SQL FUNCTIONS

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

ANSWER:
```
SELECT
  student_id,
  name,
  score,
  RANK() OVER (ORDER BY score DESC) AS score_rank
FROM Student_Performance;
```

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

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

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

ANSWER:
```
SELECT
  UPPER(name) AS student_name,
  MONTHNAME(join_date) AS join_month
FROM Student_Performance;
```

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

ANSWER:

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

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

ANSWER:

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

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

ANSWER:

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

QUESTION 7: Calculate the number of days each student has been enrolled (from join_date to today). (Assume current date = ‘2025-01-01’)

ANSWER:

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

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

ANSWER:

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

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

ANSWER:

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

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

ANSWER:

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