



fondo
sociale europeo

DISPENSA

SGQS modulo rev 01 09/09/2019

MATERIA:
BASI DI DATI

ARGOMENTO:
ISTRUZIONI NIDIFICATE



per una crescita intelligente,
sostenibile ed inclusiva
www.regenone.piemonte.it/europa2020
INIZIATIVA CO-FINANZIATA CON FSE



Linguaggio SQL: fondamenti

Interrogazioni nidificate

Interrogazioni nidificate

- Introduzione
- Operatore IN
- Operatore NOT IN
- Costruttore di tupla
- Operatore EXISTS
- Operatore NOT EXISTS
- Correlazione tra interrogazioni
- Operazione di divisione
- Table functions



Interrogazioni nidificate

Introduzione

Introduzione

- ▷ Un'interrogazione nidificata è un'istruzione SELECT contenuta all'interno di un'altra interrogazione
 - la nidificazione di interrogazioni permette di suddividere un problema complesso in sottoproblemi più semplici
- ▷ È possibile introdurre istruzioni SELECT
 - in un predicato nella clausola WHERE
 - in un predicato nella clausola HAVING
 - nella clausola FROM

DB forniture prodotti (1/2)

- P (CodP, NomeP, Colore, Taglia, Magazzino)
- F (CodF, NomeF, NSoci, Sede)
- FP (CodF, CodP, Qta)

DB forniture prodotti (2/2)

P

<u>CodP</u>	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

<u>CodF</u>	<u>CodP</u>	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

F

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

Interrogazioni nidificate (n.1)

- ▷ Trovare il codice dei fornitori che hanno sede nella stessa città di F1

- ▷ La formulazione mediante interrogazioni nidificate consente di separare il problema in due sottoproblemi
 - sede del fornitore F1
 - codici dei fornitori con la stessa sede

Interrogazioni nidificate (n.1)

- ▷ Trovare il codice dei fornitori che hanno sede nella stessa città di F1

```
SELECT CodF  
FROM F  
WHERE Sede = (SELECT Sede  
              FROM F  
              WHERE CodF='F1');
```



- ▷ È possibile utilizzare '=' esclusivamente se è noto a priori che il risultato della SELECT nidificata è sempre un solo valore

Formulazione equivalente (n.1)

- Trovare il codice dei fornitori che hanno sede nella stessa città di F1
- È possibile definire una formulazione equivalente con il join

Formulazione equivalente

- ▷ La formulazione equivalente con il join è caratterizzata da
- Clausola FROM contenente le tabelle referenziate nelle FROM di tutte le SELECT
 - Opportune condizioni di join nella clausola WHERE
 - Eventuali predicati di selezione aggiunti nella clausola WHERE

Clausola FROM (n.1)

- ▷ Trovare il codice dei fornitori che hanno sede nella stessa città di F1

```
SELECT CodF  
FROM F
```

```
WHERE Sede = (SELECT Sede  
FROM F  
WHERE CodF='F1');
```

Clausola FROM (n.1)

- ▷ Trovare il codice dei fornitori che hanno sede nella stessa città di F1

```
SELECT ...  
FROM F AS FX, F AS FY
```

...

Condizione di join (n.1)

- ▷ Trovare il codice dei fornitori che hanno sede nella stessa città di F1

```
SELECT CodF  
FROM F  
WHERE Sede = (SELECT Sede  
              FROM F  
              WHERE CodF='F1');
```

Condizione di join (n.1)

- ▷ Trovare il codice dei fornitori che hanno sede nella stessa città di F1

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

...

Predicato di selezione (n.1)

- ▷ Trovare il codice dei fornitori che hanno sede nella stessa città di F1

```
SELECT CodF  
FROM F  
WHERE Sede = (SELECT Sede  
              FROM F  
              WHERE CodF='F1');
```

Clausola SELECT (n.1)

- ▷ Trovare il codice dei fornitori che hanno sede nella stessa città di F1

```
SELECT FY.CodF  
FROM F AS FX, F AS FY  
WHERE FX.Sede=FY.Sede AND  
FX.CodF='F1';
```

Formulazione equivalente (n.2)

- ▷ Trovare il codice dei fornitori il cui numero di soci è minore del numero massimo di soci

```
SELECT CodF  
FROM F  
WHERE NSoci < (SELECT MAX(NSoci)  
                  FROM F);
```

- ▷ È possibile definire una formulazione equivalente con il join?

Formulazione equivalente (n.2)

- ▷ Trovare il codice dei fornitori il cui numero di soci è minore del numero massimo di soci

```
SELECT CodF  
FROM F  
WHERE NSoci < (SELECT MAX(NSoci)  
                  FROM F);
```

- ▷ Non è possibile definire una formulazione equivalente con il join



Interrogazioni nidificate

Operatore IN

Operatore IN (n.1)

- Trovare il nome dei fornitori che forniscono il prodotto P2
- Scomposizione del problema in due sottoproblemi
 - codici dei fornitori del prodotto P2
 - nome dei fornitori aventi quei codici

Operatore IN (n.1)

- ▷ Trovare il nome dei fornitori che forniscono il prodotto P2

FP

CodF	<u>CodP</u>	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



CodF
F1
F2
F3

```
SELECT CodF  
FROM FP  
WHERE CodP='P2'
```

*Codici
dei
fornitori
di P2*

Operatore IN (n.1)

- ▷ Trovare il nome dei fornitori che forniscono il prodotto P2

```
SELECT NomeF  
FROM F  
WHERE CodF    (SELECT CodF  
                  FROM FP  
                  WHERE CodP='P2')
```

Operatore IN (n.1)

- ▷ Trovare il nome dei fornitori che forniscono il prodotto P2

```
SELECT NomeF  
FROM F  
WHERE CodF ○ (SELECT CodF  
                FROM FP  
                WHERE CodP='P2')
```



Operatore IN (n.1)

- ▷ Trovare il nome dei fornitori che forniscono il prodotto P2

```
SELECT NomeF  
FROM F  
WHERE CodF IN (SELECT CodF  
                FROM FP  
                WHERE CodP='P2');
```

Appartenenza all'insieme



Operatore IN

- ▷ Esprime il concetto di appartenenza ad un insieme di valori
 - *NomeAttributo IN (InterrogazioneNidificata)*
- ▷ Permette di scrivere l'interrogazione
 - scomponendo il problema in sottoproblemi
 - seguendo un procedimento “bottom-up”

Formulazione equivalente

- ▷ La formulazione equivalente con il join è caratterizzata da
- clausola FROM contenente le tabelle referenziate nelle FROM di tutte le SELECT
 - opportune condizioni di join nella clausola WHERE
 - eventuali predicati di selezione aggiunti nella clausola WHERE

Operatore IN (n.1)

- ▷ Trovare il nome dei fornitori che forniscono il prodotto P2

```
SELECT NomeF  
FROM F  
WHERE CodF IN (SELECT CodF  
                FROM FP  
                WHERE CodP='P2');
```

Formulazione equivalente (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';
```

Operatore IN (n.2)

- Trovare il nome dei fornitori che forniscono almeno un prodotto rosso
- Scomposizione del problema in sottoproblemi
 - codici dei prodotti rossi
 - codici dei fornitori di quei prodotti
 - nomi dei fornitori aventi quei codici

Operatore IN (n.2)

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

```
SELECT NomeF  
FROM F  
WHERE CodF IN (SELECT CodF  
                FROM FP  
                WHERE CodP IN (SELECT CodP  
                                FROM P  
                                WHERE Colore='Rosso'));
```

Formulazione equivalente (n.2)

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

```
SELECT NomeF  
FROM F  
WHERE CodF IN (SELECT CodF  
                FROM FP  
                WHERE CodP IN (SELECT CodP  
                                FROM P  
                                WHERE Colore='Rosso'));
```

Clausola FROM (n.2)

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

```
SELECT NomeF  
FROM F  
WHERE CodF IN (SELECT CodF  
                FROM FP  
                WHERE CodP IN (SELECT CodP  
                                FROM P  
                                WHERE Colore='Rosso'));
```

Clausola FROM (n.2)

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

SELECT ...

FROM F, FP, P

...

Condizioni di join (n.2)

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

```
SELECT NomeF      1  
FROM F  
WHERE CodF IN (SELECT CodF  
                FROM FP  
                WHERE CodP IN (SELECT CodP  
                                FROM P  
                                WHERE Colore='Rosso'));
```

Condizioni di join (n.2)

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

SELECT ...

FROM F, FP, P

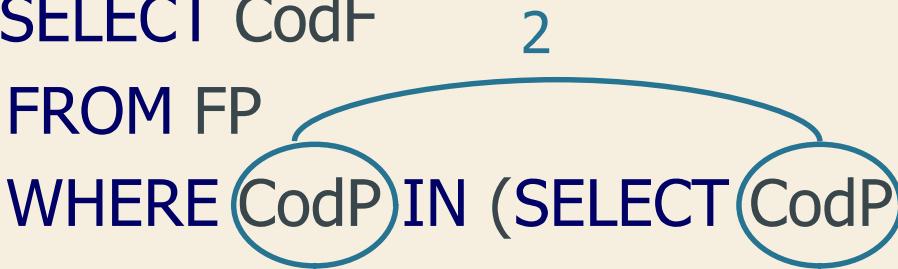
WHERE P.CodF=F.CodF

1

Condizioni di join (n.2)

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

```
SELECT NomeF  
FROM F  
WHERE CodF IN (SELECT CodF  
                FROM FP  
                WHERE CodP IN (SELECT CodP  
                                FROM P  
                                WHERE Colore='Rosso'));
```



The diagram illustrates the structure of the SQL query. It shows two instances of the variable 'CodP' circled in blue. A curved arrow originates from the first circled 'CodP' and points to the second, with the number '2' written above the arrow, indicating that there are two occurrences of this variable within the nested subquery.

Condizioni di join (n.2)

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

SELECT ...

FROM F, FP, P

WHERE P.CodF=F.CodF AND

FP.CodP=P.CodP

2

...

Predicato di selezione (n.2)

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

```
SELECT NomeF  
FROM F  
WHERE CodF IN (SELECT CodF  
                FROM FP  
                WHERE CodP IN (SELECT CodP  
                                FROM P  
                                WHERE Colore='Rosso'));
```

Clausola SELECT (n.2)

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

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

Esempio complesso (n.3)

- ▷ Trovare il nome dei fornitori che forniscono almeno un prodotto fornito da fornitori di prodotti rossi

Esempio complesso (n.3)

- ▷ Trovare il nome dei fornitori che forniscono almeno un prodotto fornito da fornitori di prodotti rossi



Esempio complesso (n.3)

- Trovare il nome dei fornitori che forniscono almeno un prodotto fornito da fornitori di prodotti rossi
- La formulazione con il join è difficile
 - è più semplice scomporre il problema in sottoproblemi mediante interrogazioni nidificate

Esempio complesso (n.3)

- ▷ Trovare il nome dei fornitori che forniscono almeno un prodotto fornito da fornitori di prodotti rossi

*Codici dei
prodotti rossi*

{
SELECT CodP
FROM P
WHERE Colore='Rosso'

Esempio complesso (n.3)

- ▷ Trovare il nome dei fornitori che forniscono almeno un prodotto fornito da fornitori di prodotti rossi

```
SELECT CodF  
FROM FP  
WHERE CodP IN  
(SELECT CodP  
FROM P  
WHERE Colore='Rosso')
```

*Codici dei fornitori
di prodotti rossi*

Esempio complesso (n.3)

- ▷ Trovare il nome dei fornitori che forniscono almeno un prodotto fornito da fornitori di prodotti rossi

*Codici dei prodotti
forniti da fornitori
di prodotti rossi*

```
SELECT CodP
FROM FP
WHERE CodF IN
  (SELECT CodF
   FROM FP
   WHERE CodP IN
     (SELECT CodP
      FROM P
      WHERE Colore='Rosso'))
```

Esempio complesso (n.3)

```
SELECT CodF  
FROM FP  
WHERE CodP IN  
(SELECT CodP  
FROM FP  
WHERE CodF IN  
(SELECT CodF  
FROM FP  
WHERE CodP IN  
(SELECT CodP  
FROM P  
WHERE Colore='Rosso')))
```

*Codici dei fornitori
di prodotti forniti
da fornitori di
prodotti rossi*

Interrogazione completa (n.3)

```
SELECT NomeF  
FROM F  
WHERE CodF IN  
    (SELECT CodF  
     FROM FP  
     WHERE CodP IN  
         (SELECT CodP  
          FROM FP  
          WHERE CodF IN  
              (SELECT CodF  
               FROM FP  
               WHERE CodP IN  
                   (SELECT CodP  
                    FROM P  
                    WHERE Colore='Rosso'))));
```

Formulazione con il join (n.3)



Formulazione con il join (n.3)

```
SELECT NomeF  
FROM F  
WHERE CodF IN  
    (SELECT CodF  
     FROM FP  
     WHERE CodP IN  
         (SELECT CodP  
          FROM FP  
          WHERE CodF IN  
              (SELECT CodF  
               FROM FP  
               WHERE CodP IN  
                   (SELECT CodP  
                    FROM P  
                    WHERE Colore='Rosso'))));
```

Clausola FROM (n.3)

```
SELECT NomeF  
FROM F  
WHERE CodF IN  
    (SELECT CodF  
     FROM FP  
     WHERE CodP IN  
         (SELECT CodP  
          FROM FP  
          WHERE CodF IN  
              (SELECT CodF  
               FROM FP  
               WHERE CodP IN  
                   (SELECT CodP  
                    FROM P  
                    WHERE Colore='Rosso'))));
```

Clausola FROM (n.3)

```
SELECT NomeF  
FROM F  
WHERE CodF IN  
    (SELECT CodF  
     FROM FP) FPA  
    WHERE CodP IN  
        (SELECT CodP  
         FROM FP) FPB  
        WHERE CodF IN  
            (SELECT CodF  
             FROM FP) FPC  
            WHERE CodP IN  
                (SELECT CodP  
                 FROM P)  
                 WHERE Colore='Rosso'))));
```

Clausola FROM (n.3)

SELECT ...

FROM F, FP AS FPA, FP AS FPB, FP AS FPC, P

...

Condizioni di join (n.3)

```
SELECT NomeF  
FROM F  
WHERE CodF IN  
      (SELECT CodF  
       FROM FP  
       WHERE CodP IN  
             (SELECT CodP  
              FROM FP  
              WHERE CodF IN  
                    (SELECT CodF  
                     FROM FP  
                     WHERE CodP IN  
                           (SELECT CodP  
                            FROM P  
                            WHERE Colore='Rosso'))));
```

The diagram illustrates the nesting of joins. It shows two occurrences of the column 'CodF'. The first 'CodF' is part of a WHERE clause in the outer query. An arrow labeled '1' points from this 'CodF' to a second 'CodF' which is part of a subquery's SELECT statement. This subquery itself contains a 'CodF' column, which is further refined by a WHERE clause that references a 'CodP' column from another subquery. This 'CodP' is highlighted with a red circle and an arrow labeled 'FPA' points to it from the 'FP' table in the outer query.

Condizioni di join (n.3)

SELECT ...

FROM F, FP AS FPA, FP AS FPB, FP AS FPC, P
WHERE F.CodF=FPA.CodF

1

...

Condizioni di join (n.3)

```
SELECT NomeF
```

```
FROM F
```

```
WHERE CodF IN
```

```
(SELECT CodF  
FROM FP  
WHERE CodP IN
```

```
2
```

```
(SELECT CodP  
FROM FP  
WHERE CodP IN
```

```
WHERE CodF IN
```

```
(SELECT CodF
```

```
FROM FP
```

```
WHERE CodP IN
```

```
(SELECT CodP
```

```
FROM P
```

```
WHERE Colore='Rosso'))));
```

Condizioni di join (n.3)

SELECT ...

FROM F, FP AS FPA, FP AS FPB, FP AS FPC, P

WHERE F.CodF=FPA.CodF AND

FPA.CodP=FPB.CodP

2

...

Condizioni di join (n.3)

```
SELECT NomeF  
FROM F  
WHERE CodF IN  
    (SELECT CodF  
     FROM FP  
     WHERE CodP IN  
         (SELECT CodP  
          FROM FP  
          WHERE CodF IN  
              (SELECT CodF  
               FROM FP  
               WHERE CodP IN  
                   (SELECT CodP  
                    FROM P  
                    WHERE Colore='Rosso'))));
```

The diagram shows a hierarchical structure of nested queries. At the top level, there is a large oval containing the text 'CodF IN'. Inside this oval, there are three smaller ovals: one at the top labeled 'FPB', one in the middle labeled '3', and one at the bottom labeled 'FPC'. Red arrows point from the word 'FP' in the 'FROM FP' clause of each nested query to the corresponding oval: one arrow from 'FP' in the innermost query to 'FPC', another from 'FP' in the middle query to '3', and a third from 'FP' in the outermost query to 'FPB'.

Condizioni di join (n.3)

SELECT ...

FROM F, FP AS FPA, FP AS FPB, FP AS FPC, P

WHERE F.CodF=FPA.CodF AND

FPA.CodP=FPB.CodP AND

FPB.CodF=FPC.CodF

3

...

Condizioni di join (n.3)

```
SELECT NomeF  
FROM F  
WHERE CodF IN  
    (SELECT CodF  
     FROM FP  
     WHERE CodP IN  
         (SELECT CodP  
          FROM FP  
          WHERE CodF IN  
              (SELECT CodF  
               FROM FP  
               WHERE CodP IN  
                   (SELECT CodP  
                    FROM P  
                    WHERE Colore='Rosso'))));
```

The diagram illustrates a self-join operation on the table FP. It shows two instances of the table FP joined together. A red arrow labeled "FPC" points from the inner join condition "CodF" to the outer join condition "CodF", indicating a self-referencing foreign key relationship. A blue circle labeled "4" is placed near the join line, indicating the cardinality of the join.

Condizioni di join (n.3)

SELECT ...

FROM F, FP AS FPA, FP AS FPB, FP AS FPC, P

WHERE F.CodF=FPA.CodF AND

FPA.CodP=FPB.CodP AND

FPB.CodF=FPC.CodF AND

FPC.CodP=P.CodP

...

4

Predicato di selezione (n.3)

```
SELECT NomeF  
FROM F  
WHERE CodF IN  
    (SELECT CodF  
     FROM FP  
     WHERE CodP IN  
         (SELECT CodP  
          FROM FP  
          WHERE CodF IN  
              (SELECT CodF  
               FROM FP  
               WHERE CodP IN  
                   (SELECT CodP  
                    FROM P  
                    WHERE Colore='Rosso'))));
```

Predicato di selezione (n.3)

```
SELECT ...  
FROM F, FP AS FPA, FP AS FPB, FP AS FPC, P  
WHERE F.CodF=FPA.CodF AND  
      FPA.CodP=FPB.CodP AND  
      FPB.CodF=FPC.CodF AND  
      FPC.CodP=P.CodP AND  
      Colore='Rosso'
```

Clausola SELECT (n.3)

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



Interrogazioni nidificate

Operatore **NOT IN**

Concetto di esclusione (n.1)

- ▷ Trovare il nome dei fornitori che *non* forniscono il prodotto P2
- è possibile esprimere l'interrogazione mediante il join?

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

Soluzione errata (n.1)

- ▷ Trovare il nome dei fornitori che *non* forniscono il prodotto P2
- non è possibile esprimere l'interrogazione mediante il join

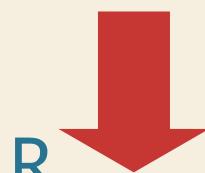
~~SELECT NomeF
FROM F, FP
WHERE F.CodF=FP.CodF
AND CodP<>'P2';~~

Soluzione errata (n.1)

▷ Trovare il nome dei fornitori che *non* forniscono il prodotto P2

F

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



<u>NomeF</u>
Andrea
Luca
Gabriele

FP

<u>CodF</u>	<u>CodP</u>	<u>Qta</u>
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

Soluzione errata (n.1)

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

⇒ A che interrogazione corrisponde?

Soluzione errata (n.1)

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



Trovare il nome dei fornitori che forniscono
almeno un prodotto diverso da P2

Concetto di esclusione (n.1)

- Trovare il nome dei fornitori che *non* forniscono il prodotto P2
- Occorre escludere dal risultato
 - i fornitori che forniscono il prodotto P2

Concetto di esclusione (n.1)

- ▷ Trovare il nome dei fornitori che *non* forniscono il prodotto P2

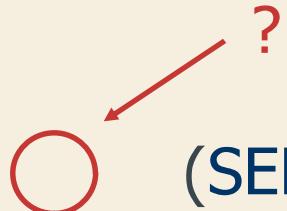
```
SELECT CodF  
FROM FP  
WHERE CodP='P2'
```

} *Codici dei fornitori
che forniscono P2*

Concetto di esclusione (n.1)

- ▷ Trovare il nome dei fornitori che *non* forniscono il prodotto P2

```
SELECT NomeF  
FROM F  
WHERE CodF
```



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

*Codici dei fornitori
che forniscono P2*

Operatore NOT IN (n.1)

- ▷ Trovare il nome dei fornitori che *non* forniscono il prodotto P2

```
SELECT NomeF  
FROM F  
WHERE CodF NOT IN (SELECT CodF  
                     FROM FP  
                     WHERE CodP='P2');
```

Non appartiene

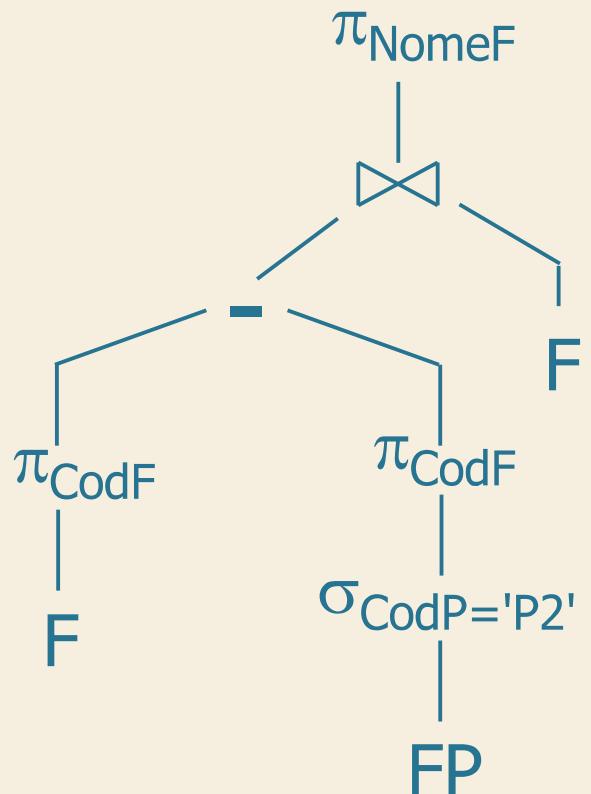
*Codici dei fornitori
che forniscono P2*

Operatore NOT IN

- ▷ Esprime il concetto di esclusione da un insieme di valori
 - *NomeAttributo NOT IN (InterrogazioneNidificata)*
- ▷ Richiede di individuare in modo appropriato *l'insieme da escludere*
 - definito dall'interrogazione nidificata

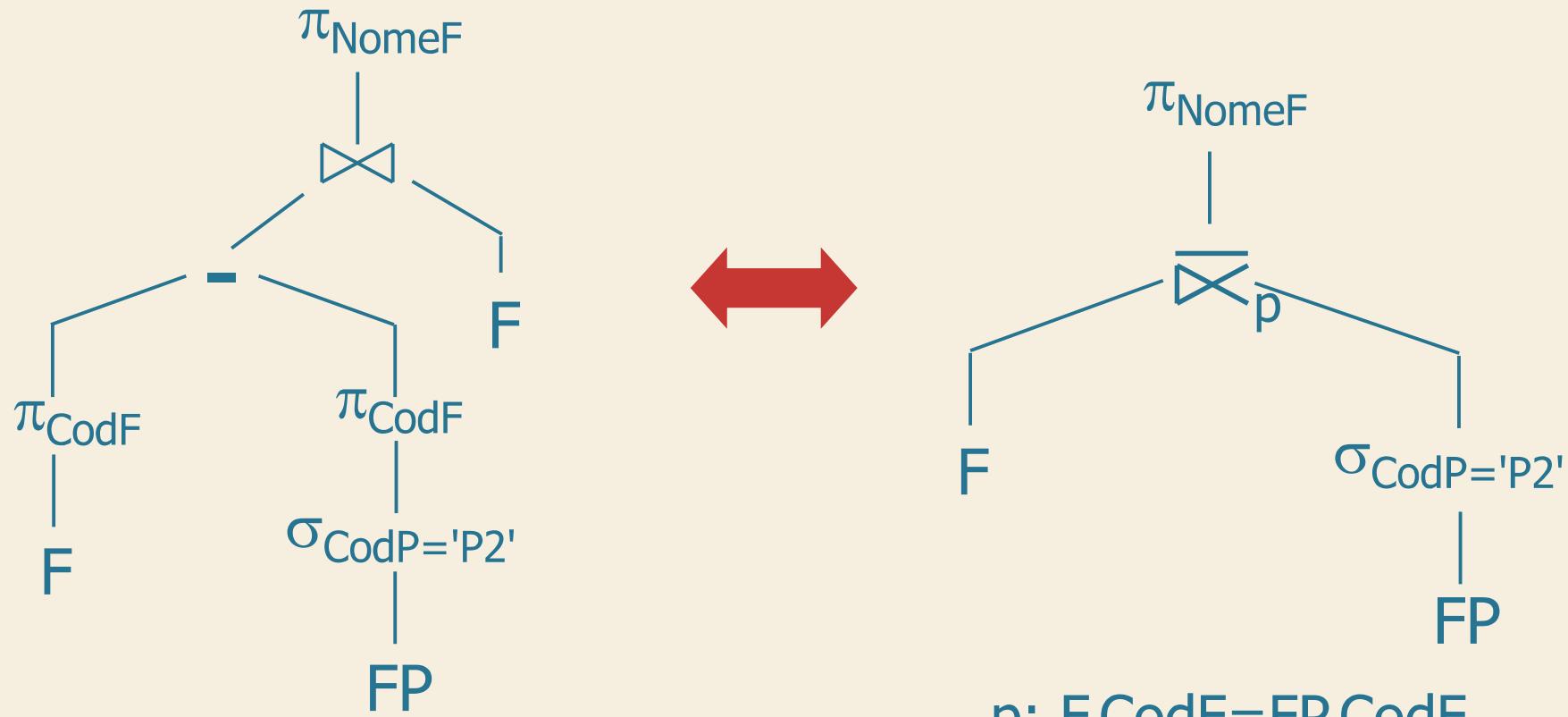
NOT IN e algebra relazionale (n.1)

▷ Trovare il nome dei fornitori che *non* forniscono il prodotto P2



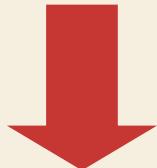
NOT IN e algebra relazionale (n.1)

▷ Trovare il nome dei fornitori che *non* forniscono il prodotto P2



Operatore NOT IN (n.2)

- ▷ Trovare il nome dei fornitori che forniscono *solo* il prodotto P2



Trovare il nome dei fornitori di P2 che non hanno mai fornito prodotti diversi da P2

- ▷ Insieme da escludere
- fornitori di prodotti diversi da P2

Operatore NOT IN (n.2)

- ▷ Trovare il nome dei fornitori che forniscono solo il prodotto P2

```
SELECT CodF  
FROM FP  
WHERE CodP <> 'P2'
```

*Codici dei fornitori
che forniscono
almeno un
prodotto diverso
da P2*

Operatore NOT IN (n.2)

- ▷ Trovare il nome dei fornitori che forniscono solo il prodotto P2

```
SELECT NomeF  
FROM F  
WHERE CodF NOT IN (SELECT CodF  
                     FROM FP  
                     WHERE CodP<>'P2')
```

...

Operatore NOT IN (n.2)

- ▷ Trovare il nome dei fornitori che forniscono solo il prodotto P2

```
SELECT NomeF  
FROM F, FP  
WHERE F.CodF NOT IN (SELECT F.CodF  
                      FROM FP  
                      WHERE CodP<>'P2')  
      AND F.CodF=FP.CodF;
```

Soluzione alternativa (n.2)

- ▷ Trovare il nome dei fornitori che forniscono solo il prodotto P2

```
SELECT NomeF  
FROM F  
WHERE F.CodF NOT IN (SELECT CodF  
                      FROM FP  
                      WHERE CodP<>'P2')  
      AND F.CodF IN (SELECT CodF  
                      FROM FP);
```

Operatore NOT IN (n.3)

- ▷ Trovare il nome dei fornitori che *non* forniscono prodotti rossi

Operatore NOT IN (n.3)

- Trovare il nome dei fornitori che *non* forniscono prodotti rossi
- Insieme da escludere?
 - i fornitori di prodotti rossi, identificati dal loro codice

Operatore NOT IN (n.3)

- ▷ Trovare il nome dei fornitori che *non* forniscono prodotti rossi

*Codici dei fornitori
di prodotti rossi*

{
(SELECT CodF
FROM FP
WHERE CodP IN (SELECT CodP
FROM P
WHERE Colore='Rosso'))

Operatore NOT IN (n.3)

- ▷ Trovare il nome dei fornitori che *non* forniscono prodotti rossi

```
SELECT NomeF  
FROM F  
WHERE CodF NOT IN (SELECT CodF  
                     FROM FP  
                     WHERE CodP IN (SELECT CodP  
                                     FROM P  
                                     WHERE Colore='Rosso'));
```

Alternativa (corretta?) (n.3)

- ▷ Trovare il nome dei fornitori che *non* forniscono prodotti rossi

*Codici dei fornitori
che forniscono
almeno un
prodotto non rosso*

```
SELECT CodF
FROM FP
WHERE CodP NOT IN (SELECT CodP
                      FROM P
                     WHERE Colore='Rosso')
```

Alternativa (corretta?) (n.3)

- ▷ Trovare il nome dei fornitori che *non* forniscono prodotti rossi

```
SELECT NomeF  
FROM F  
WHERE CodF IN (SELECT CodF  
                FROM FP  
                WHERE Colore='Rosso'))  
      WHERE CodP NOT IN (SELECT CodP  
                          FROM P  
                          WHERE Colore='Rosso'));
```

Alternativa errata (n.3)

- ▷ Trovare il nome dei fornitori che *non* forniscono prodotti rossi

```
SELECT NomeF  
FROM F  
WHERE CodF IN (SELECT CodF  
                FROM FP  
                WHERE CodP NOT IN (SELECT CodP  
                                    FROM P  
                                    WHERE Colore='Rosso'));
```

Alternativa errata (n.3)

- ▷ Trovare il nome dei fornitori che *non* forniscono prodotti rossi

```
SELECT NomeF  
FROM F  
WHERE CodF IN (SELECT CodF  
                FROM FP  
                WHERE CodP NOT IN (SELECT CodP  
                                    FROM P  
                                    WHERE Colore='Rosso'));
```

*Codici dei fornitori
di prodotti
non rossi*

Alternativa errata (n.3)

▷ Trovare il nome dei fornitori che *non* forniscono prodotti rossi

P



<u>CodP</u>	<u>NomeP</u>	<u>Colore</u>	<u>Taglia</u>	<u>Magazzino</u>
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

F



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

B
M
G

FP

<u>CodF</u>	<u>CodP</u>	<u>Qta</u>
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

Alternativa errata (n.3)

▷ Trovare il nome dei fornitori che *non* forniscono prodotti rossi

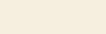
P

<u>CodP</u>	<u>NomeP</u>	<u>Colore</u>	<u>Taglia</u>	<u>Magazzino</u>
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



F

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



B
M
G

FP

<u>CodF</u>	<u>CodP</u>	<u>Qta</u>
F1	P1	300
F1	P2	200
F1	P3	400
F1	P4	200
F1	P5	100
F1	P6	100
F2	P1	300
F2	P2	400
	P2	200
	P3	200
	P4	300
	P5	400



Alternativa errata (n.3)

- ▷ Trovare il nome dei fornitori che *non* forniscono prodotti rossi

```
SELECT NomeF  
FROM F  
WHERE CodF IN (SELECT CodF  
                FROM FP  
                WHERE CodP NOT IN (SELECT CodP  
                                    FROM P  
                                    WHERE Colore='Rosso'));
```

- ▷ L'insieme di elementi da escludere non è corretto



Interrogazioni nidificate

Costruttore di tupla

Costruttore di tupla

- ▷ Permette di definire la struttura temporanea di una tupla
 - si elencano gli attributi che ne fanno parte tra ()

(NomeAttributo₁, NomeAttributo₂, ...)

- ▷ Permette di estendere il poter espressivo degli operatori IN e NOT IN

Esempio (n.1)

VIAGGIO (CodV, LuogoPartenza, LuogoArrivo,
OraPartenza, OraArrivo)

- Trovare le coppie luogo di partenza e luogo di arrivo per cui nessun viaggio dura più di 2 ore

Esempio (n.1)

VIAGGIO (CodV, LuogoPartenza, LuogoArrivo,
OraPartenza, OraArrivo)

- ▷ Trovare le coppie luogo di partenza e luogo di arrivo per cui nessun viaggio dura più di 2 ore

```
SELECT LuogoPartenza, LuogoArrivo  
FROM VIAGGIO  
WHERE (LuogoPartenza, LuogoArrivo) NOT IN  
      (SELECT LuogoPartenza, LuogoArrivo  
       FROM VIAGGIO  
       WHERE OraArrivo-OraPartenza>2);
```

*Costruttore
di tupla*



Interrogazioni nidificate

Operatore **EXISTS**

Operatore EXISTS (n.1)

▷ Trovare il nome dei fornitori del prodotto P2



*Trovare il nome dei fornitori per cui esiste
una fornitura del prodotto P2*

Condizione di correlazione (n.1)

▷ Trovare il nome dei fornitori del prodotto P2

```
SELECT NomeF  
FROM F  
WHERE EXISTS (SELECT *  
              FROM FP  
              WHERE CodP='P2'  
              AND FP.CodF=F.CodF );
```

Condizione di correlazione

Funzionamento di EXISTS (n.1)

➤ Trovare il nome dei fornitori del prodotto P2

F

CodF	NomeF	NSoci	Città
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

SELECT *

FROM FP

WHERE CodP='P2'

AND FP.CodF=[F1']



Valore di CodF nella
riga corrente di F

Funzionamento di EXISTS (n.1)

➤ Trovare il nome dei fornitori del prodotto P2

F

CodF	NomeF	NSoci	Città
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

➤ Il predicato con EXISTS è
vero per F1 poiché esiste una
fornitura di P2 per F1

- F1 fa parte del risultato
dell'interrogazione

Funzionamento di EXISTS (n.1)

➤ Trovare il nome dei fornitori del prodotto P2

F

<u>CodF</u>	NomeF	NSoci	Città
F1	Andrea	2	Torino
F2	Luca	1	Milano
F3	Antonio	3	Milano
F4	Gabriele	2	Torino
F5	Matteo	3	Venezia

FP

<u>CodF</u>	<u>CodP</u>	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

➤ Il predicato con EXISTS è falso per F4 poiché non esiste una fornitura di P2 per F4

- F4 non fa parte del risultato dell'interrogazione

Risultato dell'interrogazione (n.1)

▷ Trovare il nome dei fornitori del prodotto P2

R
NomeF
Andrea
Luca
Antonio

Predicati con EXISTS

▷ Il predicato contenente EXISTS è

- vero se l'interrogazione interna restituisce almeno una tupla
- falso se l'interrogazione interna restituisce l'insieme vuoto

Predicati con EXISTS

- ▷ Il predicato contenente EXISTS è
 - vero se l'interrogazione interna restituisce almeno una tupla
 - falso se l'interrogazione interna restituisce l'insieme vuoto
- ▷ Nell'interrogazione interna a EXISTS, la clausola SELECT è obbligatoria, ma irrilevante, perché gli attributi non sono visualizzati
- ▷ La condizione di correlazione lega l'esecuzione dell'interrogazione interna al valore di attributi della tupla corrente nell'interrogazione esterna

Visibilità degli attributi

- ▷ Un'interrogazione nidificata può far riferimento ad attributi definiti in interrogazioni più esterne
- ▷ Un'interrogazione non può far riferimento ad attributi referenziati
 - in un'interrogazione nidificata al suo interno
 - in un'interrogazione allo stesso livello



Interrogazioni nidificate

Operatore NOT EXISTS

Operatore NOT EXISTS (n.1)

- ▷ Trovare il nome dei fornitori che *non* forniscono il prodotto P2



Trovare il nome dei fornitori per cui non esiste una fornitura del prodotto P2

Operatore NOT EXISTS (n.1)

- ▷ Trovare il nome dei fornitori che *non* forniscono il prodotto P2

```
SELECT NomeF  
FROM F  
WHERE NOT EXISTS (SELECT *  
                   FROM FP  
                   WHERE CodP='P2'  
                   AND FP.CodF=F.CodF );
```

Condizione di correlazione

Funzionamento di NOT EXISTS (n.1)

▷ Trovare il nome dei fornitori che *non* forniscono il prodotto P2

F



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

```
SELECT *
FROM FP
WHERE CodP='P2' AND
```

FP.CodF=F1'

*Valore di CodF nella
riga corrente di F*

Funzionamento di NOT EXISTS (n.1)

▷ Trovare il nome dei fornitori che *non* forniscono il prodotto P2

F



CodF	NomeF	NSoci	Città
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

```
SELECT *
FROM FP
WHERE CodP='P2' AND
      FP.CodF='F1'
```

Funzionamento di NOT EXISTS (n.1)

▷ Trovare il nome dei fornitori che *non* forniscono il prodotto P2

F

<u>CodF</u>	<u>NomeF</u>	<u>NSoci</u>	<u>Città</u>
F1	Andrea	2	Torino
F2	Luca	1	Milano
F3	Antonio	3	Milano
F4	Gabriele	2	Torino
F5	Matteo	3	Venezia



FP

<u>CodF</u>	<u>CodP</u>	<u>Qta</u>
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

▷ Il predicato con NOT EXISTS è falso per F1 perché esiste una fornitura di P2 per F1

- F1 *non* fa parte del risultato dell'interrogazione

Funzionamento di NOT EXISTS (n.1)

▷ Trovare il nome dei fornitori che *non* forniscono il prodotto P2

F

<u>CodF</u>	NomeF	NSoci	Città
F1	Andrea	2	Torino
F2	Luca	1	Milano
F3	Antonio	3	Milano
F4	Gabriele	2	Torino
F5	Matteo	3	Venezia



FP

<u>CodF</u>	<u>CodP</u>	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
	P2	200
	P3	200
	P4	300
	P5	400



Funzionamento di NOT EXISTS (n.1)

▷ Trovare il nome dei fornitori che *non* forniscono il prodotto P2

F

<u>CodF</u>	NomeF	NSoci	Città
F1	Andrea	2	Torino
F2	Luca	1	Milano
F3	Antonio	3	Milano
F4	Gabriele	2	Torino
F5	Matteo	3	Venezia



Funzionamento di NOT EXISTS (n.1)

▷ Trovare il nome dei fornitori che *non* forniscono il prodotto P2

F

<u>CodF</u>	NomeF	NSoci	Città
F1	Andrea	2	Torino
F2	Luca	1	Milano
F3	Antonio	3	Milano
F4	Gabriele	2	Torino
F5	Matteo	3	Venezia



FP

<u>CodF</u>	<u>CodP</u>	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



Funzionamento di NOT EXISTS (n.1)

▷ Trovare il nome dei fornitori che *non* forniscono il prodotto P2

F

<u>CodF</u>	NomeF	NSoci	Città
F1	Andrea	2	Torino
F2	Luca	1	Milano
F3	Antonio	3	Milano
F4	Gabriele	2	Torino
F5	Matteo	3	Venezia



FP

<u>CodF</u>	<u>CodP</u>	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

▷ Il predicato con NOT EXISTS è
vero per F4 perché non esiste
una fornitura di P2 per F4

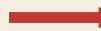
- F4 fa parte del risultato
dell'interrogazione

Funzionamento di NOT EXISTS (n.1)

▷ Trovare il nome dei fornitori che *non* forniscono il prodotto P2

F

<u>CodF</u>	NomeF	NSoci	Città
F1	Andrea	2	Torino
F2	Luca	1	Milano
F3	Antonio	3	Milano
F4	Gabriele	2	Torino
F5	Matteo	3	Venezia



FP

<u>CodF</u>	<u>CodP</u>	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

Risultato dell'interrogazione (n.1)

- ▷ Trovare il nome dei fornitori che *non* forniscono il prodotto P2

NomeF
Gabriele
Matteo

Predicato con NOT EXISTS

- ▷ Il predicato contenente NOT EXISTS è
 - vero se l'interrogazione interna restituisce l'insieme vuoto
 - falso se l'interrogazione interna restituisce almeno una tupla
- ▷ La condizione di correlazione lega l'esecuzione dell'interrogazione interna al valore di attributi della tupla corrente nell'interrogazione esterna



Interrogazioni nidificate

Correlazione tra interrogazioni

Correlazione tra interrogazioni

- ▷ Può essere necessario legare la computazione di un'interrogazione nidificata al valore di uno o più attributi in un'interrogazione più esterna
 - il legame è espresso da una o più condizioni di correlazione

Condizione di correlazione

➤ Una condizione di correlazione

- è indicata nella clausola WHERE dell'interrogazione nidificata che la richiede
- è un predicato che lega attributi di tabelle nella FROM dell'interrogazione nidificata con attributi di tabelle nella FROM di interrogazioni più esterne

➤ Non si possono esprimere condizioni di correlazione

- in interrogazioni allo stesso livello di nidificazione
- contenenti riferimenti ad attributi di una tabella nella FROM di un'interrogazione nidificata

Correlazione tra interrogazioni (n.1)

- ▷ Per ogni prodotto, trovare il codice del fornitore che ne fornisce la quantità massima

Correlazione tra interrogazioni (n.1)

- ▷ Per ogni prodotto, trovare il codice del fornitore che ne fornisce la quantità massima

```
SELECT CodP, CodF  
FROM FP AS FPX  
WHERE Qta = (...
```

)

Quantità massima per il prodotto corrente

Correlazione tra interrogazioni (n.1)

- ▷ Per ogni prodotto, trovare il codice del fornitore che ne fornisce la quantità massima

```
SELECT CodP, CodF  
FROM FP AS FPX  
WHERE Qta = (SELECT MAX(Qta)  
              FROM FP AS FPY  
              ...  
            )
```

Quantità massima

Correlazione tra interrogazioni (n.1)

- ▷ Per ogni prodotto, trovare il codice del fornitore che ne fornisce la quantità massima

```
SELECT CodP, CodF  
FROM FP AS FPX  
WHERE Qta = (SELECT MAX(Qta)  
              FROM FP AS FPY  
              WHERE FPY.CodP=FPX.CodP);
```

*Quantità
massima
per il
prodotto
corrente*

Correlazione tra interrogazioni (n.1)

- ▷ Per ogni prodotto, trovare il codice del fornitore che ne fornisce la quantità massima

```
SELECT CodP, CodF  
FROM FP AS FPX  
WHERE Qta = (SELECT MAX(Qta)  
              FROM FP AS FPY  
              WHERE FPY.CodP=FPX.CodP);
```

Condizione di correlazione

Correlazione tra interrogazioni (n.2)

VIAGGIO (CodV, LuogoPartenza, LuogoArrivo,
OraPartenza, OraArrivo)

- Trovare il codice dei viaggi che hanno una durata inferiore alla durata media dei viaggi sullo stesso percorso (caratterizzato dallo stesso luogo di partenza e di arrivo)

```
SELECT CodV  
FROM VIAGGIO AS VA  
WHERE OraArrivo-OraPartenza < (...
```

} *Durata
media
dei viaggi
sul percorso
corrente*

Correlazione tra interrogazioni (n.2)

VIAGGIO (CodV, LuogoPartenza, LuogoArrivo,
OraPartenza, OraArrivo)

- Trovare il codice dei viaggi che hanno una durata inferiore alla durata media dei viaggi sullo stesso percorso (caratterizzato dallo stesso luogo di partenza e di arrivo)

```
SELECT CodV  
FROM VIAGGIO AS VA  
WHERE OraArrivo-OraPartenza <  
    (SELECT AVG(OraArrivo-OraPartenza)  
     FROM VIAGGIO AS VB  
     ... )
```

*Durata
media
dei viaggi*

Correlazione tra interrogazioni (n.2)

VIAGGIO (CodV, LuogoPartenza, LuogoArrivo,
OraPartenza, OraArrivo)

- Trovare il codice dei viaggi che hanno una durata inferiore alla durata media dei viaggi sullo stesso percorso (caratterizzato dallo stesso luogo di partenza e di arrivo)

```
SELECT CodV  
FROM VIAGGIO AS VA  
WHERE OraArrivo-OraPartenza <  
    (SELECT AVG(OraArrivo-OraPartenza)  
     FROM VIAGGIO AS VB      Condizioni di correlazione  
     WHERE VB.LuogoPartenza=VA.LuogoPartenza  
           AND VB.LuogoArrivo=VA.LuogoArrivo);
```



Interrogazioni nidificate

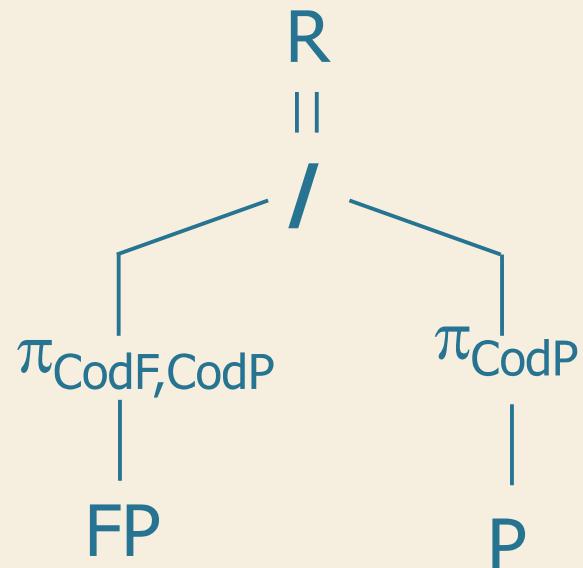
Operazione di divisione

Operazione di divisione (n.1)

- Trovare il codice dei fornitori che forniscono *tutti* i prodotti
- In algebra si utilizza l'operatore di divisione

Operazione di divisione (n.1)

- Trovare il codice dei fornitori che forniscono *tutti* i prodotti
- In algebra si utilizza l'operatore di divisione



Divisione in SQL (n.1)

➤ Trovare il codice dei fornitori che forniscono *tutti* i prodotti

➤ Osservazione

- tutti i prodotti che possono essere forniti sono contenuti nella tabella P



- un fornitore fornisce tutti i prodotti se fornisce un numero di prodotti diversi pari alla cardinalità di P

Divisione in SQL (n.1)

- ▷ Trovare il codice dei fornitori che forniscono *tutti* i prodotti

```
SELECT COUNT(*)  
FROM P
```

Divisione in SQL (n.1)

- ▷ Trovare il codice dei fornitori che forniscono *tutti* i prodotti

```
SELECT COUNT(*)  
FROM P
```

} *Numero totale di prodotti*

Divisione in SQL (n.1)

- ▷ Trovare il codice dei fornitori che forniscono *tutti* i prodotti

```
SELECT CodF  
FROM FP  
GROUP BY CodF
```

...

```
(SELECT COUNT(*)  
FROM P)
```

Divisione in SQL (n.1)

- ▷ Trovare il codice dei fornitori che forniscono *tutti* i prodotti

```
SELECT CodF  
FROM FP  
GROUP BY CodF  
HAVING COUNT(*)=(SELECT COUNT(*)  
                  FROM P);
```

Divisione in SQL: procedimento (n.2)

- Trovare il codice dei fornitori che forniscono almeno *tutti* i prodotti forniti dal fornitore F2
- Si esegue
 - il conteggio del numero di prodotti forniti da F2
 - il conteggio del numero di prodotti forniti da un fornitore arbitrario e anche da F2
- I due conteggi devono essere uguali

Divisione in SQL (n.2)

- ▷ Trovare il codice dei fornitori che forniscono almeno *tutti* i prodotti forniti dal fornitore F2

```
SELECT COUNT(*)  
FROM FP  
WHERE CodF='F2'
```

Divisione in SQL (n.2)

- ▷ Trovare il codice dei fornitori che forniscono almeno *tutti* i prodotti forniti dal fornitore F2

```
SELECT COUNT(*)  
FROM FP  
WHERE CodF='F2'
```

*Numero
di prodotti
forniti da F2*

Divisione in SQL (n.2)

- ▷ Trovare il codice dei fornitori che forniscono almeno *tutti* i prodotti forniti dal fornitore F2

```
SELECT CodF  
FROM FP  
WHERE CodP IN (SELECT CodP  
                FROM FP  
                WHERE CodF='F2')  
  
GROUP BY CodF  
...  
      (SELECT COUNT(*)  
       FROM FP  
       WHERE CodF='F2')
```

Divisione in SQL (n.2)

- ▷ Trovare il codice dei fornitori che forniscono almeno *tutti* i prodotti forniti dal fornitore F2

```
SELECT CodF  
FROM FP  
WHERE CodP IN (SELECT CodP  
                FROM FP  
                WHERE CodF='F2')  
GROUP BY CodF  
HAVING COUNT(*)=(SELECT COUNT(*)  
                     FROM FP  
                     WHERE CodF='F2');
```



Interrogazioni nidificate

Table functions

Schema di esempio

STUDENTE (Matricola, AnnoIscrizione)

ESAME-SUPERATO (Matricola, CodC, Data, Voto)

CORSO (CodC, NomeC)

Calcolo di aggregati a due livelli (n.1)

STUDENTE (Matricola, AnnoIscrizione)

ESAME-SUPERATO (Matricola, CodC, Data, Voto)

- ▷ Trovare la media massima (conseguita da uno studente)
- ▷ Risoluzione in 2 passi
 - trovare la media per ogni studente
 - trovare il valore massimo della media

Calcolo di aggregati a due livelli (n.1)

STUDENTE (Matricola, AnnoIscrizione)

ESAME-SUPERATO (Matricola, CodC, Data, Voto)

- ▷ Trovare la media massima (conseguita da uno studente)
 - passo 1: media per ogni studente

```
SELECT Matricola, AVG(Voto) AS MediaStudenti  
FROM ESAME-SUPERATO  
GROUP BY Matricola
```

Calcolo di aggregati a due livelli (n.1)

STUDENTE (Matricola, AnnoIscrizione)

ESAME-SUPERATO (Matricola, CodC, Data, Voto)

- ▷ Trovare la media massima (conseguita da uno studente)
 - passo 1: media per ogni studente

```
(SELECT Matricola, AVG(Voto) AS MediaStudenti  
FROM ESAME-SUPERATO  
GROUP BY Matricola) AS MEDIE
```

Calcolo di aggregati a due livelli (n.1)

STUDENTE (Matricola, AnnoIscrizione)

ESAME-SUPERATO (Matricola, CodC, Data, Voto)

- ▷ Trovare la media massima (conseguita da uno studente)
 - passo 2: valore massimo della media

SELECT ...

```
FROM (SELECT Matricola, AVG(Voto) AS MediaStudenti  
      FROM ESAME-SUPERATO  
      GROUP BY Matricola) AS MEDIE
```

Calcolo di aggregati a due livelli (n.1)

STUDENTE (Matricola, AnnoIscrizione)

ESAME-SUPERATO (Matricola, CodC, Data, Voto)

- ▷ Trovare la media massima (conseguita da uno studente)
 - passo 2: valore massimo della media

```
SELECT MAX(MediaStudenti)
```

```
FROM (SELECT Matricola, AVG(Voto) AS MediaStudenti
```

```
    FROM ESAME-SUPERATO
```

```
    GROUP BY Matricola) AS MEDIE;
```

Table functions (n.1)

STUDENTE (Matricola, AnnoIscrizione)

ESAME-SUPERATO (Matricola, CodC, Data, Voto)

- Trovare la media massima (conseguita da uno studente)

```
SELECT MAX(MediaStudenti)
```

```
FROM (SELECT Matricola, AVG(Voto) AS MediaStudenti  
      FROM ESAME-SUPERATO  
      GROUP BY Matricola) AS MEDIE;
```

Table function

Table function

- Definisce una tabella temporanea che può essere utilizzata per ulteriori operazioni di calcolo
- La table function
 - ha la struttura di una SELECT
 - è definita all'interno di una clausola FROM
 - può essere referenziata come una normale tabella
- La table function permette di
 - calcolare più livelli di aggregazione
 - formulare in modo equivalente le interrogazioni che richiedono la correlazione

Table functions (n.2)

STUDENTE (Matricola, AnnoIscrizione)

ESAME-SUPERATO (Matricola, CodC, Data, Voto)

- Per ogni anno di iscrizione, trovare la media massima (conseguita da uno studente)
- Risoluzione in 2 passi
 - trovare la media per ogni studente
 - raggruppare gli studenti per anno di iscrizione e calcolare la media massima

Table functions (n.2)

STUDENTE (Matricola, AnnoIscrizione)

ESAME-SUPERATO (Matricola, CodC, Data, Voto)

- ▷ Per ogni anno di iscrizione, trovare la media massima (conseguita da uno studente)
 - passo 1

```
(SELECT Matricola, AVG(Voto) AS MediaStudente  
FROM ESAME-SUPERATO  
GROUP BY Matricola) AS MEDIE
```

Table functions (n.2)

STUDENTE (Matricola, AnnoIscrizione)

ESAME-SUPERATO (Matricola, CodC, Data, Voto)

- ▷ Per ogni anno di iscrizione, trovare la media massima (conseguita da uno studente)

- passo 2

SELECT ...

FROM STUDENTE,

(SELECT Matricola, AVG(Voto) AS MediaStudente

FROM ESAME-SUPERATO

GROUP BY Matricola) AS MEDIE

Table function

WHERE STUDENTE.Matricola=MEDIE.Matricola

Table functions (n.2)

STUDENTE (Matricola, AnnoIscrizione)

ESAME-SUPERATO (Matricola, CodC, Data, Voto)

- ▷ Per ogni anno di iscrizione, trovare la media massima (conseguita da uno studente)

- passo 2

SELECT ...

FROM STUDENTE,

*Condizione
di join* (SELECT Matricola, AVG(Voto) AS MediaStudente

FROM ESAME-SUPERATO

GROUP BY Matricola) AS MEDIE

WHERE STUDENTE.Matricola=MEDIE.Matricola

Table functions (n.2)

STUDENTE (Matricola, AnnoIscrizione)

ESAME-SUPERATO (Matricola, CodC, Data, Voto)

- ▷ Per ogni anno di iscrizione, trovare la media massima (conseguita da uno studente)

- passo 2

```
SELECT ...
```

```
FROM STUDENTE,
```

```
(SELECT Matricola, AVG(Voto) AS MediaStudente
```

```
FROM ESAME-SUPERATO
```

```
GROUP BY Matricola) AS MEDIE
```

```
WHERE STUDENTE.Matricola=MEDIE.Matricola
```

```
GROUP BY AnnoIscrizione
```

Table functions (n.2)

STUDENTE (Matricola, AnnoIscrizione)

ESAME-SUPERATO (Matricola, CodC, Data, Voto)

- ▷ Per ogni anno di iscrizione, trovare la media massima (conseguita da uno studente)

- passo 2

```
SELECT AnnoIscrizione, MAX(MediaStudente)
FROM STUDENTE,
     (SELECT Matricola, AVG(Voto) AS MediaStudente
      FROM ESAME-SUPERATO
      GROUP BY Matricola) AS MEDIE
WHERE STUDENTE.Matricola=MEDIE.Matricola
      GROUP BY AnnoIscrizione;
```