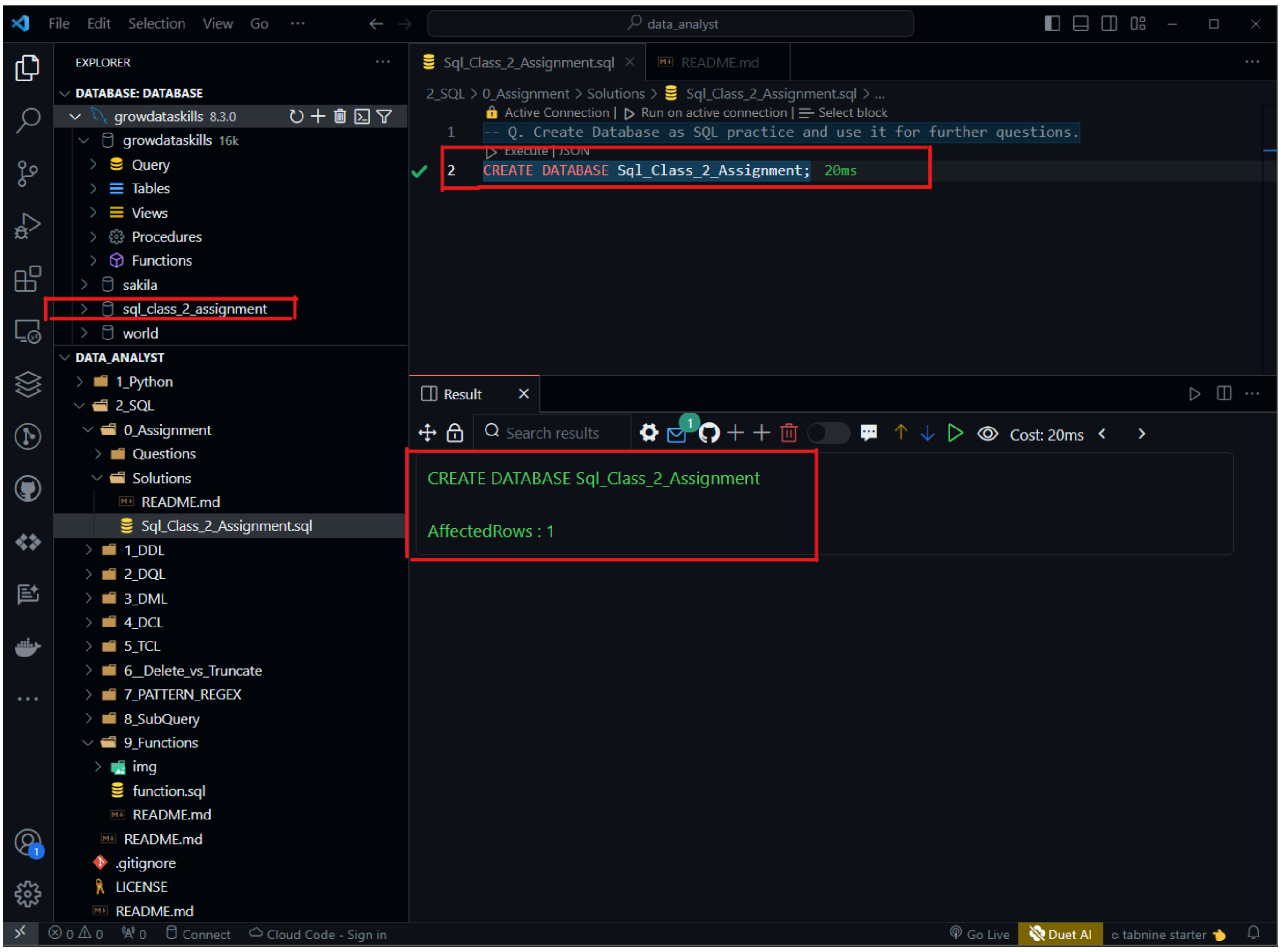
Q1. Create Database as SQL practice and use it for further questions.

- Creating a database name `Sql_Class_2_Assignment` and will use it for further questions.

Query:

```sql
CREATE DATABASE Sql_Class_2_Assignment;
```

Execution:

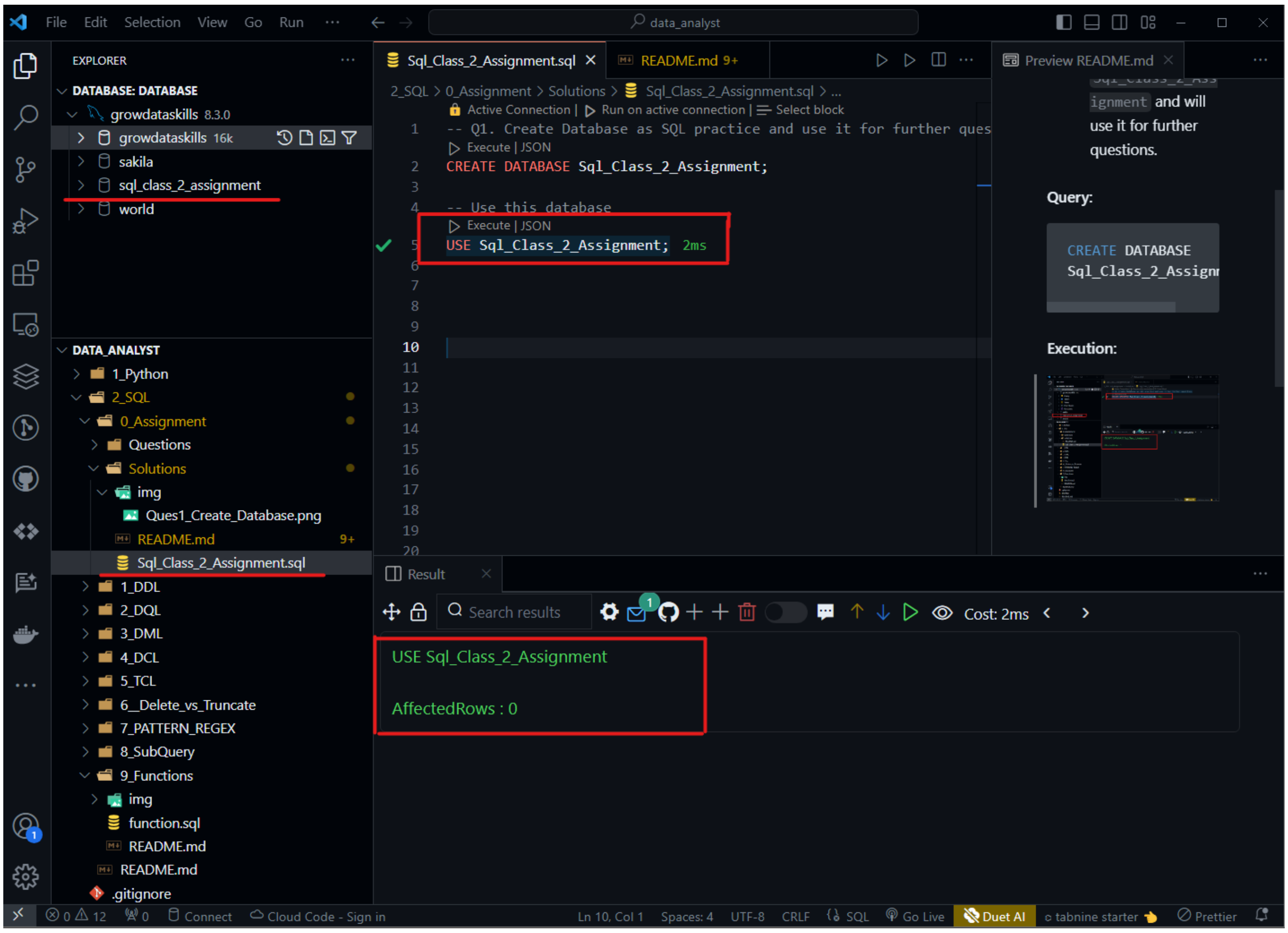


```
-- Q. Create Database as SQL practice and use it for further questions.
CREATE DATABASE Sql_Class_2_Assignment;  20ms
```

```
CREATE DATABASE Sql_Class_2_Assignment

AffectedRows : 1
```

```sql
USE Sql_Class_2_Assignment;
```



```
-- Q1. Create Database as SQL practice and use it for further ques
CREATE DATABASE Sql_Class_2_Assignment;

-- Use this database
USE Sql_Class_2_Assignment;  2ms
```

```
USE Sql_Class_2_Assignment

AffectedRows : 0
```

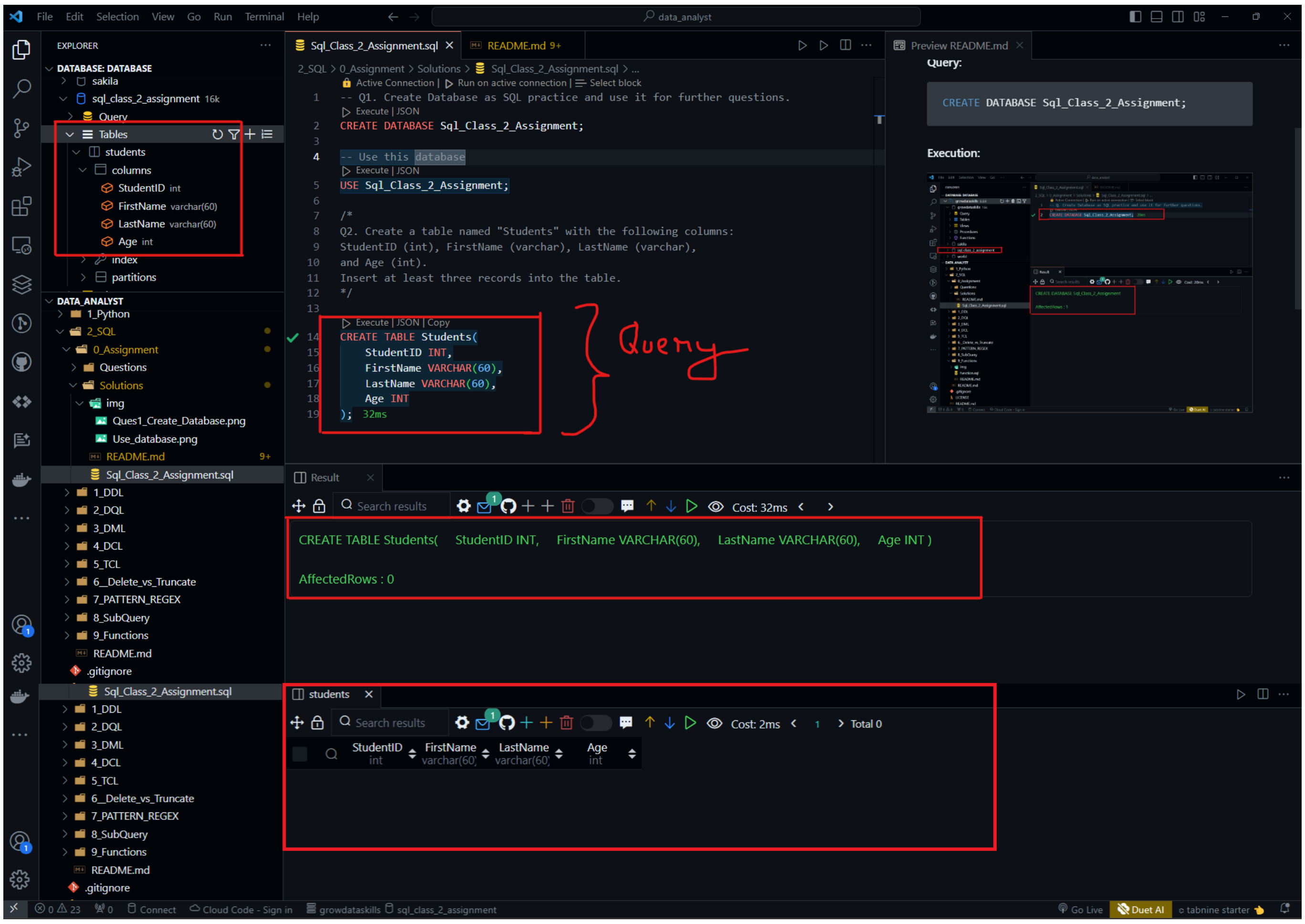
Q2. Create a table named "Students" with the following columns: StudentID (int), FirstName (varchar), LastName (varchar), and Age (int). Insert at least three records into the table.

- Creating a table named `Students` with the following columns: StudentID (int), FirstName (varchar), LastName (varchar), and Age (int)

Query:

```
CREATE TABLE Students(
    StudentID INT,
    FirstName VARCHAR(60),
    LastName VARCHAR(60),
    Age INT
);
```

Execution:



```
-- Q1. Create Database as SQL practice and use it for further questions.
CREATE DATABASE Sql_Class_2_Assignment;

-- Use this database
USE Sql_Class_2_Assignment;

/*
Q2. Create a table named "Students" with the following columns:
StudentID (int), FirstName (varchar), LastName (varchar),
and Age (int).
Insert at least three records into the table.
*/

CREATE TABLE Students(
    StudentID INT,
    FirstName VARCHAR(60),
    LastName VARCHAR(60),
    Age INT
); 32ms
```

```
CREATE TABLE Students(    StudentID INT,    FirstName VARCHAR(60),    LastName VARCHAR(60),    Age INT )

AffectedRows : 0
```

students — Cost: 2ms — Total 0

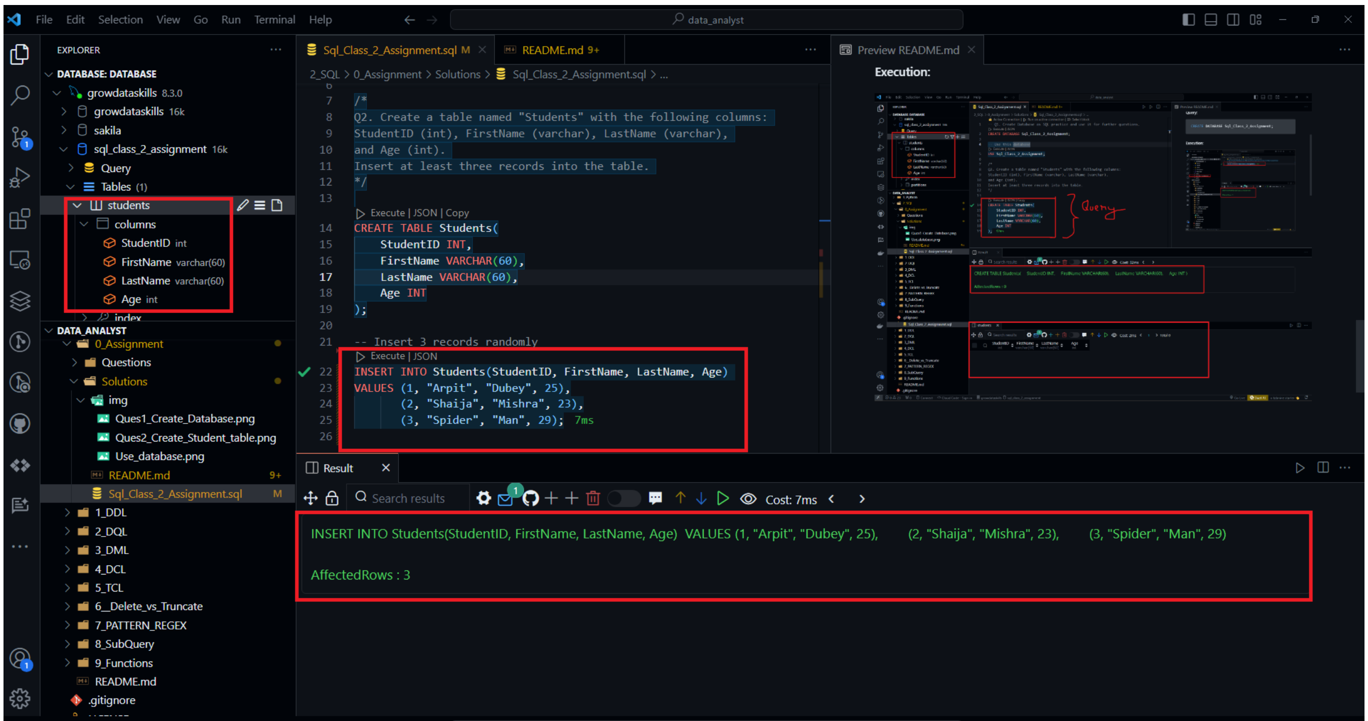
StudentID int	FirstName varchar(60)	LastName varchar(60)	Age int

- Inserting 3 random records into the table `Students`

Query:

```sql
INSERT INTO Students(StudentID, FirstName, LastName, Age)
VALUES (1, "Arpit", "Dubey", 25),
       (2, "Shaija", "Mishra", 23),
       (3, "Spider", "Man", 29);
```

Execution:



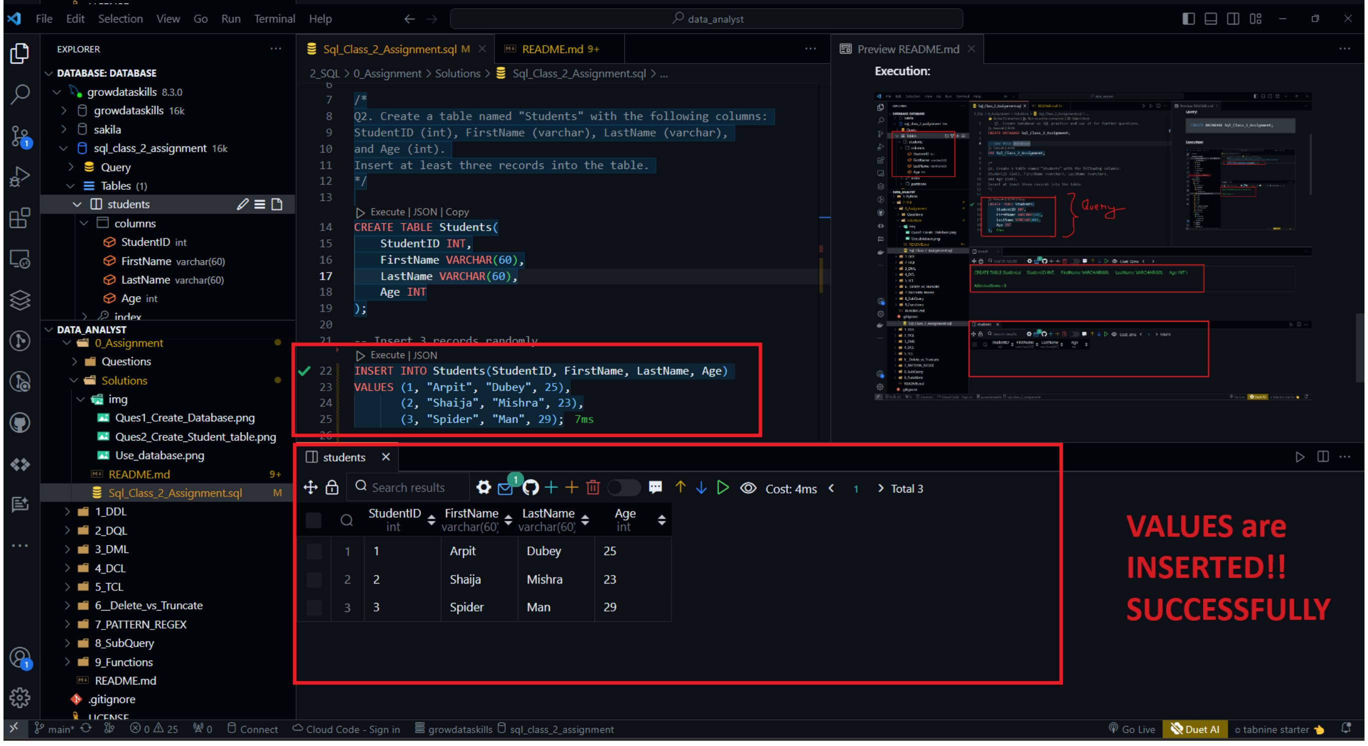
```
/*
Q2. Create a table named "Students" with the following columns:
StudentID (int), FirstName (varchar), LastName (varchar),
and Age (int).
Insert at least three records into the table.
*/

CREATE TABLE Students(
    StudentID INT,
    FirstName VARCHAR(60),
    LastName VARCHAR(60),
    Age INT
);

-- Insert 3 records randomly
INSERT INTO Students(StudentID, FirstName, LastName, Age)
VALUES (1, "Arpit", "Dubey", 25),
       (2, "Shaija", "Mishra", 23),
       (3, "Spider", "Man", 29);
```

```
INSERT INTO Students(StudentID, FirstName, LastName, Age)  VALUES (1, "Arpit", "Dubey", 25),      (2, "Shaija", "Mishra", 23),      (3, "Spider", "Man", 29)

AffectedRows : 3
```



```
/*
Q2. Create a table named "Students" with the following columns:
StudentID (int), FirstName (varchar), LastName (varchar),
and Age (int).
Insert at least three records into the table.
*/

CREATE TABLE Students(
    StudentID INT,
    FirstName VARCHAR(60),
    LastName VARCHAR(60),
    Age INT
);

-- Insert 3 records randomly
INSERT INTO Students(StudentID, FirstName, LastName, Age)
VALUES (1, "Arpit", "Dubey", 25),
       (2, "Shaija", "Mishra", 23),
       (3, "Spider", "Man", 29);
```

	StudentID int	FirstName varchar(60)	LastName varchar(60)	Age int
1	1	Arpit	Dubey	25
2	2	Shaija	Mishra	23
3	3	Spider	Man	29

VALUES are INSERTED!! SUCCESSFULLY

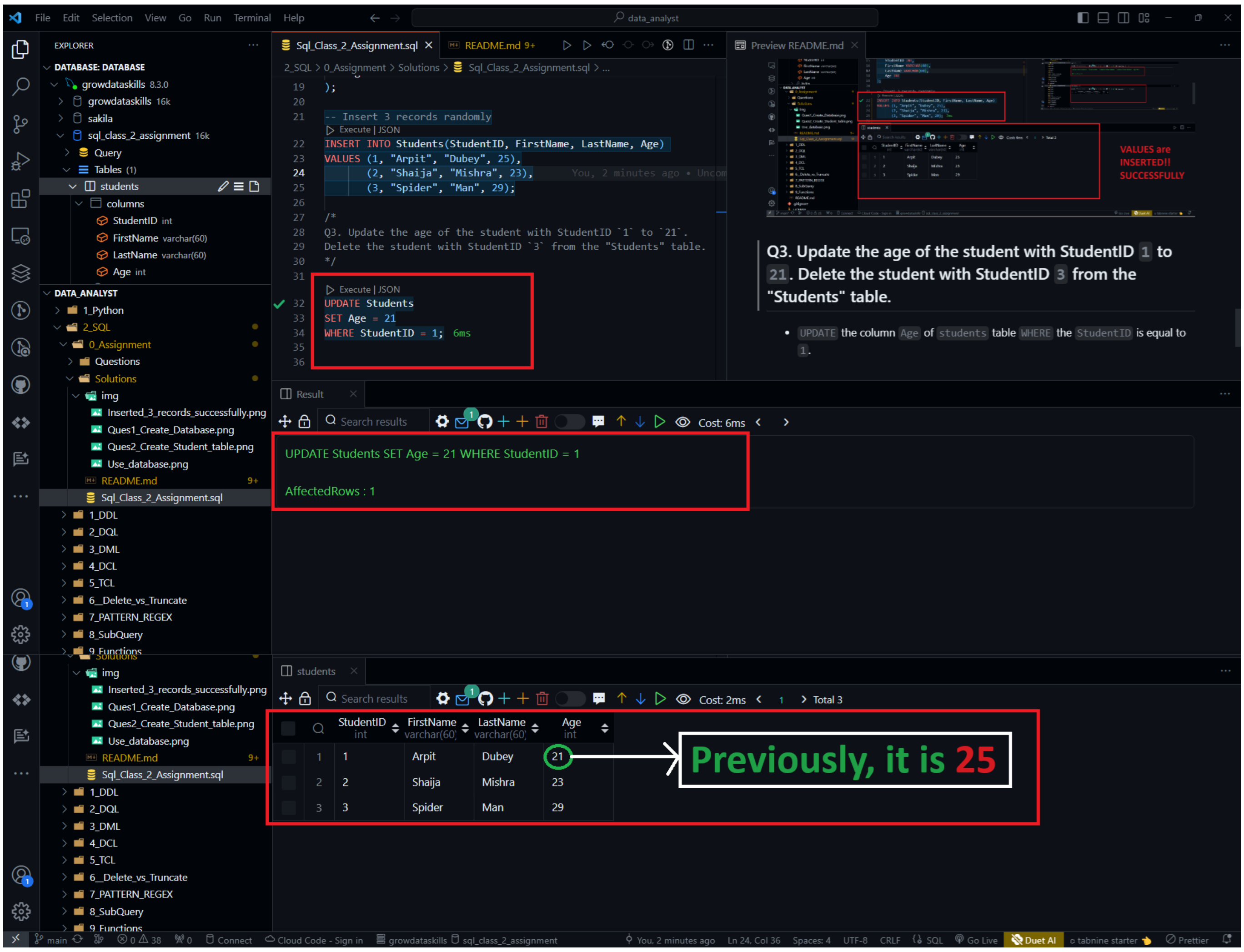
Q3. Update the age of the student with StudentID 1 to 21. Delete the student with StudentID 3 from the "Students" table.

- UPDATE the column Age of students table WHERE the StudentID is equal to 1.

Query:

```
UPDATE Students
SET Age = 21
WHERE StudentID = 1;
```

Execution:



```
);

-- Insert 3 records randomly
INSERT INTO Students(StudentID, FirstName, LastName, Age)
VALUES (1, "Arpit", "Dubey", 25),
       (2, "Shaija", "Mishra", 23),
       (3, "Spider", "Man", 29);

/*
Q3. Update the age of the student with StudentID `1` to `21`.
Delete the student with StudentID `3` from the "Students" table.
*/

UPDATE Students
SET Age = 21
WHERE StudentID = 1;  6ms
```

```
UPDATE Students SET Age = 21 WHERE StudentID = 1

AffectedRows : 1
```

Q3. Update the age of the student with StudentID `1` to `21`. Delete the student with StudentID `3` from the "Students" table.

- `UPDATE` the column `Age` of `students` table `WHERE` the `StudentID` is equal to `1`.

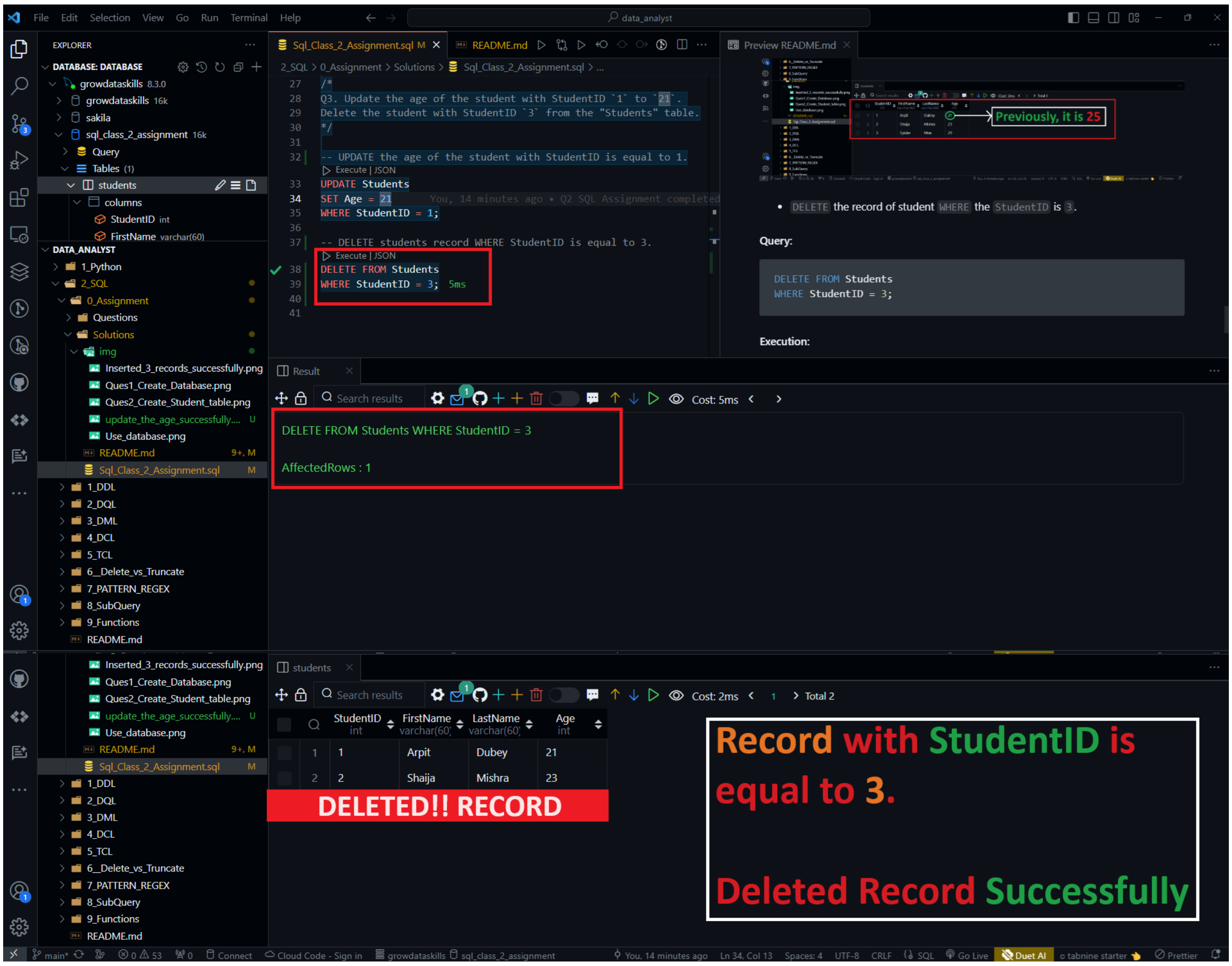
	StudentID int	FirstName varchar(60)	LastName varchar(60)	Age int
1	1	Arpit	Dubey	21
2	2	Shaija	Mishra	23
3	3	Spider	Man	29

- `DELETE` the record of student `WHERE` the `StudentID` is `3`.

Query:

```
DELETE FROM Students
WHERE StudentID = 3;
```

Execution:



```sql
/*
Q3. Update the age of the student with StudentID `1` to `21`.
Delete the student with StudentID `3` from the "Students" table.
*/

-- UPDATE the age of the student with StudentID is equal to 1.
UPDATE Students
SET Age = 21
WHERE StudentID = 1;

-- DELETE students record WHERE StudentID is equal to 3.
DELETE FROM Students
WHERE StudentID = 3;  5ms
```

```
DELETE FROM Students WHERE StudentID = 3

AffectedRows : 1
```

	StudentID int	FirstName varchar(60)	LastName varchar(60)	Age int
1	1	Arpit	Dubey	21
2	2	Shaija	Mishra	23

DELETED!! RECORD
Record with StudentID is equal to 3.
Deleted Record Successfully
Previously, it is 25

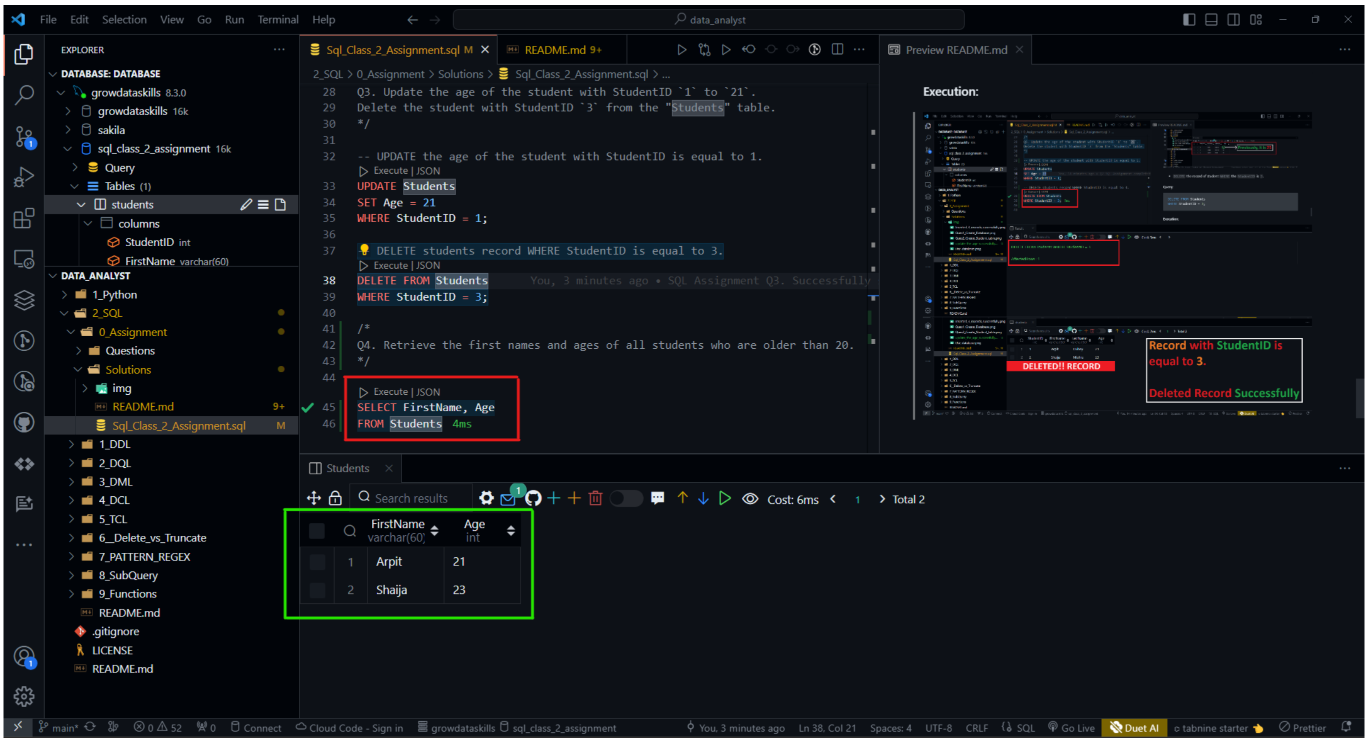
Q4. Retrieve the first names and ages of all students who are older than 20.

- Retrieving the `FirstName` and `Age` of all students without any condition.

Query:

```sql
SELECT FirstName, Age
FROM Students
```

Execution:



```
Q3. Update the age of the student with StudentID `1` to `21`.
Delete the student with StudentID `3` from the "Students" table.
*/

-- UPDATE the age of the student with StudentID is equal to 1.
UPDATE Students
SET Age = 21
WHERE StudentID = 1;

-- DELETE students record WHERE StudentID is equal to 3.
DELETE FROM Students
WHERE StudentID = 3;

/*
Q4. Retrieve the first names and ages of all students who are older than 20.
*/

SELECT FirstName, Age
FROM Students
```

	FirstName varchar(60)	Age int
1	Arpit	21
2	Shaija	23

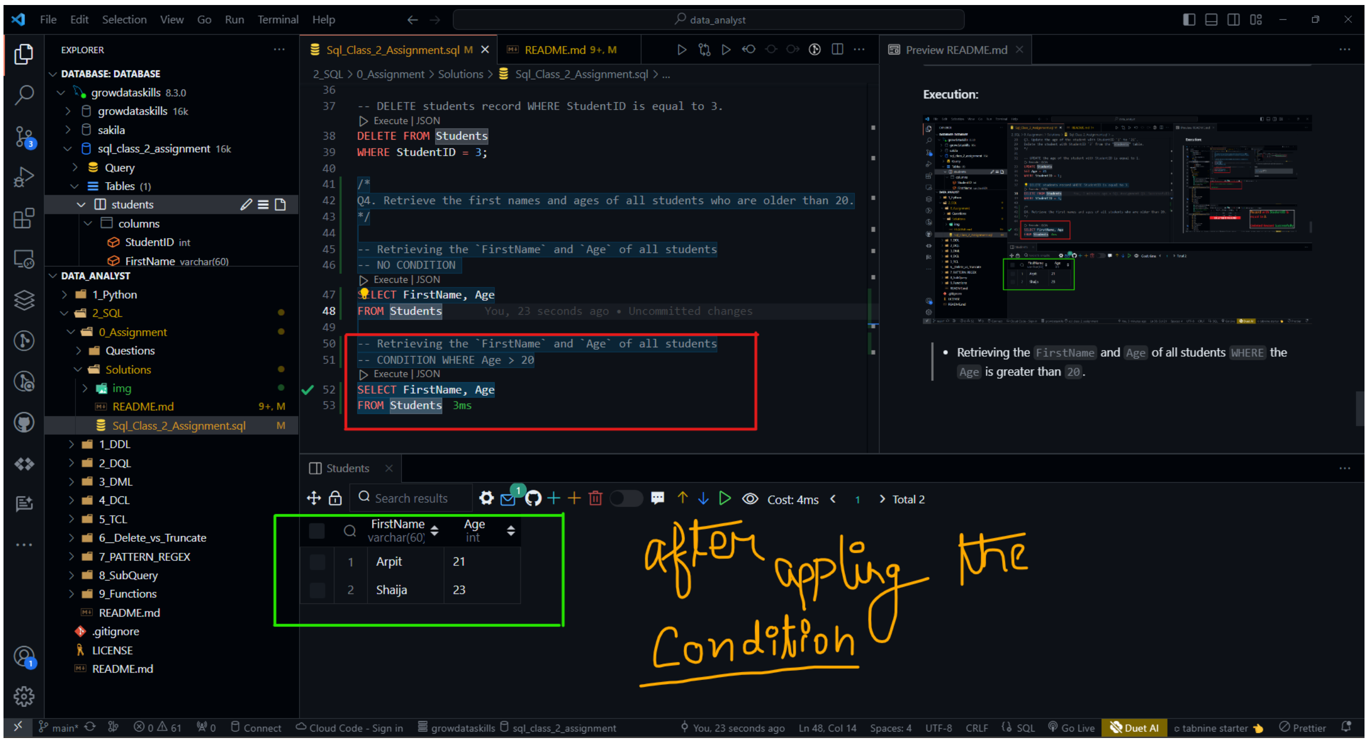
Execution:
Record with StudentID is equal to 3.
DELETED!! RECORD
Deleted Record Successfully

- Retrieving the `FirstName` and `Age` of all students `WHERE` the `Age` is greater than `20`..

Query:

```sql
SELECT FirstName, Age
FROM Students
WHERE Age > 20;
```

Execution:



```
-- DELETE students record WHERE StudentID is equal to 3.
DELETE FROM Students
WHERE StudentID = 3;

/*
Q4. Retrieve the first names and ages of all students who are older than 20.
*/

-- Retrieving the `FirstName` and `Age` of all students
-- NO CONDITION
SELECT FirstName, Age
FROM Students

-- Retrieving the `FirstName` and `Age` of all students
-- CONDITION WHERE Age > 20
SELECT FirstName, Age
FROM Students
```

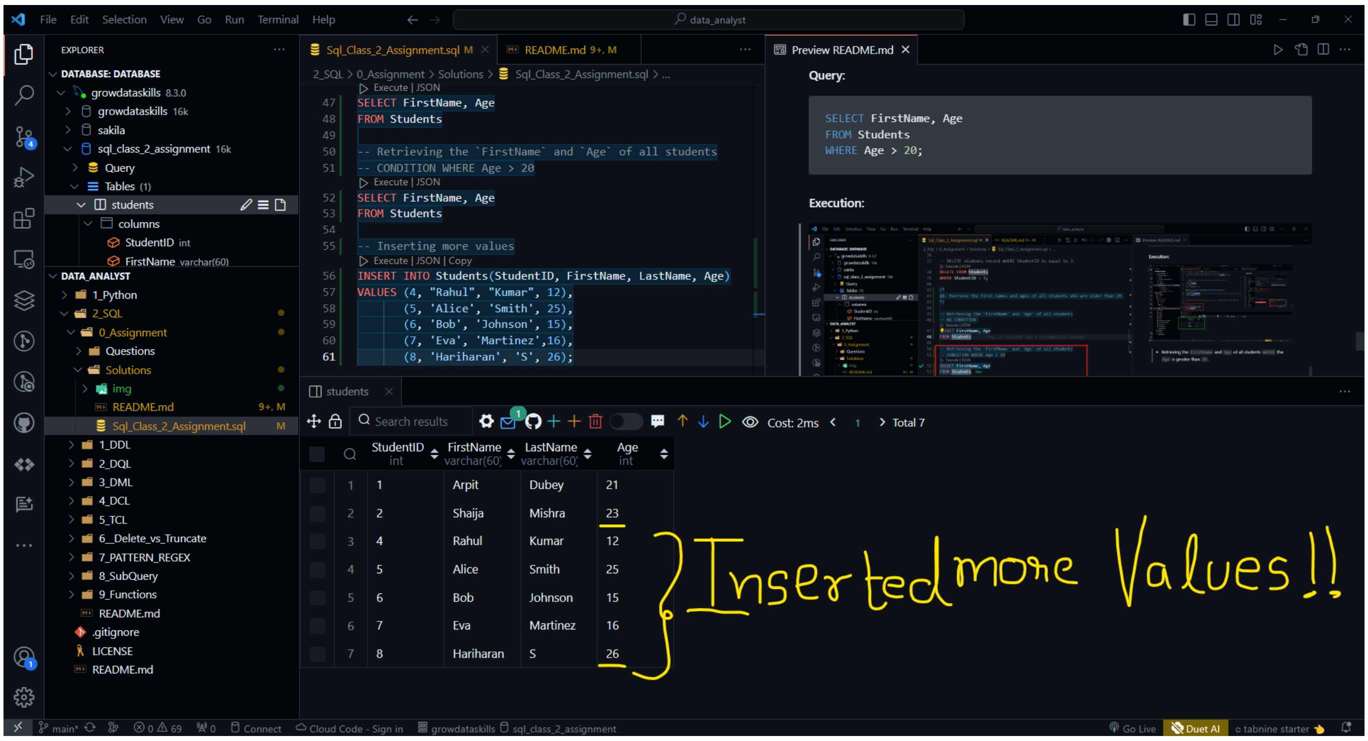
	FirstName varchar(60)	Age int
1	Arpit	21
2	Shaija	23

NOTE : To Perform more operation we required more values so I just inserted more values to the table.

Query:

```
INSERT INTO Students(StudentID, FirstName, LastName, Age)
VALUES (4, "Rahul", "Kumar", 12),
       (5, 'Alice', 'Smith', 25),
       (6, 'Bob', 'Johnson', 15),
       (7, 'Eva', 'Martinez',16),
       (8, 'Hariharan', 'S', 26);
```

Execution:



```
SELECT FirstName, Age
FROM Students

-- Retrieving the `FirstName` and `Age` of all students
-- CONDITION WHERE Age > 20
SELECT FirstName, Age
FROM Students

-- Inserting more values
INSERT INTO Students(StudentID, FirstName, LastName, Age)
VALUES (4, "Rahul", "Kumar", 12),
       (5, 'Alice', 'Smith', 25),
       (6, 'Bob', 'Johnson', 15),
       (7, 'Eva', 'Martinez',16),
       (8, 'Hariharan', 'S', 26);
```

	StudentID int	FirstName varchar(60)	LastName varchar(60)	Age int
1	1	Arpit	Dubey	21
2	2	Shaija	Mishra	23
3	4	Rahul	Kumar	12
4	5	Alice	Smith	25
5	6	Bob	Johnson	15
6	7	Eva	Martinez	16
7	8	Hariharan	S	26

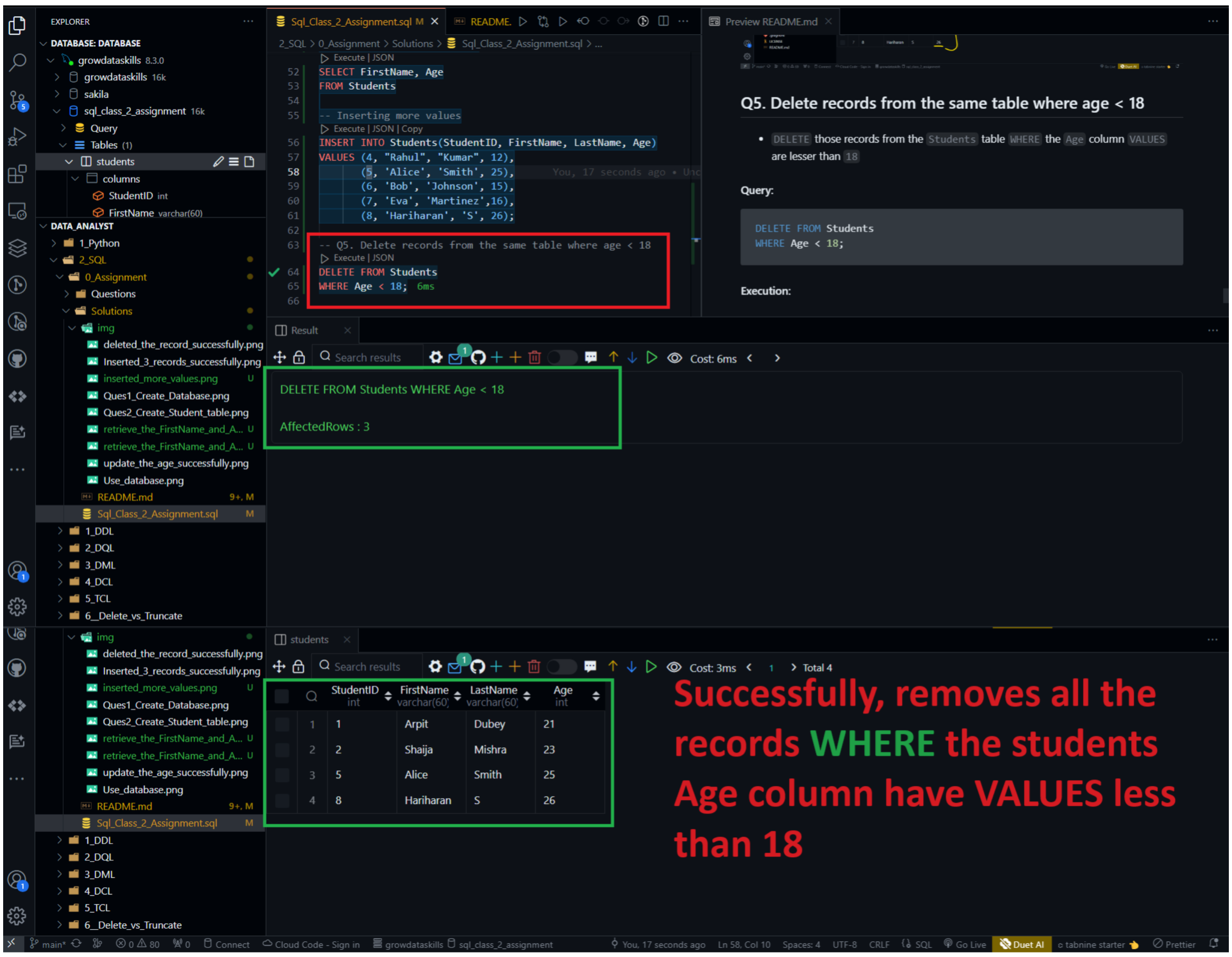
Q5. Delete records from the same table where age < 18

- `DELETE` those records from the `Students` table `WHERE` the `Age` column `VALUES` are lesser than `18`

Query:

```
DELETE FROM Students
WHERE Age < 18;
```

Execution:



```sql
SELECT FirstName, Age
FROM Students

-- Inserting more values
INSERT INTO Students(StudentID, FirstName, LastName, Age)
VALUES (4, "Rahul", "Kumar", 12),
       (5, 'Alice', 'Smith', 25),
       (6, 'Bob', 'Johnson', 15),
       (7, 'Eva', 'Martinez',16),
       (8, 'Hariharan', 'S', 26);

-- Q5. Delete records from the same table where age < 18
DELETE FROM Students
WHERE Age < 18;  6ms
```

Q5. Delete records from the same table where age < 18

- DELETE those records from the Students table WHERE the Age column VALUES are lesser than 18

Query:

```sql
DELETE FROM Students
WHERE Age < 18;
```

Execution:

```
DELETE FROM Students WHERE Age < 18

AffectedRows : 3
```

	StudentID int	FirstName varchar(60)	LastName varchar(60)	Age int
1	1	Arpit	Dubey	21
2	2	Shaija	Mishra	23
3	5	Alice	Smith	25
4	8	Hariharan	S	26

Successfully, removes all the records WHERE the students Age column have VALUES less than 18

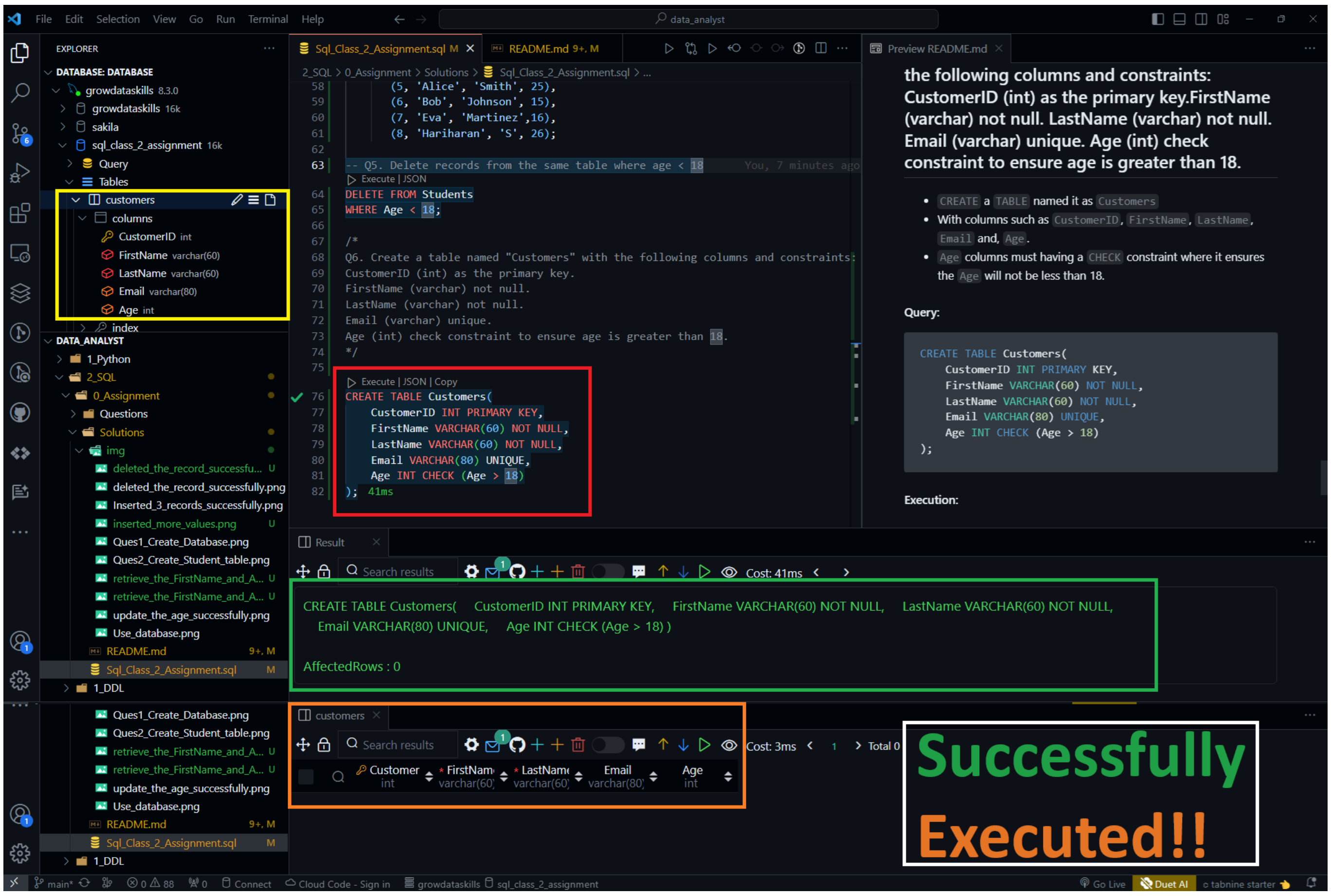
Q6. Create a table named "Customers" with the following columns and constraints: CustomerID (int) as the primary key.FirstName (varchar) not null. LastName (varchar) not null. Email (varchar) unique. Age (int) check constraint to ensure age is greater than 18.

- `CREATE` a `TABLE` named it as `Customers`
- With columns such as `CustomerID`, `FirstName`, `LastName`, `Email` and, `Age`.
- `Age` columns must having a `CHECK` constraint where it ensures the `Age` will not be less than 18.

Query:

```sql
CREATE TABLE Customers(
    CustomerID INT PRIMARY KEY,
    FirstName VARCHAR(60) NOT NULL,
    LastName VARCHAR(60) NOT NULL,
    Email VARCHAR(80) UNIQUE,
    Age INT CHECK (Age > 18)
);
```

Execution:



```
    (5, 'Alice', 'Smith', 25),
    (6, 'Bob', 'Johnson', 15),
    (7, 'Eva', 'Martinez',16),
    (8, 'Hariharan', 'S', 26);

-- Q5. Delete records from the same table where age < 18
DELETE FROM Students
WHERE Age < 18;

/*
Q6. Create a table named "Customers" with the following columns and constraints:
CustomerID (int) as the primary key.
FirstName (varchar) not null.
LastName (varchar) not null.
Email (varchar) unique.
Age (int) check constraint to ensure age is greater than 18.
*/

CREATE TABLE Customers(
    CustomerID INT PRIMARY KEY,
    FirstName VARCHAR(60) NOT NULL,
    LastName VARCHAR(60) NOT NULL,
    Email VARCHAR(80) UNIQUE,
    Age INT CHECK (Age > 18)
); 41ms
```

the following columns and constraints: CustomerID (int) as the primary key.FirstName (varchar) not null. LastName (varchar) not null. Email (varchar) unique. Age (int) check constraint to ensure age is greater than 18.

- `CREATE` a `TABLE` named it as `Customers`
- With columns such as `CustomerID`, `FirstName`, `LastName`, `Email` and, `Age`.
- `Age` columns must having a `CHECK` constraint where it ensures the `Age` will not be less than 18.

Query:

```
CREATE TABLE Customers(
    CustomerID INT PRIMARY KEY,
    FirstName VARCHAR(60) NOT NULL,
    LastName VARCHAR(60) NOT NULL,
    Email VARCHAR(80) UNIQUE,
    Age INT CHECK (Age > 18)
);
```

Execution:

```
CREATE TABLE Customers(    CustomerID INT PRIMARY KEY,    FirstName VARCHAR(60) NOT NULL,    LastName VARCHAR(60) NOT NULL,
    Email VARCHAR(80) UNIQUE,    Age INT CHECK (Age > 18) )

AffectedRows : 0
```

Customer int	FirstNam varchar(60)	LastName varchar(60)	Email varchar(80)	Age int

Cost: 3ms Total 0

```
DATABASE: DATABASE
  growdataskills 8.3.0
    growdataskills 16k
    sakila
    sql_class_2_assignment 16k
      Query
      Tables
        customers
          columns
            CustomerID int
            FirstName varchar(60)
            LastName varchar(60)
            Email varchar(80)
            Age int
          index
```