# **Assignment-5**

Name- Ajay Chaudhary Batch- Data Engineering (Batch-1)

### **Data cleaning & Transformation queries**

Table for data cleaning

```
d • ○ create table studentdatal(
    id int,
    name varchar(250),
    age int,
    grade varchar(20)
    );

10 • select * from studentdatal;

11 • insert into studentdatal(id,name,age,grade) values(null,'stella',20,'A+');

12 • insert into studentdatal(id,name,age,grade) values(1,'appu',20,'A+');

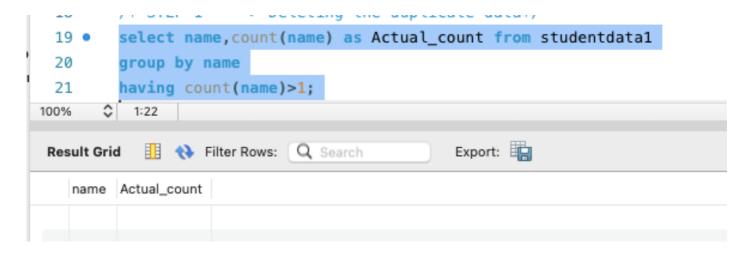
13 • insert into studentdatal(id,name,age,grade) values(5,'bob',21,'C');

14 • insert into studentdatal(id,name,age,grade) values(6,'sunny',21,null);

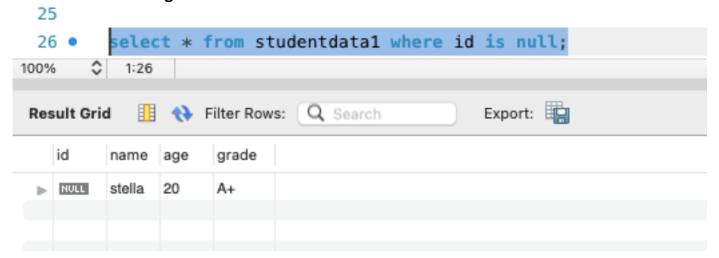
15 • insert into studentdatal(id,name,age,grade) values(7,null,21,'C');
```

### Perfoming data cleaning

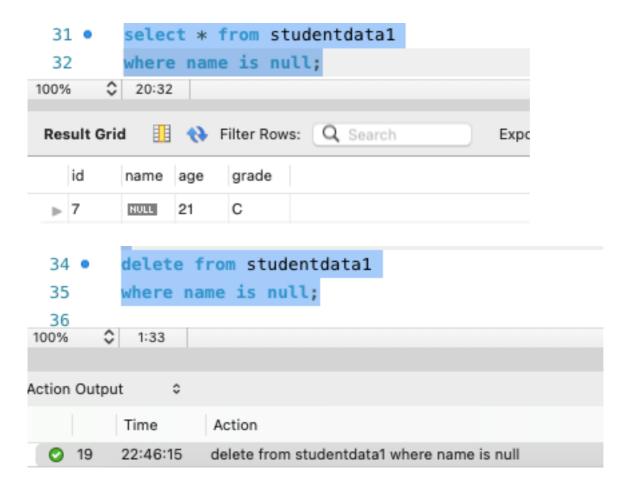
### STEP-1 ----> Deleting the duplicate data



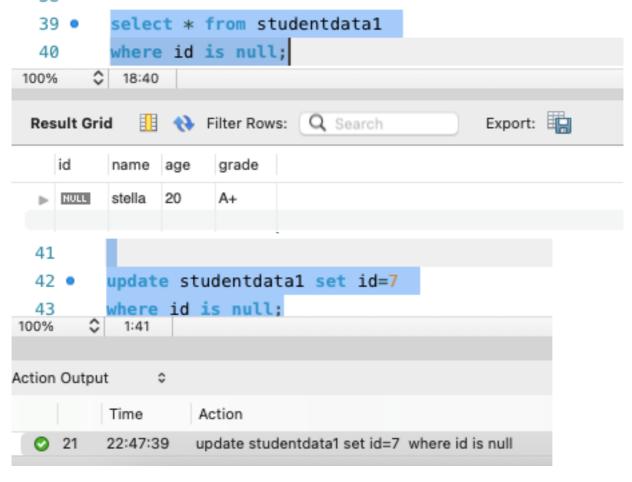
#### STEP-2 ---> removing the null values over here



### selecting the data where student name is null

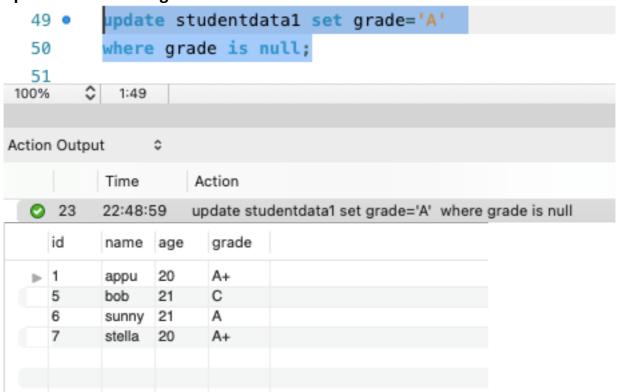


# updating the null values where id and grade column is null

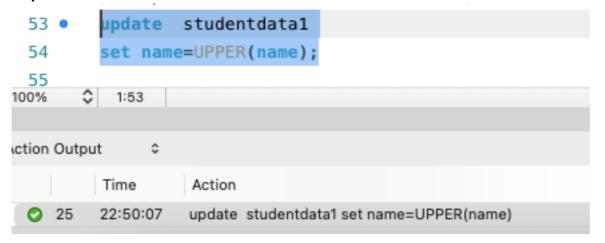


	id	name	age	grade
⊳	1	арри	20	A+
	5	bob	21	С
	6	sunny	21	NULL
	7	stella	20	A+

# update the value of grade also



### Capitalization of names in data



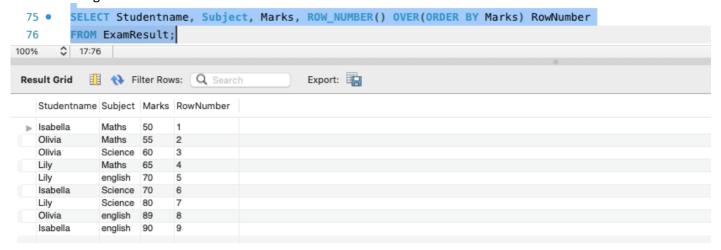
	id	name	age	grade
<b> </b>	1	APPU	20	A+
	5	BOB	21	C
	6	SUNNY	21	A
	7	STELLA	20	A+

### **Ranking Functions**

```
CREATE TABLE ExamResult
57 •
58
    ⊝ (
59
       StudentName VARCHAR(70),
60
        Subject
                    VARCHAR(20),
        Marks
                     INT
61
62
     -);
63
64 •
       INSERT INTO ExamResult VALUES('Lily', 'Maths', 65);
       INSERT INTO ExamResult VALUES('Lily', 'Science', 80);
65 •
       INSERT INTO ExamResult VALUES('Lily', 'english',70);
       INSERT INTO ExamResult VALUES('Isabella', 'Maths', 50);
67 •
       INSERT INTO ExamResult VALUES('Isabella', 'Science', 70);
68 •
       INSERT INTO ExamResult VALUES('Isabella', 'english', 90);
69 •
       INSERT INTO ExamResult VALUES('Olivia', 'Maths', 55);
70 •
71 •
       INSERT INTO ExamResult VALUES('Olivia', 'Science', 60);
72 •
       INSERT INTO ExamResult VALUES('Olivia', 'english', 89);
```

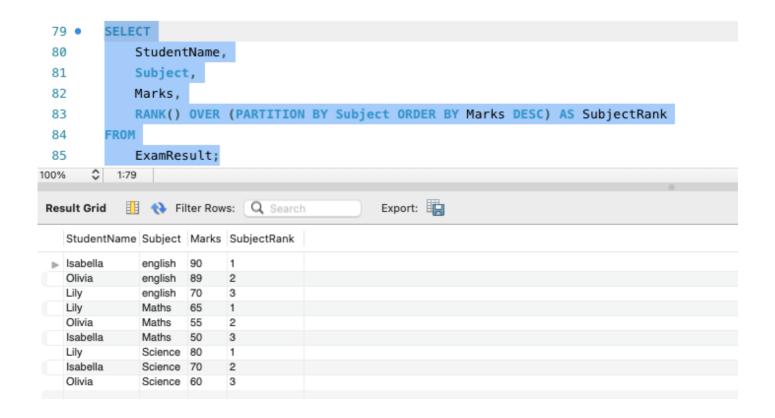
# **ROW\_Number() SQL RANK function**

We use ROW\_Number() SQL RANK function to get a unique sequential number for each row in the specified data. It gives the rank one for the first row and then increments the value by one for each row. We get different ranks for the row having similar values as well.



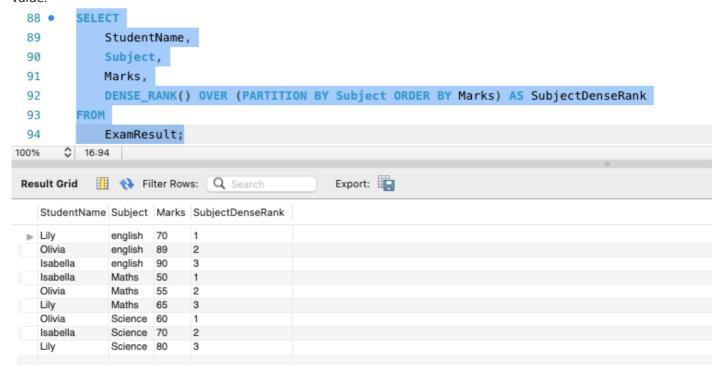
# **RANK() SQL RANK Function**

We use RANK() SQL Rank function to specify rank for each row in the result set.



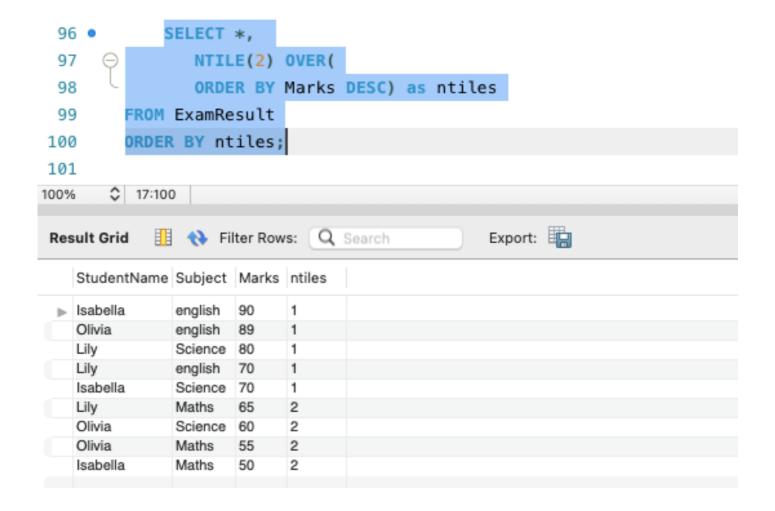
### DenseRANK() sql function

We use DENSE\_RANK() function to specify a unique rank number within the partition as per the specified column value.



## NTILE(N) SQL RANK function

We use the NTILE(N) function to distribute the number of rows in the specified (N) number of groups. Each row group gets its rank as per the specified condition. We need to specify the value for the desired number of groups.



#### **Stored Procedures-**

### **Tables used for stored procedures**

```
104 •
        CREATE TABLE Product
105
        (ProductID INT, ProductName VARCHAR(100));
106
       CREATE TABLE ProductDescription
107 •
        (ProductID INT, ProductDescription VARCHAR(800) )
108
109
       INSERT INTO Product VALUES (680, 'HL Road Frame - Black, 58')
110 🛚
        ,(706,'HL Road Frame - Red, 58')
111
112
        ,(707,'Sport-100 Helmet, Red')
113
       INSERT INTO ProductDescription VALUES (680, 'Replacement mountain wheel for entry-level rider.')
114
115
       ,(706, 'Sturdy alloy features a quick-release hub.')
       ,(707,'Aerodynamic rims for smooth riding.')
116
```

```
139
        DELIMITER //
        CREATE PROCEDURE GetProductInfo(IN p_ProductID INT)
140 •
141

→ BEGIN

            SELECT
142
                p.ProductID,
143
144
                p.ProductName,
145
                pd.ProductDescription
            FROM
146
                Product p
147
            JOIN
148
                ProductDescription pd ON p.ProductID = pd.ProductID
149
150
                p.ProductID = p_ProductID;
151
152
        END //
153
154
        DELIMITER ;
```

### Execute the stored procedure with ProductID = 680

