



Interrogazioni nidificate

Linguaggio SQL

Linguaggio SQL: fondamenti

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

Introduzione

Interrogazioni nidificate

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**

Base dati di esempio: Forniture-Prodotti

P

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

F

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

Chiave esterna

Chiave esterna

FP

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

Esempio 1: Interrogazioni nidificate

- 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

Esempio 1: Interrogazioni nidificate

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

*Codici dei fornitori
con sede nella stessa
città di S1*

```
SELECT CodF  
FROM F  
WHERE Sede = (
```

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

*Città del
fornitore di S1*

- È possibile utilizzare '=' esclusivamente se è noto a priori che il risultato della SELECT nidificata è sempre **un solo** valore
- È possibile definire una formulazione equivalente con il join

Formulazione equivalente con il join

- 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**

Esempio 1: Formulazione equivalente

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

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

Esempio 1: Formulazione equivalente

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

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

Esempio 1: Formulazione equivalente

- 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');
```



Esempio 1: Formulazione equivalente

- 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
```

...

Esempio 1: Formulazione equivalente

- 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');
```

Esempio 1: Formulazione equivalente

- 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';
```

Esempio 2: Interrogazioni nidificate

- 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

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”
- L'interrogazione nidificata può essere sostituita con una lista di valori
- 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**

Esempio 1: Operatore IN

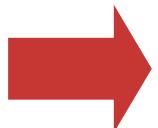
- Trovare il nome dei fornitori che forniscono il prodotto P2
- Scomposizione del problema in due sottoproblemi
 - codici dei fornitori del prodotto P2
 - nome dei fornitori aventi quei codici

Esempio 1: Operatore IN

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

FP

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



CodF
F1
F2
F3

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

Codici dei fornitori di P2

Esempio 1: Operatore IN

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

F

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

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

Appartenenza all'insieme

Codici dei fornitori di P2

CodF
F1
F2
F3

Formulazione equivalente con operatore Join

- 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**

Esempio 1: Formulazione equivalente con Join

- Trovare il nome dei fornitori che forniscono il prodotto P2

IN

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

JOIN

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

Esempio 2: Operatore IN

- 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

Esempio 2: Operatore IN

- 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'));
```



Esempio 2: Formulazione equivalente

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

IN

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

JOIN

```
SELECT NomeF  
FROM F, FP 1  
WHERE FP.CodF=F.CodF
```

Esempio 2: Formulazione equivalente

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

IN

```
SELECT NomeF  
FROM F  
WHERE CodF IN (  
    SELECT CodF 2  
    FROM FP  
    WHERE CodP IN (  
        SELECT CodP  
        FROM P  
        WHERE Colore='Rosso'));
```

JOIN

```
SELECT NomeF  
FROM F, FP, P  
WHERE FP.CodF=F.CodF AND  
      FP.CodP=P.CodP 2
```

Esempio 2: Formulazione equivalente

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

IN

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

JOIN

```
SELECT NomeF  
FROM F, FP, P  
WHERE FP.CodF=F.CodF AND  
FP.CodP=P.CodP AND  
Colore='Rosso' 3
```

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 da
 - interrogazione nidificata
 - lista di valori
- Non esiste una formulazione equivalente con il join

Esempio 1: Concetto di esclusione

- 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

- La query corrisponde alla richiesta:
 - Trovare il nome dei fornitori che forniscono almeno un prodotto diverso da P2

Esempio 1: Soluzione errata

- 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

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



NomeF
Andrea
Luca
Gabriele

Esempio 1: Operatore NOT IN

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

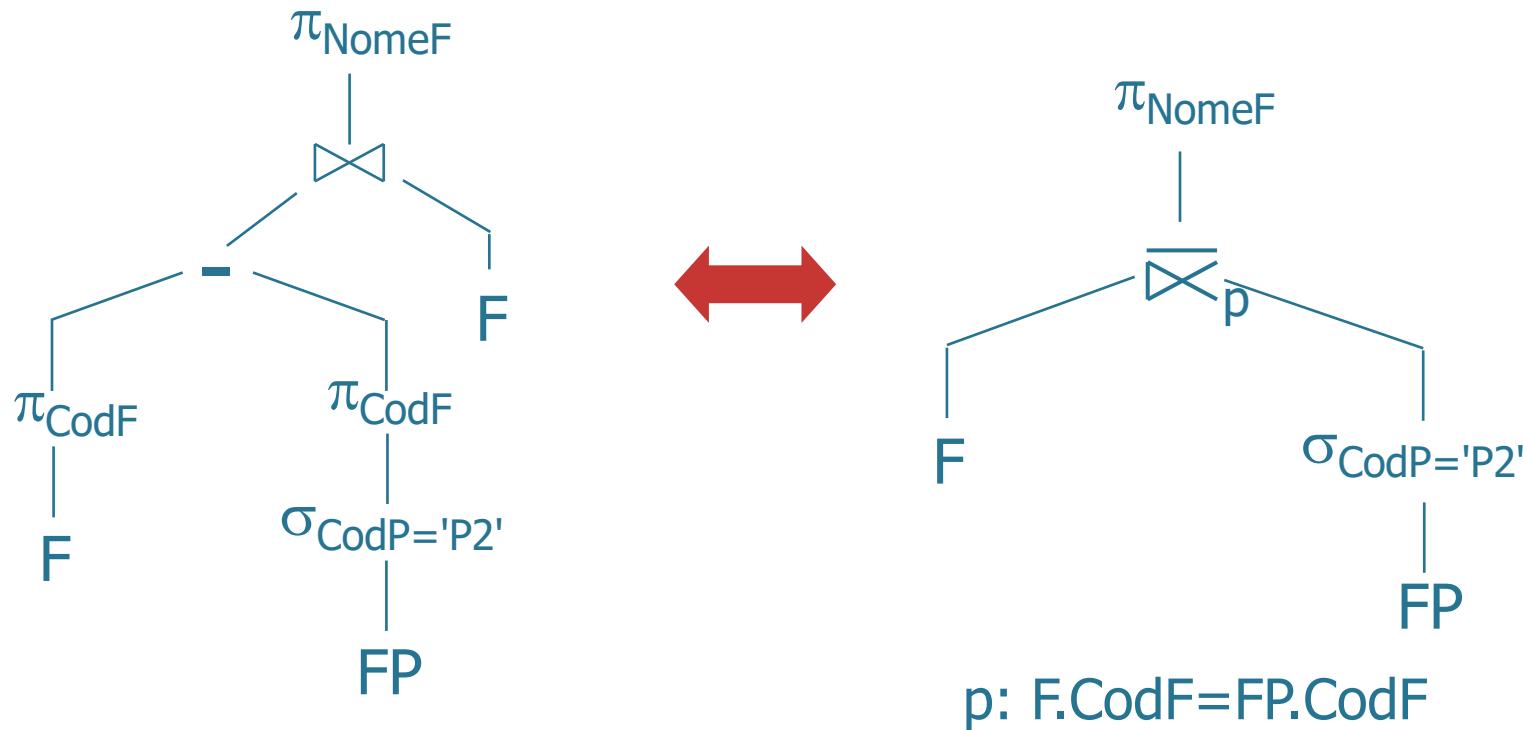
- Esprime il concetto di esclusione da un insieme di valori

*NomeAttributo **NOT IN** (InterrogazioneNidificata)*

- Richiede di individuare in modo appropriato *l'insieme da escludere*
 - definito dall'interrogazione nidificata

Operatore NOT IN e algebra relazionale (n.1)

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



Esempio 2: Operatore NOT IN

- 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

Esempio 2: Operatore NOT IN

- 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;
```

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

Esempio 1: soluzione alternativa

- 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);
```

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

Esempio 3: Operatore NOT IN

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

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

*Codici dei fornitori
di prodotti rossi*

Esempio 3: Alternativa (corretta?)

- 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')
```

Esempio 3: Alternativa (corretta?)

- 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*

- L'insieme di elementi da escludere non è corretto

Esempio 3: Alternativa errata

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

P

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

F

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

FP

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

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

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

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

Esempio

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

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

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

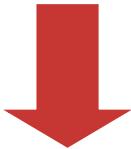
*Costruttore
di tupla*

OPERATORE EXISTS

- L'operatore **EXISTS** ammette come **parametro** una **interrogazione nidificata** e **restituisce**
 - il valore **vero** solo se l'interrogazione nidificata fornisce un **insieme non vuoto** (ossia restituisce almeno una tupla)
 - il valore **falso** se l'interrogazione interna restituisce **l'insieme vuoto** (ossia non restituisce nessuna tupla)
- 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

Operatore EXISTS

- Trovare il nome dei fornitori del prodotto P2



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

Condizione di correlazione

- 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

- La condizione di correlazione **lega la computazione dell'interrogazione nidificata** al valore di uno o più attributi dell'interrogazione più esterna

Funzionamento di operatore EXISTS

- 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 operatore EXISTS

- 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 operatore EXISTS

- 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 falso per F4 poiché non esiste una fornitura di P2 per F4
 - F4 non fa parte del risultato dell'interrogazione

Risultato dell'interrogazione

- Trovare il nome dei fornitori del prodotto P2

R
NomeF
Andrea
Luca
Antonio

Predicati con operatore 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, perchè 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

OPERATORE NOT EXISTS

- L'operatore **NOT EXISTS** ammette come **parametro** una interrogazione nidificata e **restituisce**
 - il valore **vero** se l'interrogazione interna restituisce **l'insieme vuoto** (ossia non restituisce nessuna tupla)
 - il valore **falso** se l'interrogazione nidificata fornisce **un insieme non vuoto** (ossia restituisce almeno una tupla)
- Nell'interrogazione interna a NOT 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

Operatore NOT EXISTS

- 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

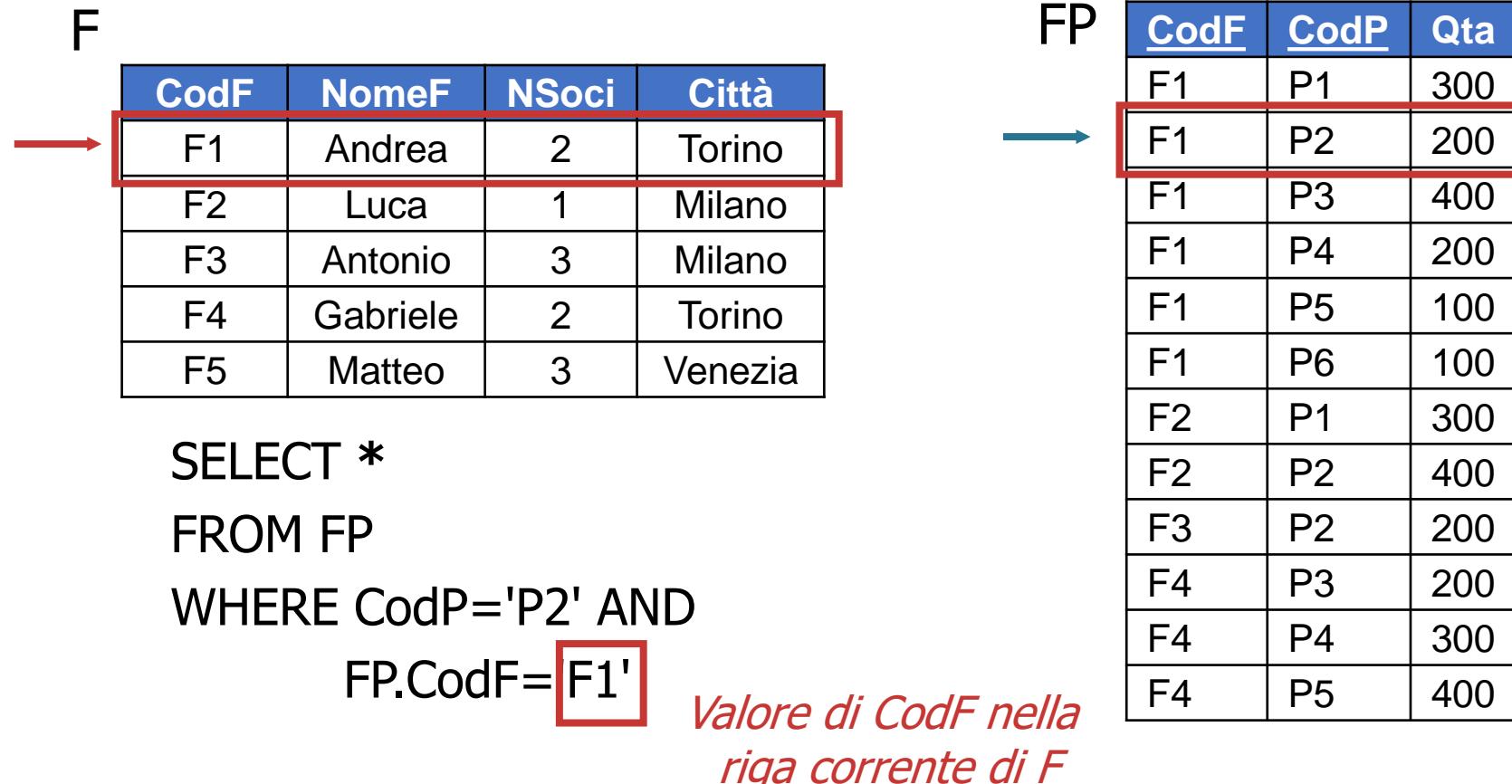
- Trovare il nome dei fornitori che *non* forniscono il prodotto P2

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

- La condizione di correlazione **lega la computazione dell'interrogazione nidificata** al valore di uno o più attributi dell'interrogazione più esterna

Funzionamento di operatore NOT EXISTS

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



Funzionamento di operatore NOT EXISTS

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



Funzionamento di operatore NOT EXISTS

- 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 F2 perché esiste una fornitura di P2 per F2
 - F2 *non* fa parte del risultato dell'interrogazione

Funzionamento di operatore NOT EXISTS

- 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 vero per F4 perché non esiste una fornitura di P2 per F4
 - F4 *fa parte* del risultato dell'interrogazione

Funzionamento di operatore NOT EXISTS

- 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 vero per F5 perché non esiste una fornitura di P2 per F5
 - F5 *fa parte* del risultato dell'interrogazione

Risultato dell'interrogazione

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

NomeF
Gabriele
Matteo

Predicato con operatore 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

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
- 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

```
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);
```

Condizione di correlazione

*Quantità
massima
per il
prodotto
corrente*

Correlazione tra interrogazioni (n.2)

- 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)

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

Correlazione tra interrogazioni (n.2)

- 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)

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

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

)

*Durata
media
dei viaggi
sul percorso
corrente*

Correlazione tra interrogazioni (n.2)

- 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)

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

```
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)

- 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)

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

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

Condizioni di correlazione

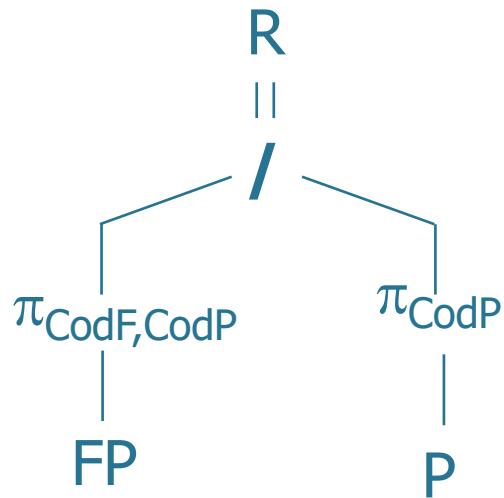
*Durata
media
dei viaggi
sul percorso
corrente*

OPERAZIONE DI DIVISIONE

- Nel linguaggio SQL, l'operazione di divisione può essere realizzata mediante l'operatore **COUNT**, per verificare che gli elementi di interesse appartengano **tutti** all'insieme di riferimento

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
```

*Numero
totale di
prodotti*

Divisione in SQL (n.1)

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

*Per ogni fornitore,
numero totale di
prodotti forniti*

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

*Numero
totale di
prodotti*

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'
```

*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

Per ogni fornitore, numero totale di prodotti forniti , considerando solo i prodotti forniti da 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');
```

Prodotti forniti da F2

Numero di prodotti forniti da F2