

## UNIRE LE TABELLE

JOIN combina i dati di due tabelle.

GIOCHI			GATTI	
id_gioco	nome_gioco	id_gatto	id_gatto	nome_gatto
1	palla	3	1	Kitty
2	molla	NULL	2	Hugo
3	topo	1	3	Sam
4	topo	4	4	Misty
5	palla	1		

JOIN combina tipicamente righe con valori uguali per le colonne specificate. **Di solito**, una tabella contiene una **chiave primaria**, ovvero una o più colonne che identificano in modo univoco le righe della tabella (la colonna `id_gatto` nella tabella `gatti`). L'altra tabella ha una o più colonne che fanno **riferimento alle colonne della chiave primaria** della prima tabella (la colonna `id_gatto` nella tabella `giochi`). Tali colonne sono **chiavi esterne**. La condizione di JOIN è l'uguaglianza tra le colonne della chiave primaria di una tabella e quelle che vi fanno riferimento nell'altra tabella.

## JOIN

JOIN restituisce tutte le righe che corrispondono alla condizione ON. JOIN è anche chiamato INNER JOIN

```
SELECT *
FROM giochi
JOIN gatti
ON giochi.id_gatto = gatti.id_gatto;
```

id_gioco	nome_gioco	id_gatto	id_gatto	nome_gatto
5	palla	1	1	Kitty
3	topo	1	1	Kitty
1	palla	3	3	Sam
4	topo	4	4	Misty

Esiste anche un'altra sintassi, più vecchia, ma non è **raccomandata**. È possibile elencare le tabelle da unire nella clausola FROM e inserire le condizioni nella clausola WHERE.

```
SELECT *
FROM giochi, gatti
WHERE giochi.id_gatto = gatti.id_gatto;
```

## CONDIZIONI DI JOIN

La condizione di JOIN non deve essere necessariamente un'uguaglianza, ma può essere qualsiasi condizione. JOIN non interpreta la condizione di JOIN, ma controlla solo se le righe soddisfano la condizione data.

Per fare riferimento a una colonna nella query JOIN, è necessario utilizzare il nome completo della colonna: prima il nome della tabella, poi un punto (.) e il nome della colonna:

```
ON gatti.id_gatto = giochi.id_gatto
```

È possibile omettere il nome della tabella e utilizzare solo il nome della colonna se il nome della colonna è unico per tutte le colonne delle tabelle unite.

## NATURAL JOIN

Se le tabelle hanno colonne con lo stesso nome, puoi usare NATURAL JOIN invece di JOIN.

```
SELECT *
FROM giochi
NATURAL JOIN gatti;
```

La colonna comune appare una sola volta nella tabella dei risultati.

**Nota:** NATURAL JOIN è raramente utilizzato nei problemi concreti.

id_gatto	id_gioco	nome_gioco	nome_gatto
1	5	palla	Kitty
1	3	topo	Kitty
3	1	palla	Sam
4	4	topo	Misty

## LEFT JOIN

LEFT JOIN restituisce tutte le righe della **tabella di sinistra** con le righe corrispondenti della tabella di destra. Le righe senza corrispondenza vengono riempite con valori NULL. LEFT JOIN è chiamato anche LEFT OUTER JOIN.

```
SELECT *
FROM giochi
LEFT JOIN gatti
ON giochi.id_gatto = gatti.id_gatto;
```

id_gioco	nome_gioco	id_gatto	id_gatto	nome_gatto
5	palla	1	1	Kitty
3	topo	1	1	Kitty
1	palla	3	3	Sam
4	topo	4	4	Misty
2	molla	NULL	NULL	NULL

intera tabella sinistra

## RIGHT JOIN

RIGHT JOIN restituisce tutte le righe della **tabella di destra** con le righe corrispondenti della tabella di sinistra. Le righe senza corrispondenza vengono riempite con valori NULL. RIGHT JOIN è anche chiamato RIGHT OUTER JOIN.

```
SELECT *
FROM giochi
RIGHT JOIN gatti
ON giochi.id_gatto = gatti.id_gatto;
```

id_gioco	nome_gioco	id_gatto	id_gatto	nome_gatto
5	palla	1	1	Kitty
3	topo	1	1	Kitty
NULL	NULL	NULL	2	Hugo
1	palla	3	3	Sam
4	topo	4	4	Misty

intera tabella destra

## FULL JOIN

FULL JOIN restituisce tutte le righe della **tabella di sinistra** e tutte le righe della **tabella di destra**. Riempie le righe non corrispondenti con valori NULL. FULL JOIN è anche chiamato FULL OUTER JOIN.

```
SELECT *
FROM giochi
FULL JOIN gatti
ON giochi.id_gatto = gatti.id_gatto;
```

id_gioco	nome_gioco	id_gatto	id_gatto	nome_gatto
5	palla	1	1	Kitty
3	topo	1	1	Kitty
NULL	NULL	NULL	2	Hugo
1	palla	3	3	Sam
4	topo	4	4	Misty
2	molla	NULL	NULL	NULL

intera tabella sinistra    intera tabella destra

## CROSS JOIN

CROSS JOIN restituisce **tutte le possibili combinazioni** di righe delle tabelle di sinistra e di destra.

```
SELECT *
FROM giochi
CROSS JOIN gatti;
```

Altri sintassi:  
SELECT \*  
FROM giochi, gatti;

id_gioco	nome_gioco	id_gatto	id_gatto	nome_gatto
1	palla	3	1	Kitty
2	molla	NULL	1	Kitty
3	topo	1	1	Kitty
4	topo	4	1	Kitty
5	palla	1	1	Kitty
1	palla	3	2	Hugo
2	molla	NULL	2	Hugo
3	topo	1	2	Hugo
4	topo	4	2	Hugo
5	palla	1	2	Hugo
1	palla	3	3	Sam
...	...	...	...	...

## ALIAS DI COLONNE E TABELLE

Gli alias danno un nome temporaneo a una **tabella** o a una **colonna** di una tabella.

GATTI AS g				PADRONI AS p	
id_gatto	nome_gatto	id_genitore	id_padrone	id	nome
1	Kitty	5	1	1	John Smith
2	Hugo	1	2	2	Danielle Davis
3	Sam	2	2		
4	Misty	1	NULL		

Un **alias di colonna** rinomina una colonna nel risultato. Un **alias di tabella** rinomina una tabella all'interno della query. Se si definisce un alias di tabella, è necessario utilizzarlo al posto del nome della tabella ovunque nella query. La parola chiave AS è facoltativa nella definizione degli alias.

SELECT

```
p.nome AS nome_padrone,
g.nome_gatto
FROM gatti AS g
JOIN padroni AS p
ON g.id_padrone = p.id;
```

nome_gatto	nome_padrone
Kitty	John Smith
Sam	Danielle Davis
Hugo	Danielle Davis

## SELF JOIN

È possibile unire una tabella a se stessa, ad esempio per mostrare una relazione genitore-figlio.

GATTI AS figli				GATTI AS genitori			
id_gatto	nome_gatto	id_padrone	id_genitore	id_gatto	nome_gatto	id_padrone	id_genitore
1	Kitty	1	5	1	Kitty	1	5
2	Hugo	2	1	2	Hugo	2	1
3	Sam	2	2	3	Sam	2	2
4	Misty	NULL	1	4	Misty	NULL	1

A ogni occorrenza della tabella deve essere assegnato un **alias diverso**. Ogni riferimento a una colonna deve essere preceduto da un **alias di tabella appropriato**.

SELECT

```
figli.nome_gatto AS nome_figlio,
genitori.nome_gatto AS nome_genitore
FROM gatti AS figli
JOIN gatti AS genitori
ON figli.id_genitore = genitori.id_gatto;
```

nome_figlio	nome_genitore
Hugo	Kitty
Sam	Hugo
Misty	Kitty

## NON-EQUI SELF JOIN

È possibile utilizzare una **non-uguaglianza** nella condizione ON, ad esempio, per mostrare **tutte le diverse coppie** di righe.

GIOCHI AS a			GIOCHI AS b		
id_gioco	nome_gioco	id_gatto	id_gatto	id_gioco	nome_gioco
3	topo	1	1	3	topo
5	palla	1	1	5	palla
1	palla	3	3	1	palla
4	topo	4	4	4	topo
2	molla	NULL	NULL	2	molla

SELECT

```
a.nome_gioco AS gioco_a,
b.nome_gioco AS gioco_b
FROM giochi a
JOIN giochi b
ON a.id_gatto < b.id_gatto;
```

id_gatto_a	gioco_a	id_gatto_b	gioco_b
1	topo	3	palla
1	palla	3	palla
1	topo	4	topo
1	palla	4	topo
3	palla	4	topo

## JOIN MULTIPLI

È possibile unire più di due tabelle. Prima si uniscono due tabelle, poi si unisce la terza al risultato dell'unione precedente.

GIOCHI AS t			GATTI AS g				PADRONI AS p	
id_gioco	nome_gioco	id_gatto	id_gatto	nome_gatto	id_genitore	id_padrone	id	nome
1	palla	3	1	Kitty	5	1	1	John Smith
2	molla	NULL	2	Hugo	1	2	2	Danielle Davis
3	topo	1	3	Sam	2	2		
4	topo	4	4	Misty	1	NULL		
5	palla	1						

### JOIN E JOIN

SELECT

```
t.nome_gioco,
g.nome_gatto,
p.nome AS nome_padrone
FROM giochi t
JOIN gatti g
ON t.id_gatto = g.id_gatto
JOIN padroni p
ON g.id_padrone = p.id;
```

nome_gioco	nome_gatto	nome_padrone
palla	Kitty	John Smith
topo	Kitty	John Smith
palla	Sam	Danielle Davis

### JOIN E LEFT JOIN

SELECT

```
t.nome_gioco,
g.nome_gatto,
p.nome AS nome_padrone
FROM giochi t
JOIN gatti g
ON t.id_gatto = g.id_gatto
LEFT JOIN padroni p
ON g.id_padrone = p.id;
```

nome_gioco	nome_gatto	nome_padrone
palla	Kitty	John Smith
topo	Kitty	John Smith
palla	Sam	Danielle Davis
topo	Misty	NULL

### LEFT JOIN E LEFT JOIN

SELECT

```
t.nome_gioco,
g.nome_gatto,
p.nome AS nome_padrone
FROM giochi t
LEFT JOIN gatti g
ON t.id_gatto = g.id_gatto
LEFT JOIN padroni p
ON g.id_padrone = p.id;
```

nome_gioco	nome_gatto	nome_padrone
palla	Kitty	John Smith
topo	Kitty	John Smith
palla	Sam	Danielle Davis
topo	Misty	NULL
molla	NULL	NULL

## JOIN CON PIÙ CONDIZIONI

È possibile utilizzare più condizioni di JOIN utilizzando la parola chiave **ON** una volta e le parole chiave **AND** quante volte si vuole.

GATTI AS g					PADRONI AS p		
id_gatto	nome_gatto	id_genitore	id_padrone	eta	id	eta	nome
1	Kitty	5	1	17	1	18	John Smith
2	Hugo	1	2	10	2	10	Danielle Davis
3	Sam	2	2	5			
4	Misty	1	NULL	11			

SELECT

```
nome_gatto,
p.nome AS nome_padrone,
g.eta AS eta_gatto,
p.eta AS eta_padrone
FROM gatti g
JOIN padroni p
ON g.id_padrone = p.id
AND g.eta < p.eta;
```

nome_gatto	nome_padrone	eta_gatto	eta_padrone
Kitty	John Smith	17	18
Sam	Danielle Davis	5	10