Soluzione Esercitazione 5

Esercizio 1
Trovare gli studenti che hanno preso più 27 che 24

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

Esercizio 2
Trovare le informazioni sugli studenti e sugli esami che hanno sostenuto. Devono essere inclusi nel risultato anche gli studenti che non hanno sostenuto esami.

```
SELECT *
FROM Studente LEFT JOIN Esame ON Matricola=MatrStud
```

Esercizio 3
Trovare la matricola, nome e cognome degli studenti che hanno superato gli esami di tutti i corsi.

```
SELECT Matricola, Nome, Cognome
FROM Studente, Esame
WHERE MatrStud=Matricola
GROUP BY Matricola, Nome, Cognome
HAVING COUNT(*) = (SELECT COUNT(*)
                   FROM Corso)
```

Esercizio 4
Calcolare la media delle medie dei voti ottenuti dagli studenti.

```
CREATE VIEW Media(MatrStud, VotoMedio) AS (
    SELECT MatrStud, AVG(Voto)
    FROM Esame
    GROUP BY MatrStud)

SELECT AVG(VotoMedio)
FROM Media
```

Esercizio 5
Trovare voto medio e matricola degli studenti con voto medio maggiore del voto medio complessivo.

```
CREATE VIEW Media(MatrStud, VotoMedio) AS (
    SELECT MatrStud, AVG(Voto)
    FROM Esame
    GROUP BY MatrStud)

SELECT MatrStud, VotoMedio
FROM Media
WHERE VotoMedio > (SELECT AVG(Voto)
                   FROM Esame)
```

Oppure, senza vista:

```
SELECT MatrStud, AVG(Voto)
FROM Esame
GROUP BY MatrStud
HAVING AVG(Voto) > (SELECT AVG(Voto)
                    FROM Esame)
```

Esercizio 6

Trovare gli studenti (matricola, nome e cognome) che hanno superato almeno 3 esami del secondo anno ma meno di 3 esami del primo

```
SELECT S.Matricola, S.Nome, S.Cognome
FROM Studente AS S, Esame AS E, Corso AS C
WHERE S.Matricola = E.MatrStud
      AND E.CodCorso = C.Codice
      AND C.AnnoDiCorso = 2
      AND S.Matricola NOT IN (SELECT E2.MatrStud
                              FROM Esame AS E2, Corso AS C2
                              WHERE E2.CodCorso=C2.Codice
                              AND C2.AnnoDiCorso=1
                              GROUP BY E2.MatrStud
                              HAVING COUNT(*) >=3)
GROUP BY S.Matricola, S.Nome, S.Cognome
HAVING COUNT(*) >=3
```

Si può fare anche con un EXCEPT:

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

EXCEPT

SELECT S.Matricola, S.Nome, S.Cognome
FROM Studente AS S, Esame AS E, Corso AS C
```

```
WHERE S.Matricola = E.MatrStud
      AND E.CodCorso = C.Codice
      AND C.AnnoDiCorso = 1
GROUP BY S.Matricola, S.Nome, S.Cognome
HAVING COUNT(*) >=3
```

Esercizio 7

Trovare gli studenti (con anche nome e cognome) che hanno ottenuto 3 voti distinti tra quelli che hanno svolto almeno 10 esami

```
SELECT Matricola, Nome, Cognome
FROM Studente
JOIN Esame ON Matricola = MatrStud
GROUP BY Matricola, Nome, Cognome
HAVING COUNT(*) >=10
       AND COUNT(DISTINCT Voto)=3
```

Esercizio 8

Trovare gli studenti con la media pesata più alta

```
SELECT E.MatrStud, SUM(E.Voto * C.NumCrediti)/SUM(C.NumCrediti)
FROM Esame AS E
JOIN Corso AS C
ON E.CodCorso = C.Codice
GROUP BY E.MatrStud
HAVING SUM(E.Voto * C.NumCrediti)/SUM(C.NumCrediti) >= ALL (
    SELECT SUM(E1.Voto *C1.NumCrediti)/SUM(C1.NumCrediti)
    FROM Esame AS E1
    JOIN Corso AS C1 ON E1.CodCorso = C1.Codice
    GROUP BY E1.MatrStud)
```

a sinistra dell'HAVING si possono usare aggregati sui gruppi.

Oppure, con vista:

```
CREATE VIEW MediaStudente (Matricola, Media) AS
    (SELECT MatrStud, SUM(Voto*NumCrediti)/SUM(NumCrediti)
    FROM Esame JOIN Corso ON CodCorso = Codice
    GROUP BY MatrStud)

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

Esercizio 9

Trovare per ogni studente l'anno di corso in cui ha avuto la media pesata più alta

```
CREATE VIEW MediaStudentePerAnno (Matricola, Anno, Media) AS
   (SELECT MatrStud, AnnoDiCorso,
           SUM(Voto*NumCrediti)/SUM(NumCrediti)
```

```
    FROM Esame, Corso
    WHERE CodCorso = Codice
    GROUP BY MatricolaStudente, AnnoDiCorso)

SELECT MS1.Matricola, MS1.Anno
FROM MediaStudentePerAnno AS MS1
WHERE MS1.Media = (SELECT MAX(MS2.Media)
                   FROM MediaStudentePerAnno AS MS2
                   WHERE MS2.Matricola = MS1.Matricola)
```

La query annidata utilizza la variabile definita esternamente nel WHERE per filtrare le righe che si riferiscono a ciascuna matricola: non è necessario fare un group by!

Esercizio 10

Trovare gli studenti più regolari, ovvero quelli con la minima differenza tra il voto migliore e il voto peggiore

```
CREATE VIEW StudenteMinMax(Matricola, VotoMigl, VotoPegg) AS
        (SELECT MatricolaStudente, MAX(Voto), MIN(Voto)
        FROM Esame
        GROUP BY MatricolaStudente)

SELECT Matricola
FROM StudenteMinMax
WHERE (VotoMigl-VotoPegg) = (SELECT MIN(VotoMigl - VotoPegg)
                             FROM StudenteMinMax)
```

Esercizio 11

Corsi in cui almeno il 50% degli studenti ha preso un voto maggiore di 25.

```
SELECT CodCorso
FROM Esame E
WHERE Voto > 25
GROUP BY CodCorso
HAVING COUNT(*) >= 0.5 * ( SELECT COUNT(*)
                           FROM Esame
                           WHERE CodCorso = E.CodCorso )
```

Esercizio 12

Studenti che hanno preso lo stesso voto in più di due terzi degli esami sostenuti.

```
SELECT MatrStud
FROM Esame E
GROUP BY MatrStud, Voto
HAVING COUNT(*) >= 2/3 * ( SELECT COUNT(*)
                           FROM Esame
                           WHERE Matricola = E.Matricola )
```

Esercizio 13

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

Usando solo gli operatori visti a lezione:

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

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

dove la condizione nel WHERE permette di filtrare le righe della tabella e tenere solo e soltanto gli studenti top ten. Ad esempio, instanziando il where, per lo studente migliore si avrà che la query annidata darà risultato 0 (non si conteranno studenti con media maggiore), per il secondo studente migliore si conta un solo studente con media maggiore, ecc ecc: in tutti questi casi (e fino al decimo studente) la condizione nel where è verificata, dunque gli studenti saranno tenuti nel risultato.

Oppure usando operatori non visti a teoria:

```
CREATE VIEW MediePeggiori(MediaVoto) AS (
    SELECT DISTINCT MediaVoto
    FROM MediaStudentePesata
    ORDER BY MediaVoto DESC
    OFFSET 10)

SELECT Matricola, MediaVoto
FROM MediaStudentePesata
WHERE MediaVoto >= ALL(SELECT MediaVoto
                  FROM MediePeggiori)
```

Questa seconda formulazione sarebbe più precisa, dato che tiene conto di possibili medie uguali tra studenti top ten (che implica, controintuitivamente, che i top ten sono più di 10).

Esercizio 14
Trovare CF, Nome e Cognome degli atleti che hanno partecipato ad almeno due gare ma mai ad una finale.

```
SELECT CF_Atleta, Nome, Cognome
FROM ATLETA
JOIN PARTECIPAZIONE ON CF_Atleta=CF_Atleta
WHERE CF_Atleta NOT IN ( SELECT CF_Atleta
                         FROM PARTECIPAZIONE
                         JOIN GARA ON ID_Gara=ID_Gara
                         WHERE EFinale=True)
GROUP BY CF_Atleta, Nome, Cognome
HAVING COUNT(*)>=2
```

Esercizio 15

Mostrare per ogni nazione il numero di medaglie d'oro vinte (si vince una medaglia d'oro quando si ottiene il tempo minore in una finale).

```
SELECT A1.Nazione, COUNT(*)
FROM ATLETA AS A1
JOIN PARTECIPAZIONE AS P1 ON A1.CF_Atleta=P1.CF_Atleta
JOIN GARA AS G1 ON P1.ID_Gara=G1.ID_Gara
WHERE G1.EFinale=True AND P1.Tempo = ( SELECT MIN(P2.TEMPO)
                                       FROM PARTECIPAZIONE AS P2
                                       WHERE P1.ID_Gara=P2.ID_Gara)
GROUP BY A1.Nazione
```

Esercizio 16

Trovare il nome e l'ID del piatto (o dei piatti) più calorico.

```
CREATE VIEW CaloriePiatti (IdPiatto, CalorieTotali)
        SELECT IdPiatto, SUM(KCaloriePerGrammo*QuantitaInGrammi)
        FROM Ricetta NATURAL JOIN Ingrediente
        GROUP BY IdPiatto

SELECT IdPiatto, Nome
FROM CaloriePiatti NATURAL JOIN Piatto
WHERE CalorieTotali = (SELECT MAX(CalorieTotali)
FROM CaloriePiatti)
```

Possiamo creare una vista che ci fornisce per ogni piatto il totale delle calorie. Dopodiché vediamo quale è il massimo

Esercizio 17

Trovare il nome degli studenti che, considerando tutti i loro ordini complessivamente, hanno ordinato al massimo due diversi tipi di piatti contenenti solo ingredienti con un numero di calorie per grammo superiore a 2.

```
CREATE VIEW PiattiConIngredientiPiu2Cal (IdPiatto) AS
        SELECT DISTINCT IdPiatto
        FROM Ricetta
        WHERE IdPiatto NOT IN (SELECT IdPiatto
        FROM Ricetta NATURAL JOIN Ingrediente
        WHERE KCaloriePerGrammo <= 2)

SELECT Nome, IdStud
FROM Studente
LEFT JOIN Ordine ON IdStud = IdStudente
NATURAL LEFT JOIN PiattiConIngredientiPiu2Cal
GROUP BY IdStud, Nome
HAVING COUNT(DISTINCT IdPiatto) <= 2
```

Esercizio 18

Trovare per ogni piatto, la data e l'ora del suo primo ordine. (Nota: idOrdine NON ha valori generati in ordine cronologico)

```
SELECT DISTINCT IDPiatto, DataOrd, OraOrd
FROM Ordine O
WHERE DataOrd = (SELECT MIN(O2.DataOrd)
        FROM Ordine AS O2
        WHERE O.IdPiatto = O2.IdPiatto)
        AND OraOrd = (SELECT MIN(O3.OraOrd)
                              FROM Ordine AS O3
                              WHERE O.IdPiatto = O2.IdPiatto
                              AND O.DataOrd=O3.DataOrd)
```