

Basi di Dati

Esercitazione #3

Tutor

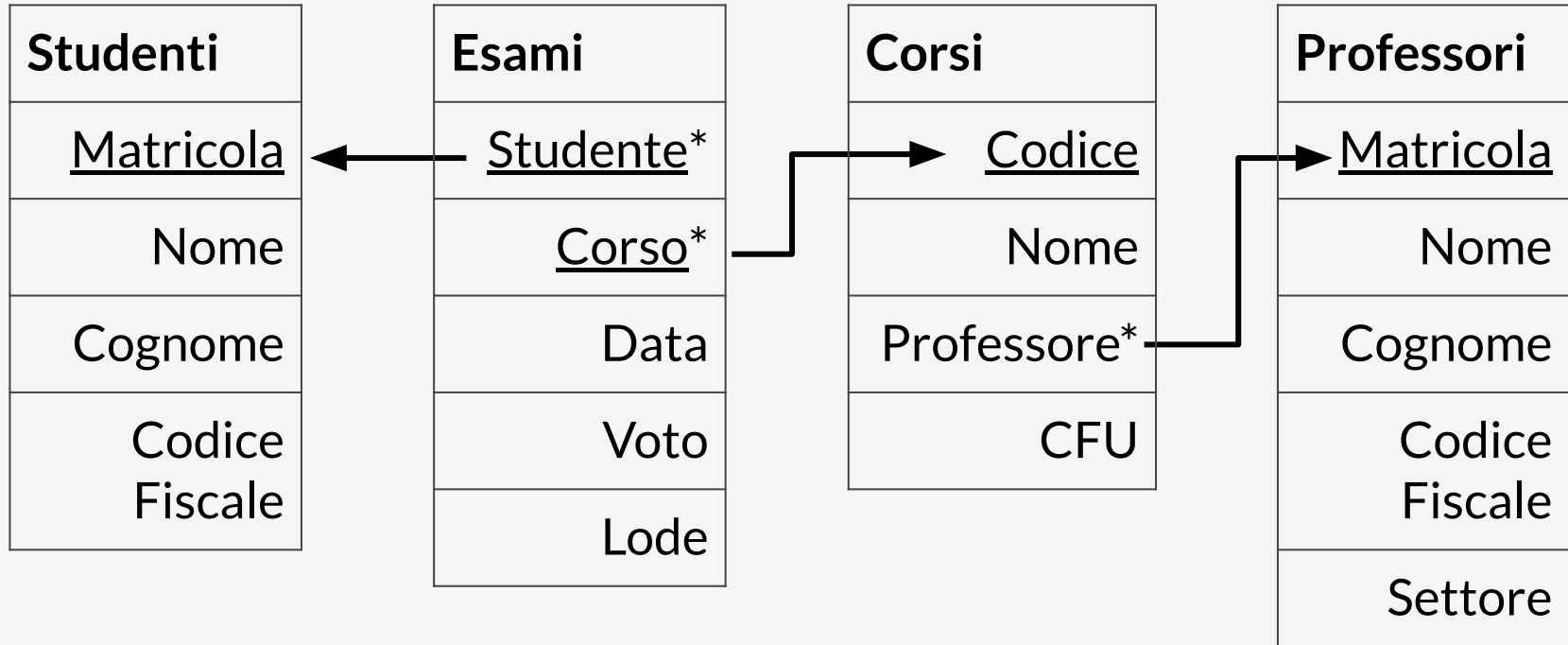
Mauro Farina

- Dottorando in ingegneria industriale e dell'informazione
- 1° anno
- Cybersecurity e misurazioni di Internet
- mauro.farina@phd.units.it

Materiale su GitHub: github.com/mauro-farina/database-exercises

Feedback: <https://forms.gle/yAYRAvC3iHarHJyL6>

"Modello logico"



Database

[unidb.sql](https://github.com/mauro-farina/database-exercises) → <https://github.com/mauro-farina/database-exercises>

- 301 studenti
- 52 professori
- 31 corsi
- 2203 esami

Per caricare lo script...

1. Copia-incolla, oppure
2. Da MySQL Workbench: File > Open SQLScript

Esercizi extra

EE1: Tutti i corsi per i quali nessuno ha passato l'esame a gennaio 2020

EE2: Ci sono professori che non tengono alcun corso?

EE3: Restituire l'anno nel quale il maggior numero di studenti ha sostenuto l'esame di Geometria

EE4: Elencare tutti i corsi per cui la media voti all'esame è superiore a 26

EE5: Se 1 CFU sono 8h di lezione, qual è il professore che passa meno tempo a insegnare?

Query sul Database

Esercizio 8

Elencare nome, cognome e media voti **ponderata** degli studenti con media voti **aritmetica** > 28 e che hanno sostenuto **almeno** 5 esami.

- Risultato atteso:

Zoe	Rudel	28.3258
Simone	D'amico	28.2143
Vida	La bella	28.0351
Cloe	Oliva	28.0000

Esercizio 8: soluzione

Elencare nome, cognome e media voti **ponderata** degli studenti con media voti **aritmetica** > 28 e che hanno sostenuto **almeno** 5 esami.

```
SELECT s.nome, s.cognome,  
       sum(e.voto*c.cfu)/sum(c.cfu) AS media_ponderata  
FROM studenti s  
INNER JOIN esami e ON e.studente = s.matricola  
INNER JOIN corsi c ON e.corso = c.codice  
GROUP BY e.studente  
HAVING count(e.voto) >= 5 AND avg(e.voto) > 28
```



```
SELECT t1.nome, t1.cognome,  
       t1.parzMedia/t1.cfuTot AS mediaPonderata  
FROM esami  
INNER JOIN (
```

1. Abbreviazioni di tabelle

```
SELECT COUNT(*) AS nEsami, studenti.matricola AS stud,  
       AVG(esami.voto) AS media, studenti.nome AS nome,  
       studenti.cognome AS cognome, SUM(corsi.cfu) AS  
       cfuTot, SUM(esami.voto * corsi.cfu) AS parzMedia  
FROM studenti  
INNER JOIN esami ON esami.studente = studenti.matricola  
INNER JOIN corsi ON corsi.codice = esami.corso  
GROUP BY studenti.matricola HAVING COUNT(*) > 5) AS t1  
ON esami.studente = t1.stud  
WHERE t1.media > 28  
GROUP BY t1.stud;
```

```
SELECT t1.nome, t1.cognome,  
       t1.parzMedia/t1.cfuTot AS mediaPonderata  
FROM esami e1  
INNER JOIN (  
    SELECT COUNT(*) AS nEsami, s.matricola AS stud,  
           AVG(e2.voto) AS media, studenti.nome AS nome,  
           s.cognome AS cognome, SUM(c.cfu) AS cfuTot,  
           SUM(e2.voto * c.cfu) AS parzMedia  
    FROM studenti s  
    INNER JOIN esami e2 ON e2.studente = s.matricola  
    INNER JOIN corsi c ON c.codice = e2.corso  
    GROUP BY s.matricola HAVING COUNT(*) > 5) AS t1  
ON e1.studente = t1.stud  
WHERE t1.media > 28  
GROUP BY t1.stud;
```

```
SELECT t1.nome, t1.cognome,  
       t1.parzMedia/t1.cfuTot AS mediaPonderata
```

```
FROM esami e1
```

```
INNER JOIN (
```

2. Alias delle colonne

```
SELECT COUNT(*) AS nEsami, s.matricola AS stud,  
       AVG(e2.voto) AS media, studenti.nome AS nome,  
       s.cognome AS cognome, SUM(c.cfu) AS cfuTot,  
       SUM(e2.voto * c.cfu) AS parzMedia
```

```
FROM studenti s
```

```
INNER JOIN esami e2 ON e2.studente = s.matricola
```

```
INNER JOIN corsi c ON c.codice = e2.corso
```

```
GROUP BY s.matricola HAVING COUNT(*) > 5) AS t1
```

```
ON e1.studente = t1.stud
```

```
WHERE t1.media > 28
```

```
GROUP BY t1.stud;
```

```
SELECT t1.nome, t1.cognome,  
       t1.parzMedia/t1.cfuTot AS mediaPonderata  
FROM esami e1  
INNER JOIN (  
    SELECT COUNT(*) AS nEsami, s.matricola,  
           AVG(e2.voto) AS media, s.nome, s.cognome,  
           SUM(c.cfu) AS cfuTot, SUM(e2.voto * c.cfu) AS  
parzMedia  
FROM studenti s  
    INNER JOIN esami e2 ON e2.studente = s.matricola  
    INNER JOIN corsi c ON c.codice = e2.corso  
    GROUP BY s.matricola HAVING COUNT(*) > 5) AS t1  
ON e1.studente = t1.matricola  
WHERE t1.media > 28  
GROUP BY t1.matricola;
```

```
SELECT t1.nome, t1.cognome,  
       t1.parzMedia/t1.cfuTot AS mediaPonderata
```

```
FROM esami e1
```

```
INNER JOIN (
```

3. Colonne inutilizzate

```
SELECT COUNT(*) AS nEsami, s.matricola,  
       AVG(e2.voto) AS media, s.nome, s.cognome,  
       SUM(c.cfu) AS cfuTot, SUM(e2.voto * c.cfu) AS
```

```
parzMedia
```

```
FROM studenti s
```

```
INNER JOIN esami e2 ON e2.studente = s.matricola
```

```
INNER JOIN corsi c ON c.codice = e2.corso
```

```
GROUP BY s.matricola HAVING COUNT(*) > 5) AS t1
```

```
ON e1.studente = t1.matricola
```

```
WHERE t1.media > 28
```

```
GROUP BY t1.matricola;
```

```
SELECT t1.nome, t1.cognome,  
       t1.parzMedia/t1.cfuTot AS mediaPonderata  
FROM esami e1  
INNER JOIN (  
    SELECT s.matricola, AVG(e2.voto) AS media,  
           s.nome, s.cognome, SUM(c.cfu) AS cfuTot,  
           SUM(e2.voto * c.cfu) AS parzMedia  
    FROM studenti s  
    INNER JOIN esami e2 ON e2.studente = s.matricola  
    INNER JOIN corsi c ON c.codice = e2.corso  
    GROUP BY s.matricola HAVING COUNT(*) > 5) AS t1  
ON e1.studente = t1.matricola  
WHERE t1.media > 28  
GROUP BY t1.matricola;
```

```
SELECT t1.nome, t1.cognome,  
       t1.parzMedia/t1.cfuTot AS mediaPonderata
```

```
FROM esami e1
```

```
INNER JOIN (
```

4. Scelta delle colonne

```
SELECT s.matricola, AVG(e2.voto) AS media,  
       s.nome, s.cognome, SUM(c.cfu) AS cfuTot,  
       SUM(e2.voto * c.cfu) AS parzMedia
```

```
FROM studenti s
```

```
INNER JOIN esami e2 ON e2.studente = s.matricola
```

```
INNER JOIN corsi c ON c.codice = e2.corso
```

```
GROUP BY s.matricola HAVING COUNT(*) > 5) AS t1
```

```
ON e1.studente = t1.matricola
```

```
WHERE t1.media > 28
```

```
GROUP BY t1.matricola;
```

```
SELECT t1.nome, t1.cognome, t1.mediaPonderata
FROM esami e1
INNER JOIN (
    SELECT s.matricola, AVG(e2.voto) AS media,
           s.nome, s.cognome,
           SUM(e2.voto * c.cfu)/SUM(c.cfu) AS mediaPonderata
    FROM studenti s
    INNER JOIN esami e2 ON e2.studente = s.matricola
    INNER JOIN corsi c ON c.codice = e2.corso
    GROUP BY s.matricola HAVING COUNT(*) > 5) AS t1
ON e1.studente = t1.matricola
WHERE t1.media > 28
GROUP BY t1.matricola;
```



```
SELECT t1.nome, t1.cognome, t1.mediaPonderata  
FROM esami e1
```

```
INNER JOIN (
```

5. Attenzione alla consegna

```
    SELECT s.matricola, AVG(e2.voto) AS media,  
           s.nome, s.cognome,  
           SUM(e2.voto * c.cfu)/SUM(c.cfu) AS mediaPonderata  
    FROM studenti s  
    INNER JOIN esami e2 ON e2.studente = s.matricola  
    INNER JOIN corsi c ON c.codice = e2.corso  
    GROUP BY s.matricola HAVING COUNT(*) > 5) AS t1
```


```
ON e1.studente = t1.matricola
```

```
WHERE t1.media > 28
```

```
GROUP BY t1.matricola;
```

```
SELECT t1.nome, t1.cognome, t1.mediaPonderata
FROM esami e1
INNER JOIN (
    SELECT s.matricola, AVG(e2.voto) AS media,
           s.nome, s.cognome,
           SUM(e2.voto * c.cfu)/SUM(c.cfu) AS mediaPonderata
    FROM studenti s
    INNER JOIN esami e2 ON e2.studente = s.matricola
    INNER JOIN corsi c ON c.codice = e2.corso
    GROUP BY s.matricola HAVING COUNT(*) >= 5) AS t1
ON e1.studente = t1.matricola
WHERE t1.media > 28
GROUP BY t1.matricola;
```

```
SELECT t1.nome, t1.cognome, t1.mediaPonderata
FROM esami e1
INNER JOIN (
    SELECT s.matricola, AVG(e2.voto) AS media,
           s.nome, s.cognome,
           SUM(e2.voto * c.cfu)/SUM(c.cfu) AS mediaPonderata
    FROM studenti s
    INNER JOIN esami e2 ON e2.studente = s.matricola
    INNER JOIN corsi c ON c.codice = e2.corso
    GROUP BY s.matricola HAVING COUNT(*) >= 5) AS t1
ON e1.studente = t1.matricola
WHERE t1.media > 28
GROUP BY t1.matricola;
```



```
SELECT t1.nome, t1.cognome, t1.mediaPonderata
FROM esami e1
INNER JOIN (
    SELECT s.matricola, AVG(e2.voto) AS media,
           s.nome, s.cognome,
           SUM(e2.voto * c.cfu)/SUM(c.cfu) AS mediaPonderata
    FROM studenti s
    INNER JOIN esami e2 ON e2.studente = s.matricola
    INNER JOIN corsi c ON c.codice = e2.corso
    GROUP BY s.matricola
    HAVING COUNT(*) >= 5 AND media > 28) AS t1
ON e1.studente = t1.matricola
GROUP BY t1.matricola;
```

```
SELECT t1.nome, t1.cognome, t1.mediaPonderata
FROM esami e1
INNER JOIN (
```

```
    SELECT s.matricola, AVG(e2.voto) AS media,
           s.nome, s.cognome,
           SUM(e2.voto * c.cfu)/SUM(c.cfu) AS mediaPonderata
    FROM studenti s
    INNER JOIN esami e2 ON e2.studente = s.matricola
    INNER JOIN corsi c ON c.codice = e2.corso
    GROUP BY s.matricola
    HAVING COUNT(*) >= 5 AND media > 28) AS t1
```

```
ON e1.studente = t1.matricola
GROUP BY t1.matricola;
```

```
SELECT s.matricola, AVG(e2.voto) AS media,  
       s.nome, s.cognome,  
       SUM(e2.voto * c.cfu)/SUM(c.cfu) AS mediaPonderata  
FROM studenti s  
INNER JOIN esami e2 ON e2.studente = s.matricola  
INNER JOIN corsi c ON c.codice = e2.corso  
GROUP BY s.matricola  
HAVING COUNT(*) >= 5 AND media > 28
```

Prepared Statements

Query riutilizzabili

- Vivono e muoiono nella **sessione**

Possono ricevere parametri

- Ogni '?' nella query corrisponde a un parametro d'ingresso

Sintassi

- `PREPARE nome_statement FROM "query SQL";`
- `EXECUTE nome_statement USING var1, ..., varN;`
- `DEALLOCATE PREPARE nome_statement;`

Esercizio 9

Creare un **prepared statement** che mostri tutti gli studenti iscritti a un corso di laurea (e.g., IN05, SM32) che viene dato come parametro

- Suggerimento: servirà la funzione "**concat**"

Esercizio 9: soluzione (usando **concat**)

Creare un **prepared statement** che mostri tutti gli studenti iscritti a un corso di laurea (e.g., IN05, SM32) che viene dato come parametro

```
PREPARE studenti_cd1 FROM  
"SELECT *  
FROM studenti s  
WHERE matricola LIKE concat(?, '%')";  
  
SET @cd1 = "IN05";  
EXECUTE studenti_cd1 USING @cd1;
```

Esercizio 9: soluzione (usando **substring**)

Creare un **prepared statement** che mostri tutti gli studenti iscritti a un corso di laurea (e.g., IN05, SM32) che viene dato come parametro

```
PREPARE studenti_cd1 FROM  
"SELECT *  
FROM studenti s  
WHERE substring(matricola, 1, 4) = ?";  
  
SET @cd1 = "IN05";  
EXECUTE studenti_cd1 USING @cd1;
```

Esercizio 10

Qual è il voto più comunemente assegnato da ciascun professore?

1. Creare la Vista **dist_voti** contenente la distribuzione di voti assegnati da ciascun professore
2. Selezionare il voto più frequente per ogni docente

Esercizio 10: soluzione (1)

1. Creare la Vista **dist_voti** contenente la distribuzione di voti assegnati da ciascun professore

```
CREATE VIEW dist_voti AS
SELECT p.matricola, p.nome, p.cognome, e.voto,
       count(*) AS n_voti
FROM esami e
INNER JOIN corsi c ON e.corso=c.codice
INNER JOIN professori p ON p.matricola=c.professore
GROUP BY p.matricola, e.voto;
```

Esercizio 10: soluzione (2)

2. Selezionare il voto più frequente per ogni docente

```
SELECT dv1.nome, dv1.cognome, dv1.voto
FROM dist_voti dv1
WHERE dv1.n_voti = (
    SELECT max(dv2.n_voti)
    FROM dist_voti dv2
    WHERE dv2.matricola=dv1.matricola
);
```

Esercizio Extra 6 (EE6)

Chi è lo studente/ssa con media voti ponderata più alta di ciascun corso di laurea (cdl)?

- Cdl → Primi 4 caratteri della matricola
 - Suggerimento: servirà la funzione SUBSTRING
 - https://www.w3schools.com/SQL/func_mysql_substring.asp

Esercizio 11

Quali studenti hanno migliorato almeno una volta la loro media fra un anno (**solare**) e l'altro?

1. Creare la Vista **media_per_anno**
2. Selezionare gli studenti che migliorano la media fra un anno e l'altro

Esercizio 11: soluzione (1)

1. Creare la Vista `media_per_anno`

```
CREATE VIEW media_per_anno AS
SELECT s.matricola, s.nome, s.cognome, year(e.data)
       AS anno, sum(e.voto*c.cfu)/sum(c.cfu) AS media
FROM studenti s
INNER JOIN esami e ON e.studente=s.matricola
INNER JOIN corsi c ON c.codice=e.corso
GROUP BY s.matricola, anno;
```


Esercizio 11: soluzione (2)

2. Selezionare gli studenti che migliorano la media fra un anno e l'altro

```
SELECT DISTINCT m1.matricola, m1.nome, m1.cognome
FROM media_per_anno m1
WHERE m1.media > (
    SELECT m2.media
    FROM media_per_anno m2
    WHERE m2.matricola=m1.matricola
    AND m2.anno < m1.anno
    ORDER BY m2.anno LIMIT 1
);
```

Esercizio Extra 7 (EE7)

Creare una **stored procedure** (**sp_trend_esame**) che, dato in ingresso un corso (codice, char(5)), elenchi nome e cognome di tutti gli studenti che hanno passato l'esame, aggiungendo la colonna "Trend" che indica se lo studente ha preso un voto "Sopra", "Sotto" o "Uguale" alla media voti del corso.

Risultato atteso con **CALL sp_trend_esame("079IN")**:

Enea	Ghini	Sopra
Diego	Vaccaro	Sopra
Sofia	Cagnotti	Sopra
Federico	Pastore	Sotto