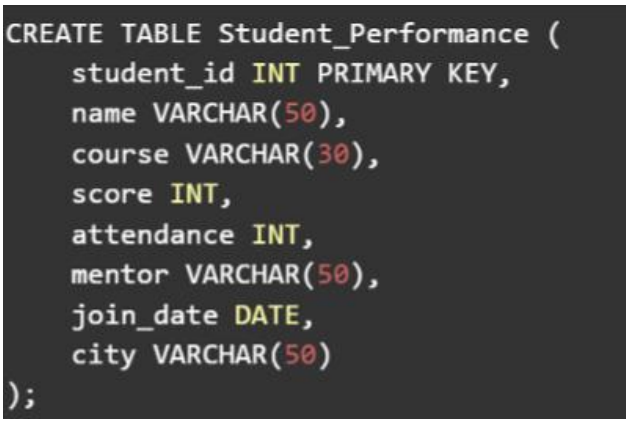
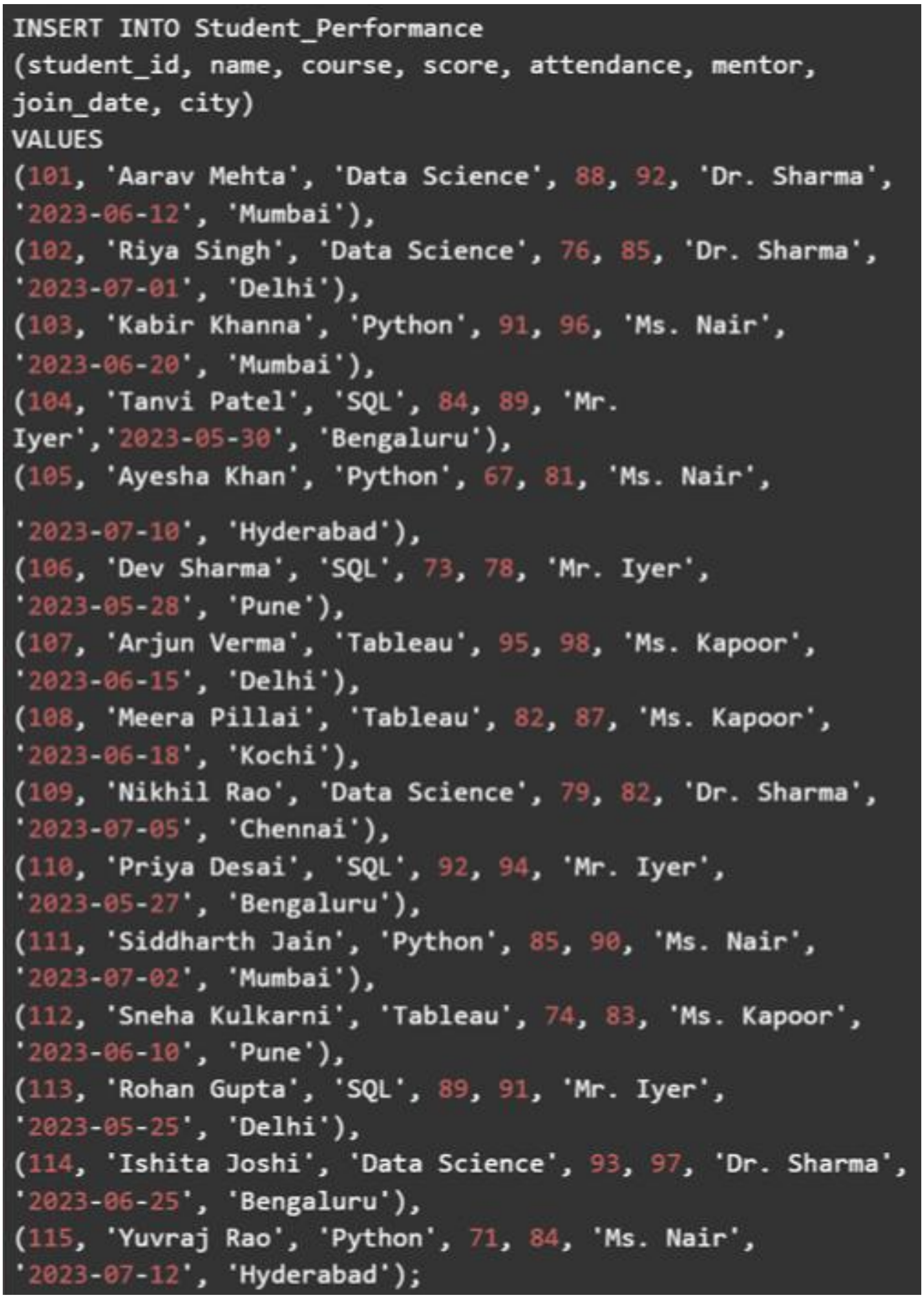
Note : Create a table in MySQL Workbench using the following code :

- **SQL Code to Create the Table**



```
CREATE TABLE Student_Performance (
    student_id INT PRIMARY KEY,
    name VARCHAR(50),
    course VARCHAR(30),
    score INT,
    attendance INT,
    mentor VARCHAR(50),
    join_date DATE,
    city VARCHAR(50)
);
```

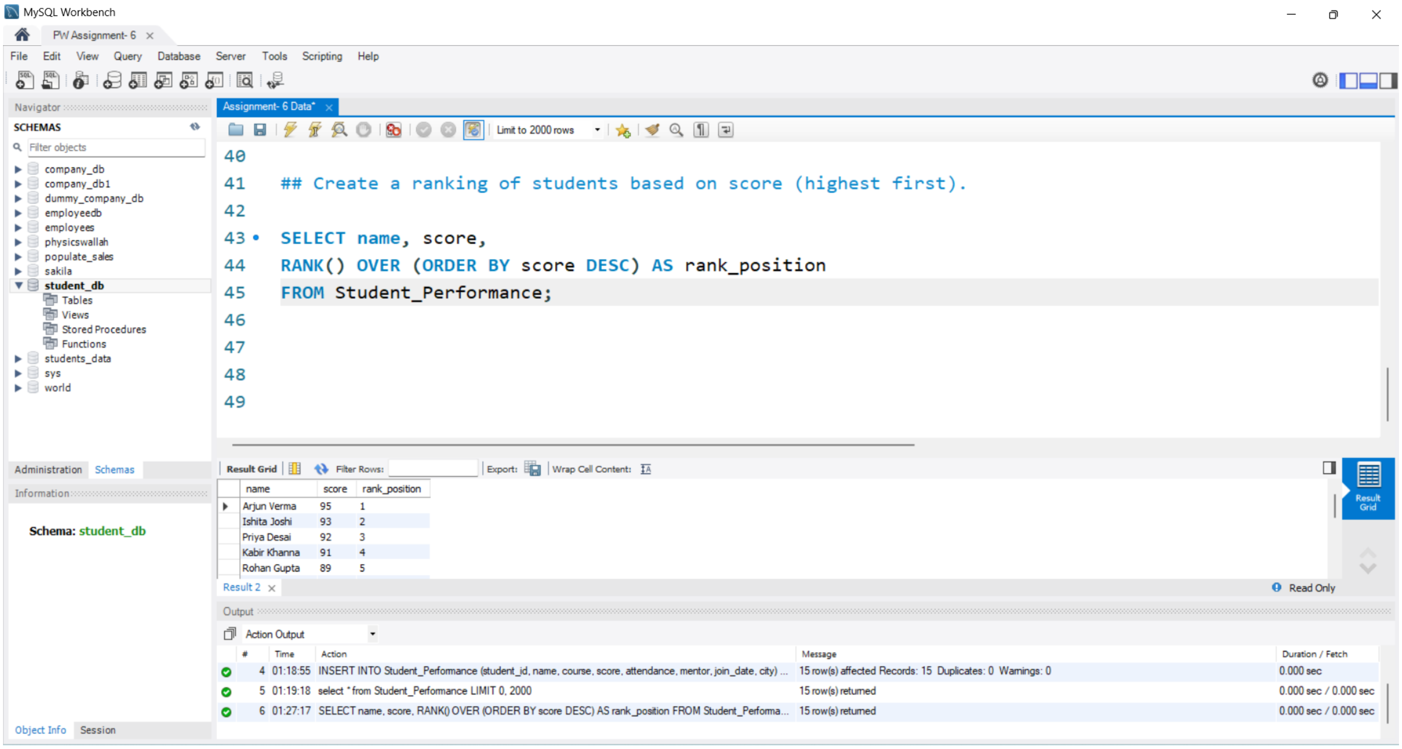
- **Insert the Data**



```
INSERT INTO Student_Performance
(student_id, name, course, score, attendance, mentor,
join_date, city)
VALUES
(101, 'Aarav Mehta', 'Data Science', 88, 92, 'Dr. Sharma',
'2023-06-12', 'Mumbai'),
(102, 'Riya Singh', 'Data Science', 76, 85, 'Dr. Sharma',
'2023-07-01', 'Delhi'),
(103, 'Kabir Khanna', 'Python', 91, 96, 'Ms. Nair',
'2023-06-20', 'Mumbai'),
(104, 'Tanvi Patel', 'SQL', 84, 89, 'Mr.
Iyer','2023-05-30', 'Bengaluru'),
(105, 'Ayesha Khan', 'Python', 67, 81, 'Ms. Nair',

'2023-07-10', 'Hyderabad'),
(106, 'Dev Sharma', 'SQL', 73, 78, 'Mr. Iyer',
'2023-05-28', 'Pune'),
(107, 'Arjun Verma', 'Tableau', 95, 98, 'Ms. Kapoor',
'2023-06-15', 'Delhi'),
(108, 'Meera Pillai', 'Tableau', 82, 87, 'Ms. Kapoor',
'2023-06-18', 'Kochi'),
(109, 'Nikhil Rao', 'Data Science', 79, 82, 'Dr. Sharma',
'2023-07-05', 'Chennai'),
(110, 'Priya Desai', 'SQL', 92, 94, 'Mr. Iyer',
'2023-05-27', 'Bengaluru'),
(111, 'Siddharth Jain', 'Python', 85, 90, 'Ms. Nair',
'2023-07-02', 'Mumbai'),
(112, 'Sneha Kulkarni', 'Tableau', 74, 83, 'Ms. Kapoor',
'2023-06-10', 'Pune'),
(113, 'Rohan Gupta', 'SQL', 89, 91, 'Mr. Iyer',
'2023-05-25', 'Delhi'),
(114, 'Ishita Joshi', 'Data Science', 93, 97, 'Dr. Sharma',
'2023-06-25', 'Bengaluru'),
(115, 'Yuvraj Rao', 'Python', 71, 84, 'Ms. Nair',
'2023-07-12', 'Hyderabad');
```

Question 1 : Create a ranking of students based on score (highest first).

Ans:-



Schema: student_db

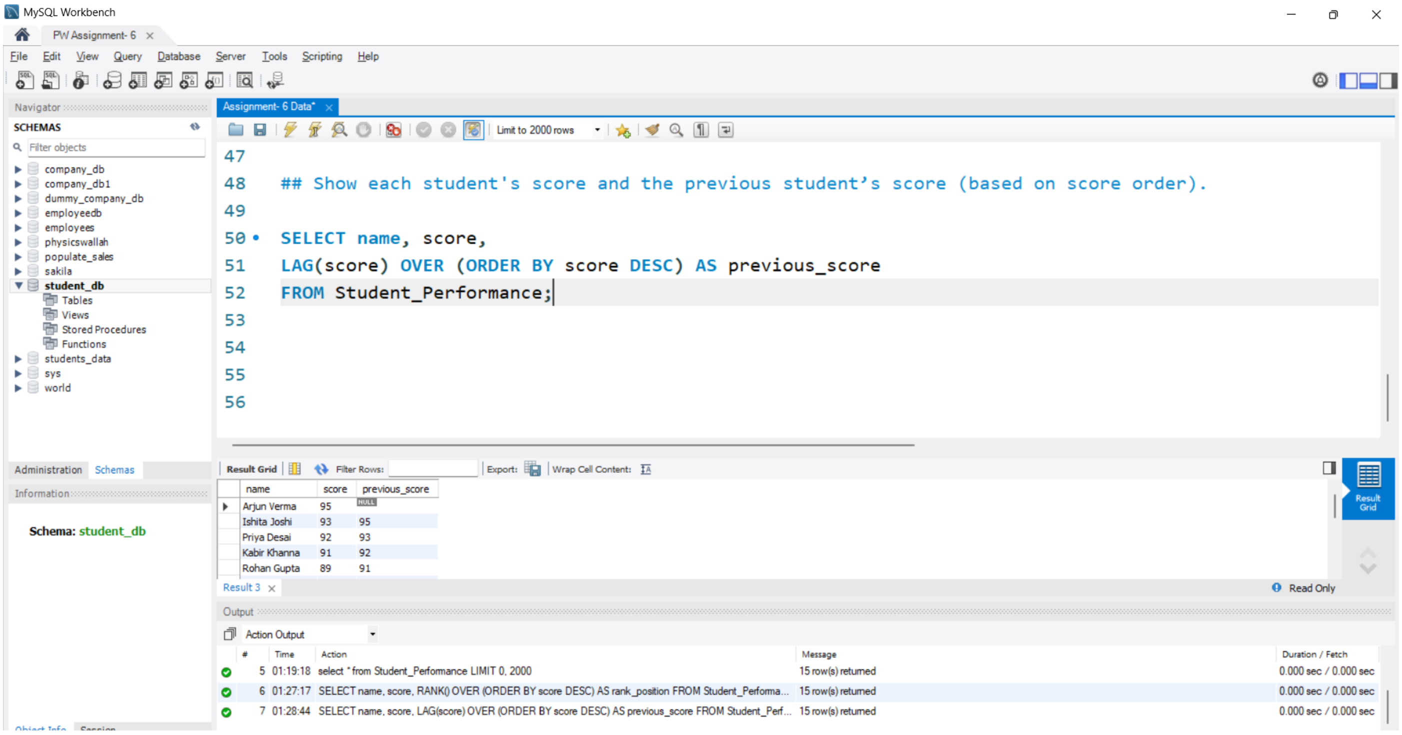
```sql
## Create a ranking of students based on score (highest first).

SELECT name, score,
RANK() OVER (ORDER BY score DESC) AS rank_position
FROM Student_Performance;
```

name	score	rank_position
Arjun Verma	95	1
Ishita Joshi	93	2
Priya Desai	92	3
Kabir Khanna	91	4
Rohan Gupta	89	5

#	Time	Action	Message	Duration / Fetch
4	01:18:55	INSERT INTO Student_Performance (student_id, name, course, score, attendance, mentor, join_date, city) ...	15 row(s) affected Records: 15 Duplicates: 0 Warnings: 0	0.000 sec
5	01:19:18	select * from Student_Performance LIMIT 0, 2000	15 row(s) returned	0.000 sec / 0.000 sec
6	01:27:17	SELECT name, score, RANK() OVER (ORDER BY score DESC) AS rank_position FROM Student_Performa...	15 row(s) returned	0.000 sec / 0.000 sec

Question 2 : Show each student's score and the previous student’s score (based on score order).

Ans:-



Schema: student_db

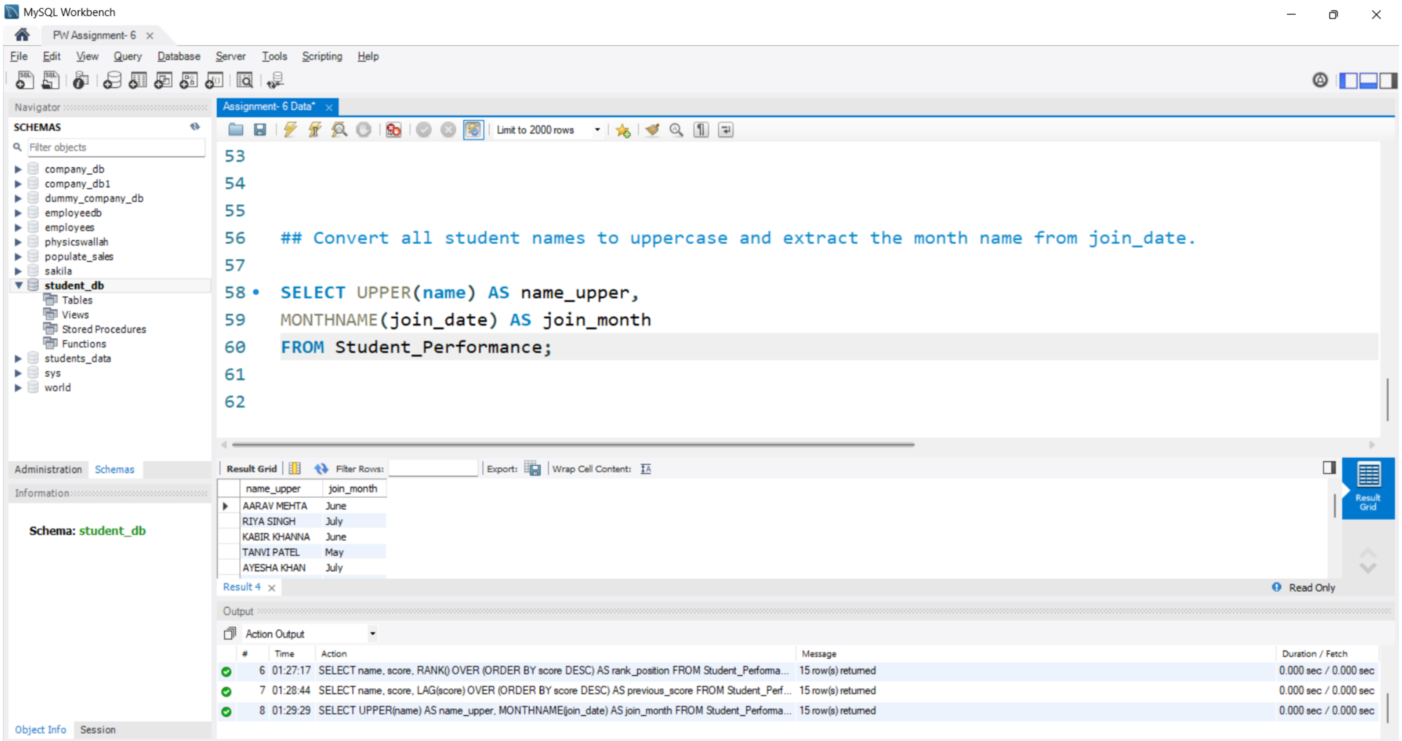
```sql
## Show each student's score and the previous student’s score (based on score order).

SELECT name, score,
LAG(score) OVER (ORDER BY score DESC) AS previous_score
FROM Student_Performance;
```

name	score	previous_score
Arjun Verma	95	NULL
Ishita Joshi	93	95
Priya Desai	92	93
Kabir Khanna	91	92
Rohan Gupta	89	91

#	Time	Action	Message	Duration / Fetch
5	01:19:18	select * from Student_Performance LIMIT 0, 2000	15 row(s) returned	0.000 sec / 0.000 sec
6	01:27:17	SELECT name, score, RANK() OVER (ORDER BY score DESC) AS rank_position FROM Student_Performa...	15 row(s) returned	0.000 sec / 0.000 sec
7	01:28:44	SELECT name, score, LAG(score) OVER (ORDER BY score DESC) AS previous_score FROM Student_Perf...	15 row(s) returned	0.000 sec / 0.000 sec

Question 3 : Convert all student names to uppercase and extract the month name from join_date.

Ans:-



Schema: student_db

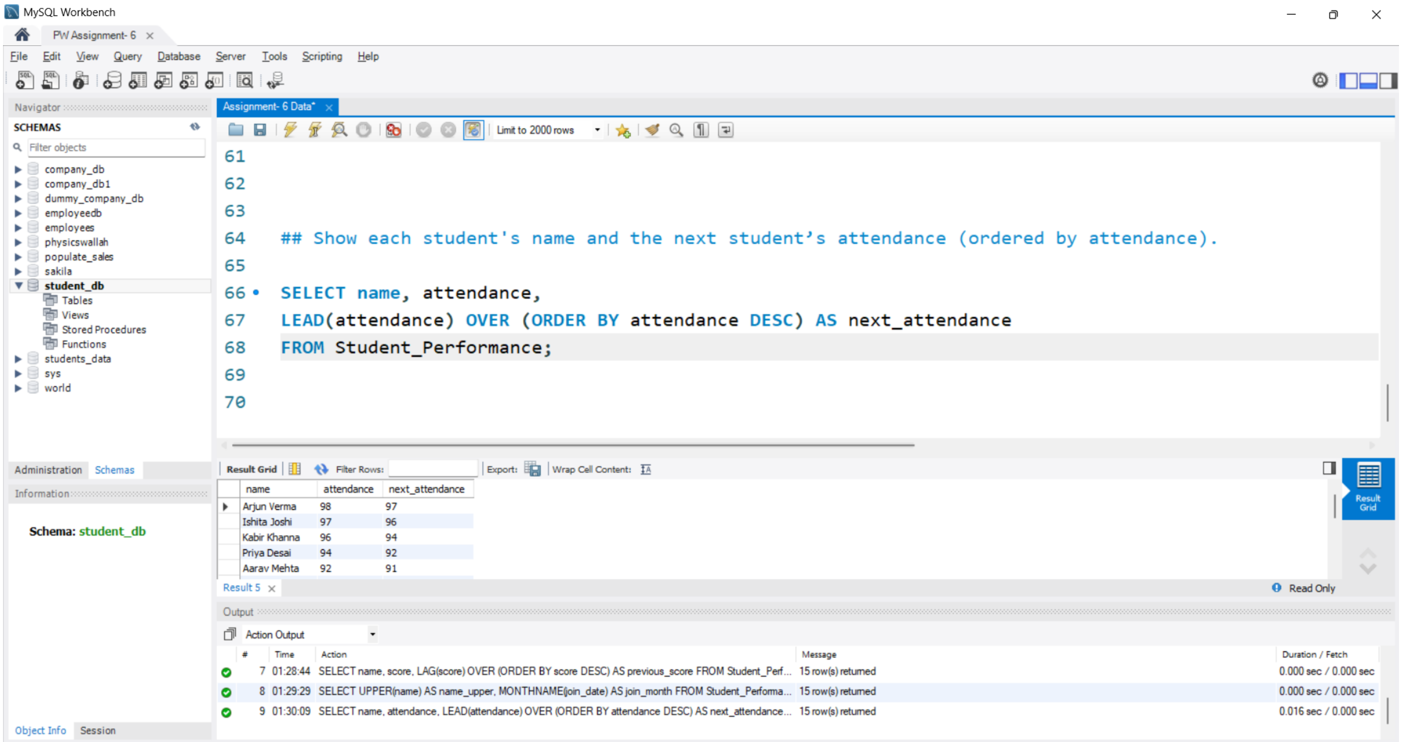
```sql
## Convert all student names to uppercase and extract the month name from join_date.

SELECT UPPER(name) AS name_upper,
MONTHNAME(join_date) AS join_month
FROM Student_Performance;
```

name_upper	join_month
AARAV MEHTA	June
RIYA SINGH	July
KABIR KHANNA	June
TANVI PATEL	May
AYESHA KHAN	July

```
#  Time      Action                                                                                  Message              Duration / Fetch
6  01:27:17  SELECT name, score, RANK() OVER (ORDER BY score DESC) AS rank_position FROM Student_Performa...  15 row(s) returned  0.000 sec / 0.000 sec
7  01:28:44  SELECT name, score, LAG(score) OVER (ORDER BY score DESC) AS previous_score FROM Student_Perf...  15 row(s) returned  0.000 sec / 0.000 sec
8  01:29:29  SELECT UPPER(name) AS name_upper, MONTHNAME(join_date) AS join_month FROM Student_Performa...  15 row(s) returned  0.000 sec / 0.000 sec
```

Question 4 : Show each student's name and the next student's attendance (ordered by attendance).

Ans:-



Schema: student_db

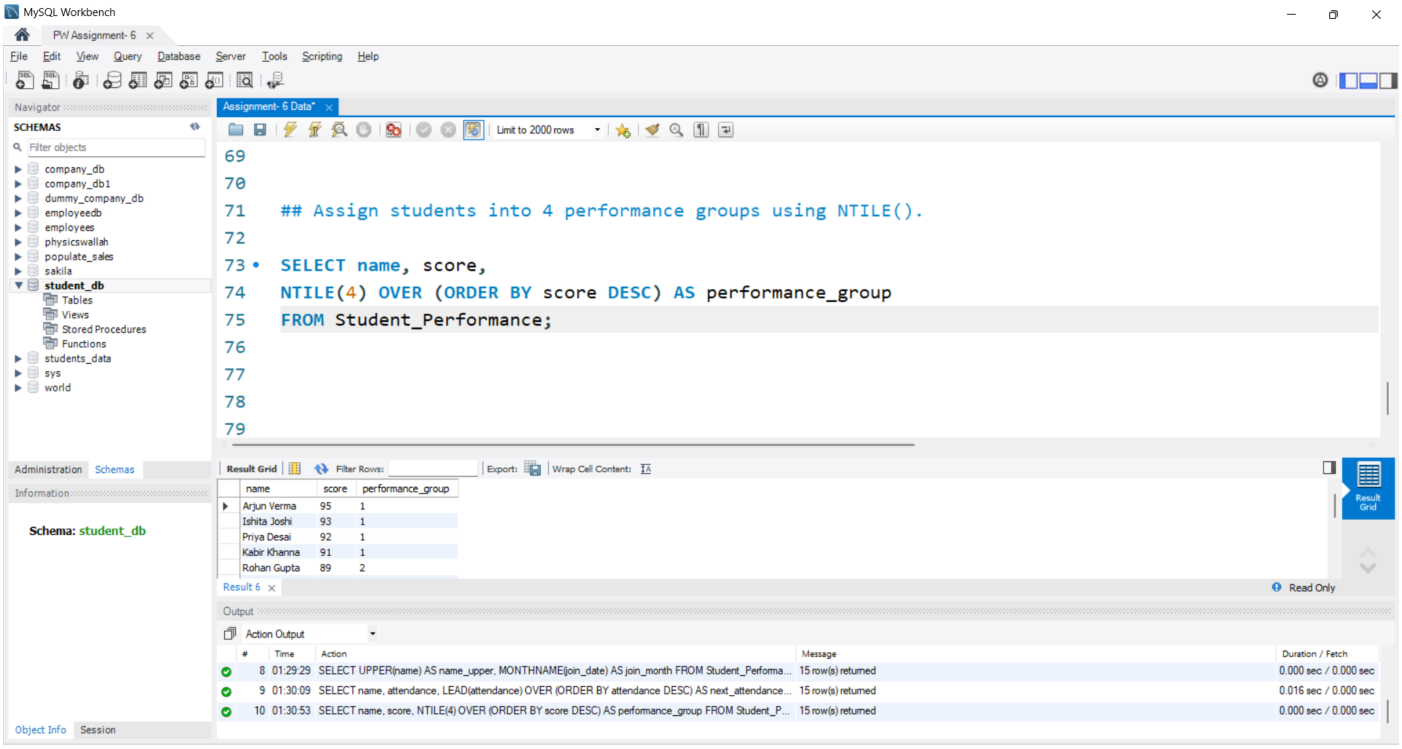
```sql
## Show each student's name and the next student’s attendance (ordered by attendance).

SELECT name, attendance,
LEAD(attendance) OVER (ORDER BY attendance DESC) AS next_attendance
FROM Student_Performance;
```

name	attendance	next_attendance
Arjun Verma	98	97
Ishita Joshi	97	96
Kabir Khanna	96	94
Priya Desai	94	92
Aarav Mehta	92	91

```
#  Time      Action                                                                                  Message              Duration / Fetch
7  01:28:44  SELECT name, score, LAG(score) OVER (ORDER BY score DESC) AS previous_score FROM Student_Perf...  15 row(s) returned  0.000 sec / 0.000 sec
8  01:29:29  SELECT UPPER(name) AS name_upper, MONTHNAME(join_date) AS join_month FROM Student_Performa...  15 row(s) returned  0.000 sec / 0.000 sec
9  01:30:09  SELECT name, attendance, LEAD(attendance) OVER (ORDER BY attendance DESC) AS next_attendance...  15 row(s) returned  0.016 sec / 0.000 sec
```

Question 5 : Assign students into 4 performance groups using NTILE().

Ans:-



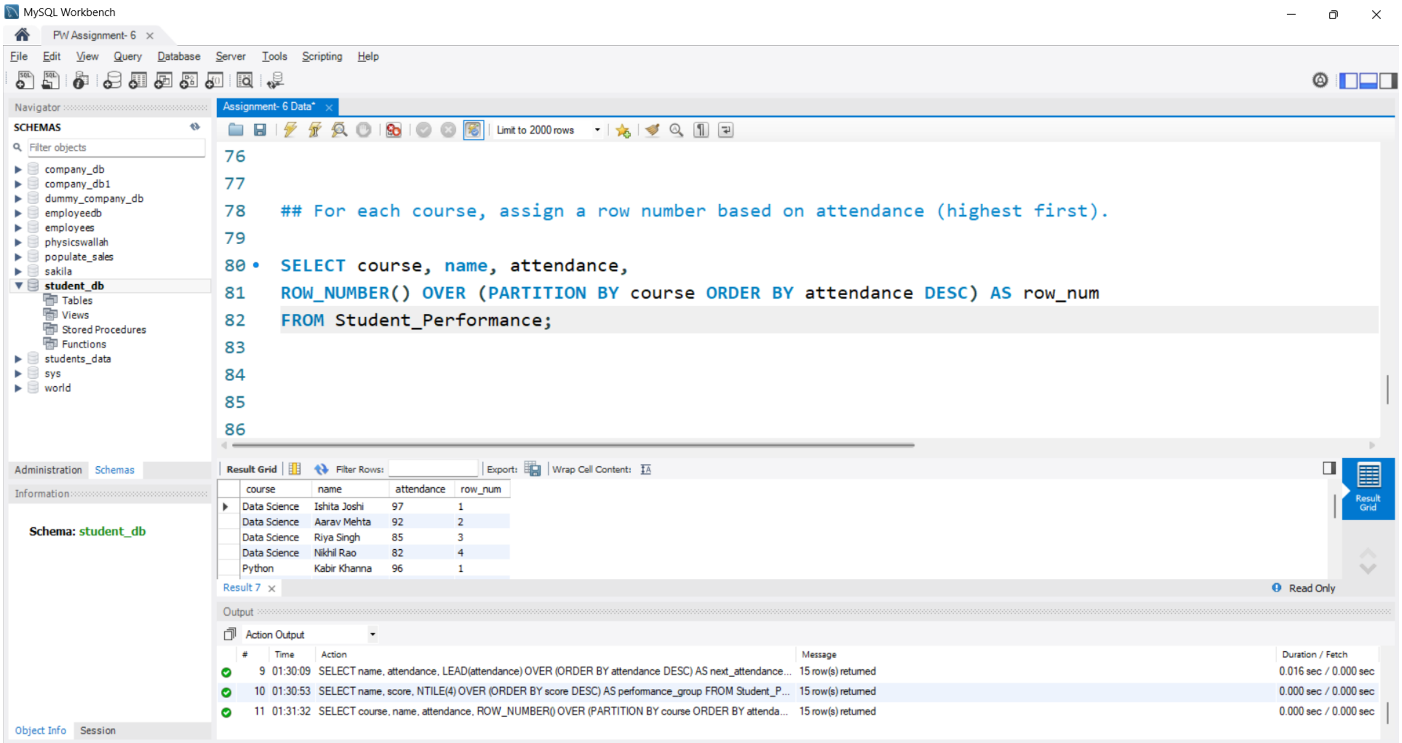
```
## Assign students into 4 performance groups using NTILE().

SELECT name, score,
NTILE(4) OVER (ORDER BY score DESC) AS performance_group
FROM Student_Performance;
```

name	score	performance_group
Arjun Verma	95	1
Ishita Joshi	93	1
Priya Desai	92	1
Kabir Khanna	91	1
Rohan Gupta	89	2

```
#   Time      Action                                                                                  Message              Duration / Fetch
8   01:29:29  SELECT UPPER(name) AS name_upper, MONTHNAME(join_date) AS join_month FROM Student_Performa...   15 row(s) returned   0.000 sec / 0.000 sec
9   01:30:09  SELECT name, attendance, LEAD(attendance) OVER (ORDER BY attendance DESC) AS next_attendance...  15 row(s) returned   0.016 sec / 0.000 sec
10  01:30:53  SELECT name, score, NTILE(4) OVER (ORDER BY score DESC) AS performance_group FROM Student_P...    15 row(s) returned   0.000 sec / 0.000 sec
```

Question 6 : For each course, assign a row number based on attendance (highest first).

Ans:-



```
## For each course, assign a row number based on attendance (highest first).

SELECT course, name, attendance,
ROW_NUMBER() OVER (PARTITION BY course ORDER BY attendance DESC) AS row_num
FROM Student_Performance;
```

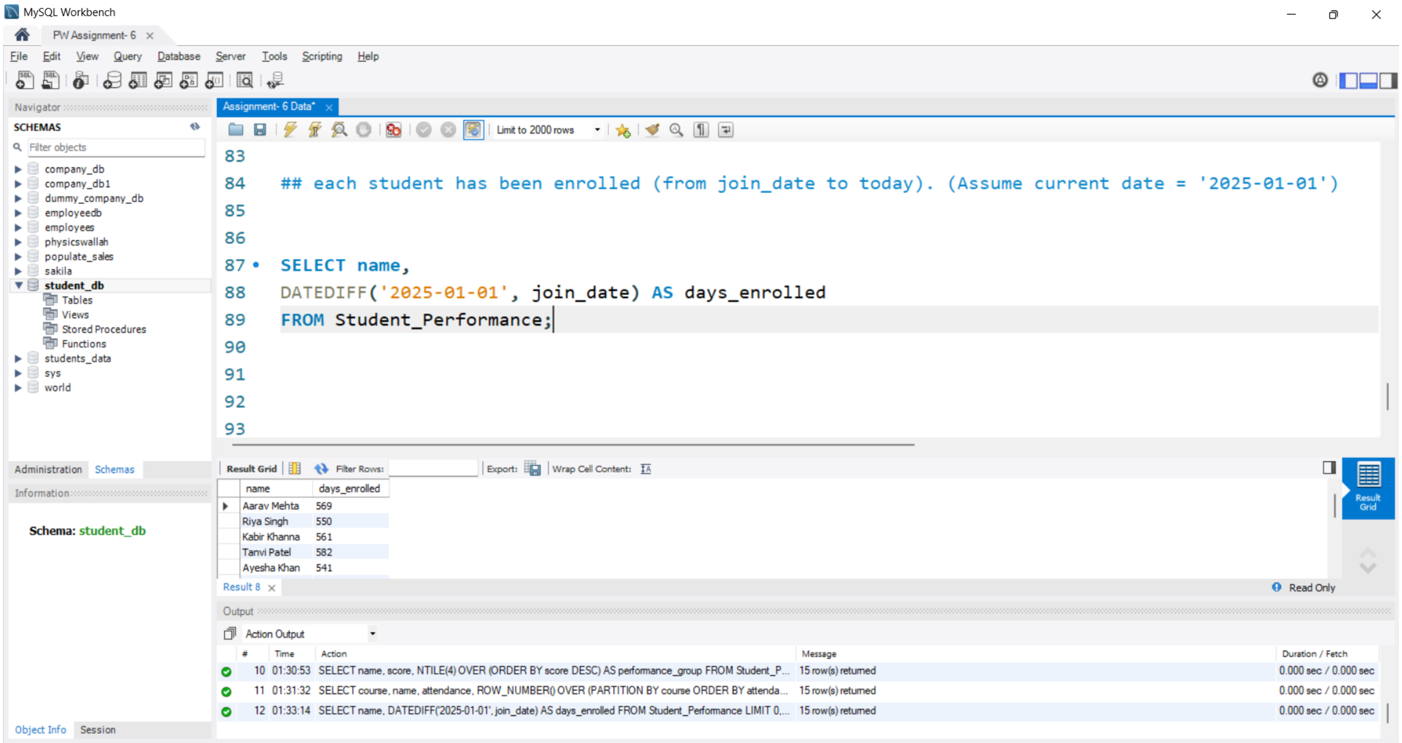
course	name	attendance	row_num
Data Science	Ishita Joshi	97	1
Data Science	Aarav Mehta	92	2
Data Science	Riya Singh	85	3
Data Science	Nikhil Rao	82	4
Python	Kabir Khanna	96	1

```
#   Time      Action                                                                                  Message              Duration / Fetch
9   01:30:09  SELECT name, attendance, LEAD(attendance) OVER (ORDER BY attendance DESC) AS next_attendance...  15 row(s) returned   0.016 sec / 0.000 sec
10  01:30:53  SELECT name, score, NTILE(4) OVER (ORDER BY score DESC) AS performance_group FROM Student_P...    15 row(s) returned   0.000 sec / 0.000 sec
11  01:31:32  SELECT course, name, attendance, ROW_NUMBER() OVER (PARTITION BY course ORDER BY attenda...      15 row(s) returned   0.000 sec / 0.000 sec
```

Question 7 : Calculate the number of days each student has been enrolled (from join_date to today).

(Assume current date = '2025-01-01')

Ans:-



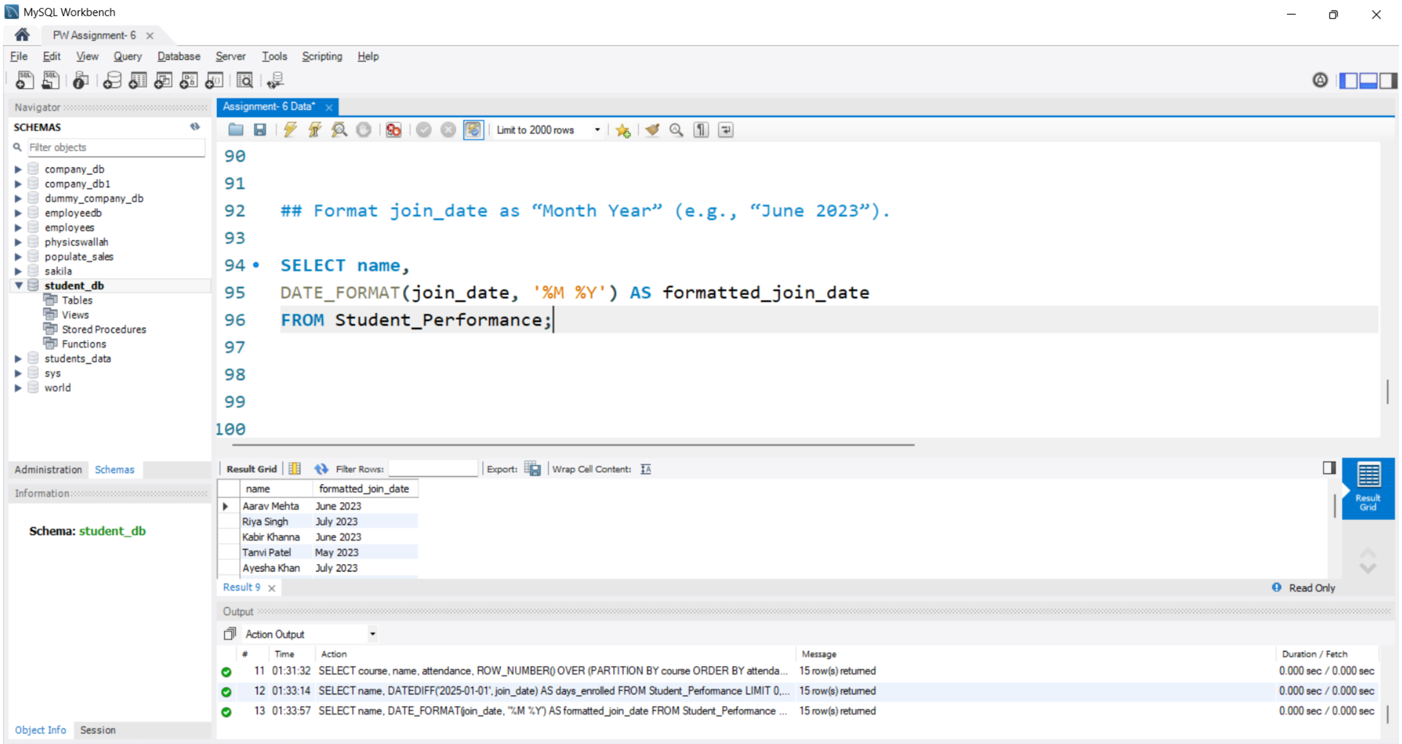
```sql
## each student has been enrolled (from join_date to today). (Assume current date = '2025-01-01')

SELECT name,
DATEDIFF('2025-01-01', join_date) AS days_enrolled
FROM Student_Performance;
```

name	days_enrolled
Aarav Mehta	569
Riya Singh	550
Kabir Khanna	561
Tanvi Patel	582
Ayesha Khan	541

```
#   Time      Action                                                                                    Message              Duration / Fetch
10  01:30:53  SELECT name, score, NTILE(4) OVER (ORDER BY score DESC) AS performance_group FROM Student_P...  15 row(s) returned  0.000 sec / 0.000 sec
11  01:31:32  SELECT course, name, attendance, ROW_NUMBER() OVER (PARTITION BY course ORDER BY attenda...  15 row(s) returned  0.000 sec / 0.000 sec
12  01:33:14  SELECT name, DATEDIFF('2025-01-01', join_date) AS days_enrolled FROM Student_Performance LIMIT 0,...  15 row(s) returned  0.000 sec / 0.000 sec
```

Question 8 : Format join_date as “Month Year” (e.g., “June 2023”).

Ans:-



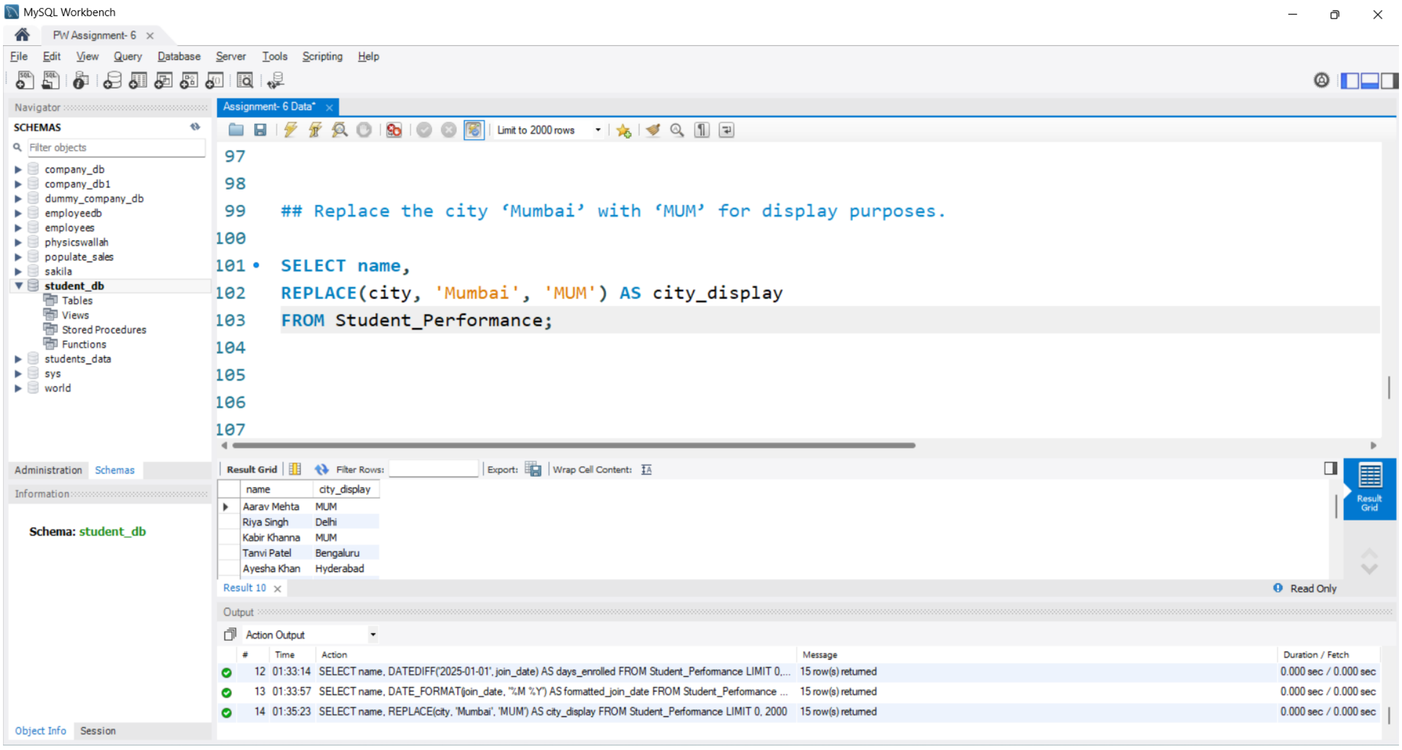
```sql
## Format join_date as “Month Year” (e.g., “June 2023”).

SELECT name,
DATE_FORMAT(join_date, '%M %Y') AS formatted_join_date
FROM Student_Performance;
```

name	formatted_join_date
Aarav Mehta	June 2023
Riya Singh	July 2023
Kabir Khanna	June 2023
Tanvi Patel	May 2023
Ayesha Khan	July 2023

```
#   Time      Action                                                                                    Message              Duration / Fetch
11  01:31:32  SELECT course, name, attendance, ROW_NUMBER() OVER (PARTITION BY course ORDER BY attenda...  15 row(s) returned  0.000 sec / 0.000 sec
12  01:33:14  SELECT name, DATEDIFF('2025-01-01', join_date) AS days_enrolled FROM Student_Performance LIMIT 0,...  15 row(s) returned  0.000 sec / 0.000 sec
13  01:33:57  SELECT name, DATE_FORMAT(join_date, '%M %Y') AS formatted_join_date FROM Student_Performance ...  15 row(s) returned  0.000 sec / 0.000 sec
```

Question 9 : Replace the city ‘Mumbai’ with ‘MUM’ for display purposes.

Ans:-



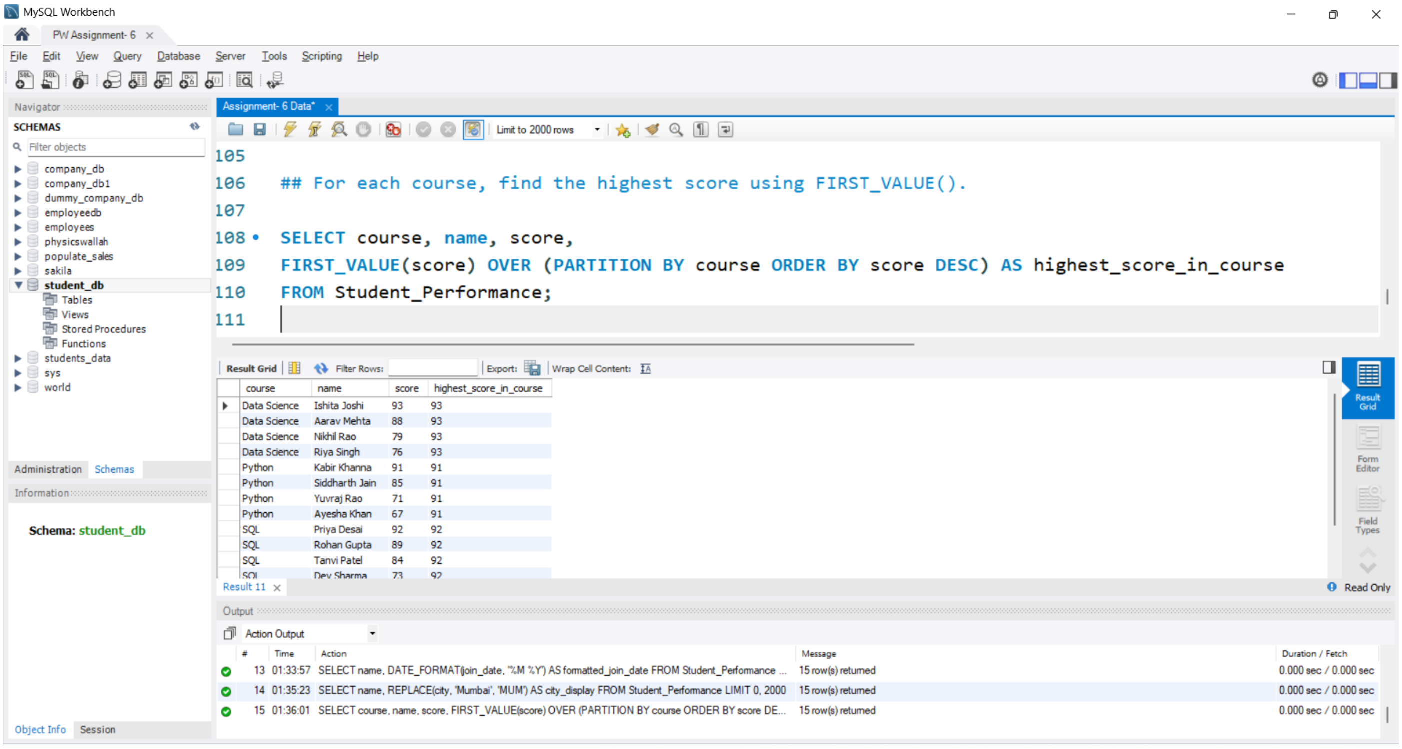
```
## Replace the city ‘Mumbai’ with ‘MUM’ for display purposes.

SELECT name,
REPLACE(city, 'Mumbai', 'MUM') AS city_display
FROM Student_Performance;
```

name	city_display
Aarav Mehta	MUM
Riya Singh	Delhi
Kabir Khanna	MUM
Tanvi Patel	Bengaluru
Ayesha Khan	Hyderabad

```
#   Time      Action                                                                              Message               Duration / Fetch
12  01:33:14  SELECT name, DATEDIFF('2025-01-01', join_date) AS days_enrolled FROM Student_Performance LIMIT 0,...  15 row(s) returned  0.000 sec / 0.000 sec
13  01:33:57  SELECT name, DATE_FORMAT(join_date, '%M %Y') AS formatted_join_date FROM Student_Performance ...  15 row(s) returned  0.000 sec / 0.000 sec
14  01:35:23  SELECT name, REPLACE(city, 'Mumbai', 'MUM') AS city_display FROM Student_Performance LIMIT 0, 2000  15 row(s) returned  0.000 sec / 0.000 sec
```

Question 10 : For each course, find the highest score using FIRST_VALUE().

Ans:-



```
## For each course, find the highest score using FIRST_VALUE().

SELECT course, name, score,
FIRST_VALUE(score) OVER (PARTITION BY course ORDER BY score DESC) AS highest_score_in_course
FROM Student_Performance;
```

course	name	score	highest_score_in_course
Data Science	Ishita Joshi	93	93
Data Science	Aarav Mehta	88	93
Data Science	Nikhil Rao	79	93
Data Science	Riya Singh	76	93
Python	Kabir Khanna	91	91
Python	Siddharth Jain	85	91
Python	Yuvraj Rao	71	91
Python	Ayesha Khan	67	91
SQL	Priya Desai	92	92
SQL	Rohan Gupta	89	92
SQL	Tanvi Patel	84	92
SQL	Dev Sharma	73	92

```
#   Time      Action                                                                              Message               Duration / Fetch
13  01:33:57  SELECT name, DATE_FORMAT(join_date, '%M %Y') AS formatted_join_date FROM Student_Performance ...  15 row(s) returned  0.000 sec / 0.000 sec
14  01:35:23  SELECT name, REPLACE(city, 'Mumbai', 'MUM') AS city_display FROM Student_Performance LIMIT 0, 2000  15 row(s) returned  0.000 sec / 0.000 sec
15  01:36:01  SELECT course, name, score, FIRST_VALUE(score) OVER (PARTITION BY course ORDER BY score DE...  15 row(s) returned  0.000 sec / 0.000 sec
```