo **stesso numero di colonne**, e le colonne corrispondenti devono avere lo stesso dominio (non è richiesto che abbiano la stessa lunghezza) o domini compatibili

la corrispondenza si basa sulla **posizione delle colonne**, indipendentemente dal loro nome

Intersezione e differenza



Differenza

impiegato

dip

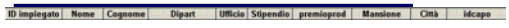
Query: Estrarre il nome degli impiegati esclusi quelli che hanno il nome uguale al cognome di un altro impiegato

```
select Nome
from Impiegato
      except
select Cognome as Nome
from Impiegato
```

- Si può fare con join
- vedremo che si può esprimere con select nidificate

Intersezione

impiegato



dip

Query: Estrarre gli impiegati che hanno il nome uguale al cognome di un altro impiegato

```
select Nome
from Impiegato
      intersect
select Cognome as Nome
from Impiegato
```

equivale a

```
select I.Nome
from Impiegato I, Impiegato J
where I.Nome = J.Cognome
```

Differenza

Estrarre i codici degli ordini i cui importi superano 500 euro ma in cui nessun prodotto è presente con quantità superiore a 1000

```
select CodOrd
from Ordine
where Importo > 500
      except
select CodOrd
from Dettaglio
where Qta > 1000
```

Cliente(CODCLI, INDIRIZZO, P_IVA);
Ordine(CODORD,CODCLI,DATA,IMPORTO)
Dettaglio(CODORD,CODPROD,QTA)
Prodotto(CODPROD,NOME, PREZZO)

Intersezione

Query: Estrarre i codici degli ordini i cui importi superano 500 euro e in cui qualche prodotto è presente con quantità superiore a 1000

```
select CodOrd
from Ordine
where Importo > 500
intersect
select CodOrd
from Dettaglio
where Qta > 1000
```

Cliente(CODCLI, INDIRIZZO, P_IVA);
Ordine(CODORD,CODCLI,DATA,IMPORTO)
Dettaglio(CODORD,CODPROD,QTA)
Prodotto(CODPROD,NOME, PREZZO)

Interrogazioni nidificate

Interrogazioni nidificate

Una delle ragioni che rendono SQL un linguaggio potente è la possibilità di esprimere interrogazioni più complesse in termini di interrogazioni più semplici, tramite il meccanismo delle **subqueries** (sottointerrogazioni)

La clausola **WHERE** di una query (detta query esterna) può infatti contenere un'altra query (detta subquery)

La subquery viene usata per determinare uno o più valori da usare come **valori di confronto** in un predicato della query esterna

Interrogazioni nidificate

Nella clausola **where** possono comparire predicati che:

– confrontano un attributo (o un'espressione sugli attributi) con il risultato di una query SQL; sintassi:

ScalarValue Operator < **any** | **all** > *SelectSQL*

- **any**: il predicato è vero se almeno una riga restituita dalla query *SelectSQL* soddisfa il confronto
- **all**: il predicato è vero se tutte le righe restituite dalla query *SelectSQL* soddisfano il confronto

- *Operator*: uno qualsiasi tra **=, <>, <, <=, >, >=**

- La query che appare nella clausola **where** è chiamata query nidificata

Interrogazioni nidificate

impiegato

ID impiegato	Nome	Cognome	Dipart	Ufficio	Stipendio	premiopred	Mansione	Città	idcapo

dip

Nome	Indirizzo	Città

Query: Estrarre il cognome degli impiegati che lavorano in dipartimenti con sede a Milano

```
SELECT cognome
FROM Impiegato
WHERE dipart IN (SELECT nome
FROM dip
WHERE Citta = 'Milano')
```

nome
Amministrazione
direzione
ricerca

Cognome
Rossi
Verdi
Rossi
Lanzi
Borroni
Oscar

```
SELECT cognome
FROM Impiegato
WHERE dipart IN ('amministrazione','direzione','ricerca')
```

Interrogazioni nidificate

impiegato

ID impiegato	Nome	Cognome	Dipart	Ufficio	Stipendio	premioprod	Mansione	Città	idcapo

dip

Nome	Indirizzo	Città

Query: Estrarre il cognome degli impiegati che lavorano in dipartimenti con sede a Milano

```
SELECT cognome
FROM Impiegato
WHERE dipart IN (SELECT nome
FROM dip
WHERE Citta = 'Milano')
```

Cognome
Rossi
Verdi
Rossi
Lanzi
Borroni
Oscar

```
SELECT Cognome
FROM impiegato INNER JOIN dip
        ON dip.Nome = impiegato.Dipart
WHERE dip.Città='Milano';
```

Interrogazioni nidificate, commenti

- La forma nidificata è "meno dichiarativa", ma talvolta più leggibile (richiede meno variabili)
- La forma piana e quella nidificata possono essere combinate
- Osservazione:
  Le sottointerrogazioni non possono contenere operatori insiemistici ("l'unione si fa solo al livello esterno"); la limitazione non è significativa

Interrogazioni nidificate

Gli operatori di confronto =, <,… si possono usare solo se la subquery restituisce **non più di una tupla**
---subquery "scalare"----

esempio: si vogliono elencare tutti gli impiegati che hanno uno stipendio superiore alla media degli stipendi di tutti gli impiegati

```
SELECT Nome, Stipendio FROM Impiegato
WHERE Stipendio >  (SELECT AVG(Stipendio)
                    FROM Impiegato);
```

stipendio_medio
51,6666666666667

Nome	Stipendio
Carlo	80
Lorenzo	73
Francesca	60

Interrogazioni nidificate

Se la subquery può restituire **più di un valore** si devono usare le forme

> **ANY**: la relazione > vale per almeno uno dei valori
> **ALL** : la relazione > vale per tutti i valori

Selezionare le città in cui almeno un impiegato guadagna piu' di 40

```
SELECT distinct città
FROM dip
WHERE nome = ANY  (SELECT dipart
                   FROM Impiegato
                   WHERE Stipendio > 40)
```

La forma = ANY equivale a IN

dipart
Amministrazione
Distribuzione
Direzione
Direzione
Produzione
Vendite

città
Milano
Roma
Torino

Selezionare gli impiegati con stipendio minimo

```
SELECT cognome
FROM Impiegato
WHERE Stipendio <= ALL (SELECT Stipendio FROM Impiegato)
```

Interrogazioni nidificate

impiegato

ID impiegato	Nome	Cognome	Dipart	Ufficio	Stipendio	premioprod	Mansione	Città	idcapo

dip

Nome	Indirizzo	Città

Query: elencare il nome e la mansione degli impiegati del dipartimento Amministrazione che hanno una qualche mansione svolta anche nel dipartimento Direzione

```
SELECT Nome, Mansione
FROM Impiegato
WHERE Dipart='amministrazione'
AND Mansione  IN  (SELECT Mansione
                   FROM Impiegato
                   WHERE Dipart='direzione');
```

Nome	Mansione
Giovanni	dirigente

max e min con query nidificate

Gli operatori aggregati max e min possono essere espressi tramite query nidificate.

Estrarre l'ordine con il massimo importo

Cliente(CODCLI, INDIRIZZO, P_IVA);
Ordine(CODORD,CODCLI,DATA,IMPORTO)
Dettaglio(CODORD,CODPROD,QTA)
Prodotto(CODPROD,NOME, PREZZO)

```
select CodOrd
from Ordine
where Importo in (select max(Importo)
                  from Ordine)

select CodOrd
from Ordine
where Importo >= all (select Importo
                      from Ordine)
```

Interrogazioni nidificate

impiegato

ID impiegato	Nome	Cognome	Dipart	Ufficio	Stipendio	premioprod	Mansione	Città	idcapo

Query: Trovare l'impiegato che ha lo stipendio piu' alto

```
select Cognome, Nome, max(Stipendio)
from Impiegato
where Dipart='Amministrazione'
```

NO

```
SELECT nome, cognome
FROM Impiegato
WHERE Stipendio =  (SELECT max(Stipendio)
                    FROM Impiegato)
SELECT nome, cognome
FROM Impiegato
WHERE Stipendio >= ALL (SELECT Stipendio
                        FROM Impiegato)
```

Nome	Cognome
Carlo	Rossi

Interrogazioni nidificate

Estrarre gli impiegati che hanno il nome uguale al cognome di un impiegato del dipartimento di Produzione

impiegato

ID impiegato	Nome	Cognome	Dipart	Ufficio	Stipendio	premioprod	Mansione	Città	idcapo

```
select nome, cognome, dipart
from Impiegato
where nome = any (select Cognome
                  from Impiegato
                  where Dipart = 'Produzione')
```

Nome	Cognome	dipart
Franco	Neri	Distribuzione

```
SELECT I1.Nome, I1.Cognome, I1.Dipart
FROM Impiegato AS I1 INNER JOIN Impiegato AS I2 ON
I1.Nome = I2.Cognome
WHERE I2.Dipart='Produzione';
```

Interrogazioni nidificate

impiegato

ID impiegato	Nome	Cognome	Dipart	Ufficio	Stipendio	premioprod	Mansione	Città	idcapo

dip

Nome	Indirizzo	Città

Estrarre i nomi dei dipartimenti in cui non lavorano persone di nome Rossi

```
select Nome
from Dip
where Nome <> all (select Dipart
                   from Impiegato
                   where Cognome = 'Rossi')
```

<> all equivale a not in

```
select Nome
from Dip
except
select Dipart
from Impiegato
where Cognome = 'Rossi'
```

Nome
produzione
distribuzione
ricerca

Interrogazioni nidificate

nome e reddito del padre di Franco

Persone Nome Età Reddito

Maternità Madre Figlio

Paternità Padre Figlio

```
select Nome, Reddito
from Persone
where Nome = (select Padre
              from Paternita
              where Figlio = 'Franco')
```

```
select Nome, Reddito
from Persone, Paternita
where Nome = Padre and Figlio = 'Franco'
```

Interrogazioni nidificate

Nome e reddito dei padri di persone che guadagnano più di 20 milioni

Persone Nome Età Reddito

Maternità Madre Figlio

Paternità Padre Figlio

```
select Nome, Reddito
from Persone
where Nome in (select Padre
               from Paternita
               where Figlio = any (select Nome
                                   from Persone
                                   where Reddito > 20))
```

```
select distinct P.Nome, P.Reddito
from Persone P, Paternita, Persone F
where P.Nome = Padre and Figlio = F.Nome
      and F.Reddito > 20
```

Interrogazioni nidificate

Nome e reddito dei padri di persone che guadagnano più di 20 milioni

Persone Nome Età Reddito

Maternità Madre Figlio

Paternità Padre Figlio

```
select Nome, Reddito
from Persone
where Nome in (select Padre
               from Paternita, Persone
               where Figlio = Nome
               and Reddito > 20)
```

```
select distinct P.Nome, P.Reddito
from Persone P, Paternita, Persone F
where P.Nome = Padre and Figlio = F.Nome
      and F.Reddito > 20
```

Interrogazioni nidificate

impiegato

ID impiegato	Nome	Cognome	Dipart	Ufficio	Stipendio	premioprod	Mansione	Città	idcapo

dip

Nome	Indirizzo	Città

è inoltre possibile selezionare **più di una colonna** tramite una sottointerrogazione; in tal caso è necessario apporre delle parentesi alla lista delle colonne a sinistra dell'operatore di confronto (Costruttore di tupla)

```
SELECT Nome, cognome
FROM Impiegato
WHERE (Mansione,Stipendio) = (SELECT mansione,
        Stipendio
        FROM Impiegato
        WHERE Cognome = 'Rossi' and nome='Mario');
```

Un altro esempio

STUDENTE(matricola, nome, cognome)

ESAME(studente, materia, voto, data)

Nome e cognome degli studenti che hanno superato l'esame di Fisica con 28 *oppure* 30.

```
select nome, cognome
from STUDENTE, ESAME
where matricola = studente
and materia = 'Fisica' and
(voto = 28 or voto = 30)
```

```
select nome, cognome
from STUDENTE
where matricola = ANY( select studente
                       from ESAME
                       where materia = 'Fisica'
                       and (voto = 28 or voto = 30))
```

Un altro esempio

STUDENTE(matricola, nome, cognome)

ESAME(studente, materia, voto, data)

Materie in cui almeno 5 studenti hanno conseguito 30.

```
select materia
from ESAME
where voto = 30
group by materia
having count(*) >= 5
```

Un altro esempio

STUDENTE(matricola, nome, cognome)

ESAME(studente, materia, voto, data)

Date in cui è stato sostenuto il maggior numero di esami.

```
select data
from ESAME
group by data
having count(*) >= all (select count(*)
        from ESAME
        group by data)
```

Esempio

Estrarre gli ordini di prodotti con un prezzo superiore a 100

Cliente(CODCLI, INDIRIZZO, P_IVA);
Ordine(CODORD,CODCLI,DATA,IMPORTO)
Dettaglio(CODORD,CODPROD,QTA)
Prodotto(CODPROD,NOME, PREZZO)c

```
select CodOrd
from Dettaglio
where CodProd = any (select CodProd
                     from Prodotto
                     where Prezzo > 100)
```

Equivalente a:

```
select CodOrd
from Dettaglio D, Prodotto P
where D.CodProd = P.CodProd
      and Prezzo > 100
```

Esempio

Estrarre i prodotti ordinati assieme al prodotto avente codice 'ABC'

Cliente(CODCLI, INDIRIZZO, P_IVA);
Ordine(CODORD,CODCLI,DATA,IMPORTO)
Dettaglio(CODORD,CODPROD,QTA)
Prodotto(CODPROD,NOME, PREZZO)c

– senza query nidificate:

```
select D1.CodProd
from Dettaglio D1, Dettaglio D2
where D1.CodOrd = D2.CodOrd and D2.CodProd = 'ABC'
```

– con una query nidificata:

```
select CodProd
from Dettaglio
where CodOrd = any (select CodOrd
                    from Dettaglio
                    where CodProd = 'ABC')
```

Esempio

Estrarre nome e indirizzo dei clienti che hanno emesso qualche ordine che comprende il prodotto "Pneumatico"

Cliente(CODCLI, NOME, INDIRIZZO, P_IVA);
Ordine(CODORD,CODCLI,DATA,IMPORTO)
Dettaglio(CODORD,CODPROD,QTA)
Prodotto(CODPROD,NOME, PREZZO)

```
select Nome, Indirizzo
from Cliente
where CodCli in select CodCli
                from Ordine
                where CodOrd in select CodOrd
                        from Dettaglio
                        where CodProd in select CodProd
                                from Prodotto
                                where Nome = 'Pneumatico'
```

Interrogazioni nidificate

Interpretazione semplice: l'interrogazione viene eseguita prima dell'interrogazione esterna. Il risultato può essere salvato in una tabella temporanea.

In questo modo l'interrogazione nidificata è eseguita una volta sola.

Negli esempi visti ogni subquery viene eseguita una volta per tutte ed il valore (o insieme di valori) è usato nella clausola WHERE della query esterna

Interrogazioni nidificate: query correlate

A volte la query più interna fa riferimento a una variabile definita nella query più esterna (*passaggio di binding*).

l'interpretazione semplice di prima non va più bene. Bisogna riconsiderare l'interpretazione standard: Si costruisce prima il prodotto cartesiano delle tabelle coinvolte dalla clausola from e poi a ciascuna riga si applica la clausola where, quindi le subqueries sono eseguite ripetutamente per ogni *tupla candidata* considerata nella valutazione della query esterna.

L'interrogazione interna viene eseguita una volta per ciascuna tupla dell'interrogazione esterna

Interrogazioni nidificate: query correlate

esempio: *si vogliono determinare gli impiegati che guadagnano più dello stipendio medio **del proprio dipartimento***

SE FOSSE: si vogliono elencare tutti gli impiegati che hanno uno stipendio superiore alla media degli stipendi **di tutti gli impiegati:**

```
SELECT Nome, Stipendio FROM Impiegato
WHERE Stipendio >  (SELECT AVG(Stipendio)
                    FROM Impiegato);
```

Interrogazioni nidificate: query correlate

è pertanto necessaria una query esterna che selezioni gli impiegati dalla relazione Impiegati in base ad un predicato su stipendio; tale query avrebbe la forma:

SELECT Nome, cognome FROM Impiegato
WHERE Stipendio > (*media degli stipendi nel dipartimento dell'impiegato candidato*);

è quindi necessaria una subquery che calcoli la media degli stipendi del dipartimento di ogni tupla candidata della relazione Impiegati; tale subquery avrebbe la forma:

(SELECT AVG(Stipendio) FROM Impiegato
WHERE Dipart=(*valore di Dipart nella tupla candidata*));

Interrogazioni nidificate: query correlate

ogni volta che la query esterna considera una tupla candidata, deve invocare la subquery e "passare" il dipartimento dell'impiegato in esame

la subquery calcola quindi la media degli stipendi nel dipartimento che è stato passato e restituisce tale valore alla query esterna

la query esterna può quindi confrontare lo stipendio dell'impiegato in esame con il valore restituito dalla subquery

Interrogazioni nidificate: query correlate

impiegato

ID impiegato	Nome	Cognome	Dipart	Ufficio	Stipendio	premiopred	Mansione	Città	idcapo

dip

Nome	Indirizzo	Città

esempio: si vogliono determinare gli impiegati che guadagnano più dello stipendio medio del proprio dipartimento; si richiede inoltre l'ordinamento delle tuple del risultato

```
SELECT Dipart, Nome, Stipendio
FROM Impiegato X
WHERE Stipendio > (SELECT AVG (Stipendio)
                   FROM Impiegato
                   WHERE X.Dipart = Dipart)
ORDER BY Dipart;
```

Interrogazioni nidificate: query correlate

Questo tipo di interrogazioni è chiamato **correlato**, perchè ogni esecuzione della subquery è correlata al valore di uno o più attributi delle tuple candidate nella interrogazione principale.

Per poter riferire le colonne delle tuple candidate nella query esterna si fa uso degli **alias** di relazione; un alias di relazione è definito nella query esterna e riferito nella query interna

Interrogazioni nidificate: query correlate

due sono i concetti principali che sono alla base della correlazione:

a) una subquery correlata fa riferimento ad un attributo selezionato dalla query esterna

b) se una subquery seleziona tuple dalla stessa relazione riferita dalla query esterna, è necessario definire un alias per tale relazione della query esterna;

la subquery deve usare l'alias per riferire i valori di attributo nelle tuple candidate nella query principale.

Interrogazioni nidificate: visibilità delle variabili

Una variabile è usabile solo nell'ambito della query in cui è definita o nell'ambito di una query nidificata (a qualsiasi livello) all'interno di essa.

Se una interrogazione possiede interrogazioni nidificate allo stesso livello, le variabili introdotte dalla clausola from di una query NON possono essere usate dall'altra query.

Interrogazioni nidificate: visibilità delle variabili

- scorretta:

```
select *
from Impiegato
where Dipart in (select Nome
                 from Dipartimento D1
                 where Nome = 'Produzione')
or
      Dipart in (select Nome
                 from Dipartimento D2
                 where D2.Citta = D1.Citta)
```

Interrogazioni nidificate: visibilità delle variabili

```
select *                                                    I1
from Impiegato
     where Dipart in (select D1.Nome                        D1
                      from Dip as D1
                      where D1.Città = 'Firenze')
     or
           Dipart in (select Nome                           D2
                      from Dip as D2
                      where D2.Nome = 'Produzione')
```

Interrogazioni nidificate

EXISTS e NOT EXISTS ammettono come parametro le query nidificate.

L'operatore logico EXISTS (sq) restituisce il valore Booleano **True** se la subquery restituisce almeno una tupla;

restituisce il valore Booleano **False** altrimenti

L'operatore NOT EXISTS (sq) restituisce il valore Booleano **True** se la subquery non restituisce alcuna tupla;

restituisce il valore Booleano **False** altrimenti

nota: la valutazione di predicati con questi due operatori non restituisce mai il valore Unknown

quantificatore esistenziale

Sintassi:

exists *SelectStar*

il predicato è vero se la query *SelectStar* restituisce un risultato non nullo
(sempre **select** * perché è irrilevante la proiezione)

Interrogazioni nidificate

impiegato

ID impiegato	Nome	Cognome	Dipart	Ufficio	Stipendio	premioprod	Mansione	Città	idcapo

dip

Nome	Indirizzo	Città

Mediante **EXISTS** (SELECT * …) è possibile verificare se il risultato di una subquery restituisce almeno una tupla

Dipartimenti in cui lavorano dei programmatori

```
SELECT nome
FROM dip
WHERE EXISTS (SELECT *
              FROM impiegato
              WHERE Mansione = 'Programmatore')
```

Facendo uso di NOT EXISTS il predicato è vero se la subquery non restituisce alcuna tupla

Interrogazioni nidificate: query correlate

impiegato

ID impiegato	Nome	Cognome	Dipart	Ufficio	Stipendio	premioprod	Mansione	Città	idcapo

dip

Nome	Indirizzo	Città

Dipartimenti con almeno un programmatore

```
SELECT nome
FROM dip S
WHERE EXISTS (SELECT *
              FROM Impiegato
              WHERE mansione = 'Programmatore'
              AND dipart = S.nome)
```

Adesso il risultato della query innestata dipende dal dipart specifico, e la semantica quindi diventa:
Per ogni tupla del blocco esterno, considera il valore di s.nome e risolvi la query innestata

Interrogazioni nidificate: query correlate

impiegato

ID impiegato	Nome	Cognome	Dipart	Ufficio	Stipendio	premioprod	Mansione	Città	idcapo

dip

Nome	Indirizzo	Città

È spesso possibile ricondursi a una forma "piatta", ma la cosa non è sempre così ovvia. Ad esempio, nell'esempio precedente si può anche scrivere

```
SELECT S.nome
FROM dip S, Impiegato I
WHERE S.nome = I.dipart
AND I.mansione = 'Programmatore'
```

•La forma innestata è "più procedurale" di quella piatta e, a seconda dei casi, può risultare più semplice da derivare.

interrogazioni nidificate: commenti

- In una subquery non si possono usare operatori insiemistici (UNION, INTERSECT e EXCEPT)
- una subquery può comparire solo come operando destro in un predicato
- in una subquery non ci possono essere le clausole group by e having
- le subquery possono comparire anche nelle espressioni della clausola select (non solo nella clausola where)

interrogazioni nidificate nelle espressioni

- Le subquery nella forma scalare (che restituisce cioè un solo valore) possono essere usate nelle espressioni:

Articoli(Art_Cod, Art_Prezzo, Descrizione, Art_IVA)

```
SELECT Art_Cod, Art_prezzo,
        Art_Prezzo – (SELECT min(Art_Prezzo)
                      FROM Articoli)  'Differenza'
FROM Articoli
```

Esempio

•I dati delle persone che hanno almeno un figlio

Persone(Nome, Età, Reddito)
Maternità(Madre, Figlio)
Paternità(Padre, Figlio)

```
select *
from Persone
where exists (select *
              from Paternita
              where Padre = Nome)
      or
       exists (select *
               from Maternita
               where Madre = Nome)
```

Esempio

I padri i cui figli guadagnano tutti più di venti milioni

Persone(Nome, Età, Reddito)
Maternità(Madre, Figlio)
Paternità(Padre, Figlio)

```
select distinct Padre
from Paternita Z
where not exists (
                  select *
                  from Paternita W, Persone
                  where W.Padre = Z.Padre
                        and W.Figlio = Nome
                        and Reddito <= 20)
```

Esempio

Persone che hanno omonimi (stesso nome e cognome, diverso CF)

Persone(CF, Cognome, Nome, Età, Reddito)

```
select *
from persone P
where exists (select *
              from Persone P1
              where P1.nome= P.nome
                    and P1.cognome= P.cognome
                    and P1.CF<> P.CF )
```

senza interrogazioni nidificate si poteva risolvere con un self join sulla tabella persone

Esempio

Persone che non hanno omonimi (stesso nome e cognome, diverso CF)

Persone(CF, Cognome, Nome, Età, Reddito)

```
select *
from persone P
where not exists (select *
                  from Persone P1
                  where P1.nome= P.nome
                  and P1.cognome= P.cognome
                  and P1.CF<> P.CF )
```

Esempio

In alternativa con costruttore di tupla:

Persone che non hanno omonimi (stesso nome e cognome, diverso CF)

```
select *
from Persone P
where (Nome,Cognome) not in (select Nome, Cognome
                             from Persone P1
                             where P1.CF<> P.CF )
```

Esempio

Estrarre tutti i clienti che hanno emesso più di un ordine nella stessa giornata:

Cliente(CODCLI, NOME, INDIRIZZO, P_IVA);
Ordine(CODORD,CODCLI,DATA,IMPORTO)
Dettaglio(CODORD,CODPROD,QTA)
Prodotto(CODPROD,NOME, PREZZO)

```
select CodCli
from Ordine O
where exists (select *
   from Ordine O1
   where O1.CodCli = O.CodCli
   and O1.Data = O.Data
   and O1.CodOrd <> O.CodOrd)
```

esercizio

Cliente(CODCLI, NOME, INDIRIZZO, P_IVA);
Ordine(CODORD,CODCLI,DATA,IMPORTO)
Dettaglio(CODORD,CODPROD,QTA)
Prodotto(CODPROD,NOME, PREZZO)

```
select *
from Cliente
where CodCli in (select CodCli
                 from Ordine O1
                 where CodOrd = 'AZ1020')
or
      CodCli in (select CodCli
                 from Ordine O2
                 where O2.Data = O1.Data)
```

esercizio

Cliente(CODCLI, NOME, INDIRIZZO, P_IVA);
Ordine(CODORD,CODCLI,DATA,IMPORTO)
Dettaglio(CODORD,CODPROD,QTA)
Prodotto(CODPROD,NOME, PREZZO)

```
select *
from Cliente
where CodCli in (select CodCli
                 from Ordine O1, Ordine O2
                 where CodOrd = 'AZ1020'
                       or O2.Data = O1.Data)
```

Esercizio

AEROPORTO(Città, Nazione,NumPiste)
VOLO(IdVolo,GiornoSett,CittàPart,OraPart, CittàArr,OraArr,TipoAereo)
AEREO(TipoAereo,NumPasseggeri,QtaMerci)

Gli aeroporti italiani che hanno solo voli interni.
a) con operatori insiemistici;

```
select CittàPart
from VOLO join AEROPORTO on CittàPart=Città
where Nazione = 'Italia'
EXCEPT
select CittàPart
from AEROPORTO as A1 join VOLO on A1.Città=CittàPart
join AEROPORTO as A2 on CittàArr=A2.Città
where (A1.Nazione=' Italia ' and A2.Nazione<>' Italia ' )
```

Esercizio

AEROPORTO(Città, Nazione,NumPiste)
VOLO(IdVolo,GiornoSett,CittàPart,OraPart, CittàArr,OraArr,TipoAereo)
AEREO(TipoAereo,NumPasseggeri,QtaMerci)

con un interrogazione nidificata con l'operatore not in

```
select CittàPart
from VOLO join AEROPORTO on CittàPart=Città
where Nazione= 'Italia' and CittàPart NOT IN
        ( select CittàPart
        from AEROPORTO as A1 join VOLO on
                A1.Città=CittàPart join AEROPORTO as A2 on
                        CittàArr=A2.Città
        where A1.Nazione='Italia' and A2.Nazione<> 'Italia' )
```

Esercizio

AEROPORTO(Città, Nazione,NumPiste)
VOLO(IdVolo,GiornoSett,CittàPart,OraPart, CittàArr,OraArr,TipoAereo)
AEREO(TipoAereo,NumPasseggeri,QtaMerci)

con un interrogazione nidificata con l'operatore not exists;

```
select CittàPart
from VOLO join AEROPORTO as A1 on CittàPart=Città
where Nazione= 'Italia' and not exists ( select *
   from VOLO join AEROPORTO as A2
        on A2.Città=CittàArr
   where A1.Città=CittàPart and A2.Nazione<>'Italia' )
```

Esercizio

AEROPORTO(Città, Nazione,NumPiste)
VOLO(IdVolo,GiornoSett,CittàPart,OraPart, CittàArr,OraArr,TipoAereo)
AEREO(TipoAereo,NumPasseggeri,QtaMerci)

con l'outer join e l'operatore di conteggio

```
select CittàPart
from AEROPORTO as A1 join VOLO on
        A1.Città=CittàPart left join AEROPORTO as A2 on
        (CittàArr=A2.Città and A2.Nazione='Italia')
where A1.Nazione='Italia'
group by CittàPart
having count (district A2.Nazione)= 1 )
```

Esercizio

AEROPORTO(Città, Nazione,NumPiste)
VOLO(IdVolo,GiornoSett,CittàPart,OraPart, CittàArr,OraArr,TipoAereo)
AEREO(TipoAereo,NumPasseggeri,QtaMerci)

Le città che sono servite dall'aereo caratterizzato dal massimo numero di passeggeri;

```
select CittàPart
from VOLO join AEREO on VOLO.TipoAereo=AEREO.TipoAereo
where NumPasseggeri= (select max( NumPasseggeri )
                      from AEREO )
union
select CittàArr
from VOLO join AEREO on VOLO.TipoAereo=AEREO.TipoAereo
where NumPasseggeri= (select max( NumPasseggeri )
                      from AEREO)
```

Esercizio

CANTANTE(Nome, Canzone)
AUTORE(Nome, Canzone)

Estrarre i cantanti che hanno eseguito solo canzoni di cui sono anche autori

```
select Nome
from Cantante
where Nome in (select Nome
          from Cantante C
          where Nome in (select Nome
               from Autore
               where Autore.Canzone=C.Canzone))
```

Esercizio

CANTANTE(Nome, Canzone)
AUTORE(Nome, Canzone)

Estrarre i cantanti che hanno eseguito solo canzoni di cui sono anche autori

```
select Nome
from Cantante
except
select Nome
from Cantante C
where Nome not in (select Nome
          from Autore
          where Autore.Canzone=C.Canzone )
```