All SQL queries and results are listed under the answers to question 1.F.

Insights :

1. Age of students distribution :
    - There are 50% of students (2 students) are 14 years old of the total students : 4
    - There are 25% of students (1 student) are 15 years old of the total students : 4
    - There are 25% of students (1 student) are 16 years old of the total students : 4

    14 year old students being the most numerous.

2. Summary statistical of the age of students :

	Count	Mean	Standard Deviation	Min	Quartile 1 (Q1)	Quartile 2 (Q2)	Quartile 3 (Q3)	Max
Age	4.00	14.75	0.96	14.00	14.00	14.50	15.25	16.00

    Total students are 4.
    The minimum age is 14 years.
    The average age of students is 14.75
    The maximum age is 16 years, that is the oldest student with the name marquee.
    The age standard deviation is 0.96

3. Distribution of students :
    - There are 75% of students are Middle School, that is as much as 3 students.
    - There are 25% of students are High School, that is as much as 1 student.

    We can see that the majority of students are Middle School students.

4. Distribution of classroom :
    - There are 50% of students are in classroom 123, that is as much as 2 students.
    - There are 25% of students are in classroom 234, that is as much as 1 student.
    - There are 25% of students are have no classroom, that is as much as 1 student.

    The classroom with the most students is classroom 123.

5. There is no data of students who are under 13 years old, so there are only 2 groups of students based on age, namely Middle School and High School. There is no group for Elementary School. Grouping students based on age will be wrong if there is data of students who are under 13 years.

6. And there will be some exceptions for grouping students based on age as follows :
    1) There are several cases of students entering school earlier than the supposed minimum age or students who experience accelerated classes, so that the minimum age limit for the student category cannot be applied.
       Example:
         - Students who enter school earlier than the supposed minimum age, so that at the age of 14 or 15 years they have entered High School.
           If categorized by age, those students will be categorized into Middle School.

- Students who experience accelerated classes during Middle School, so that they graduate at the age of 14 and then enter the High School category.
  If categorized by age, the students will still be enrolled in Middle School.

2) Likewise for the opposite case, students who are late entering school than the supposed minimum age or ever failing a grade, so that the maximum age limit for the student category also cannot be applied.
  Example : Students who ever failing a grade in Middle School, so that they just graduated at the age of 16.
  If categorized by age, those students is categorized as High School.

7. In one classroom there can be students of different ages and in a classroom they're all likely to be more or less the same age.

Questions :

1. What are the factors that cause students not to have classrooms?
   Because every student must have a classroom.
   Is it possible that the student is on school leave or something like that?

2. What if there is data of students who are under 13 years old, this data will be included in the Middle School group?

3. What about the student group based on age if there is an exception case for students who experience accelerated classes or enter school earlier than the supposed minimum age?
   Likewise for the opposite case, students who are late entering to school or ever failing a grade, so that the maximum age limit for the student category cannot be applied.

Recommendations :

- The student group should be inputted by the user with the default state based on the age group (minimum & maximum supposed age limit), not determined through coding based on age.

- And a warning/alert message appears if the minimum or maximum age limit for a student is not within the supposed age limit for the student group, and we can add an input field for the description of the exception case.

SQL Queries and Results :

Link File SQL :
https://felicebenita.github.io/vidio/Felice-Online_Test_Data_Analyst-Answer_Queries.sql
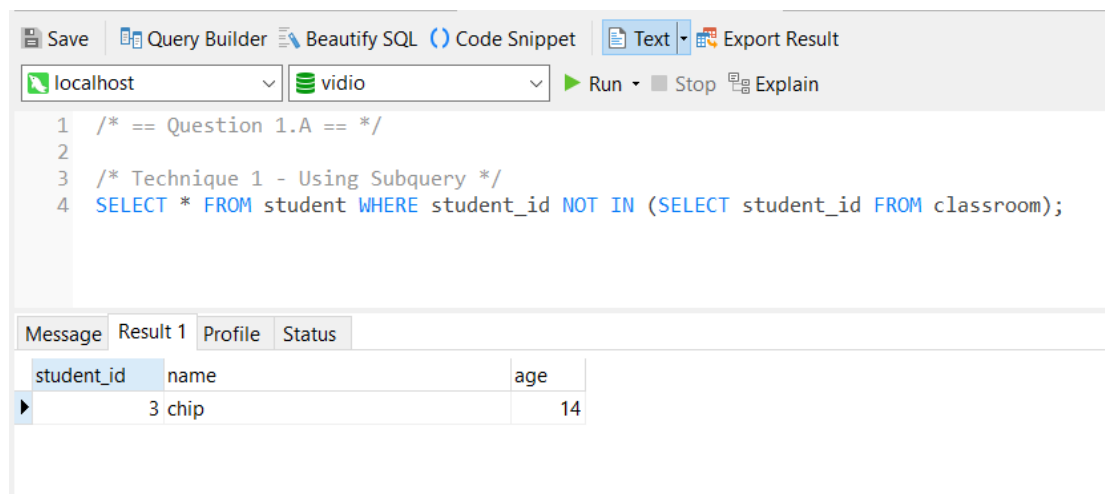
Tools used : MySQL.

1.A. Find out student that has no classroom :

/* Technique 1 - Using Subquery */

```
SELECT * FROM student WHERE student_id NOT IN (SELECT student_id FROM classroom);
```

Result :

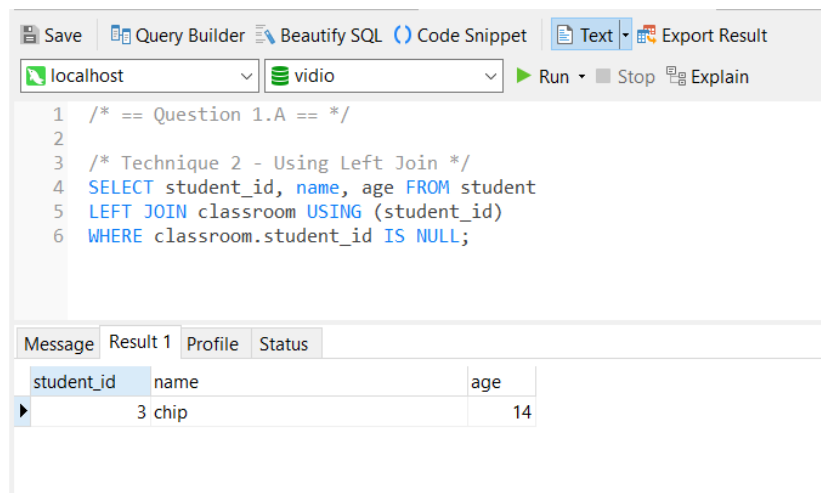/* Technique 2 - Using Left Join */

```
SELECT student_id, name, age FROM student
LEFT JOIN classroom USING (student_id)
WHERE classroom.student_id IS NULL;
```
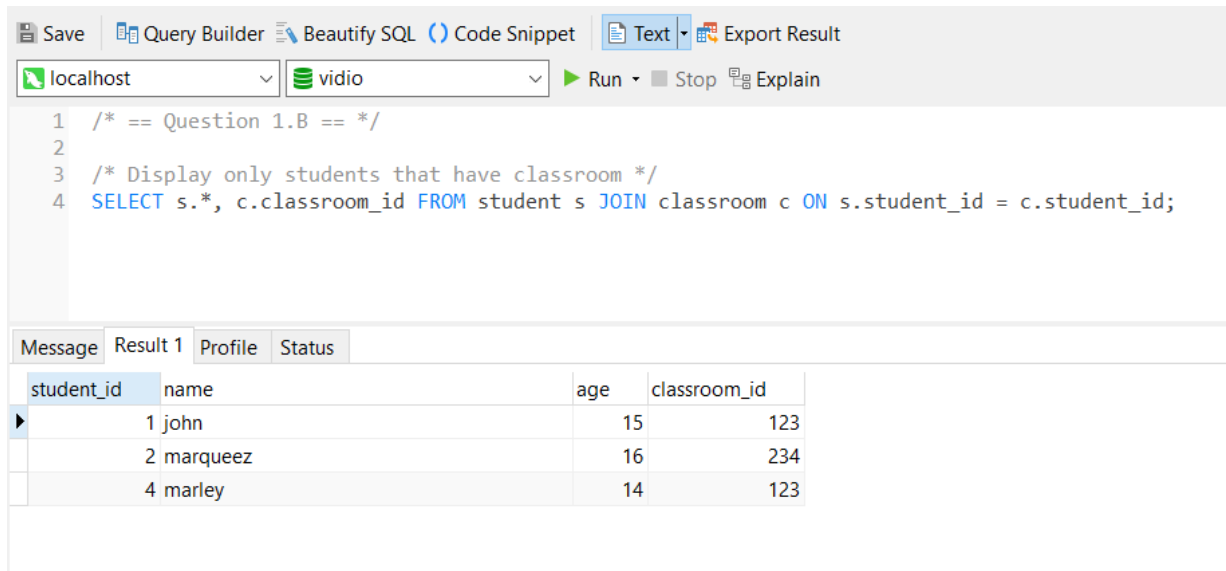
Result :

1.B. Display classroom_id that each student has :

/* Display only students that have classroom */

```
SELECT s.*, c.classroom_id FROM student s JOIN classroom c ON s.student_id = c.student_id;
```
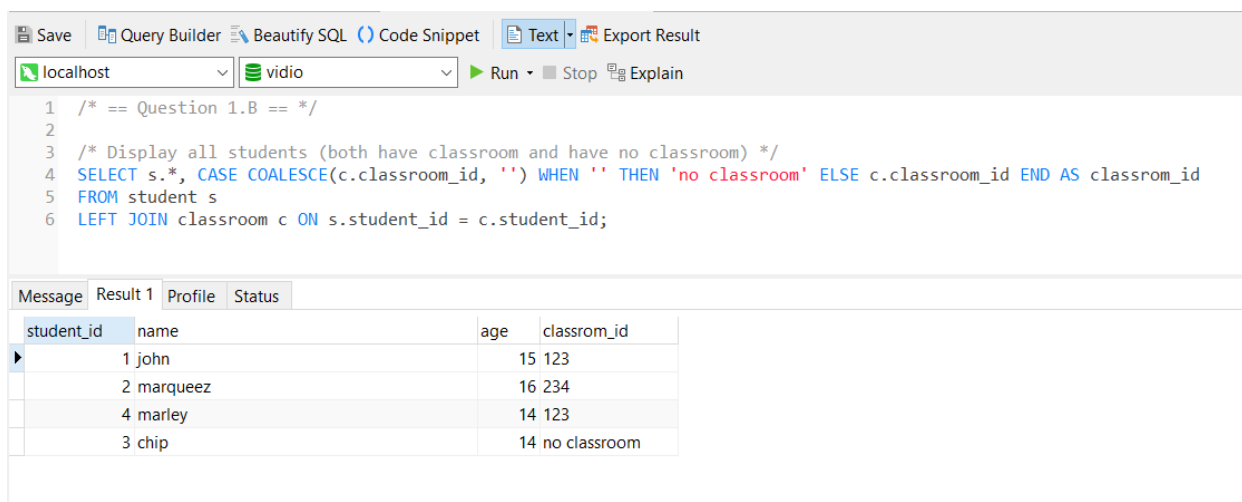
Result :

/* Display all students (both have classroom and have no classroom) */

```
SELECT s.*, CASE COALESCE(c.classroom_id, '') WHEN '' THEN 'no classroom' ELSE
c.classroom_id END AS classrom_id
FROM student s
LEFT JOIN classroom c ON s.student_id = c.student_id;
```

Result :

1.C. Create a group for students that have age > 15 is called "high school" and age < 16 called "middle school", and count how many students that belong to that group :

```
SELECT student_group, COUNT(student_id) 'total_student' FROM
(
    SELECT *, CASE WHEN age > 15 THEN 'high school' WHEN age < 16 THEN 'middle school'
    ELSE 'unspecified' END AS student_group
    FROM student
) AS s
GROUP BY student_group;
```

Result :

1.D Create SQL that give output that marqueez is the oldest student :

```
SELECT * FROM student WHERE age = (SELECT MAX(age) FROM student);
```

Result :

/* This method can be used if the student_id is auto increment, so the rows below always have the larger id (and are the later data), so comparison can be made based on student_id to calculate the cumulative_age. */

```
SELECT *, (SELECT COALESCE(SUM(age),0) FROM student s1 WHERE s1.student_id <=
s.student_id) 'cumulative_age'
FROM student s;
```

Result :

/* If student_id is not auto-increment, it can be done by sorting the data based on the data order determinant column, for example: entry_date, and doing a comparison based on entry_date to calculate the cumulative_age. */

Example :

```
SELECT *,
    (SELECT COALESCE(SUM(age),0) FROM student_copy s1 WHERE s1.entry_date <=
    s.entry_date ORDER BY s1.entry_date) 'cumulative_age'
FROM student_copy s
ORDER BY s.entry_date;
```

Result :

```
1   /* == Question 1.E == */
2
3   /* If student_id is not auto-increment, it can be done by sorting the data based on the data order determinant column, for example:
    entry_date, and doing a comparison based on entry_date to calculate the cumulative_age. */
4   /* Example : */
5   SELECT *,
6     (SELECT COALESCE(SUM(age),0) FROM student_copy s1 WHERE s1.entry_date <= s.entry_date ORDER BY s1.entry_date) 'cumulative_age'
7   FROM student_copy s
8   ORDER BY s.entry_date;
```

Message | Result 1 | Profile | Status

student_id	name	age	entry_date	cumulative_age
3	john	15	2022-04-20 13:46:57	15
2	marqueez	16	2022-04-21 13:46:57	31
4	marley	14	2022-04-23 13:46:57	45
1	chip	14	2022-04-24 13:46:57	59