Lecture Notes – String Functions-CaseChange

Section 1: Lecture Summary

The lecture covered **case-changing string functions** in MySQL—specifically **UPPER()** and **LOWER()**—and showed how to use them in SELECT and WHERE clauses, plus how to apply them to concatenated columns (e.g., full names) and to make string comparisons case-insensitive.

Section 2: Key Concepts and Explanations

- **UPPER(string)**: returns the string converted to uppercase.

- **LOWER(string)**: returns the string converted to lowercase.

- **Concatenation with CONCAT(...)**: combine columns and literals (for example, first name, a space, and last name) before applying UPPER/LOWER.

- **Using aliases in ORDER BY**: you can assign an alias (e.g., full_name) to an expression in the SELECT list and reference that alias in ORDER BY.

- **Why use UPPER/LOWER in WHERE**: convert stored values and the comparison literal to the same case to perform case-insensitive matching when you cannot guarantee the stored text's casing.

Section 3: Example Code and Use Cases (must use only companyDB)

Example 1 — Show employees' full name in uppercase and order alphabetically:

```
SELECT UPPER(CONCAT(FirstName, ' ', LastName)) AS full_name
FROM companyDB.Employees
ORDER BY full_name;
```

Example 2 — Show employees and department names for departments located in Delhi (case-insensitive match on Location):

```
SELECT e.FirstName,
       e.LastName,
       d.DeptName,
       d.Location
FROM companyDB.Employees e
```

```
NATURAL JOIN companyDB.Departments d
WHERE LOWER(d.Location) = 'delhi';
```

Notes for Example 2:

- Uses **NATURAL JOIN** because Departments.DeptID is the primary/foreign key linking Employees and Departments in the provided schema (DeptID exists in both tables).

- Applies **LOWER(d.Location)** so that the comparison is insensitive to how "Delhi" is cased in the table.

Section 4: Key Takeaways

- Use **UPPER()** or **LOWER()** to normalize text for display or comparison.

- Apply these functions to concatenated expressions (e.g., CONCAT(FirstName,' ',LastName)) to change case for combined values.

- For case-insensitive filtering, convert the column value and compare to a constant in the same case (commonly LOWER(column) = 'value').

- You may reference a SELECT alias (like full_name) in ORDER BY to sort by the computed value.