HOMEWORK BUSINESS INTELLIGENCE

ALESSANDRO MOTTA – MATR. 812309

MATTEO PAPARELLA – MATR. 812561

GABRIELE ZOTTOLA – MATR. 812363

QUERY 1 - NORMALIZZATO

Abbiamo raggruppato e contato gli studenti per ogni corso di studi, codice appello e relativo nome esame. Per poter isolare l'anno dalla data di appello abbiamo utilizzato la funzione SUBSTR(), che permette di estrapolare da una stringa un numero 'n' di valori, da una posizione 'x' da noi indicata. Inoltre, si è deciso di raggruppare per la data di ogni appello, in modo tale da ottenere una visione completa della distribuzione richiesta.

Visualizzazione:
https://public.tableau.com/profile/gabrielezottola#!/vizhome/Query1parte1/Dashboard1
https://public.tableau.com/profile/gabrielezottola#!/vizhome/Query1parte2/Dashboard1?publish=yes

```
SELECT COUNT(DISTINCT studenti.studente) AS numerostudenti,
       iscrizioni.appcod,
       ad.ad,
       cds,
       SUBSTR(appelli.dtappello, -4, 4) AS Annoappello,
       appelli.dtappello
  FROM appelli,
       iscrizioni,
       studenti,
       cds,
       ad
 WHERE appelli.appcod = iscrizioni.appcod AND
       iscrizioni.studente = studenti.studente AND
       cds.cdscod = appelli.cdscod AND
       appelli.adcod = ad.adcod
 GROUP BY Annoappello,
          dtappello,
          cds,
          ad.ad,
          iscrizioni.appcod
 ORDER BY Annoappello ASC;
```

QUERY 1 - DENORMALIZZATO

Per il trattamento della Query 1 denormalizzata, abbiamo seguito lo stesso approccio applicato al modello normalizzato, fatta eccezione per due aspetti:

- L'assenza di join tra tabelle;
- Il trattamento dell'anno delle date appello (abbiamo cambiato i parametri della funzione SUBSTR() e concatenato con il valore 20, poiché la data si presentava in formato '%dd-%mm-%YY')

```
SELECT COUNT(DISTINCT studente) AS numerostudenti,
       ad,
       Cds,
       '20' || SUBSTR(DtAppello, -2, 2) AS Annoappello,
       DtAppello
  FROM bos_denormalizzato
 GROUP BY Annoappello,
          dtappello,
          Cds,
          AD
 ORDER BY Annoappello ASC;
```

QUERY 11 - NORMALIZZATO

Per poter svolgere la richiesta, si è partiti calcolando il tasso di superamento, si sono quindi contati gli studenti come il numero totale di iscritti e ci si è avvalsi della funzione CASE() per contare unicamente gli studenti che hanno come condizione 'superamento' uguale a 1. Si è fatto coincidere l'anno accademico con l'anno solare (utilizzo della funzione SUBSTR()).
Inoltre, si è notata un'incongruenza tra il numero di studenti iscritti ed il numero di studenti che hanno passato l'esame. Motivo per cui, si è deciso di inserire la condizione HAVING superato < totiscritti.
Una volta calcolato il tasso di superamento (arrotondato alla seconda cifra decimale), abbiamo selezionato la Top 10 degli esami più 'difficili'.

Visualizzazione:
https://public.tableau.com/profile/gabrielezottola#!/vizhome/dashboardquery2/Foglio1

```
SELECT ROUND( (superato / totiscritti) * 100, 2) AS tasso,
       ad,
       cds,
       Annoappello
  FROM (
           SELECT ad,
                  cds,
                  appcod,
                  CAST (superato AS DOUBLE) AS superato,
                  CAST (totiscritti AS DOUBLE) AS totiscritti,
                  Annoappello
             FROM (
                      SELECT appelli.appcod,
                             ad.ad,
                             cds.cds,
                             count(DISTINCT studente) AS totiscritti,
                             count(DISTINCT CASE superamento WHEN 1 THEN studente ELSE 0 END) AS superato,
                             SUBSTR(appelli.dtappello, -4, 4) AS Annoappello
                        FROM iscrizioni,
                             appelli,
                             ad,
                             cds
                       WHERE iscrizioni.appcod = appelli.appcod AND
                             cds.cdscod = appelli.cdscod AND
                             appelli.adcod = ad.adcod
                       GROUP BY ad.ad,
                                cds.cds,
                                Annoappello
                      HAVING superato < totiscritti
                  )
                  A
           GROUP BY ad,
                    cds,
                    Annoappello
       )
       B
 GROUP BY ad,
          cds,
          Annoappello
 ORDER BY tasso ASC;
```

QUERY 11 - DENORMALIZZATO

Nello svolgimento della Query II denormalizzata abbiamo seguito il modello normalizzato, eccetto come variazioni:

- Non sono state effettuate join tra tabelle;
- Metodologia di estrazione dell'anno di riferimento di ogni data di appello;
- Non è presente l'attributo 'Appcode', quindi non è stato selezionato.

```
SELECT ROUND( (superato / totiscritti) * 100, 3) AS tasso,
       AD,
       Cds,
       Annoappello
  FROM (
           SELECT AD,
                  Cds,
                  CAST (superato AS DOUBLE) AS superato,
                  CAST (totiscritti AS DOUBLE) AS totiscritti,
                  Annoappello
             FROM (
                      SELECT AD,
                             Cds,
                             count(DISTINCT Studente) AS totiscritti,
                             count(DISTINCT CASE Superamento WHEN 1 THEN Studente ELSE 0 END) AS superato,
                             '20' || SUBSTR(DtAppello, -2, 2) AS Annoappello
                        FROM bos_denormalizzato
                       GROUP BY AD,
                                Cds,
                                Annoappello
                      HAVING superato < totiscritti
                  )
                  A
            GROUP BY AD,
                     Cds,
                     Annoappello
       )
       B
 GROUP BY AD,
          Cds,
          Annoappello
 ORDER BY tasso ASC;
```

QUERY 111 - NORMALIZZATO

Per il calcolo del tasso di commitment si è partiti dal conteggio del numero di appelli diversi che sono stati sostenuti il medesimo giorno, per ogni corso di laurea. Il tasso di commitment è stato calcolato come rapporto tra la somma degli appelli diversi ed il numero di appelli totali sostenuti per corso di studi.

Visualizzazione:
https://public.tableau.com/views/visualizzazione_query_3/Foglio4?:language=en&:display_count=y&:origin=viz_share_link

```
SELECT cds,
       numeroesamidiversi,
       conteggiocds,
       numeroesamidiversi / conteggiocds AS tassocommitment
  FROM (
           SELECT cds,
                  SUM(num_corsi_diversi) AS numeroesamidiversi,
                  count(cds) AS conteggiocds
             FROM (
                      SELECT CAST (num_corsi_diversi AS DOUBLE) AS num_corsi_diversi,
                             cds
                        FROM (
                                 SELECT cds,
                                        dtappello,
                                        count(DISTINCT appelli.adcod) AS num_corsi_diversi
                                   FROM appelli,
                                        cds,
                                        ad
                                  WHERE appelli.adcod = ad.adcod AND
                                        cds.cdscod = appelli.cdscod
                                  GROUP BY dtappello,
                                           appelli.cdscod
                             )
                             A
                  )
            GROUP BY cds
       )
 GROUP BY cds
 ORDER BY tassocommitment DESC
 LIMIT 20;
```

QUERY 111 - DENORMALIZZATO

L'unica differenza rispetto al modello normalizzato è stata l'impossibilità di effettuare join tra tabelle.

```
SELECT Cds,
       sommanumerocorsi,
       conteggiocds,
       sommanumerocorsi / conteggiocds AS tassocommitment
  FROM (
           SELECT CdS,
                  SUM(num_corsi_diversi) AS sommanumerocorsi,
                  count(CdS) AS conteggiocds
             FROM (
                      SELECT CAST (num_corsi_diversi AS DOUBLE) AS num_corsi_diversi,
                             cds
                        FROM (
                                 SELECT CdS,
                                        DtAppello,
                                        count(DISTINCT AdCod) AS num_corsi_diversi
                                   FROM bos_denormalizzato
                                  GROUP BY DtAppello,
                                           CdScod
                             )
                             A
                  )
            GROUP BY CdS
       )
 GROUP BY Cds
 ORDER BY tassocommitment DESC
 LIMIT 20;
```

QUERY IV - NORMALIZZATO

Si è calcolata la media dei voti per ogni esame e corso di studi, affiancando a tale valore il tasso di superamento calcolato come rapporto tra il numero di studenti che hanno superato l'esame e il numero di studenti iscritti.
Abbiamo notato la presenza di alcuni studenti che hanno superato l'esame, ma il voto non è stato loro registrato, di conseguenza abbiamo inserito la condizione 'mediavoto IS NOT NULL'.

Visualizzazione:
https://public.tableau.com/views/visualizzazione_query_4/Foglio5?:language=en&:display_count=y&:origin=viz_share_link
https://public.tableau.com/profile/alessandro.motta3078#!/vizhome/EsamifacilieDifficili/Dashboard1?publish=yes

```
SELECT ad,
       cds,
       ROUND(mediavoto, 3) AS mediavoto, ROUND( (superato / totiscritti) * 100, 3) AS tassosuperamento
  FROM (
            SELECT ad,
                   cds,
                   voto,
                   mediavoto,
                   CAST (superato AS DOUBLE) AS superato,
                   CAST (totiscritti AS DOUBLE) AS totiscritti
              FROM (
                        SELECT ad.ad,
                               cds.cds,
                               count(DISTINCT studente) AS totiscritti,
                               count(DISTINCT CASE superamento WHEN 1 THEN studente ELSE 0 END) AS superato,
                               voto,
                               AVG(voto) AS mediavoto
                          FROM iscrizioni,
                               appelli,
                               ad,
                               cds
                         WHERE iscrizioni.appcod = appelli.appcod AND
                               cds.cdscod = appelli.cdscod AND
                               appelli.adcod = ad.adcod
                         GROUP BY ad.ad,
                                  cds.cds
                        HAVING superato < totiscritti
                   )
                   A
             GROUP BY ad,
                      cds
       )
       B
 WHERE mediavoto IS NOT NULL
 GROUP BY ad,
          cds,
          tassosuperamento
 ORDER BY cds, mediavoto ASC;
```

QUERY IV - DENORMALIZZATO

L'unica differenza rispetto al modello normalizzato è stata l'impossibilità di effettuare join tra tabelle.

```
SELECT AD,
       Cds,
       mediavoto,
       ROUND( (superato / totiscritti) * 100, 3) AS tassosuperamento
  FROM (
           SELECT AD,
                  Cds,
                  Voto,
                  mediavoto,
                  CAST (superato AS DOUBLE) AS superato,
                  CAST (totiscritti AS DOUBLE) AS totiscritti
             FROM (
                      SELECT AD,
                             Cds,
                             count(DISTINCT Studente) AS totiscritti,
                             count(DISTINCT CASE Superamento WHEN 1 THEN Studente ELSE 0 END) AS superato,
                             Voto,
                             AVG(Voto) AS mediavoto
                        FROM bos_denormalizzato
                       GROUP BY AD,
                                Cds
                      HAVING superato < totiscritti
                  )
            GROUP BY AD,
                     Cds
       )
 WHERE mediavoto IS NOT NULL
 GROUP BY AD,
          Cds,
          tassosuperamento
 ORDER BY cds,
          mediavoto ASC;
```

QUERY V - NORMALIZZATO

Abbiamo estrapolato da ogni data appello il giorno, mese ed anno, in modo tale da ottenere il primo ed ultimo appello dato da ogni studente. Una volta trovato il primo ed ultimo appello, abbiamo calcolato la differenza in giorni. Successivamente, il tasso di Fast & Furious è stato calcolato come media voto di ogni studente sul numero giorni trascorsi tra il primo appello dato e l'ultimo. Abbiamo inserito la condizione 'WHERE differenzagiorni > 0' poiché la differenza, in alcuni casi, risultava 0 dato che alcuni studenti hanno sostenuto solamente un appello.
Visualizzazione:
https://public.tableau.com/profile/alessandro.motta3078#!/vizhome/FastFurious/Dashboard1?publish=yes

```
SELECT count(DISTINCT studente) AS numerostudenti,
       cds,
       ROUND(rapporto, 2) AS fastandfurious,
       ROUND(mediavoto, 4) AS mediavoto,
       differenzagiorni,
       esamisostenuti
  FROM (
           SELECT cds,
                  studente,
                  esamisostenuti,
                  mediavoto / (differenzagiorni) AS rapporto,
                  differenzagiorni,
                  mediavoto
             FROM (
                      SELECT CAST ( (JulianDay(maxdata) - JulianDay(mindata) ) AS INTEGER) AS differenzagiorni,
                             cds,
                             studente,
                             esamisostenuti,
                             mediavoto
                        FROM (
                                 SELECT max(data_concat) AS maxdata,
                                        min(data_concat) AS mindata,
                                        avg(voto) AS mediavoto,
                                        cds,
                                        studente,
                                        count( * ) AS esamisostenuti
                                   FROM (
                                            SELECT anno || '-' || mese_concat || '-' || giorno_concat AS data_concat,
                                                   studente,
                                                   cds,
                                                   voto
                                              FROM (
                                                       SELECT CASE length(mese) WHEN 1 THEN '0' || mese ELSE mese END AS mese_concat,
                                                              CASE length(giorno) WHEN 1 THEN '0' || giorno ELSE giorno END AS giorno_concat,
                                                          anno,
                                                          studente,
                                                          cds,
                                                          voto
                                                     FROM (
                                                              SELECT studente,
                                                                     cds,
                                                                     voto,
                                                                     dtappello,
                                                                     substr(rimanenzadata, instr(rimanenzadata, '/'), -2) AS mese,
                                                                     anno,
                                                                     giorno
                                                                FROM (
                                                                         SELECT studente,
                                                                                cds,
                                                                                voto,
                                                                                dtappello,
                                                                                substr(dtappello, instr(dtappello, '/') + 1) AS rimanenzadata,
                                                                                SUBSTR(dtappello, -4, 4) AS anno,
                                                                                giorno
                                                                           FROM (
                                                                                    SELECT studente,
                                                                                           cds,
                                                                                           voto,
                                                                                           dtappello,
                                                                                           SUBSTR(appelli.dtappello, -4, 4) AS anno,
                                                                                           SUBSTR(dtappello, instr(dtappello, '/'), -2) AS giorno
                                                                                      FROM iscrizioni,
                                                                                           appelli,
                                                                                           cds
                                                                                     WHERE appelli.appcod = iscrizioni.appcod AND
                                                                                           appelli.cdscod = cds.cdscod
                                                                                )
                                                                                A
                                                                     )
                                                                     B
                                                          )
                                                          C
                                                   )
                                                   D
                                            )
                                            E
                                      WHERE Voto IS NOT NULL
                                      GROUP BY studente,
                                               cds
                                     HAVING esamisostenuti > 1
                                 )
                                 F
                       WHERE differenzagiorni > 0
                       GROUP BY studente,
                                cds
                  )
                  G
       )
GROUP BY cds,
         fastandfurious,
         mediavoto,
         differenzagiorni
ORDER BY cds,
         fastandfurious DESC;
```

QUERY V - DENORMALIZZATO

Le differenze rispetto al modello normalizzato sono state:

- Non vi sono state join tra tabelle;
- L' attributo Dtappello è stato gestito in maniera differente a causa del diverso formato;

```
SELECT count(DISTINCT studente) AS numerostudenti,
       Cds,
       mediavoto,
       ROUND(mediavoto / differenzagiorni, 2) AS fastandfurious,
       differenzagiorni,
       esamisostenuti
  FROM (
           SELECT julianDay(maxdata) - julianDay(mindata) AS differenzagiorni,
                  Cds,
                  Studente,
                  esamisostenuti,
                  mediavoto
             FROM (
                      SELECT max(data_concat) AS maxdata,
                             min(data_concat) AS mindata,
                             avg(Voto) AS mediavoto,
                             Cds,
                             Studente,
                             count( * ) AS esamisostenuti
                        FROM (
                                 SELECT anno || '-' || mese || '-' || giorno AS data_concat,
                                        Studente,
                                        Cds,
                                        Voto
                                   FROM (
                                            SELECT giorno,
                                                   mese,
                                                   anno,
                                                   Studente,
                                                   Cds,
                                                   Voto
                                              FROM (
                                                       SELECT Studente,
                                                              Cds,
                                                              Voto,
                                                              Dtappello,
                                                              substr(rimanenzadata, instr(rimanenzadata, '/'), -2) AS mese,
                                                              anno,
                                                              giorno
                                                         FROM (
                                                                  SELECT Studente,
                                                                         Cds,
                                                                         Voto,
                                                                         Dtappello,
                                                                         substr(dtappello, instr(dtappello, '/') + 1) AS rimanenzadata,
                                                                         anno,
                                                                         giorno
                                                                    FROM (
                                                                             SELECT Studente,
                                                                                    Cds,
                                                                                    Voto,
                                                                                    Dtappello,
                                                                                    '20' || SUBSTR(Dtappello, -2, 2) AS anno,
                                                                                    SUBSTR(dtappello, instr(dtappello, '/'), -2) AS giorno
                                                                               FROM bos_denormalizzato
                                                                         )
                                                              )
                                                   )
                                            )
                                   )
                             )
                       WHERE Voto IS NOT NULL
                       GROUP BY Studente
                      HAVING esamisostenuti > 1
                  )
       )
GROUP BY Cds,
         fastandfurious,
         mediavoto,
         differenzagiorni
ORDER BY cds,
         fastandfurious DESC;
```

QUERY V1 - NORMALIZZATO

Abbiamo contato il numero di bocciature per ogni studente iscritto in ogni corso di studi ed esame. Il risultato è stato ottenuto mediante l'inserimento della condizione 'WHERE insufficienza = '1''. E' stata calcolata la media del numero di bocciature, affiancando a quest'ultima il numero totale di appelli per ogni esame.

Visualizzazione:
https://public.tableau.com/profile/alessandro.motta3078#!/vizhome/Top3TE/Dashboard1?publish=yes

```
SELECT A.ad,
       cds,
       ROUND(avg(A.numerodibocciature), 3) AS mediabocciature,
       numerototaliappelli
  FROM (
           SELECT iscrizioni.studente,
                  ad,
                  cds,
                  count(iscrizioni.iscrizione) AS numerodibocciature,
                  count(appelli.dtappello) AS numerototaliappelli
             FROM iscrizioni,
                  appelli,
                  cds,
                  ad
            WHERE iscrizioni.appcod = appelli.appcod AND
                  cds.cdscod = appelli.cdscod AND
                  ad.adcod = appelli.adcod AND
                  insufficienza = '1'
            GROUP BY studente,
                     ad,
                     cds
       )
       A
 GROUP BY A.ad,
          A.cds
 ORDER BY mediabocciature DESC;
```

QUERY V1 - DENORMALIZZATO

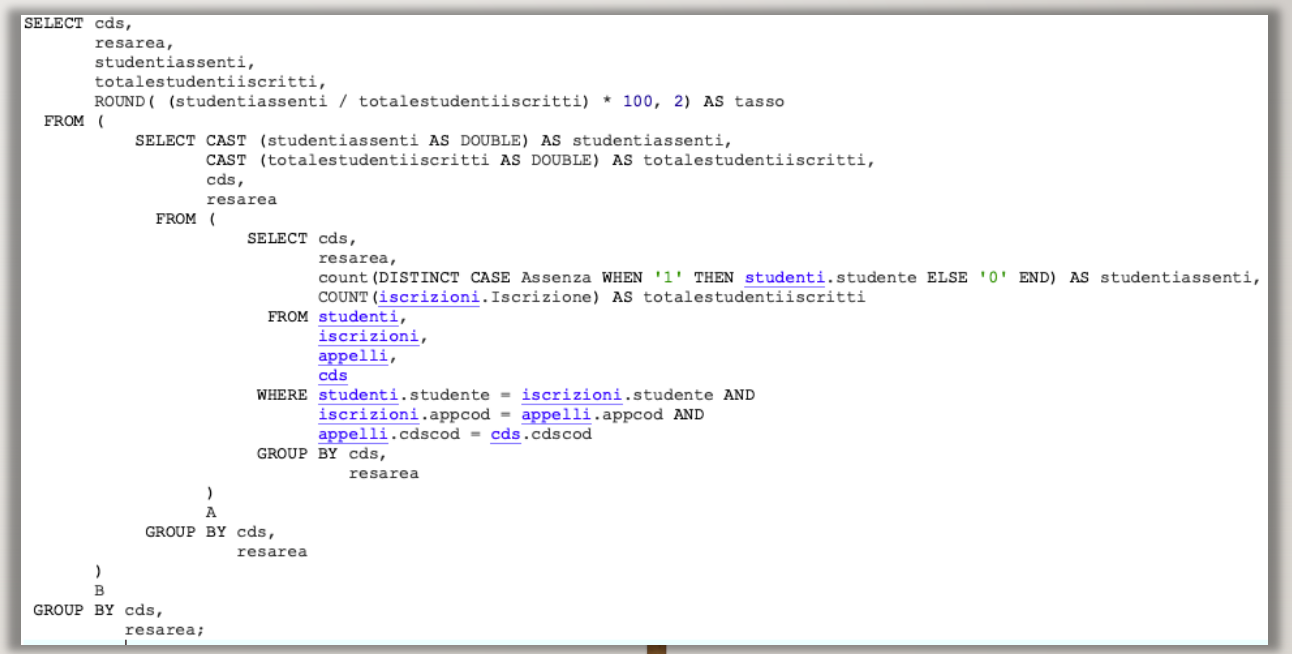
L'unica differenza rispetto al modello normalizzato è stata l'impossibilità di effettuare join tra tabelle.

```
SELECT A.AD,
       CdS,
       ROUND(avg(A.numerodibocciature), 3) AS mediabocciature,
       numerototaleappelli
  FROM (
           SELECT studente,
                  AD,
                  CdS,
                  count(iscrizione) AS numerodibocciature,
                  count(DtAppello) AS numerototaleappelli
             FROM bos_denormalizzato
            WHERE insufficienza = '1'
            GROUP BY studente,
                     AD,
                     CdS
       )
       A
 GROUP BY A.AD,
          A.CdS
 ORDER BY mediabocciature DESC;
```

QUERY V11 – NORMALIZZATO

E’ stato eseguito il conteggio del numero di studenti assenti e del totale studenti iscritti per ogni corso di studi divisi per zona geografica di provenienza. Abbiamo calcolato il tasso di assenza come rapporto tra il numero di studenti assenti ed il numero di studenti iscritti per corso di studi. E’ stata mantenuta la distinzione per zona geografica di provenienza poiché il nostro obiettivo era capire quali fossero i corsi di studio con il più alto tasso di assenza in base alla provenienza degli studenti.

Visualizzazione:
https://public.tableau.com/profile/alessandro.motta3078#!/vizhome/StudentiAssenti/Dashboard1



```
SELECT cds,
       resarea,
       studentiassenti,
       totalestudentiiscritti,
       ROUND( (studentiassenti / totalestudentiiscritti) * 100, 2) AS tasso
  FROM (
           SELECT CAST (studentiassenti AS DOUBLE) AS studentiassenti,
                  CAST (totalestudentiiscritti AS DOUBLE) AS totalestudentiiscritti,
                  cds,
                  resarea
             FROM (
                      SELECT cds,
                             resarea,
                             count(DISTINCT CASE Assenza WHEN '1' THEN studenti.studente ELSE '0' END) AS studentiassenti,
                             COUNT(iscrizioni.Iscrizione) AS totalestudentiiscritti
                        FROM studenti,
                             iscrizioni,
                             appelli,
                             cds
                       WHERE studenti.studente = iscrizioni.studente AND
                             iscrizioni.appcod = appelli.appcod AND
                             appelli.cdscod = cds.cdscod
                       GROUP BY cds,
                                resarea
                  )
                  A
           GROUP BY cds,
                    resarea
       )
       B
 GROUP BY cds,
          resarea;
```

QUERY V11 - DENORMALIZZATO

L'unica differenza rispetto al modello normalizzato è stata l'impossibilità di effettuare join tra tabelle.

```
SELECT Cds,
       StuResArea,
       studentiassenti,
       totalestudentiiscritti,
       ROUND( (studentiassenti / totalestudentiiscritti) * 100, 2) AS tasso
  FROM (
           SELECT CAST (studentiassenti AS DOUBLE) AS studentiassenti,
                  CAST (totalestudentiiscritti AS DOUBLE) AS totalestudentiiscritti,
                  Cds,
                  StuResArea
             FROM (
                      SELECT Cds,
                             StuResArea,
                             count(DISTINCT CASE Assenza WHEN '1' THEN Studente ELSE '0' END) AS studentiassenti,
                             COUNT(Iscrizione) AS totalestudentiiscritti
                        FROM bos_denormalizzato
                       GROUP BY Cds,
                                StuResArea
                  )
                  A
            GROUP BY Cds,
                     StuResArea
       )
       B
 GROUP BY Cds,
          StuResArea;
```

TOOL UTILIZZATI:

SQLiteStudio
TableauDesktop
TableauPublic

FUNZIONI SQLITE UTILIZZATE PER IL PROGETTO:

- SELECT;
- FROM;
- WHERE;
- GROUP BY;
- HAVING;
- LIMIT;
- ORDER BY;
- SUBSTR();
- ROUND();
- INSTR();
- AVG();
- SUM();
- COUNT();
- CAST();
- CASE();
- LENGTH()