

STUDENTE (Matricola, Nome, Cognome, Indirizzo, Città)

ESAME (CodCorso, MatrStud, Data, Voto, Lode)

Lode può valere 'yes' se e solo se il voto è 30

CORSO (Codice, Nome, AnnoDiCorso, Facoltà, NumeroCrediti)

INSEGNAMENTO (CodCorso, AnnoAccademico, MatrProf, NumeroStudenti)

PROFESSORE (Matricola, Nome, Cognome, Città, Telefono, Stipendio)

Formulare in SQL le interrogazioni per estrarre:

1. Gli studenti che hanno sostenuto esattamente 10 esami.
2. Gli studenti che non hanno mai sostenuto nessun esame.
3. Le matricole dei professori che hanno insegnato in corsi di tutte le facoltà.
4. I corsi in cui qualche voto non è mai stato assegnato.
5. Di ogni corso estratto dalla query precedente, i voti che non sono stati assegnati.
6. Matricola, nome e cognome degli studenti di Milano che hanno superato esami per un totale di almeno 20 crediti e non hanno mai preso un voto minore di 28.
7. L'edizione di corso (l'insegnamento) con il maggior numero di studenti in assoluto.
8. L'edizione di corso del primo anno con il maggior numero di studenti.
9. Di ogni facoltà, corsi con il minor numero di studenti.
10. Studenti che hanno preso più 27 che 24.
11. Studenti che hanno preso più 30L che 30.
12. Studenti che hanno preso più volte 27 rispetto a qualsiasi altro voto (complessivamente).
13. Studenti che hanno preso più volte 27 rispetto a qualsiasi altro voto (individualmente).
14. Matricola, nome e cognome degli studenti che hanno superato almeno 3 esami del secondo anno ma meno di 3 esami del primo.
15. Matricola, nome e cognome degli studenti che hanno superato più esami del terzo anno che del secondo.
16. Il corso con la media più bassa.
17. Studenti con la media più alta (pesata rispetto ai CFU)
18. Matricola e media degli studenti che hanno una media maggiore della "media" media di ateneo.
19. Per ogni studente, l'anno di corso in cui ha avuto la media più alta.
20. Gli studenti più regolari, ovvero quelli con la minima differenza tra il voto migliore e il voto peggiore.
21. Corsi in cui almeno il 50% degli studenti ha preso un voto maggiore di 25.
22. Studenti che hanno preso lo stesso voto in più di due terzi degli esami sostenuti.
23. Trovare i "top ten" studenti in base alla media pesata, tra quelli che abbiano sostenuto almeno 10 esami.
24. Corsi tenuti da professori di Torino che non sono stati superati da nessuno studente di Torino.
25. Studenti che non hanno mai sostenuto esami per corsi tenuti da docenti con il loro stesso cognome.
26. Studenti che hanno sostenuto al più 5 esami di corsi di una stessa facoltà.
27. Studenti che hanno sostenuto almeno due esami di corsi tenuti dallo stesso docente.
28. Studenti che non hanno mai sostenuto due esami di corsi tenuti dallo stesso docente.
29. Trovare i codici e i nomi dei corsi con il minimo numero di crediti.
30. Trovare i codici e nomi dei corsi del primo anno con il minimo numero di crediti.
31. Trovare, per ogni facoltà, i codici e i nomi dei corsi con il massimo numero di crediti di quella facoltà.
32. Facoltà che forniscono il maggior numero di corsi.
33. Professori che hanno tenuto il maggior numero di corsi.
34. Professori che hanno tenuto il maggior numero di corsi da almeno 5 crediti.
35. Corsi in cui almeno uno studente che ha superato l'esame aveva lo stesso cognome del docente.

STUDENTE (Matricola, Nome, Cognome, Indirizzo, Città)

ESAME (CodCorso, MatrStud, Data, Voto, Lode)

Lode può valere 'yes' se e solo se il voto è 30

CORSO (Codice, Nome, AnnoDiCorso, Facoltà, NumeroCrediti)

INSEGNAMENTO (CodCorso, AnnoAccademico, MatrProf, NumeroStudenti)

PROFESSORE (Matricola, Nome, Cognome, Città, Telefono, Stipendio)

1. Trovare gli studenti che hanno sostenuto esattamente 10 esami.

```
SELECT MatrStud
FROM Esame
GROUP BY MatrStud
HAVING COUNT(*)=10
```

2. Trovare gli studenti che non hanno mai sostenuto nessun esame.

```
SELECT Matricola
FROM Studente
WHERE Matricola NOT IN ( SELECT MatrStud FROM Esame )
```

Assolutamente NON:

```
SELECT MatrStud
FROM Esame
GROUP BY MatrStud
HAVING COUNT(*)=0
```

*Questa query è scorretta perché la clausola GROUP BY consente di raggruppare le tuple che sono presenti nella relazione a cui è applicata. Di conseguenza le tuple che non sono presenti **non fanno parte di nessun gruppo** e un gruppo, per esistere, deve contenere almeno una tupla! Quindi la condizione HAVING COUNT(*)=0 è intrinsecamente falsa, sempre, per ogni gruppo.*

3. Trovare le matricole dei professori che hanno insegnato in corsi di tutte le facoltà.

```
SELECT MatrProf
FROM Insegnamento join Corso on Codice = CodCorso
GROUP BY MatrProf
HAVING COUNT( DISTINCT Facoltà ) = ( SELECT COUNT( DISTINCT Facoltà ) FROM Corso )
```

4. Corsi in cui qualche voto non è mai stato assegnato.

```
SELECT CodCorso
FROM Esame
GROUP BY CodCorso
HAVING COUNT ( DISTINCT Voto ) < 13
```

ATTENZIONE : questa soluzione non estrae eventuali corsi di cui non sia (mai/ancora) stato sostenuto alcun esame (magari perché alla loro prima edizione). Per includere anche loro si può:

- a) Con un outer join

```
SELECT Codice
FROM Esame RIGHT JOIN Corso ON CodCorso = Codice
GROUP BY Codice
HAVING COUNT ( DISTINCT Voto ) < 13
```

b) *Utilizzare un passaggio al complemento*

```
SELECT Codice
FROM Corso
WHERE Codice NOT IN ( SELECT CodCorso
                        FROM Esame
                        GROUP BY CodCorso
                        HAVING COUNT ( DISTINCT Voto ) = 13 )
```

c) *Usare una union con una query dedicata ad estrarli*

```
SELECT CodCorso
FROM Esame
GROUP BY CodCorso
HAVING COUNT ( DISTINCT Voto ) < 13
UNION
SELECT Codice
FROM Corso
WHERE Codice NOT IN ( SELECT CodCorso
                        FROM Esame )
```

5. Di ogni corso estratto dalla query precedente, i voti che non sono stati assegnati.

```
SELECT DISTINCT Codice, Voto
FROM Corso, Esame
WHERE (Codice, Voto) NOT IN ( SELECT CodCorso, Voto
                              FROM Esame)
```

Oppure

```
SELECT Codice, Voto
FROM Corso, Esame
EXCEPT
SELECT CodCorso, Voto
FROM Esame
```

6. Matricola, nome e cognome degli studenti di Milano che hanno superato esami per un totale di almeno 20 crediti e non hanno mai preso un voto minore di 28.

```
SELECT Matricola, Nome, Cognome
FROM ( Studente join Esame on Matricola = MatrStud ) join Corso on CodCorso = Codice
WHERE Città='Milano'
GROUP BY Matricola, Nome, Cognome
HAVING SUM( NumeroCrediti ) >= 20 AND min( Voto ) >= 28
```

Oppure:

```

SELECT Matricola, Nome, Cognome
FROM ( Studente join Esame on Matricola = MatrStud ) join Corso on CodCorso = Codice
WHERE Città = 'Milano'
      AND Matricola NOT IN ( SELECT MatrStud
                             FROM Esame
                             WHERE Voto<28)
GROUP BY Matricola, Nome, Cognome
HAVING SUM( NumeroCrediti ) >= 20

```

Oppure:

```

SELECT Matricola, Nome, Cognome
FROM Studente S
WHERE Città = 'Milano'
      AND 28 <= ( SELECT min( Voto )
                 FROM Esame
                 WHERE MatrStud = S.Matricola )
      AND 20 <= ( SELECT sum( NumeroCrediti)
                 FROM Esame join Corso on CodCorso = Codice
                 WHERE MatrStud = S.Matricola )

```

Assolutamente NON, invece:

```

SELECT Matricola, Nome, Cognome
FROM ( Studente join Esame on Matricola = MatrStud) join Corso on CodCorso = Codice
WHERE Città='Milano' AND Voto >= 28
GROUP BY Matricola, Nome, Cognome
HAVING SUM( NumeroCrediti ) >= 20

```

Perché questa query estrae gli studenti che hanno preso 28 o più in corsi del peso complessivo di almeno 20 CFU, ma non esclude che abbiano preso meno di 28 in qualche altro esame, scartato dalla clausola WHERE.

7. L'edizione di corso (l'insegnamento) con il maggior numero di studenti in assoluto.

```

SELECT CodCorso, AnnoAccademico
FROM Insegnamento
WHERE NumeroStudenti = ( SELECT MAX(NumeroStudenti)
                        FROM Insegnamento)

```

Oppure:

```

SELECT CodCorso, AnnoAccademico
FROM Insegnamento
WHERE NumeroStudenti >= ALL ( SELECT NumeroStudenti
                              FROM Insegnamento )

```

Di questa query sono state proposte e discusse varianti (legate a un'interpretazione forse fuorviante della formulazione iniziale ("Le edizioni dei corsi con il maggior..."), ora corretta. Le riporto nel seguito (bis e ter).

bis. *L'edizione del corso con il maggior numero di studenti di ciascun anno accademico.*

```
SELECT CodCorso, AnnoAccademico
FROM Insegnamento X
WHERE NumeroStudenti >= ALL ( SELECT NumeroStudenti
                              FROM Insegnamento
                              WHERE AnnoAccademico = X.AnnoAccademico )
```

Oppure

```
SELECT CodCorso, AnnoAccademico
FROM Insegnamento
WHERE (AnnoAccademico, NumeroStudenti) IN ( SELECT AnnoAccademico, max(NumeroStudenti)
                                           FROM Insegnamento
                                           GROUP BY AnnoAccademico )
```

ter. *L'edizione più numerosa di ciascun corso.*

```
SELECT CodCorso, AnnoAccademico
FROM Insegnamento X
WHERE NumeroStudenti >= ALL ( SELECT NumeroStudenti
                              FROM Insegnamento
                              WHERE CodCorso = X.CodCorso )
```

Oppure

```
SELECT CodCorso, AnnoAccademico
FROM Insegnamento
WHERE (CodCorso, NumeroStudenti) IN ( SELECT CodCorso, max(NumeroStudenti)
                                       FROM Insegnamento
                                       GROUP BY CodCorso )
```

8. Le edizioni dei corsi del primo anno con il maggior numero di studenti.

```
SELECT Codice, AnnoAccademico
FROM Corso join Insegnamento on Codice = CodCorso
WHERE AnnoDiCorso = 1
      AND NumeroStudenti = ( SELECT MAX( NumeroStudenti )
                            FROM Corso join Insegnamento on Codice = CodCorso
                            WHERE AnnoDiCorso = 1 )
```

9. Di ogni facoltà, i corsi con il minor numero di studenti.

```
SELECT Facoltà, Codice, Nome, NumeroStudenti, AnnoAccademico
FROM Corso C join Insegnamento on Codice = CodCorso
WHERE NumeroStudenti = ( SELECT MIN( NumeroStudenti )
                        FROM Corso join Insegnamento on Codice = CodCorso
                        WHERE Facoltà = C.Facoltà )
```

Oppure, usando il costruttore di tupla (che sostituisce la necessità del passaggio di un binding):

```
SELECT Facoltà, Codice, Nome, NumeroStudenti, AnnoAccademico
FROM Corso join Insegnamento on Codice = CodCorso
WHERE ( Facoltà, NumeroStudenti ) IN ( SELECT Facoltà, MIN( NumeroStudenti )
                                         FROM Corso join Insegnamento on Codice = CodCorso
                                         GROUP BY Facoltà )
```

10. Studenti che hanno preso più 27 che 24.

```
SELECT MatrStud
FROM Esame E
WHERE Voto = 27
GROUP BY MatrStud
HAVING COUNT(*) > ( SELECT COUNT(*)
                    FROM Esame E2
                    WHERE E.MatrStud = E2.MatrStud
                    AND Voto = 24 )
```

11. Studenti che hanno preso più 30L che 30.

```
SELECT MatrStud
FROM Esame E
WHERE Voto = 30 AND Lode = 'yes'
GROUP BY MatrStud
HAVING COUNT(*) > ( SELECT COUNT(*)
                    FROM Esame
                    WHERE MatrStud = E.MatrStud
                    AND Voto = 30 AND Lode = 'no' )
```

12. Studenti che hanno preso più volte 27 rispetto a qualsiasi altro voto (complessivamente).

```
SELECT MatrStud
FROM Esame E
WHERE Voto = 27
GROUP BY MatrStud
HAVING COUNT(*) > ( SELECT COUNT(*)
                    FROM Esame
                    WHERE MatrStud = E.MatrStud
                    AND Voto <> 27 )
```

Equivalente a

```

SELECT MatrStud
FROM Esame E
WHERE Voto = 27
GROUP BY MatrStud
HAVING COUNT(*) > 0.5 * ( SELECT COUNT(*)
                        FROM Esame
                        WHERE MatrStud = E.MatrStud )

```

13. Studenti che hanno preso più volte 27 rispetto a qualsiasi altro voto (individualmente).

```

SELECT MatrStud
FROM Esame E
WHERE Voto = 27
GROUP BY MatrStud
HAVING COUNT(*) > ALL ( SELECT COUNT(*)
                        FROM Esame
                        WHERE MatrStud = E.MatrStud AND Voto <> 27
                        GROUP BY Voto )


```

La soluzione precedente è corretta per una interpretazione che considera il 30 e il 30 e Lode come un unico voto. Come si fa a *contare separatamente i 30 e i 30 e lode*?

```

SELECT MatrStud
FROM Esame E
WHERE Voto = 27
GROUP BY MatrStud
HAVING COUNT(*) > ALL ( SELECT COUNT(*)
                        FROM Esame
                        WHERE MatrStud = E.MatrStud AND Voto <> 27
                        GROUP BY Voto, Lode )

```



In alternative, meno “elegante”, se non si è pensato al raggruppamento per lode:

```

SELECT MatrStud
FROM Esame E
WHERE Voto = 27
GROUP BY MatrStud
HAVING COUNT(*) > ALL ( SELECT COUNT(*)
                        FROM Esame
                        WHERE MatrStud = E.MatrStud AND Voto <> 27 AND Lode = 'no'
                        GROUP BY Voto
                        UNION
                        SELECT COUNT(*)
                        FROM Esame
                        WHERE MatrStud = E.MatrStud AND Lode = 'yes' )

```

14. Matricola, nome e cognome degli studenti che hanno superato almeno 3 esami del secondo anno ma meno di 3 esami del primo.

La traduzione diretta delle due condizioni in due query distinte (che formuliamo per comodità sotto forma di viste) ci permette di usare l'intersezione.

CREATE VIEW AlmenoTreDelSecondoAnno as

```
SELECT Matricola, Nome, Cognome
FROM ( Studente join Esame on Matricola = MatrStud ) join Corso on CodCorso = Codice
WHERE AnnoDiCorso = 2
GROUP BY Matricola, Nome, Cognome
HAVING COUNT(*) >= 3
```

CREATE VIEW MenoDiTreDelPrimoAnno as

```
SELECT Matricola, Nome, Cognome
FROM Studente
WHERE Matricola NOT IN ( SELECT MatrStud
                        FROM Esame join Corso on CodCorso = Codice
                        WHERE AnnoDiCorso = 1
                        GROUP BY MatrStud
                        HAVING COUNT(*) >= 3 )
```

```
SELECT *
FROM AlmenoTreDelSecondoAnno
INTERSECT
SELECT *
FROM MenoDiTreDelPrimoAnno
```

*ATTENZIONE! Si noti che anche chi non ha sostenuto nessun esame del primo anno può appartenere al risultato della query, se ne ha sostenuti 3 del secondo (chiamiamo questi studenti “**studenti estremi**”).*

Sarebbe quindi sbagliato definire la view MenoDiTreDelPrimoAnno nel seguente modo, poiché non estrarrebbe gli “studenti estremi” (per i quali non ci sarebbe alcun “gruppo” su cui contare 0 esami) :

~~**CREATE VIEW** MenoDiTreDelPrimoAnno as~~

~~```
SELECT Matricola, Nome, Cognome
FROM (Studente join Esame on Matricola = MatrStud) join Corso on CodCorso = Codice
WHERE AnnoDiCorso = 1
GROUP BY Matricola, Nome, Cognome
HAVING COUNT(*) < 3
```~~

*Volendo invece usare una query sola, si può esprimere la seconda parte attraverso una query annidata:*

```
SELECT Matricola, Nome, Cognome
FROM (Studente S join Esame on Matricola = MatrStud) join Corso on CodCorso = Codice
WHERE AnnoDiCorso = 2
AND 3 > (SELECT COUNT(*)
 FROM Esame join Corso on CodCorso = Codice
 WHERE AnnoDiCorso = 1 AND MatrStud = S.Matricola
 GROUP BY MatrStud)
GROUP Matricola, Nome, Cognome
HAVING COUNT(*) >= 3
```



Si noti che è quindi sbagliata anche la seguente versione, nella quale gli “studenti estremi” studente **non** sarebbero estratto dalla query annidata.

```
SELECT Matricola, Nome, Cognome
FROM (Studente join Esame on Matricola = MatrStud) join Corso on CodCorso = Codice
WHERE AnnoDiCorso = 2
 AND Matricola IN (SELECT MatrStud
 FROM Esame join Corso on CodCorso = Codice
 WHERE AnnoDiCorso = 1
 GROUP BY MatrStud
 HAVING COUNT(*) < 3)
GROUP Matricola, Nome, Cognome
HAVING COUNT(*) >= 3
```

Si può invece correttamente controllare la condizione in una query annidata con un NOT IN

```
SELECT Matricola, Nome, Cognome
FROM (Studente join Esame on Matricola = MatrStud) join Corso on CodCorso = Codice
WHERE AnnoDiCorso = 2
 AND Matricola NOT IN (SELECT MatrStud
 FROM Esame join Corso on CodCorso = Codice
 WHERE AnnoDiCorso = 1
 GROUP BY MatrStud
 HAVING COUNT(*) >= 3)
GROUP Matricola, Nome, Cognome
HAVING COUNT(*) >= 3
```

15. Matricola, nome e cognome degli studenti che hanno superato più esami del terzo anno che del secondo.

```
SELECT Matricola, Nome, Cognome
FROM Studente S join Esame on Matricola = MatrStud) join Corso on CodCorso = Codice
WHERE AnnoDiCorso = 3
GROUP BY Matricola, Nome, Cognome
HAVING COUNT(*) > (SELECT COUNT(*)
 FROM Esame join Corso on CodCorso = Codice
 WHERE AnnoDiCorso = 2 AND MatrStud = S.Matricola)
```

16. Il corso con la media più bassa.

```
SELECT CodCorso, AVG(Voto)
FROM Esame
GROUP BY CodCorso
HAVING AVG(Voto) <= ALL (SELECT AVG(Voto)
 FROM Esame
 GROUP BY CodCorso)
```

Oppure con una vista:

```

CREATE VIEW MediaPerCorso (Corso, Media) AS
SELECT CodCorso, AVG(Voto)
FROM Esame
GROUP BY CodCorso

SELECT Corso
FROM MediaPerCorso
WHERE Media = (SELECT min(Media) FROM MediaPerCorso)

```

17. Studenti con la media **pesata in base ai CFU** più alta.

```

CREATE VIEW MediaStudente (Matricola, Media) AS
SELECT MatrStud, SUM(Voto * NumeroCrediti) / SUM(NumeroCrediti)
FROM Esame join Corso on CodCorso = Codice
GROUP BY MatrStud

SELECT Matricola
FROM MediaStudente
WHERE Media = (SELECT MAX(Media)
 FROM MediaStudente)

```

*Oppure*

```

SELECT MatrStud, SUM(Voto * NumeroCrediti) / SUM(NumeroCrediti) AS Media
FROM Esame join Corso on CodCorso = Codice
GROUP BY MatrStud
HAVING SUM(Voto * NumeroCrediti) / SUM(NumeroCrediti)
 >= ALL
 (SELECT SUM(Voto * NumeroCrediti) / SUM(NumeroCrediti)
 FROM Esame join Corso on CodCorso = Codice
 GROUP BY MatrStud)

```

18. Matricola e media degli studenti che hanno una media maggiore della “media” media di ateneo.

```

SELECT Matricola, Media
FROM MediaStudente
WHERE Media > (SELECT AVG(Media)
 FROM MediaStudente)

```

*Si noti che si riusa la vista definita precedentemente, e che la “media media” è calcolata considerando equivalente il contributo di ogni studente (indipendentemente da numero e peso degli esami sostenuti)*

19. Per ogni studente, l’anno di corso in cui ha avuto la media più alta.

```

CREATE VIEW MediaStudentePerAnno (Matricola, Anno, Media) AS
SELECT MatrStud, AnnoDiCorso,
 SUM(Voto * NumeroCrediti) / SUM(NumeroCrediti)
FROM Esame join Corso on CodCorso = Codice
GROUP BY MatrStud, AnnoDiCorso

```



23. Trovare i “top ten” studenti in base alla media pesata, tra quelli che abbiano sostenuto almeno 10 esami.

```
SELECT Matricola, Media
FROM MediaStudente MS1
WHERE Matricola IN (SELECT MatrStud
 FROM Esame
 GROUP BY MatrStud
 HAVING COUNT(*) >= 10)
AND 9 >= (SELECT COUNT(*)
 FROM MediaStudente
 WHERE Matricola IN (SELECT MatrStud
 FROM Esame
 GROUP BY MatrStud
 HAVING COUNT(*) >= 10)
 AND Media > MS1.Media)
```

*Per evitare di ripetere due volte la query annidata e dare una formulazione più compatta si può definire la vista MediaStudenteStatisticamenteRilevante, che elenca le medie dei soli studenti con almeno 10 esami.*

```
CREATE VIEW MediaStudenteStatisticamenteRilevante (Matricola, Media) AS
SELECT Matricola, Media
FROM MediaStudente
WHERE Matricola IN (SELECT MatrStud
 FROM Esame
 GROUP BY MatrStud
 HAVING COUNT(*) >= 10)
```

*Oppure, in alternativa, per non introdurre un “doppio livello di viste” che non è strettamente necessario:*

```
CREATE VIEW MediaStudenteStatisticamenteRilevante (Matricola, Media) AS
SELECT MatrStud, SUM(Voto * NumeroCrediti) / SUM(NumeroCrediti)
FROM Esame join Corso on CodCorso = Codice
GROUP BY MatrStud
HAVING COUNT(*) >= 10
```

*A questo punto, la query si può formulare come:*

```
SELECT Matricola, Media
FROM MediaStudenteStatisticamenteRilevante MS1
WHERE 9 >= (SELECT COUNT(*)
 FROM MediaStudenteStatisticamenteRilevante
 WHERE Media > MS1.Media)
```

24. Corsi tenuti da professori di Torino che non sono stati superati da nessuno studente di Torino.

```
SELECT CodCorso
FROM Insegnamento join Professore on MatrProf = Matricola
WHERE Città = 'Torino' AND
 CodCorso NOT IN (SELECT CodCorso
 FROM Esame join Studente on MatrStud = Matricola
 WHERE Città = 'Torino')
```

*Oppure con not exists:*

```
SELECT CodCorso
FROM Insegnamento INS join Professore on MatrProf = Matricola
WHERE Città = 'Torino' AND
 NOT EXISTS (SELECT *
 FROM Esame E join Studente on MatrStud = Matricola
 WHERE Città = 'Torino' AND E.CodCorso = INS.CodCorso)
```

25. Studenti che non hanno mai sostenuto esami per corsi tenuti da docenti con il loro stesso cognome.

```
SELECT Matricola
FROM Studente S
WHERE Matricola NOT IN (SELECT MatrStud
 FROM (Esame E join Insegnamento I on E.CodCorso = I.CodCorso)
 join Professore P on I.MatrProf = P.Matricola
 WHERE P.Cognome = S.Cognome)
```

*Oppure, per chi non ama i passaggi di binding:*

```
SELECT Matricola
FROM Studente
WHERE (Matricola, Cognome) NOT IN (SELECT MatrStud, P.Cognome
 FROM (Esame E join Insegnamento I on E.CodCorso = I.CodCorso)
 join Professore P on I.MatrProf = P.Matricola)
```

26. Studenti che hanno sostenuto al più 5 esami di corsi di una stessa facoltà.

```
SELECT *
FROM Studente S
WHERE 5 <= ALL (SELECT count(*)
 FROM Esame join Corso on CodCorso = Codice
 WHERE MatrStud = S.Matricola
 GROUP BY Facoltà)
```

27. Studenti che hanno sostenuto almeno due esami di corsi tenuti dallo stesso docente.

```
SELECT E1.MatrStud
FROM Esame E1 join Insegnamento I1 on E1.CodCorso = I1.CodCorso,
 Esame E2 join Insegnamento I2 on E2.CodCorso = I2.CodCorso
WHERE E1.CodCorso <> E2.CodCorso AND I1.MatrProf = I2.MatrProf AND E1.MatrProf = E2.MatrProf
```

*Oppure, usando il raggruppamento:*

```
SELECT MatrStud
FROM Esame E join Insegnamento I on E.CodCorso = I.CodCorso
GROUP BY MatrStud, MatrProf
HAVING count(*) >= 2
```

28. Studenti che non hanno mai sostenuto due esami di corsi tenuti dallo stesso docente.

```
SELECT Matricola
FROM Studente
WHERE Matricola NOT IN (SELECT E1.MatrStud
 FROM Esame E1, Insegnamento I1, Esame E2, Insegnamento I2
 WHERE E1.CodCorso = I1.CodCorso AND E2.CodCorso = I2.CodCorso AND
 E1.CodCorso <> E2.CodCorso AND I1.MatrProf = I2.MatrProf)
```

*Oppure*

```
SELECT MatrStud
FROM Esame E join Insegnamento I on E.CodCorso = I.CodCorso
GROUP BY MatrStud
HAVING COUNT(*) = COUNT(DISTINCT MatrProf)
```

29. Trovare i codici e i nomi dei corsi con il minimo numero di crediti.

```
SELECT Codice, Nome
FROM Corso
WHERE NumeroCrediti = (SELECT MIN(NumeroCrediti)
 FROM Corso)
```

*Oppure:*

```
SELECT Codice, Nome
FROM Corso
WHERE NumeroCrediti <= ALL (SELECT NumeroCrediti
 FROM Corso)
```

30. Trovare i codici e nomi dei corsi del primo anno con il minimo numero di crediti.

```
SELECT Codice, Nome
FROM Corso
WHERE AnnoDiCorso = 1 AND NumeroCrediti = (SELECT MIN(NumeroCrediti)
 FROM Corso
 WHERE AnnoDiCorso = 1)
```

*Oppure:*

```
SELECT Codice, Nome
FROM Corso
WHERE AnnoDiCorso = 1 AND NumeroCrediti <= ALL (SELECT NumeroCrediti
 FROM Corso
 WHERE AnnoDiCorso = 1)
```

31. Trovare, per ogni facoltà, i codici e i nomi dei corsi con il massimo numero di crediti di quella facoltà.

```
SELECT Facoltà, Codice, Nome
FROM Corso C
WHERE NumeroCrediti = (SELECT MAX(NumeroCrediti)
 FROM Corso
 WHERE Facoltà = C.Facoltà)
```

*Oppure (usando il costruttore di tupla):*

```

SELECT Facoltà, Codice, Nome
FROM Corso
WHERE (Facoltà, NumeroCrediti) IN (SELECT Facoltà, MAX(NumeroCrediti)
 FROM Corso
 GROUP BY Facoltà)

```

32. Facoltà che forniscono il maggior numero di corsi.

```

SELECT Facoltà
FROM Corso
GROUP BY Facoltà
HAVING COUNT(*) >= ALL (SELECT COUNT(*)
 FROM Corso
 GROUP BY Facoltà)

```

33. Professori che hanno tenuto il maggior numero di corsi.

```

SELECT MatrProf
FROM Insegnamento
GROUP BY MatrProf
HAVING COUNT(*) >= ALL (SELECT COUNT(*)
 FROM Insegnamento
 GROUP BY MatrProf)

```

34. Professori che hanno tenuto il maggior numero di corsi da almeno 5 crediti.

```

SELECT MatrProf
FROM Insegnamento join Corso on CodCorso = Codice
WHERE NumeroCrediti >= 5
GROUP BY MatrProf
HAVING COUNT(*) >= ALL (SELECT COUNT(*)
 FROM Insegnamento join Corso on CodCorso = Codice
 WHERE NumeroCrediti >= 5
 GROUP BY MatrProf)

```

35. Corsi in cui almeno uno studente che ha superato l'esame aveva lo stesso cognome del docente.

```

SELECT E.CodCorso
FROM Studente S join Esame E on E.MatrStud = S.Matricola,
 Insegnamento I join Professore P on I.MatrProf = P.Matricola
WHERE E.CodCorso = I.CodCorso AND S.Cognome = P.Cognome

```