Cofinanziato dall'Unione europea

FinTech Software Developer

Basi di dati SQL

Docente: Roi Davide Simone

Titolo argomento: Sviluppo su Postgresql – Costrutti di querying
2° parte

Roi Davide
Dispense

Intervento realizzato da
ITS TECNOLOGIE DELL'INFORMAZIONE E DELLA COMUNICAZIONE

Il comando di JOIN

Descrizione: Il comando "JOIN" in SQL viene utilizzato per combinare informazioni da due o più tabelle in base ad una o più colonne in comune.

Ad esempio, se abbiamo una tabella contenente informazioni sulle aziende e un'altra tabella contenente informazioni sugli investimenti effettuati dalle aziende, possiamo utilizzare il comando "JOIN" per unire le due tabelle in modo da avere informazioni sulle aziende e sugli investimenti nello stesso RECORDSET. In questo modo, puoi ottenere una visione più completa delle informazioni relativa alle aziende.
Ci sono quattro tipi di JOIN comunemente utilizzati in SQL:

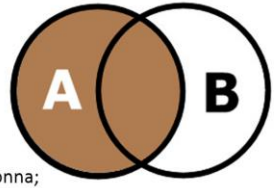
- INNER JOIN
- LEFT JOIN (o LEFT OUTER JOIN)
- RIGHT JOIN (o RIGHT OUTER JOIN)
- FULL JOIN (o FULL OUTER JOIN)

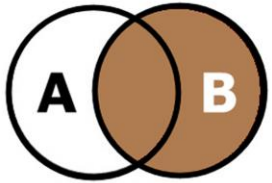
Combinando queste quattro JOIN con ulteriori condizioni di WHERE si possono ottenere i principali 7 incroci di tabelle descritti nella slide successiva:

Join

Roi Davide
Dispense

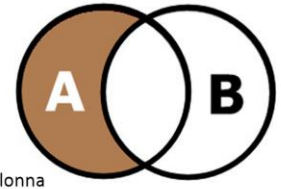
```
SELECT
  <list>
FROM
  tabella1 AS a
LEFT JOIN
  tabella2 AS b ON
    a.colonna = b.colonna;
```

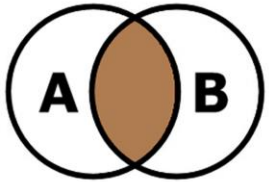




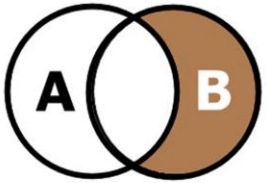

```
SELECT
  <list>
FROM
  tabella1 AS a
RIGHT JOIN
  tabella2 AS b ON
    a.colonna = b.colonna;
```

```
SELECT
  <list>
FROM
  tabella1 AS a
LEFT JOIN
  tabella2 AS b ON
    a.colonna = b.colonna
WHERE
    b.colonna IS NULL;
```



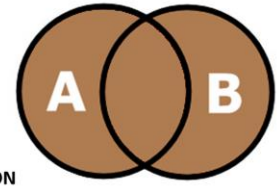


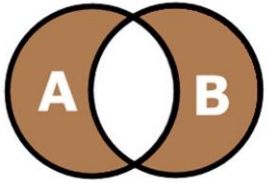

```
SELECT
  <list>
FROM
  tabella1 AS a
INNER JOIN
  tabella2 AS b ON
    a.colonna = b.colonna;
```




```
SELECT
  <list>
FROM
  tabella1 AS a
RIGHT JOIN
  tabella2 AS b ON
    a.colonna = b.colonna
WHERE
    a.colonna IS NULL;
```

```
SELECT
  <list>
FROM
  tabella1 AS a
FULL JOIN
  tabella2 AS b ON
    a.colonna = b.colonna;
```






```
SELECT
  <list>
FROM
  tabella1 AS a
FULL JOIN
  tabella2 AS b ON
    a.colonna = b.colonna
WHERE
  a.colonna IS NULL
  OR b.colonna IS NULL;
```

Intervento realizzato da
ITS TECNOLOGIE DELL'INFORMAZIONE E DELLA COMUNICAZIONE

Ordine di esecuzione di una JOIN

Roi Davide Dispense

Comprendere l'ordine con in quale il DBMS esegue le JOIN è fondamentale per capire come scriverle la query.

STEP1

```
SELECT
    a.nome_colonna_x,
    b.nome_colonna_y,
    ecc...
FROM
    nome_tabella_left AS a
[INNER|LEFT|RIGHT|FULL] JOIN
    nome_tabella_right AS b ON
        a.colonna_z = b.colonna_z
WHERE
    condizione_1,
    condizione_2,
    ecc..
```

Lo step1 restituisce un recordset eseguendo una semplice JOIN

STEP2

```
SELECT
    a.nome_colonna_x,
    b.nome_colonna_y,
    ecc...
FROM
    nome_tabella_left AS a
[INNER|LEFT|RIGHT|FULL] JOIN
    nome_tabella_right AS b ON
        a.colonna_z = b.colonna_z
WHERE
    condizione_1,
    condizione_2,
    ecc..
```

Lo step2 lavora sul recordset dello STEP1 e decide quali record prendere in base alla TIPOLOGIA di JOIN.

STEP3

```
SELECT
    a.nome_colonna_x,
    b.nome_colonna_y,
    ecc...
FROM
    nome_tabella_left AS a
[INNER|LEFT|RIGHT|FULL] JOIN
    nome_tabella_right AS b ON
        a.colonna_z = b.colonna_z
WHERE
    condizione_1,
    condizione_2,
    ecc..
```

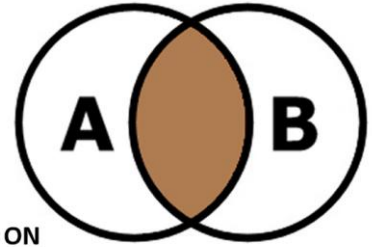
Lo step3 lavora sul recordset dello STEP2 ed applica i filtri della WHERE

Intervento realizzato da ITS Tecnologie dell'Informazione e della Comunicazione

INNER JOIN

Roi Davide
Dispense

```
SELECT
  <list>
FROM
  tabella1 AS a
INNER JOIN
  tabella2 AS b ON
    a.colonna = b.colonna;
```



A.anno_fondazione	A.ragione_sociale	A.id
1800	Banco di Napoli	1
2015	FintechLab	2
1823	Intesa Sanpaolo	3
2011	Moneyfarm	4
2013	N26	5

B.id_azienda	B.nome	B.id
3	Fondo comune	1
4	Obbligazioni	2
5	Azioni	3
6	Cambio valuta	4
7	Derivati	5

Questa Join restituisce una tabella con:

- I record di A che hanno i valori delle colonne di join in comune con B
- I record di B che hanno i valori delle colonne di join in comune con A

Risultato:

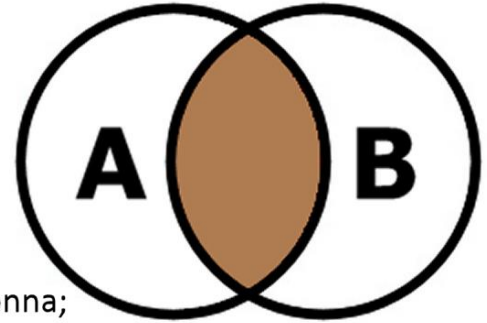
A.anno_fondazione	A.ragione_sociale	A.id	B.id_azienda	B.nome	B.id
1823	Intesa Sanpaolo	3	3	Fondo comune	1
2011	Moneyfarm	4	4	Obbligazioni	2
2013	N26	5	5	Azioni	3

Intervento realizzato da
ITS TECNOLOGIE DELL'INFORMAZIONE E DELLA COMUNICAZIONE

INNER JOIN (JOIN implicita)

Roi Davide Dispense

```
SELECT
  <list>
FROM
  tabella1 AS a,
  tabella2 AS b
WHERE
    a.colonna = b.colonna;
```



Talvolta la INNER JOIN può anchè essere scritta nella forma implicita senza dichiarare espressamente la parola INNER e usanto il costrutto di WHERE per mettere in relazione i campi. Vediamo degli esempi

Esempio di JOIN DI 4 TABELLE:

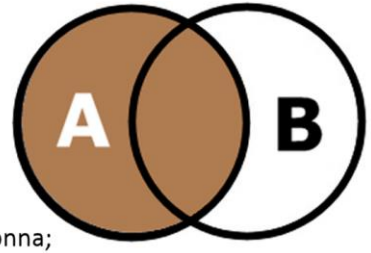
```
SELECT
  <list>
FROM
  tabella1 AS a,
  tabella2 AS b,
  tabella3 AS c,
  tabella4 AS d
WHERE
    a.colonna = b.colonna
    and b.colonna = c.colonna
    and c.colonna = d.colonna;
```

Intervento realizzato da ITS Tecnologie dell'Informazione e della Comunicazione

LEFT JOIN

Roi Davide Dispense

```
SELECT
  <list>
FROM
  tabella1 AS a
LEFT JOIN
  tabella2 AS b ON
    a.colonna = b.colonna;
```



A.anno_fondazione	A.ragione_sociale	A.id
1800	Banco di Napoli	1
2015	FintechLab	2
1823	Intesa Sanpaolo	3
2011	Moneyfarm	4
2013	N26	5

B.id_azienda	B.nome	B.id
3	Fondo comune	1
4	Obbligazioni	2
5	Azioni	3
6	Cambio valuta	4
7	Derivati	5

Questa Join restituisce una tabella con:

- Tutti i record di A
- I record di B che hanno i valori delle colonne di join in comune con A

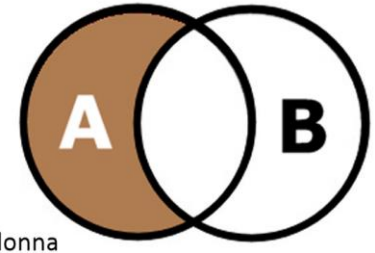
Risultato:

A.anno_fondazione	A.ragione_sociale	A.id	B.id_azienda	B.nome	B.id
1800	Banco di Napoli	1	NULL	NULL	NULL
2015	FintechLab	2	NULL	NULL	NULL
1823	Intesa Sanpaolo	3	3	Fondo comune	1
2011	Moneyfarm	4	4	Obbligazioni	2
2013	N26	5	5	Azioni	3

LEFT JOIN esclusiva

Roi Davide Dispense

```
SELECT
  <list>
FROM
  tabella1 AS a
LEFT JOIN
  tabella2 AS b ON
    a.colonna = b.colonna
WHERE
    b.colonna IS NULL;
```



A.anno_fondazione	A.ragione_sociale	A.id
1800	Banco di Napoli	1
2015	FintechLab	2
1823	Intesa Sanpaolo	3
2011	Moneyfarm	4
2013	N26	5

B.id_azienda	B.nome	B.id
3	Fondo comune	1
4	Obbligazioni	2
5	Azioni	3
6	Cambio valuta	4
7	Derivati	5

Questa Join restituisce una tabella con:

- I record di A che NON hanno i valori delle colonne di join in comune con B

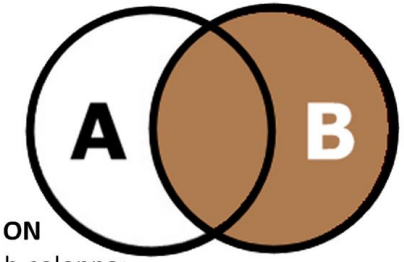
Risultato:

A.anno_fondazione	A.ragione_sociale	A.id	B.id_azienda	B.nome	B.id
1800	Banco di Napoli	1	NULL	NULL	NULL
2015	FintechLab	2	NULL	NULL	NULL

RIGHT JOIN

Roi Davide Dispense

```
SELECT
  <list>
FROM
  tabella1 AS a
RIGHT JOIN
  tabella2 AS b ON
    a.colonna = b.colonna;
```



A.anno_fondazione	A.ragione_sociale	A.id
1800	Banco di Napoli	1
2015	FintechLab	2
1823	Intesa Sanpaolo	3
2011	Moneyfarm	4
2013	N26	5

B.id_azienda	B.nome	B.id
3	Fondo comune	1
4	Obbligazioni	2
5	Azioni	3
6	Cambio valuta	4
7	Derivati	5

Questa Join restituisce una tabella con:

- Tutti i record di B
- I record di A che hanno i valori delle colonne di join in comune con B

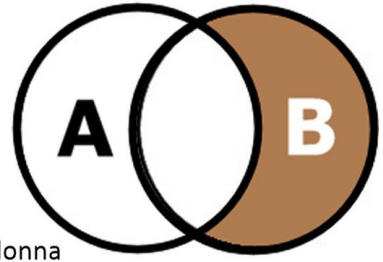
Risultato:

A.anno_fondazione	A.ragione_sociale	A.id	B.id_azienda	B.nome	B.id
1823	Intesa Sanpaolo	3	3	Fondo comune	1
2011	Moneyfarm	4	4	Obbligazioni	2
2013	N26	5	5	Azioni	3
NULL	NULL	NULL	6	Cambio valuta	4
NULL	NULL	NULL	7	Derivati	5

RIGHT JOIN esclusiva

Roi Davide Dispense

```
SELECT
  <list>
FROM
  tabella1 AS a
RIGHT JOIN
  tabella2 AS b ON
    a.colonna = b.colonna
WHERE
    a.colonna IS NULL;
```



A.anno_fondazione	A.ragione_sociale	A.id
1800	Banco di Napoli	1
2015	FintechLab	2
1823	Intesa Sanpaolo	3
2011	Moneyfarm	4
2013	N26	5

B.id_azienda	B.nome	B.id
3	Fondo comune	1
4	Obbligazioni	2
5	Azioni	3
6	Cambio valuta	4
7	Derivati	5

Questa Join restituisce una tabella con:

- I record di B che NON hanno i valori delle colonne di join in comune con A

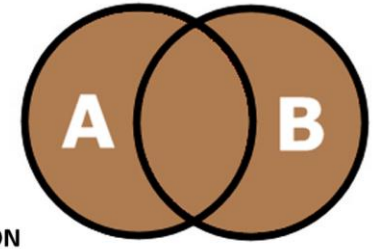
Risultato:

A.anno_fondazione	A.ragione_sociale	A.id	B.id_azienda	B.nome	B.id
NULL	NULL	NULL	6	Cambio valuta	4
NULL	NULL	NULL	7	Derivati	5

FULL JOIN

Roi Davide Dispense

```
SELECT
  <list>
FROM
  tabella1 AS a
FULL JOIN
  tabella2 AS b ON
    a.colonna = b.colonna;
```



A.anno_fondazione	A.ragione_sociale	A.id
1800	Banco di Napoli	1
2015	FintechLab	2
1823	Intesa Sanpaolo	3
2011	Moneyfarm	4
2013	N26	5

B.id_azienda	B.nome	B.id
3	Fondo comune	1
4	Obbligazioni	2
5	Azioni	3
6	Cambio valuta	4
7	Derivati	5

Risultato:

A.anno_fondazione	A.ragione_sociale	A.id	B.id_azienda	B.nome	B.id
1800	Banco di Napoli	1	NULL	NULL	NULL
2015	FintechLab	2	NULL	NULL	NULL
1823	Intesa Sanpaolo	3	3	Fondo comune	1
2011	Moneyfarm	4	4	Obbligazioni	2
2013	N26	5	5	Azioni	3
NULL	NULL	NULL	6	Cambio valuta	4
NULL	NULL	NULL	7	Derivati	5

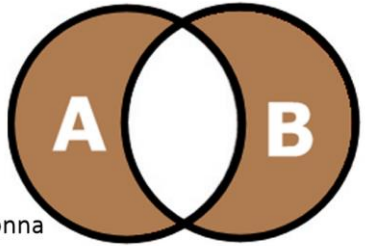
Questa Join restituisce una tabella con:

- Tutti i record di A
- Tutti i record di B

FULL JOIN esclusiva

Roi Davide Dispense

```
SELECT
  <list>
FROM
  tabella1 AS a
FULL JOIN
  tabella2 AS b ON
    a.colonna = b.colonna
WHERE
  a.colonna IS NULL
  OR b.colonna IS NULL;
```



A.anno_fondazione	A.ragione_sociale	A.id
1800	Banco di Napoli	1
2015	FintechLab	2
1823	Intesa Sanpaolo	3
2011	Moneyfarm	4
2013	N26	5

B.id_azienda	B.nome	B.id
3	Fondo comune	1
4	Obbligazioni	2
5	Azioni	3
6	Cambio valuta	4
7	Derivati	5

Risultato:

A.anno_fondazione	A.ragione_sociale	A.id	B.id_azienda	B.nome	B.id
1800	Banco di Napoli	1	NULL	NULL	NULL
2015	FintechLab	2	NULL	NULL	NULL
NULL	NULL	NULL	6	Cambio valuta	4
NULL	NULL	NULL	7	Derivati	5

Questa Join restituisce una tabella con:

- Tutti i record di A che hanno i valori delle colonne di join in comune con B
- Tutti i record di B che hanno i valori delle colonne di join in comune con A

Le SELECT annidate

Roi Davide
Dispense

L’annidamento delle select è una pratica comune nell’uso dell’ SQL. Permette di eseguire interrogazioni complesse in modo elegante e funzionale.

Le select annidate possono essere utilizzate:

- Nel costrutto di selezione **FROM**
- Nel costrutto di condizione sulla selezione **WHERE**
- Nel costrutto di condizione sul raggruppamento **HAVING**

Le SELECT annidate

- Nel costrutto di selezione **FROM**

```
SELECT
    cl.*
FROM
    (
    SELECT
        cli.*
    FROM
        ft_cliente AS cli
    WHERE
        cli.id =2
    ) AS cl
;
```

```
SELECT
    cli.*,
    rec.*
FROM
    ft_cliente as cli
INNER JOIN
    (
    SELECT
        *
    FROM
        ft_recensione
    WHERE
        voto>3
    ) AS rec ON
        cli.id =rec.id_cliente
;
```

Le SELECT annidate

- Nel costrutto di condizione sulla selezione **WHERE**

```
SELECT
    cli.*
FROM
    ft_cliente as cli
WHERE
    cli.data_nascita =
        (
        SELECT
            max(data_nascita)
        FROM
            ft_cliente
         )
;
```

```
SELECT
    cli.*
FROM
    ft_cliente as cli
WHERE
    cli.data_nascita IN
        (
        SELECT
            data_nascita
        FROM
            ft_cliente
        ORDER BY
            data_nascita DESC
        LIMIT 2
         )
;
```

Le SELECT annidate

- Nel costrutto di condizione sulla selezione **HAVING**

```
SELECT
    cli.cognome,
    max(cli.data_nascita) AS dn
FROM
    ft_cliente as cli
GROUP BY
    cli.cognome
HAVING
    cli.cognome =
        (
        SELECT
            max(cognome)
        FROM
            ft_cliente
        )
;
```

```
SELECT
    cli.cognome,
    count(cli.*) AS qta
FROM
    ft_cliente as cli
GROUP BY
    cli.cognome
HAVING
    cli.cognome in
        (
        SELECT
            cognome
        FROM
            ft_cliente
        ORDER BY
            cognome
        LIMIT 3
        )
;
```

Fonti:

- Documentazione ufficiale di Postgresql
  https://www.postgresql.org/docs/

- SQL online tutorial.org
  https://www.sqltutorial.org/

Fine della Presentazione