

Laurea in Ingegneria Informatica – SAPIENZA Università di Roma

Insegnamento di Basi di Dati

Esercitazione:

Il DBMS MySQL

Domenico Fabio Savo

Cosa vedremo

1. Presentazione del DBMS MySQL
2. Come scaricare ed installare MySQL
3. Il “client mysql”
4. Creazione e gestione di una base di dati
5. Creazione e gestione delle tabelle
6. Esercitazione sulle interrogazione di una base di dati

Cosa vedremo

1. **Presentazione del DBMS MySQL**
2. Come scaricare ed installare MySQL
3. Il “client mysql”
4. Creazione e gestione di una base di dati
5. Creazione e gestione delle tabelle
6. Esercitazione sulle interrogazione di una base di dati

Il DBMS MySQL

- ▶ MySQL è un DBMS open-source disponibile gratuitamente su <http://www.mysql.it/downloads/mysql/>
- ▶ In questa esercitazione si farà riferimento alla versione MySQL 5 per Windows.
(è possibile utilizzare MySQL anche su sistemi Linux e MacOS)

Cosa vedremo

1. Presentazione del DBMS MySQL
2. **Come scaricare ed installare MySQL**
3. Il “client mysql”
4. Creazione e gestione di una base di dati
5. Creazione e gestione delle tabelle
6. Esercitazione sulle interrogazione di una base di dati

Installazione di MySQL

Per l'installazione procedere come segue:

1. Scaricare l'ultima versione di **MySQL Community Server** dal sito (è necessario registrarsi).
2. Eseguire il file **.msi** appena scaricato e seguire la procedura di installazione.
3. Dopo il termine dell'installazione è possibile lanciare la **Configuration Wizard** per configurare immediatamente il nostro server MySQL.
4. Selezionare la configurazione **“Standard”**.
5. Selezionare le check box per:
 - a) eseguire MySQL come servizio;
 - b) lanciare automaticamente MySQL all'avvio;
 - c) includere la directory 'bin' nel path di Windows.
6. L'ultima schermata ci consente di impostare la password di **root**, tale password ci consentirà di amministrare il server.
7. Al termine l'installazione e la configurazione sono completate.

Cosa vedremo

1. Presentazione del DBMS MySQL
2. Come scaricare ed installare MySQL
3. **Il “client mysql”**
4. Creazione e gestione di una base di dati
5. Creazione e gestione delle tabelle
6. Esercitazione sulle interrogazione di una base di dati

Il “client mysql”

- ▶ **Client mysql** è il programma client a riga di comando che consente di collegarsi al server MySQL per sfruttarne le funzionalità.
(viene installato insieme al server MySQL)
- ▶ Dalla pagina web <http://www.mysql.it/downloads/workbench/> è possibile scaricare ed installare dei client grafici, chiamati **MySQL GUI Tools**, che forniscono una interfaccia grafica intuitiva per la gestione e l'interrogazione delle basi di dati gestite dal DBMS MySQL.
- ▶ In questa esercitazione utilizzeremo il client a riga di comando. Si lascia allo studente il compito di familiarizzare con i client grafici su indicati.

Il “client mysql”

- ▶ Per lanciare il client mysql è sufficiente richiamarlo dal prompt indicandogli utenza e password (nel nostro caso, useremo sempre l'utente **root**):

```
shell> mysql --user=root --password=xxx
```

oppure:

```
shell> mysql -uroot -p
```

In questo caso sarà il programma a chiedervi di introdurre la password senza visualizzarla.

- ▶ Una volta connessi al DBMS, appare il prompt di mysql:

```
mysql>
```

A questo punto si possono digitare i comandi SQL per interagire con il DBMS.

- ▶ Per chiudere il client digitiamo:

```
mysql> quit
```

Cosa vedremo

1. Presentazione del DBMS MySQL
2. Come scaricare ed installare MySQL
3. Il “client mysql”
4. **Creazione e gestione di una base di dati**
5. Creazione e gestione delle tabelle
6. Esercitazione sulle interrogazione di una base di dati

I permessi in MySQL

- ▶ Una volta connessi al server, l'utente deve possedere i permessi necessari per lavorare sulle varie basi di dati.
- ▶ Per chiedere quali basi di dati gestite dal DBMS sono accessibili dall'utente (nel nostro caso, **root**) utilizziamo il comando:

```
mysql> show databases;

+-----+
| Database |
+-----+
| information_schema |
| mysql      |
| test      |
+-----+
3 rows in set (0.05 sec)
```

Creazione di una base di dati

- ▶ Per poter creare una nuova base di dati su cui lavorare utilizziamo il comando:

```
CREATE DATABASE [IF NOT EXISTS] nome_db
```

- ▶ Con l'opzione **IF NOT EXISTS** possiamo evitare la segnalazione di errore nel caso esista già una base di dati con lo stesso nome.
- ▶ Per eliminare una base di dati si utilizza l'istruzione:

```
DROP DATABASE [IF EXISTS] nome_db
```

- ▶ Con l'opzione **IF EXISTS** possiamo evitare la segnalazione di errore nel caso non esista una base di dati chiamata **nome_db**.

ES: Creazione di un database

Creiamo il DB “esempio” utilizzando il “client mysql”.

Le istruzioni da utilizzare sono:

```
mysql> CREATE DATABASE esempio;  
  
Query OK, 1 row effected (0.06 sec)
```

Ora le basi di dati gestite dall'utente *root* sono:

```
mysql> show databases;  
  
+-----+  
| Database |  
+-----+  
| information_schema |  
| mysql |  
| test |  
| esempio |  
+-----+  
4 rows in set (0.05 sec)
```

Importare i comandi

Anziché eseguire comandi SQL digitandoli su terminale è spesso più conveniente scriverli in un file di testo e poi richiamarli dall'interprete dei comandi MySQL.

Supponiamo di aver scritto alcuni comandi SQL in un file **miaquery.sql** nella directory corrente. Possiamo eseguire il file da MySQL con il comando:

```
mysql> source miaquery.sql
```

Ovviamente è possibile anche specificare il path completo del file.

Cosa vedremo

1. Presentazione del DBMS MySQL
2. Come scaricare ed installare MySQL
3. Il “client mysql”
4. Creazione e gestione di una base di dati
5. **Creazione e gestione delle tabelle**
6. Esercitazione sulle interrogazione di una base di dati

Creazione delle tabelle (1 / 4)

- ▶ Per selezionare la base di dati su cui lavorare usare il comando `USE nome_database`
- ▶ Una volta selezionata la base di dati, l'istruzione per definire uno schema di relazione (specificando attributi e vincoli) in MySQL è

```
CREATE TABLE [IF NOT EXISTS] nome_tabella
[(
    [definizione attributi]
    [opzioni di tabella]
)]
```

- ▶ La tabella viene creata nella base di dati in uso. In generale, è possibile indicare espressamente in quale base di dati creare la tabella usando *nome_db.nome_tabella*.
- ▶ *IF NOT EXISTS* si usa per evitare messaggi di errore nel caso la tabella esista già nella base di dati in uso.

Creazione delle tabelle (2/4)

```
CREATE TABLE [IF NOT EXISTS] nome_tabella  
[(  
    [definizione attributi]  
    [opzioni di tabella]  
)]
```

- ▶ Le **definizioni attributi** si riferiscono agli attributi della tabella, la loro sintassi è:

nome_colonna TIPO

[NOT NULL | NULL] (di default può contenere valori NULL)

[DEFAULT valore] (usato per impostare un valore di default)

[AUTO_INCREMENT] (per attributi di tipo intero per avere un valore sequenziale generato automaticamente)

[UNIQUE | [PRIMARY] KEY] (UNIQUE rappresenta una superchiave, PRIMARY KEY indica la chiave primaria, oltre a non ammettere duplicati non può contenere valori NULL)

[reference_definition] ()

Creazione delle tabelle (3/4)

```
CREATE TABLE [IF NOT EXISTS] nome_tabella  
[(  
    [definizione attributi]  
    [opzioni di tabella]  
)]
```

▶ **Reference_definition**

Tramite le **reference_definition** è possibile definire vincoli di integrità referenziale, ovvero l'attributo su cui è definito può assumere solo valori specificati nell'attributo di un'altra tabella.

```
REFERENCES nome_tabella [(colonna_indice,...)]
```

Creazione delle tabelle (4/4)

```
CREATE TABLE [IF NOT EXISTS] nome_tabella
[(
  [definizione attributi]
  [opzioni di tabella]
)]
```

- ▶ Le **opzioni tabella** si riferiscono all'intera tabella e permettono di definire diverse proprietà di questa.

- ▶ Le più importanti sono:

PRIMARY KEY (nome_attributo1, nome_attributo2,...)

Permette di definire come chiave primaria della tabella un insieme di attributi di questa.

INDEX (nome_attributo1, nome_attributo2,...)

Permette di definire degli indici su uno o più attributi della tabella

FOREIGN KEY (nome_att1, nome_att2,...)

REFERENCE nome_tab(nome_att1, nome_att2,...)

Permette di definire vincoli di integrità referenziale su più attributi

ES: Creazione di una tabella (1 / 2)

Vogliamo creare le seguenti tabelle:

- **individui(nome, reddito, eta, sesso)**
 - nome è una stringa di 20 caratteri (chiave primaria)
 - reddito è un intero di 10 cifre
 - eta è un intero di 3 cifre
 - sesso è un carattere
- **genitori(figlio,genitore)**
 - figlio (stringa di 20 caratteri, chiave esterna su INDIVIDUI)
 - genitore (stringa di 20 caratteri, chiave esterna su INDIVIDUI)
 - chiave primaria formata da “figlio” e “genitore”

ES: Creazione di una tabella (2/2)

- Creazione tabella Individui:

```
mysql> CREATE TABLE Individui(  
        Nome          CHARACTER(20) PRIMARY KEY,  
        Reddito       NUMERIC(10),  
        Eta           NUMERIC(3),  
        Sesso         CHARACTER,  
        );
```

- Creazione tabella Genitori:

```
mysql> CREATE TABLE Genitori(  
        Figlio        CHARACTER(20) REFERENCES Individui(Nome),  
        Genitore      CHARACTER(20) REFERENCES Individui(Nome),  
        PRIMARY KEY (Figlio,Genitore)  
        );
```

Visualizzare le tabelle di un database

Per visualizzare le tabelle di una base di dati usare il comando:

```
mysql> show tables;
```

Dopo la creazione delle tabelle “Individui” e “Genitori” il risultato sarà

```
mysql> show tables;

+-----+
| Tables_in_esempio |
+-----+
| genitori          |
| individui         |
+-----+
2 rows in set (0.01 sec)
```

Visualizzare lo schema di una tabella

Per visualizzare lo schema della tabella `nome_tabella` si utilizza l'istruzione

`SHOW COLUMNS FROM nome_tabella`

oppure

`DESCRIBE nome_tabella`

```
mysql> Describe genitori;
+-----+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Figlio     | char(20)  | NO   | PRI |          |       |
| Genitore   | char(20)  | NO   | PRI |          |       |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.07 sec)
```

Modificare una tabella

È possibile modificare una tabella attraverso il comando **ALTER TABLE**:

```
ALTER TABLE nome_tabella
  ADD nome_attributo TIPO |
  ADD INDEX (nome_attributo,...) |
  ADD [CONSTRAINT [nome_vincolo]] PRIMARY KEY (nome_attributo,...) |
  ADD [CONSTRAINT [nome_vincolo]] UNIQUE (nome_attributo,...)
  ADD [CONSTRAINT [nome_vincolo]]
      FOREIGN KEY (colonna_indice,...) [reference_definition] |
  CHANGE vecchio_attributo nuovo_attributo TIPO |
  DROP nome_attributo
  DROP PRIMARY KEY
  DROP INDEX nome_attributo
  .....
  .....
```


Ridenominazione di una tabella

Per ridenominare una tabella usare il comando

```
ALTER TABLE Nome_Tabella RENAME Nuovo_Nome;
```

ES: Vogliamo cambiare il nome della tabella “Individui” con “Persone”

```
mysql> ALTER TABLE Individui RENAME Persone;
```

Ridenominazione di una colonna

Per ridenominare una colonna di una tabella utilizzare il comando:

```
ALTER TABLE Nome_Tabella CHANGE  
Nome_Colonna_da_cambiare Nuovo_Nome_Colonna  
Proprietà_della_Nuova_Colonna;
```

ES: Vogliamo cambiare il nome del campo “Eta” con “Anni”

```
mysql> ALTER TABLE Persone  
CHANGE Eta Anni NUMERIC(3);
```

NOTA: Per modificare il tipo di colonna utilizzare il medesimo comando cambiando solo il tipo della colonna.

Aggiungere una nuova colonna

Per aggiungere una nuova colonna ad una tabella utilizzare il comando:

```
ALTER TABLE Nome_Tabella ADD Nome_della_Nuova_Colonna  
Proprietà_Colonna;
```

ES: Aggiungiamo la colonna “n_telefono” alla tabella “Persone”

```
mysql> ALTER TABLE Persone  
ADD n_telefono NUMERIC(20);
```

Per eliminare una colonna utilizzare il comando:

```
ALTER TABLE Nome_Tabella DROP Nome_Colonna_da_canc
```

Aggiungere un vincolo di chiave esterna (1)

Per aggiungere un vincolo di chiave esterna utilizzare il comando:

```
ALTER TABLE Nome_Tabella  
ADD CONSTRAINT [nome_vincolo]  
FOREIGN KEY (nome_col_che_referenzia)  
REFERENCE Nome_Tabella_Referenziata(nome_colonna_refe);
```

Aggiungere un vincolo di chiave esterna (2)

ES: Date le tabelle:

- ▶ Aziende(Nome,Sede,Capitale)
- ▶ GruppoAziendale(Nome,Capogruppo)

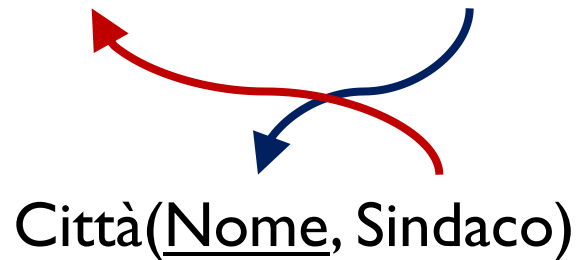
Vogliamo aggiungere alla tabella GruppoAziendale un vincolo di chiave esterna sull'attributo Capogruppo su Aziende

```
mysql> ALTER TABLE GruppoAziendale
ADD CONSTRAINT fk_capogruppo
FOREIGN KEY (Capogruppo)
REFERENCES Aziende(Nome);
```

Aggiungere un vincolo di chiave esterna (3)

La possibilità di aggiungere un vincolo di integrità referenziale permette di realizzare vincoli **ciclici**:

ES: Persona(Cod-Fiscale, Luogo-Nascita)



Dov'è il problema?

Aggiungere un vincolo di chiave esterna (4)

```
mysql> CREATE TABLE Persona (  
    Cod-Fiscale          CHARACTER(20) primary key,  
    Luogo-Nascita       CHARACTER(1),  
    FOREIGN KEY (Luogo-Nascita) REFERENCES Città(Nome));
```

Facciamo riferimento alla tabella “Città” che ancora **NON ESISTE!!**

Lo stesso accade se proviamo a creare prima la tabella “Città”

```
mysql> CREATE TABLE Città(  
    Nome                 CHARACTER(20) primary key,  
    Sindaco              CHARACTER(20),  
    FOREIGN KEY (Sindaco) REFERENCES Persona(Cod-Fiscale));
```

La tabella “Persona” ancora non esiste

Aggiungere un vincolo di chiave esterna (5)

Soluzione:

Eseguo le seguenti istruzioni in quest'ordine:

- 1- Creo la tabella “Persona” SENZA vincoli di foreign key;
- 2- Creo la tabella “Città” CON i vincoli di foreign key verso la tabella “Persona” (che ora esiste);
- 3- Aggiungo il vincolo di foreign key alla tabella “Persona” verso la tabella “Città” (che ora esiste).

Eliminare una tabella

È possibile eliminare una o più tabelle utilizzando il comando:

```
DROP TABLE [IF EXISTS] nome_tabella [, nome_tabella]
```

Con l'opzione **IF EXISTS** possiamo evitare la segnalazione di errore nel caso non esista una tabella chiamata **nome_tabella**.

ES:

```
mysql> DROP TABLE Genitori;
```

Inserimento dei dati nelle tabelle

Per inserire dei dati in una tabella si utilizza l'istruzione:

```
INSERT INTO nome_tabella [(nome_attributo1,nome_attributo2,...)]  
VALUES (valore1,valore2,...);
```

Attenzione:

- ▶ L'ordinamento degli attributi (se presente) e dei valori è significativo.
- ▶ Le due liste di attributi e di valori devono avere lo stesso numero di elementi.
- ▶ Se la lista di attributi è omessa, si fa riferimento a tutti gli attributi della relazione secondo l'ordine con cui sono stati definiti.
- ▶ Se la lista di attributi non contiene tutti gli attributi della relazione, per gli altri viene inserito un valore nullo (che deve essere permesso) o un valore di default.

ES: Inserimento dati

Inseriamo alcune tuple nella tabella

Persone(Nome, Reddito, Eta, Sesso)

```
mysql> INSERT INTO PERSONE (Nome, Reddito, Eta, Sesso)
VALUES ('Aldo', 25, 15, 'M');
mysql> INSERT INTO PERSONE (Nome, Reddito, Eta, Sesso)
VALUES ('Andrea', 27, 21, 'M');
mysql> INSERT INTO PERSONE (Nome, Reddito, Eta, Sesso)
VALUES ('Luisa', 75, 87, 'F');
mysql> INSERT INTO PERSONE (Nome, Reddito, Eta, Sesso)
VALUES ('Maria', 55, 42, 'F');
```

Eliminazione di dati dalle tabelle

Per eliminare una ennuola utilizzare il comando:

```
DELETE FROM nome_tabella [WHERE condizione]
```

ES: Eliminiamo tutte le persone con meno di 18 anni dalla tabella “Persone”

```
mysql> DELETE FROM persone WHERE eta<18;
```

Interrogare un database

Per effettuare un'interrogazione in SQL si utilizza l'istruzione **SELECT**

```
SELECT nome_attributo,...,nome_attributo  
FROM nome_tabella, ...,nome_tabella  
[WHERE condizione]
```

Le tre parti sono solitamente chiamate:

- ▶ target list
- ▶ clausola from
- ▶ clausola where

Le ridenominazioni

SQL permette di specificare un “alias” degli attributi (nella target list usando il comando **AS**) e delle tabelle (nella clausola **FROM**).

La ridenominazione è usata per:

1. Ottenere segnature più esplicitive nei risultati;
2. Creare abbreviazione ed evitare ambiguità;

```
mysql> SELECT p.nome as donne  
        FROM persone p  
        WHERE p.sesso = "F";
```

Cosa vedremo

1. Presentazione del DBMS MySQL
2. Come scaricare ed installare MySQL
3. Il “client mysql”
4. Creazione e gestione di una base di dati
5. Creazione e gestione delle tabelle
6. Esercitazioni sulle interrogazione di una base di dati

Esercizio 1

Data la tabella:

persone	<u>nome</u>	reddito	età	sex
----------------	-------------	---------	-----	-----

Effettuare le seguenti interrogazioni:

1. Trovare nome e sesso delle persone con più di 30 anni;
2. Trovare nome e sesso delle persone con più di 30 e reddito superiore a 60;
3. Trovare nome e reddito delle persone che hanno il nome che comincia per “M”;
4. Trovare i generi di sesso che compaiono nella tabella “persone”;
5. Trovare nome, età e reddito delle persone che hanno 30 o 40 anni ed un reddito maggiore o uguale a 50;
6. Trovare il nome delle persone la cui età è sconosciuta.

Soluzione (1)

Trovare il nome e sesso delle persone con più di 30 anni

persone	<u>nome</u>	reddito	eta	sexso
----------------	-------------	---------	-----	-------

```
mysql> SELECT nome, sesso FROM persone
        WHERE eta > 30;
```

```
+-----+-----+
| nome   | sesso |
+-----+-----+
| AntonGiulio | M     |
| Luigi    | M     |
| Luisa    | F     |
| Maria    | F     |
| Olga     | F     |
| Sergio   | M     |
+-----+-----+
6 rows in set (0.00 sec)
```

Soluzione (2)

Trovare nome e sesso delle persone con più di 30 anni e reddito superiore a 60

persone	<u>nome</u>	reddito	eta	sesso
----------------	-------------	---------	-----	-------

```
mysql> SELECT nome, sesso
        FROM persone
        WHERE reddito > 60 AND eta > 30 ;
```

```
+-----+-----+
| Nome   | Sesso |
+-----+-----+
| Luisa  | F     |
| Sergio | M     |
+-----+-----+
2 rows in set (0.00 sec)
```

Soluzione (3)

Trovare nome e reddito delle persone che hanno il nome che comincia per "M"

persone	<u>nome</u>	reddito	eta	sesso
----------------	-------------	---------	-----	-------

```
mysql> SELECT nome, reddito
        FROM persone
        WHERE nome LIKE 'M%' ;
```

```
+-----+-----+
| Nome          | Reddito |
+-----+-----+
| Maria        | 55      |
| Michelangelo | 79      |
+-----+-----+
2 rows in set (0.01 sec)
```

Soluzione (4)

Trovare tutti i generi di sesso che compaiono nella tabella "persone"

persone	<u>nome</u>	reddito	eta	sex
----------------	-------------	---------	-----	-----

```
mysql> SELECT DISTINCT sesso AS genere
        FROM persone;
```

```
+-----+
| genere |
+-----+
| F      |
| M      |
+-----+
```

```
2 rows in set (0.01 sec)
```

Soluzione (5)

Trovare nome, eta e reddito delle persone che hanno 30 o 40 anni ed un reddito maggiore o uguale a 50

persone

nome

reddito

eta

sexso

```
mysql> SELECT nome, reddito, eta FROM persone
        WHERE (eta=30 OR eta=40) AND reddito>=50;
```

```
+-----+-----+-----+
| nome          | reddito | eta  |
+-----+-----+-----+
| Beatrice     | 79     | 30  |
| Leonardo     | 79     | 30  |
| Luigi        | 50     | 40  |
| Michelangelo | 79     | 30  |
+-----+-----+-----+
4 rows in set (0.00 sec)
```

Soluzione (6)

Trovare il nome delle persone la cui età è sconosciuta

persone

nome

reddito

eta

sesso

```
mysql> SELECT nome FROM persone
        WHERE eta IS NULL;
```

```
+-----+
| nome  |
+-----+
| Diana |
+-----+
```

```
1 row in set (0.00 sec)
```

Esercizio 2

Date le tabelle:

persone	nome	reddito	eta	sezzo
genitori	figlio	genitore		

Effettuare le seguenti interrogazioni:

7. Trovare nome e reddito dei genitori di 'Aldo';
8. Trovare la relazione che mostra per ogni persona i nonni;
9. Trovare il reddito medio dei padri per età;
10. Trovare la relazione che mostra le coppie di fratelli (due persone sono fratelli se hanno un genitore in comune);
11. Per ogni "genitore" restituire il numero dei suoi figli;
12. Trovare la relazione che mostra per ciascun figlio i rispettivi genitori (Padre, Madre, Figlio).

Soluzione (7)

Trovare nome e reddito dei genitori di 'Aldo';

persone	nome	reddito	eta	sex
genitori	figlio	genitore		

```
mysql> SELECT genitore, reddito
FROM persone, genitori
WHERE figlio= 'Aldo' AND genitore=nome;
```


Soluzione (8)

Trovare la relazione che mostra per ogni persona i nonni;

persone	nome	reddito	eta	sexso
genitori	figlio	genitore		

```
mysql> SELECT P.Figlio AS Nipote,  
          N.Genitore AS Nonno  
FROM GENITORI P, GENITORI N  
WHERE P.Genitore=N.Figlio;
```

Soluzione (9)

Trovare il reddito medio dei padri per età;

persone	nome	reddito	eta	sezzo
genitori	figlio	genitore		

```
mysql> SELECT eta, AVG(reddito) AS AVG_REDDITO
FROM Persone, Genitori
WHERE nome = genitore AND sesso = 'M'
GROUP BY eta;
```

Soluzione (10)

Trovare la relazione che mostra le coppie di fratelli
(due persone sono fratelli se hanno un genitore in comune).

persone	nome	reddito	eta	sex
genitori	figlio	genitore		

```
mysql> SELECT distinct G1.Figlio AS Nome1,  
                        G2.Figlio AS Nome2  
FROM GENITORI G1, GENITORI G2  
WHERE G1.Genitore = G2.Genitore and  
      G1.Figlio <> G2.Figlio;
```

Soluzione (11)

Per ogni “genitore” restituire il numero dei suoi figli;

persone	nome	reddito	eta	sexso
genitori	figlio	genitore		

```
mysql> SELECT genitore,  
           count(figlio) as numero_figli  
FROM genitori  
GROUP BY genitore;
```

Soluzione (12)

Trovare la relazione che mostra per ciascun figlio i rispettivi genitori (Padre, Madre, Figlio);

persone	nome	reddito	eta	sex
genitori	figlio	genitore		

```
mysql> SELECT GP.Genitore AS Padre,  
             GM.Genitore AS Madre, GP.Figlio  
FROM genitori GP, genitori GM,  
     persone PP, persone PM  
WHERE GP.Figlio=GM.Figlio AND  
      GP.Genitore=PP.Nome AND PP.Sesso='M'  
AND  
      GM.Genitore=PM.Nome AND PM.Sesso='F';
```