

Come scrivere una query

Un'introduzione colorata a
SQL

di Nicola Iantomasi

Yimp – La scuola dei dati

www.yimp.it

Indice generale

Introduzione.....	5
Capitolo 1: Select, From e Where	7
La mia prima query	7
Un nuovo esercizio	14
Ora prova tu.....	17
Capitolo 2: Join tra tabelle	18
Dati su più tabelle.....	18
Un nuovo esercizio	23
Ora prova tu.....	27
Capitolo 3: Group by e Having	27
Somme, medie e conteggi.....	27
Dati aggregati	30
Un nuovo esercizio	32

Ora prova tu.....	35
Capitolo 4: Soluzioni degli esercizi	36
Informazioni sull'autore	41

Introduzione

L'SQL ha delle importanti peculiarità rispetto a tutti gli altri linguaggi di programmazione. In primo luogo si tratta di un linguaggio dichiarativo e non procedurale. Occorre dunque scrivere le istruzioni necessarie per esplicitare **cosa** occorre fare, non **come** farlo. Queste istruzioni sono scritte in un linguaggio English-like che rispecchia il modo di parlare tra gli esseri umani, piuttosto che il modo in cui daremmo informazioni ad un automa.

Diventa dunque importante studiare delle metodologie didattiche specifiche per l'SQL per insegnarlo a studenti che ci si avvicinano per la prima volta. Tali metodi dovranno essere diversi da quelli utilizzati per presentare un linguaggio di programmazione con un paradigma differente, come ad esempio Java.

Lo scopo di questo eBook è introdurre l'SQL mettendo in luce la stretta relazione tra i costrutti chiave di questo linguaggio informatico e le componenti di una frase espressa tramite l'usuale linguaggio verbale. Per enfatizzare ciò ho utilizzato i colori, evidenziando le corrispondenze tra porzioni di codice e richieste fatte in linguaggio naturale.

La sola lettura del testo sarà sicuramente efficace per comprendere i principi del linguaggio, tuttavia il mio consiglio è quello di provare a scrivere ed eseguire il codice su un vero Database Management System. A questo link trovate il file contenente il codice per creare il proprio database in ambiente MySql

https://github.com/iantomasinicola/DatabaseYimp/blob/master/ScriptDatabaseBanca_MySql.sql

o in ambiente Sql Server

https://github.com/iantomasinicola/DatabaseYimp/blob/master/ScriptDatabaseBanca_SqlServer.sql

Capitolo 1:

Select, From e Where

La mia prima query

La parola **Query** significa interrogazione, richiesta, domanda. È molto usata nel gergo degli sviluppatori SQL in quanto molto spesso vogliamo interrogare il nostro database per estrarre delle particolari informazioni.

Nei database **relazionali** i dati sono organizzati all'interno di **tabelle**. Il nostro database d'esempio si chiama **Banca** ed è formato da queste cinque tabelle:

- CarteCredito
- Clienti
- ClientiContoCorrente
- ContoCorrente
- Servizi

Le tabelle sono a loro volta organizzate in colonne, ognuna con la propria un'intestazione. Nella prossima pagina è riportato il riepilogo completo.

Tabella CarteCredito:

Colonne Carta,
CodiceFiscale,
Tipologia,
Circuito,
Saldo,
Valuta

Tabella Clienti:

Colonne CodiceFiscale,
Nome,
Cognome,
Eta,
Residenza,
Impiego

Tabella ClientiContiCorrenti:

Colonne CodiceFiscale,
Conto

Tabella ContoCorrente:

Colonne Conto,
Saldo,
Valuta,
DataApertura,
DataChiusura

Tabella Servizi:

Colonne NumeroServizio,
Conto,
TipologiaServizio,
DataApertura,
DataChiusura

La prima query che impariamo a scrivere è quella che ci permette di vedere tutti i dati presenti all'interno di una tabella, senza nessuna distinzione. Basterà scrivere

```
SELECT * FROM
```

seguito dal nome della tabella a cui siamo interessati.

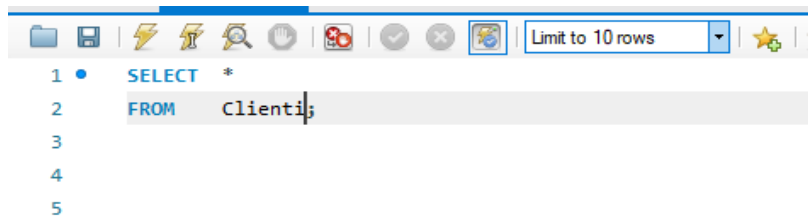
Possiamo dunque scrivere

```
SELECT * FROM CarteCredito
```

per vedere tutti i dati delle carte di credito, o

```
SELECT * FROM Clienti
```

per le informazioni sui clienti.



A screenshot of a data grid interface showing the results of the SQL query. The grid has a toolbar with 'Result Grid', 'Filter Rows', 'Edit', and 'Export/Import' options. The data is presented in a table with the following columns: CodiceFiscale, Nome, Cognome, Eta, Residenza, and Impiego.

	CodiceFiscale	Nome	Cognome	Eta	Residenza	Impiego
▶	AWNNLZ36R05E168T	Giovanni	Verdi	45	Puglia	Impiegato
	BHLVVP64C64F735E	Alice	Gianni	34	Sicilia	Dirigente
	BHLVVR64R64F635E	Cristina	Gianni	56	Lombardia	Presidente
	CDPKXR28E62A973B	Grazia	Fabrizi	59	Toscana	Disoccupato
	DWNNLZ36P05E168T	Maria	Verdi	54	Toscana	Dirigente
	FLTXXI58H51E295S	Mike	Moratti	36	Toscana	Professore
	FLTXRI58H51E295S	Mino	Moratti	56	Lombardia	Commerciante
	GMCGYC77C17D591G	Nicola	Motta	66	Toscana	Calciatore

Molto spesso però vogliamo **interrogare** il database per estrarre soltanto determinate informazioni. Ad esempio supponiamo di voler estrarre

- il codice fiscale
- la tipologia
- il numero

delle carte di credito con:

- valuta uguale a euro
- saldo strettamente maggiore di 30

La nostra nuova query avrà questa struttura

SELECT

FROM

WHERE

Osserviamo la presenza di una nuova parola chiave: **WHERE**.
Ne capiremo presto l'utilità.

Il nostro obiettivo sarà quello di completare la query riempiendo i puntini lasciati precedentemente in bianco. Molto spesso lo spazio più facile da completare è quello dopo la **FROM**: basterà aggiungere la **tabella di riferimento** dell'estrazione.

Rivediamo il nostro esercizio aggiungendo il colore Giallo per sottolineare l'informazione sulla tabella di riferimento
Selezionare:

- il codice fiscale
- la tipologia
- il numero

delle **carte di credito** con:

- valuta uguale a euro
- saldo strettamente maggiore di 30

La nostra query sarà dunque

```
SELECT .....
FROM CarteCredito
WHERE .....
```

A questo punto riscriviamo la query che abbiamo imparato precedentemente per analizzare l'intero contenuto della tabella CarteCredito

```
SELECT *
FROM CarteCredito;
```

The screenshot shows a database query editor with a toolbar at the top. The query entered is:

```
1 SELECT *
2 FROM CarteCredito
3
4
5
```

Below the query editor, a "Result Grid" is displayed with the following data:

	Carta	CodiceFiscale	Tipologia	Circuito	Saldo	Valuta
▶ 1	SRDLZQ40M58A184Y		Visa	Maestro	12.13	EUR
10	CDPKXR28E62A973B		Visa	Maestro	41.20	EUR
11	RQMNHF29T53I482R		Visa	Maestro	12.13	USD
12	VRNVVF88A50B646H		Mastercard	Maestro	26.12	EUR
13	TRRNZN61A46F853F		Visa	Maestro	12.13	EUR
14	BHLVVR64R64F635E		Mastercard	Maestro	53.10	EUR
15	MXXHRR88A02A895O		Visa	Maestro	12.13	EUR
16	RVRKTO80T64G614O		Mastercard	Maestro	12.99	EUR

Analizziamo quali condizioni devono verificare le righe della tabella CarteCredito. Coloriamole di verde per poi inserirle nella clausola **WHERE**.

Selezionare:

- il codice fiscale
- la tipologia
- il numero

delle **carte di credito** con:

- valuta uguale a euro
- saldo strettamente maggiore di 30

La condizione **valuta uguale a euro** può essere tradotta in Sql con

Valuta = 'EUR'

Mentre la condizione **saldo strettamente maggiore di 30** corrisponde a

Saldo > 30

Dobbiamo infine scegliere l'operatore logico che collegherà le due condizioni. In questo caso devono essere entrambe verificate, di conseguenza sceglieremo l'**AND**. La query diventerà dunque

SELECT

FROM CarteCredito

WHERE Valuta='EUR'
AND Saldo > 30

Valutiamo infine quali colonne devono essere riportate e coloriamole di azzurro.

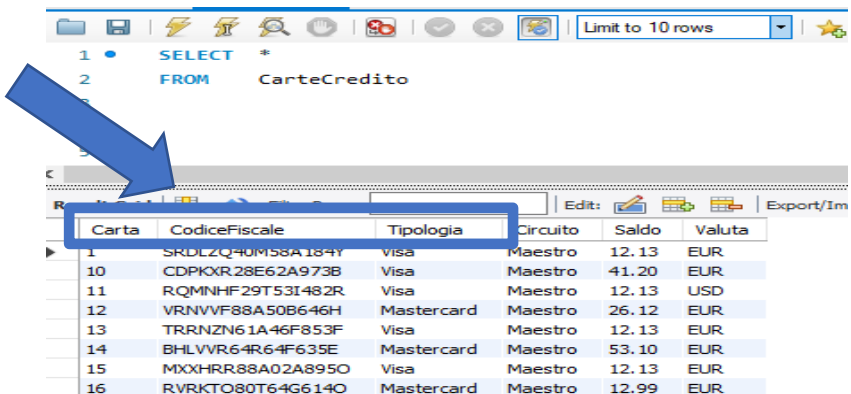
Selezionare:

- il codice fiscale
- la tipologia
- il numero

delle **carte di credito** con:

- valuta uguale a euro
- saldo strettamente maggiore di 30

Osserviamo di nuovo le colonne della tabella e selezioniamo quelle di nostro interesse



The screenshot shows a database query editor with a table named 'CarteCredito'. The table has the following columns: Carta, CodiceFiscale, Tipologia, Circuito, Saldo, and Valuta. The first three columns are highlighted in blue. A blue arrow points to this selection. The table data is as follows:

	Carta	CodiceFiscale	Tipologia	Circuito	Saldo	Valuta
1		SRDLZQ40M58A184Y	Visa	Maestro	12.13	EUR
10		CDPKXR28E62A973B	Visa	Maestro	41.20	EUR
11		RQMNHF29T53I482R	Visa	Maestro	12.13	USD
12		VRNVVF88A50B646H	Mastercard	Maestro	26.12	EUR
13		TRRNZN61A46F853F	Visa	Maestro	12.13	EUR
14		BHLVVR64R64F635E	Mastercard	Maestro	53.10	EUR
15		MXXHRR88A02A895O	Visa	Maestro	12.13	EUR
16		RVRKTO80T64G614O	Mastercard	Maestro	12.99	EUR

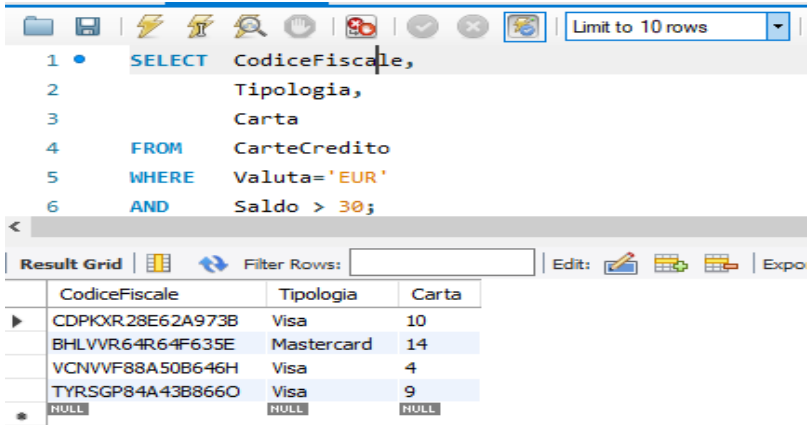
Le inseriremo nella **SELECT** separandole con una virgola

```
SELECT CodiceFiscale,  
       Tipologia,  
       Carta
```

```
FROM CarteCredito
```

```
WHERE Valuta='EUR'  
AND Saldo > 30
```

Ricordiamoci di inserire alla fine il punto e virgola.
Ecco il risultato su MySql:



The screenshot shows a MySQL query editor window. The query is as follows:

```
1 • SELECT CodiceFiscale,  
2     Tipologia,  
3     Carta  
4 FROM CarteCredito  
5 WHERE Valuta='EUR'  
6 AND Saldo > 30;
```

Below the query, the 'Result Grid' is displayed with the following data:

CodiceFiscale	Tipologia	Carta
CDPKXR.28E62A973B	Visa	10
BHLVVR64R64F635E	Mastercard	14
VCNVVF88A50B646H	Visa	4
TYRSGP84A43B866O	Visa	9
NULL	NULL	NULL

Un nuovo esercizio

Selezionare nome e cognome dei clienti con età compresa tra i 40 e i 50 anni e residenti in Puglia o in Sicilia.

Analizziamo la richiesta individuando le informazioni necessarie per le clausole **SELECT**, **FROM** e **WHERE**.

Selezionare **nome e cognome** dei **clienti** con **età compresa tra i 40 e i 50 anni e residenti in Puglia o in Sicilia**

Anche questa query avrà la solita struttura

```
SELECT .....
```

```
FROM .....
```

```
WHERE .....
```

La tabella da cui estrarre le informazioni è la tabella **Clienti**

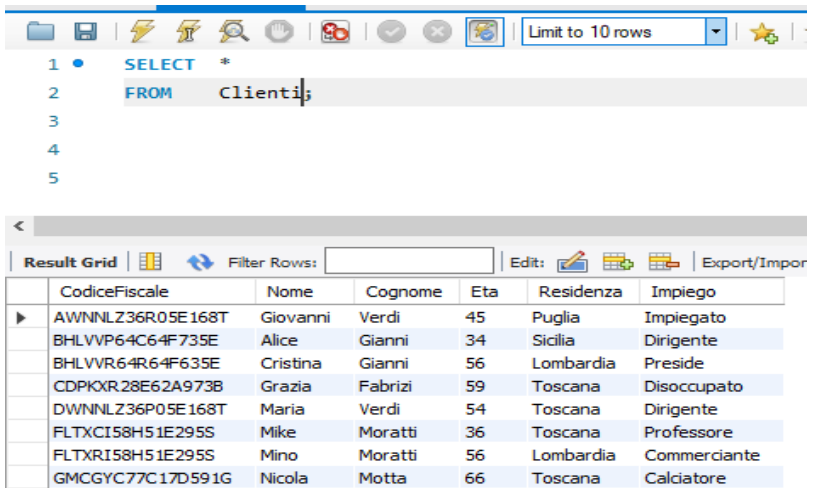
```
SELECT .....
```

```
FROM Clienti
```

```
WHERE .....
```

Analizziamo le colonne e il contenuto della tabella

```
SELECT *  
FROM Clienti;
```



The screenshot shows a database query editor interface. At the top, there is a toolbar with various icons and a dropdown menu set to "Limit to 10 rows". Below the toolbar, the SQL query is entered in a text area:

```
1 • SELECT *  
2 FROM Clienti;  
3  
4  
5
```

Below the query editor, the "Result Grid" is displayed, showing the data returned by the query. The grid has the following columns: CodiceFiscale, Nome, Cognome, Eta, Residenza, and Impiego. The data is as follows:

	CodiceFiscale	Nome	Cognome	Eta	Residenza	Impiego
▶	AWNNLZ36R05E168T	Giovanni	Verdi	45	Puglia	Impiegato
	BHLVVP64C64F735E	Alice	Gianni	34	Sicilia	Dirigente
	BHLVVR64R64F635E	Cristina	Gianni	56	Lombardia	Preside
	CDPKXR28E62A973B	Grazia	Fabrizi	59	Toscana	Disoccupato
	DWNNLZ36P05E168T	Maria	Verdi	54	Toscana	Dirigente
	FLTXXCI58H51E295S	Mike	Moratti	36	Toscana	Professore
	FLTXXRI58H51E295S	Mino	Moratti	56	Lombardia	Commerciante
	GMCGYC77C17D591G	Nicola	Motta	66	Toscana	Calciatore

La condizione età compresa tra i 40 e i 50 anni può essere tradotta in SQL con

```
eta >= 40 AND eta <= 50
```

Mentre la condizione residenti in Puglia o in Sicilia corrisponde a

```
Residenza = 'Puglia' OR Residenza = 'Sicilia'
```

Racchiudiamo ognuna delle due condizioni tra parentesi e colleghiamole con l'operatore AND, coerentemente con la richiesta

```
SELECT .....
```

```
FROM Clienti
```

```
WHERE (eta >= 40 and eta <= 50)  
AND ( Residenza = 'Puglia'  
      OR  
      Residenza = 'Sicilia')
```

Inseriamo infine le colonne richieste nella Select

```
SELECT Nome,  
       Cognome
```

```
FROM Clienti
```

```
WHERE (eta >= 40 and eta <= 50)  
AND ( Residenza = 'Puglia'  
      OR  
      Residenza = 'Sicilia');
```

Ricordiamoci di inserire alla fine il punto e virgola.

The screenshot shows a SQL query editor with the following query:

```
1 SELECT Nome,  
2     Cognome  
3 FROM Clienti  
4 WHERE (eta >= 40 and eta <= 50)  
5 AND (Residenza = 'Puglia' or  
6     Residenza = 'Sicilia');
```

Below the query, the results are displayed in a table:

Nome	Cognome
Giovanni	Verdi
Seth	Rossi
Fabrizio	Fabrizi
Maria	Paoli

Ora prova tu

1. Selezionare tutte le informazioni sui conti che rispettano almeno una delle seguenti condizioni
 - la valuta è il dollaro
 - la valuta è l'euro e l'importo è maggiore di 1000
2. Selezionare il numero dei conti correnti aperti nell'ultimo trimestre del 2018.
3. Selezionare il numero e il saldo dei conti correnti con valuta euro, aperti a ottobre 2018 e ottobre 2019, con saldo compreso tra 1000 e 2000 euro.

Le soluzioni sono presenti nell'ultimo capitolo.

Capitolo 2:

Join tra tabelle

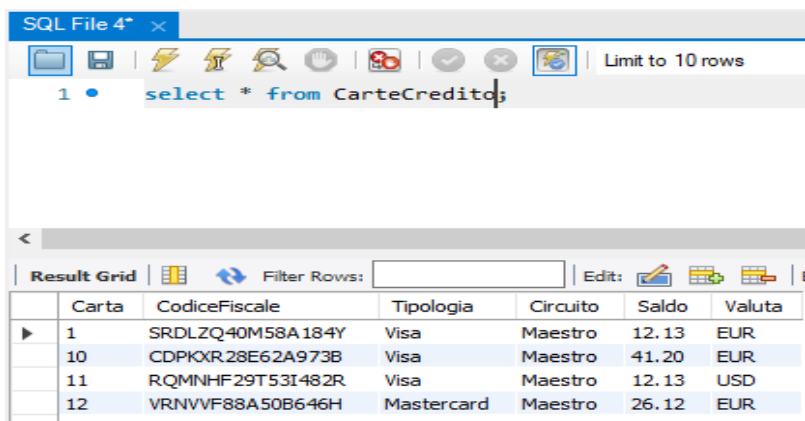
Dati su più tabelle

Nelle domande poste finora tutti i dati e i filtri richiesti facevano riferimento ad un'unica tabella.

Come ci comportiamo invece con domande di questo genere?

*Riportare per ogni **carta di credito** il suo **numero**, il suo **saldo**, il **nome** e il **cognome** del relativo **cliente**.*

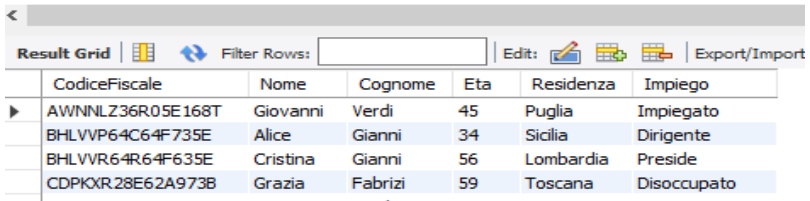
In questo caso le colonne appartengono **a due tabelle differenti**. Il numero della carta e il relativo saldo si trovano nella tabella **CarteCredito**.



The screenshot shows a SQL IDE window titled "SQL File 4* x". The query editor contains the SQL statement: `select * from CarteCredito;`. Below the editor, the "Result Grid" displays the following data:

	Carta	CodiceFiscale	Tipologia	Circuito	Saldo	Valuta
▶	1	SRDLZQ40M58A184Y	Visa	Maestro	12.13	EUR
	10	CDPKXR28E62A973B	Visa	Maestro	41.20	EUR
	11	RQMNHF29T53I482R	Visa	Maestro	12.13	USD
	12	VRNVVF88A50B646H	Mastercard	Maestro	26.12	EUR

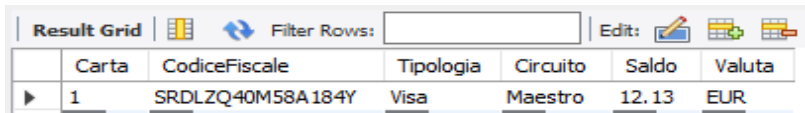
Invece il nome e il cognome dei clienti si trovano nella tabella **Clienti**.



A screenshot of a "Result Grid" window displaying a table of client data. The table has the following columns: CodiceFiscale, Nome, Cognome, Eta, Residenza, and Impiego. The data rows are:

	CodiceFiscale	Nome	Cognome	Eta	Residenza	Impiego
▶	AWNNLZ36R05E168T	Giovanni	Verdi	45	Puglia	Impiegato
	BHLVVP64C64F735E	Alice	Gianni	34	Sicilia	Dirigente
	BHLVVR64R64F635E	Cristina	Gianni	56	Lombardia	Presidente
	CDPKXR28E62A973B	Grazia	Fabrizi	59	Toscana	Disoccupato

Occorre dunque trovare un criterio per **collegare** le due tabelle. Ci chiediamo ad esempio: quale riga della tabella *Clienti* corrisponde alla *carta numero 1*?



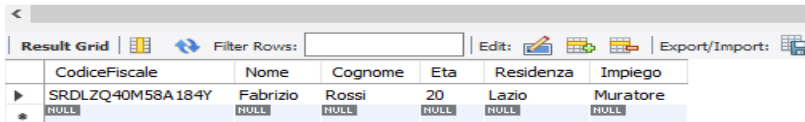
A screenshot of a "Result Grid" window displaying a table of card data. The table has the following columns: Carta, CodiceFiscale, Tipologia, Circuito, Saldo, and Valuta. The data row is:

	Carta	CodiceFiscale	Tipologia	Circuito	Saldo	Valuta
▶	1	SRDLZQ40M58A184Y	Visa	Maestro	12.13	EUR

In questo caso la risposta è abbastanza intuitiva: colleghiamo la carta con la riga della tabella *Clienti* **avente lo stesso CodiceFiscale**.



```
1 • select * from clienti where CodiceFiscale='SRDLZQ40M58A184Y';
```



	CodiceFiscale	Nome	Cognome	Eta	Residenza	Impiego
▶	SRDLZQ40M58A184Y	Fabrizio	Rossi	20	Lazio	Muratore
*	NULL	NULL	NULL	NULL	NULL	NULL

Per collegare le due tabelle secondo il criterio appena individuato dobbiamo inserire dei nuovi costrutti all'interno della clausola **FROM**: si tratta di **INNER JOIN** e **ON**.

SELECT

FROM **CarteCredito**

INNER JOIN **Clienti**

ON **CarteCredito.CodiceFiscale** =
Clienti.CodiceFiscale

Per “alleggerire” questa sintassi possiamo rinominare le tabelle nella query tramite la parola **AS**

SELECT

FROM **CarteCredito AS Cc**

INNER JOIN **Clienti AS Cl**

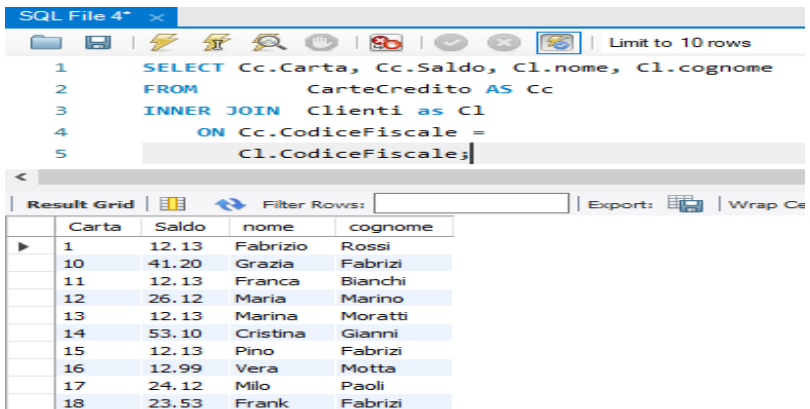
ON **Cc.CodiceFiscale** =
Cl.CodiceFiscale

Possiamo ora rispondere alla domanda di partenza:

Riportare per ogni carta di credito il suo numero, il suo saldo, il nome e il cognome del relativo cliente

```
SELECT Cc.Carta,  
       Cc.Saldo,  
       Cl.nome,  
       Cl.cognome  
  
FROM   CarteCredito AS Cc  
  
INNER JOIN Clienti AS Cl  
  
ON Cc.CodiceFiscale =  
   Cl.CodiceFiscale;
```

Ecco il risultato su MySQL.



The screenshot shows a MySQL SQL editor window titled "SQL File 4*" with a toolbar and a "Limit to 10 rows" option. The SQL query is displayed in a text area, and below it, the "Result Grid" shows the output of the query. The result grid has columns for Carta, Saldo, nome, and cognome, and rows for each record.

	Carta	Saldo	nome	cognome
▶	1	12.13	Fabrizio	Rossi
	10	41.20	Grazia	Fabrizi
	11	12.13	Franca	Bianchi
	12	26.12	Maria	Marino
	13	12.13	Marina	Moratti
	14	53.10	Cristina	Gianni
	15	12.13	Pino	Fabrizi
	16	12.99	Vera	Motta
	17	24.12	Milo	Paoli
	18	23.53	Frank	Fabrizi

Le tipologie di Join

Ma cosa succede se il codice fiscale associato alla carta di credito **non** ha nessun record corrispondente nella tabella dei clienti?

La risposta è che la query precedente **non** mostrerà queste carte di credito!

Se vogliamo visualizzare le carte di credito senza clienti associati, dobbiamo modificare la tipologia di Join passando da **INNER JOIN** a **LEFT JOIN**. La nuova query sarà dunque

```
SELECT Cc.Carta,  
       Cc.Saldo,  
       Cl.nome,  
       Cl.cognome  
  
FROM   CarteCredito AS Cc
```

LEFT JOIN Clienti as Cl

```
ON     Cc.CodiceFiscale =  
       Cl.CodiceFiscale;
```

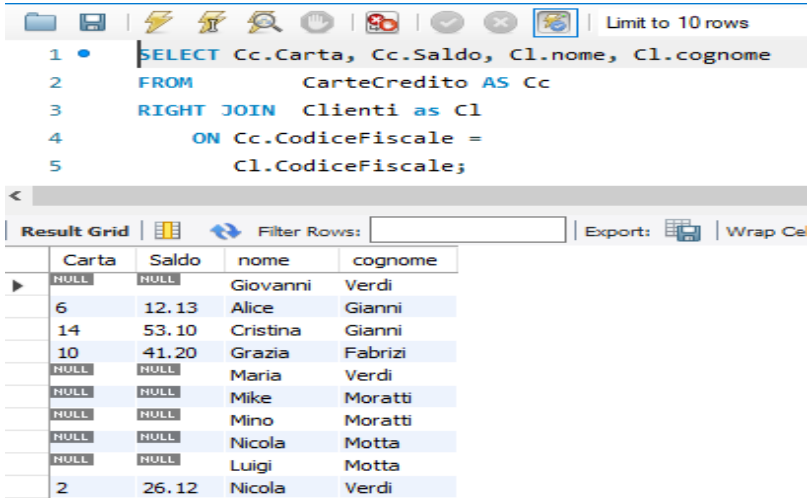
Analogamente, se volessimo visualizzare i Clienti che **non** hanno carte di credito associate, modificheremmo **INNER JOIN** in **RIGHT JOIN**

```
SELECT Cc.Carta,  
       Cc.Saldo,  
       Cl.nome,  
       Cl.cognome  
  
FROM   CarteCredito AS Cc
```

RIGHT JOIN Clienti as Cl

ON Cc.CodiceFiscale =
Cl.CodiceFiscale;

In questo caso le colonne della tabella CarteCredito senza clienti associati riporteranno dei **null**.



```
1 SELECT Cc.Carta, Cc.Saldo, Cl.nome, Cl.cognome
2 FROM CarteCredito AS Cc
3 RIGHT JOIN Clienti as Cl
4 ON Cc.CodiceFiscale =
5 Cl.CodiceFiscale;
```

Result Grid

	Carta	Saldo	nome	cognome
▶	NULL	NULL	Giovanni	Verdi
6	12.13		Alice	Gianni
14	53.10		Cristina	Gianni
10	41.20		Grazia	Fabrizi
▶	NULL	NULL	Maria	Verdi
▶	NULL	NULL	Mike	Moratti
▶	NULL	NULL	Mino	Moratti
▶	NULL	NULL	Nicola	Motta
▶	NULL	NULL	Luigi	Motta
	2	26.12	Nicola	Verdi

Un nuovo esercizio

Riportare per ogni **conto corrente** il suo **numero**, il suo **saldo**, il **codice fiscale** e la **residenza** dei **clienti**.

Il numero del conto e il saldo sono colonne della tabella **ContoCorrente**.

```
1 • select * from ContoCorrente;
```

Conto	Saldo	Valuta	DataApertura	DataChiusura
1	1000.12	EUR	2018-10-10 00:00:00	NULL
10	4001.20	EUR	2019-02-05 00:00:00	2019-08-30 00:00:00
11	412.50	USD	2018-08-08 00:00:00	NULL
12	55.10	EUR	2019-12-04 00:00:00	NULL

Il codice fiscale e la residenza dei clienti si trovano nella tabella **Clienti**.

```
1 • select * from Clienti;
```

CodiceFiscale	Nome	Cognome	Eta	Residenza	Impiego
AWNNLZ36R05E168T	Giovanni	Verdi	45	Puglia	Impiegato
BHLVVP64C64F735E	Alice	Gianni	34	Sicilia	Dirigente
BHLVVR64R64F635E	Cristina	Gianni	56	Lombardia	Preside
CDPKXR28E62A973B	Grazia	Fabrizi	59	Toscana	Disoccupato

La relazione **ContiCorrenti-Clienti** è molti a molti. Ciò significa che:

- un cliente può avere più conti correnti
- uno stesso conto corrente può essere associato a più persone (ad esempio se il conto è cointestato).

Queste associazioni sono riportate nella tabella di nome **ClientiContiCorrenti**


```
1 • select * from ClientiContiCorrente;  
2  
3
```

Result Grid | Filter Rows: | Edit: [

	CodiceFiscale	Conto
▶	JHLTTS66P17H911Y	1
	SRDLZQ40M58A184Y	1
	CDPKXR28E62A973B	10
	LTTLTT63P56Z327R	11
	DWNINLZ36P05E168T	12
	LTTLTT63P56Z327R	12

Per collegare le tabelle ContoCorrente e Cliente abbiamo dunque bisogno di **due JOIN**:

- una tra la tabella **ContoCorrente** e la tabella **ClientiContiCorrenti**
- una tra la tabella **ClientiContiCorrenti** e la tabella **Clienti**

Dobbiamo ora stabilire i criteri di JOIN. Si tratta di scelte “naturali”:

- tra **ContoCorrente** e **ClientiContiCorrenti** dobbiamo considerare l’uguaglianza tra il **numero dei conti**.
- tra **ClientiContiCorrenti** e **Clienti** dobbiamo considerare l’uguaglianza tra il **codice fiscale**.

In definitiva, al netto di riconsiderare la **tipologia** di Join in base al comportamento desiderato, scriveremo la query

```
SELECT Cc.Conto,  
       Cc.Saldo,  
       Cl.CodiceFiscale,  
       Cl.Residenza  
FROM   ContoCorrente AS Cc  
INNER JOIN ClientiContiCorrente As Associazione  
ON Cc.Conto = Associazione.Conto  
INNER JOIN Clienti As Cl  
ON Associazione.CodiceFiscale = Cl.CodiceFiscale;
```

Osserviamo attentamente l'output della query su MySql

Conto	Saldo	CodiceFiscale	Residenza
1	1000.12	JHLTTS66P17H911Y	Lazio
1	1000.12	SRDLZQ40M58A184Y	Lazio
10	4001.20	CDPKRZ8L8Z8973B	Toscana
11	412.50	LTTLTT63P56Z327R	Toscana
12	55.10	DWNNLZ36P05E168T	Toscana
12	55.10	LTTLTT63P56Z327R	Toscana
13	5672.20	JTTGNG66M45G4062	Toscana
14	234.50	KDTNTH80H51A274L	Toscana
15	12.40	FLTXXCI58H51E295S	Toscana
16	342.10	MRZQDC66P17A281I	Toscana

Notiamo che i conti numero 1 e 12 e i rispettivi saldi sono ripetuti in **due righe differenti**. Si tratta infatti di conti cointestati a due clienti con codici fiscali distinti.

Conto	Saldo	CodiceFiscale	Residenza
1	1000.12	JHLTTS66P17H911Y	Lazio
1	1000.12	SRDLZQ40M58A184Y	Lazio

A questo punto il totale della colonna **Saldo** non conterrà più il valore complessivo di tutti i conti correnti, ma un numero **più grande** perché alcuni conti sono riportati più di una volta.

In generale occorre dunque fare attenzione ad utilizzare i dati dopo un'operazione di Join poiché alcune informazioni potrebbero risultare duplicate.

Ora prova tu

- 1) Per ogni servizio, riportare il numero del servizio, il numero del conto, la data di apertura del conto e la data di apertura del servizio
- 2) Riportare per ogni carta di credito, il numero della carta, la tipologia, il circuito, il CodiceFiscale del cliente associato e il suo impiego. Se un cliente non ha carte associate, riportare comunque i suoi dati.

Le soluzioni sono presenti nell'ultimo capitolo insieme all'elenco delle colonne per ogni tabella.

Capitolo 3: Group by e Having

Somme, medie e conteggi

Vediamo ora come rispondere a domande di questa tipologia:

- *Quanti conti correnti sono presenti?*
- *Qual è la somma dei saldi dei conti correnti?*
- *Qual è la media dei saldi delle carte di credito?*

Partiamo dalla prima domanda:

Quanti conti correnti sono presenti?

Inseriamo la tabella ContoCorrente nella **From**.

```
SELECT ...
```

```
FROM ContoCorrente
```

Questa volta nella Select non dobbiamo inserire delle colonne, ma una funzione che conti il numero di righe. Il nome di questa funzione è **COUNT(*)**, coloriamola di grigio.

```
SELECT COUNT(*)
```

```
FROM ContoCorrente
```

Analogamente a Count(*), esistono altre funzioni che permettono di aggregare i dati:

SUM(<Colonna>): riporta la somma dei valori in Colonna

AVG(<Colonna>): riporta la media dei valori in Colonna

MIN(<Colonna>): riporta il minimo dei valori in Colonna

MAX(<Colonna>): riporta il massimo dei valori in Colonna

Tramite queste funzioni possiamo rispondere alle altre due domande poste all'inizio:

Qual è la somma dei saldi dei conti correnti?

```
SELECT SUM(Saldo)
```

```
FROM ContoCorrente
```

Qual è la media dei saldi delle carte di credito?

```
SELECT AVG(Saldo)
```

```
FROM CarteCredito
```

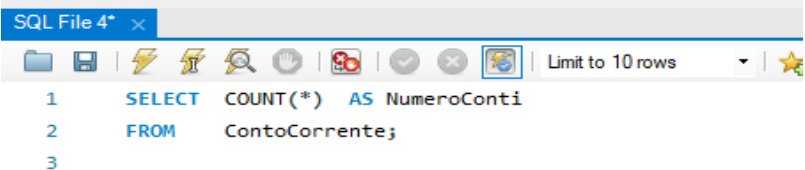
Per assegnare un nome a queste valori possiamo utilizzare **AS**

```
SELECT COUNT(*) AS NumeroConti  
FROM ContoCorrente
```

```
SELECT SUM(Saldo) AS SommaSaldoConti  
FROM ContoCorrente
```

```
SELECT AVG(Saldo) AS MediaSaldoConti  
FROM CarteCredito
```

Vediamo un esempio su MySql



The screenshot shows a MySQL SQL editor window titled "SQL File 4* x". The toolbar includes icons for file operations, execution, and search. The query text is as follows:

```
1 SELECT COUNT(*) AS NumeroConti  
2 FROM ContoCorrente;  
3
```



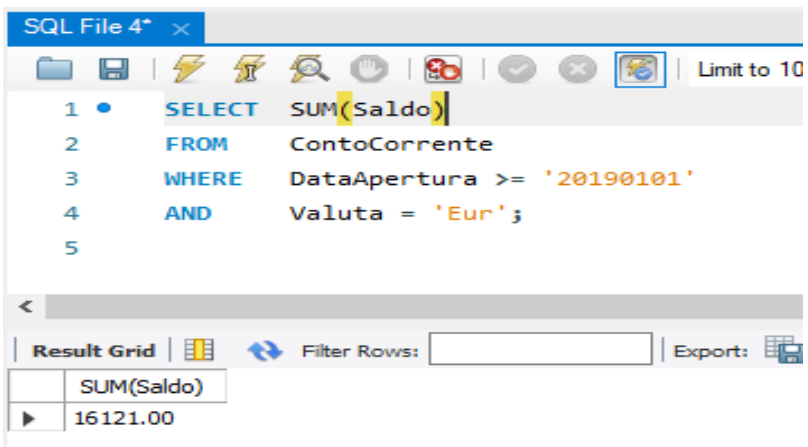
The screenshot shows the MySQL result grid window. The toolbar includes "Result Grid", "Filter Rows", "Export", and "Wrap Cell Content". The result grid contains the following data:

NumeroConti
40

Aggiungendo una clausola **where**, possiamo rispondere a domande più complesse.

Calcolare la somma dei saldi dei conti correnti aperti dal primo gennaio 2019 e con valuta Euro

```
SELECT SUM(Saldo)
FROM ContoCorrente
WHERE DataApertura >= '20190101'
AND Valuta = 'EUR';
```



The screenshot shows a SQL IDE window titled "SQL File 4* x". The query editor contains the following SQL code:

```
1 SELECT SUM(Saldo)
2 FROM ContoCorrente
3 WHERE DataApertura >= '20190101'
4 AND Valuta = 'Eur';
5
```

Below the editor, the "Result Grid" is displayed with the following data:

SUM(Saldo)
16121.00

Dati aggregati

Riportare il numero di conti correnti per ogni valuta.

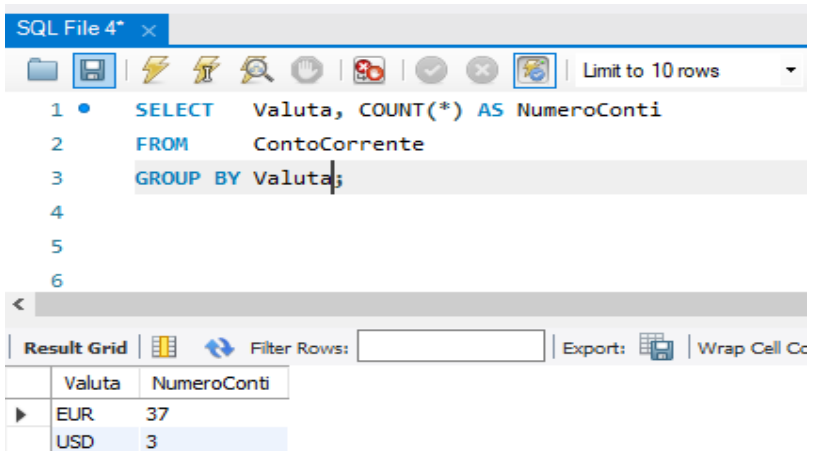
Si tratta di un caso nuovo: nell'output della query deve essere presente una riga per ogni valuta presente nella tabella dei conti correnti, con il relativo conteggio.

Abbiamo bisogno di aggiungere una nuova clausola: la **GROUP BY**.

Analizziamo la domanda colorando le varie componenti:

Riportare il numero di conti correnti per ogni valuta.

```
SELECT Valuta,  
       COUNT(*) AS NumeroConti  
  
FROM ContoCorrente  
  
GROUP BY Valuta;
```



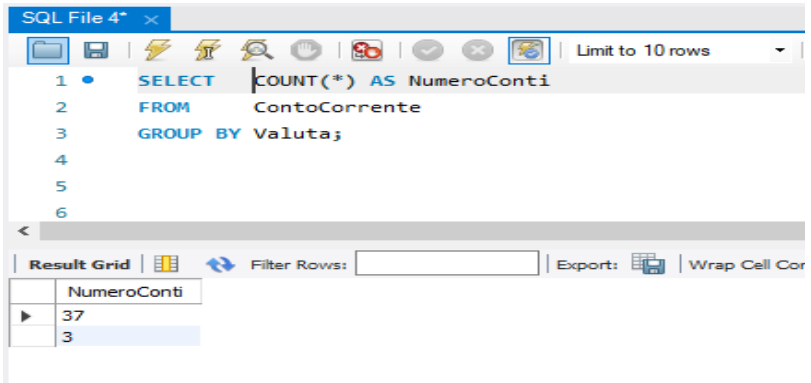
The screenshot shows a SQL IDE window titled "SQL File 4* x". The query editor contains the following SQL code:

```
1 • SELECT Valuta, COUNT(*) AS NumeroConti  
2 FROM ContoCorrente  
3 GROUP BY Valuta;  
4  
5  
6
```

Below the query editor, the "Result Grid" is displayed. It shows a table with two columns: "Valuta" and "NumeroConti". The results are as follows:

Valuta	NumeroConti
EUR	37
USD	3

Osserviamo che è necessario riportare la colonna *Valuta* anche nella **select**, altrimenti non sapremmo a quale valuta corrispondo i totali.



Un nuovo esercizio

Riportare la somma dei saldi delle carte di credito divise per Tipologia e Valuta.

In questo caso nell'output deve esserci una riga **per ogni coppia distinta di tipologia e valuta**. Procediamo con i colori.

Riportare la **somma dei saldi** delle **carte di credito** divise per **Tipologia** e **valuta**.

```

SELECT Tipologia,
       Valuta,
       SUM(Saldo) AS SommaSaldo

```

```

FROM CarteCredito

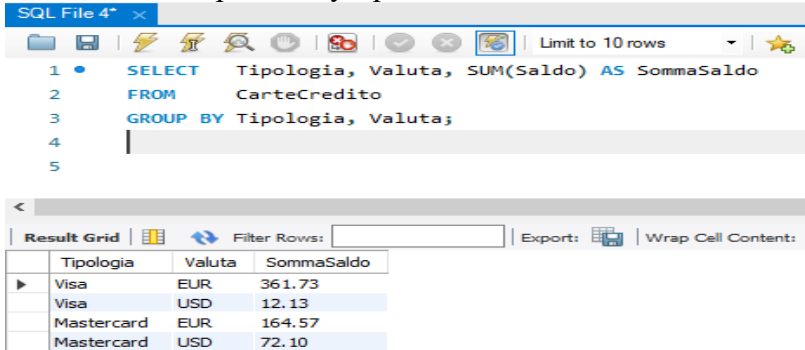
```

```

GROUP BY Tipologia,
         Valuta;

```

Vediamo l'esempio su MySQL



The screenshot shows a MySQL SQL editor window titled "SQL File 4* x". The query editor contains the following SQL code:

```
1 • SELECT Tipologia, Valuta, SUM(Saldo) AS SommaSaldo
2 FROM CarteCredito
3 GROUP BY Tipologia, Valuta;
4
5
```

Below the query editor is the "Result Grid" window. It displays a table with the following data:

Tipologia	Valuta	SommaSaldo
▶ Visa	EUR	361.73
▶ Visa	USD	12.13
▶ Mastercard	EUR	164.57
▶ Mastercard	USD	72.10

Aggiungiamo un ulteriore vincolo alla richiesta.

*Riportare la somma dei saldi delle carte di credito divise per
Tipologia e Valuta, se tale somma è superiore a 50.*

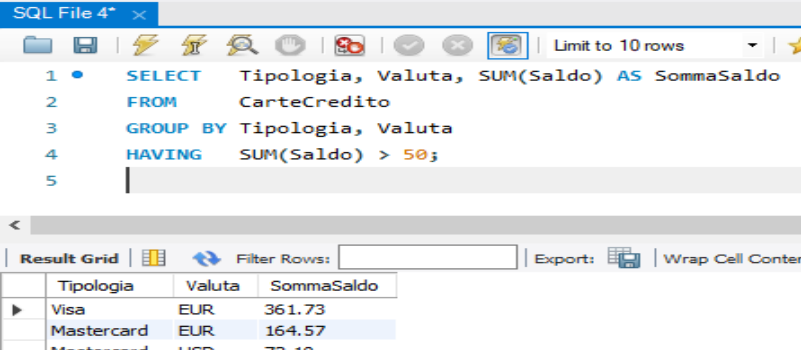
Ci viene richiesto un filtro aggiuntivo sulla somma dei saldi correnti. Tuttavia non possiamo utilizzare una **WHERE** perché si tratta di un filtro relativo ad un dato raggruppato (la **somma** dei saldi).

La nuova parola chiave da usare è **HAVING**, coloriamo di viola il filtro relativo.

*Riportare la somma dei saldi delle carte di credito divise per
Tipologia e valuta, se tale somma è superiore a 50.*

```
SELECT Tipologia,  
Valuta,  
SUM(Saldo) AS SommaSaldo  
  
FROM CarteCredito  
  
GROUP BY Tipologia,
```

HAVING Valuta SUM(Saldo) > 50;



The screenshot shows a SQL query editor window titled "SQL File 4*" with a toolbar and a "Limit to 10 rows" dropdown. The query is as follows:

```
1 SELECT Tipologia, Valuta, SUM(Saldo) AS SommaSaldo
2 FROM CarteCredito
3 GROUP BY Tipologia, Valuta
4 HAVING SUM(Saldo) > 50;
5
```

Below the query is a "Result Grid" window showing the results of the query. The grid has columns for Tipologia, Valuta, and SommaSaldo. The results are:

Tipologia	Valuta	SommaSaldo
▶ Visa	EUR	361.73
Mastercard	EUR	164.57
Mastercard	USD	72.10

Vediamo un ultimo esempio che utilizza tutti i costrutti visti finora.

Riportare il numero di conti correnti aperti dopo il primo gennaio 2019, divisi per valuta, se la media dei saldi è superiore a 100.

```
SELECT Valuta,
COUNT(*) AS NumeroConti
FROM ContoCorrente
WHERE DataApertura >= '20190101'
GROUP BY Valuta
HAVING AVG(saldo) > 100;
```

The screenshot shows a SQL IDE window titled "SQL File 4*" with a toolbar and a query editor. The query is as follows:

```
1 SELECT Valuta, COUNT(*) AS NumeroConti
2 FROM ContoCorrente
3 WHERE DataApertura>='20190101'
4 GROUP BY Valuta
5 HAVING AVG(saldo) > 100;
6
```

Below the query editor is a "Result Grid" section with a "Filter Rows" input field and "Export" and "Wrap Cell Cor" buttons. The result grid contains the following data:

Valuta	NumeroConti
▶ EUR	19

Ora prova tu

- 1) Contare i clienti con età maggiore di 30 anni
- 2) Contare, tra quelli residenti in Puglia, Sicilia, Lombardia, i clienti divisi per il loro impiego.
- 3) Calcolare il numero dei conti correnti divisi per Valuta, se la relativa somma dei saldi è maggiore di 100.

Capitolo 4:

Soluzioni degli esercizi

Struttura Database

Ripetiamo per comodità la struttura del database anche in questo capitolo

Tabella **CarteCredito:**

Colonne Carta,
CodiceFiscale,
Tipologia,
Circuito,
Saldo,
Valuta

Tabella **Clients:**

Colonne CodiceFiscale,
Nome,
Cognome,
Eta,
Residenza,
Impiego

Tabella **ClientsContiCorrenti:**

Colonne CodiceFiscale,
Conto

Tabella **ContoCorrente:**

Colonne Conto,
Saldo,
Valuta,
DataApertura,

DataChiusura

Tabella **Servizi**:

Colonne NumeroServizio,
Conto,
TipologiaServizio,
DataApertura,
DataChiusura

Capitolo 1

Selezionare tutte le informazioni sui conti che rispettano almeno una delle seguenti condizioni

la valuta è il dollaro

la valuta è l'euro e l'importo è maggiore di 1000

```
SELECT *
```

```
FROM ContiCorrente
```

```
WHERE Valuta = 'USD'
```

```
OR Importo > 1000
```

Selezionare il numero dei conti correnti aperti nell'ultimo trimestre del 2018.

```
SELECT NumeroConto
```

```
FROM ContiCorrente
```

```
WHERE DataApertura >= '20181001'
```

```
AND DataApertura < '20190101'
```

Selezionare il numero e il saldo dei conti correnti con valuta euro, aperti a ottobre 2018 o ottobre 2019, con saldo compreso tra 1000 e 2000 euro.

Fare attenzione alle parentesi:

```
SELECT NumeroConto,  
       saldo  
  
FROM   ContiCorrente  
  
WHERE  
(  
(DataApertura >= '20181001' AND DataApertura < '20181101')  
OR  
(DataApertura >= '20191001' AND DataApertura < '20191101') )  
  
AND  
  
(Saldo >= 1000 and Saldo <=2000)
```

Capitolo 2

Per ogni servizio, riportare il numero del servizio, il numero del conto, la data di apertura del conto e la data di apertura del servizio

```
SELECT s.NumeroServizio,  
       s.Conto,  
       c.DataApertura,  
       c.DataChiusura  
  
FROM   Servizi AS s
```

```
INNER JOIN Conti AS c
      ON s.Conto=c.Conto
```

Riportare per ogni carta di credito, il numero della carta, la tipologia, il circuito, il codice fiscale del cliente associato e il suo impiego. Se un cliente non ha carte associate, riportare comunque i suoi dati.

```
SELECT cc.Carta,
       cc.Tipologia,
       cl.CodiceFiscale
       cl.Impiego
FROM   CarteCredito AS cc

RIGHT JOIN Clienti AS cl
      ON cc.CodiceFiscale =
         cl. CodiceFiscale
```

Capitolo 3

Contare i clienti con età maggiore di 30 anni

```
SELECT COUNT(*) AS NumeroClienti

FROM   Clienti AS cc

WHERE  Eta > 30
```


Contare, tra quelli residenti in Puglia, Sicilia, Lombardia, i clienti divisi per il loro impiego.

```
SELECT Impiego,  
        COUNT(*) AS NumeroClienti  
  
FROM Clienti AS cc  
  
WHERE Residenza = 'Puglia'  
OR Residenza = 'Sicilia'  
OR Residenza = 'Lombardia'  
  
GROUP BY Impiego
```

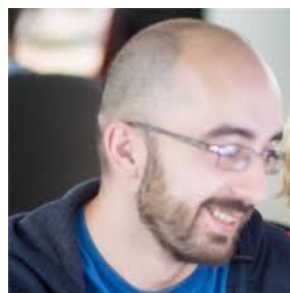
Calcolare il numero dei conti correnti divisi per Valuta, se la relativa somma dei saldi è maggiore di 100.

```
SELECT Valuta,  
        COUNT(*) AS NumeroClienti  
  
FROM ContiCorrenti AS cc  
  
GROUP BY Valuta  
  
HAVING SUM(Saldo) > 100
```

Informazioni sull'autore

Nicola Iantomasi

Mi chiamo Nicola Iantomasi e dopo cinque anni di esperienza come sviluppatore SQL e progettista Datawarehouse, ho fondato **Yimp - La scuola dei dati** per erogare corsi, consulenze e lezioni ad aziende, privati e studenti.



Il nostro focus è sul mondo dei database, l'Sql, la programmazione in Python, il machine learning, l'intelligenza artificiale, Microsoft Excel, la matematica e la statistica.

Se vuoi informazioni sui miei corsi, sulle mie attività o se hai semplicemente delle domande sul contenuto di questo libro, scrivimi alla mail **nicola.iantomasi@yimp.it** o visita il sito <https://www.yimp.it>