Experiment 7: Execution of String, Comparison, and Set Operations

Aim:

Execution of string, comparison, and set operations.

Theory

STRING FUNCTIONS

- **Concat:** Returns char1 concatenated with char2.
    - **Syntax:** `CONCAT(char1, char2)`
    - **Example:**

    ```
    SELECT CONCAT('ORACLE', 'CORPORATION') FROM DUAL;
    ```

    Result: `ORACLECORPORATION`

- **Lpad:** Returns expr1 left-padded to length n characters with the sequence of characters in expr2.
    - **Syntax:** `LPAD(expr1, n, expr2)`
    - **Example:**

    ```
    SELECT LPAD('ORACLE', 15, '*') FROM DUAL;
    ```

    Result: `*********ORACLE`

- **Rpad:** Returns expr1 right-padded to length n characters with expr2, replicated as necessary.
    - **Syntax:** `RPAD(expr1, n, expr2)`
    - **Example:**

    ```
    SELECT RPAD('ORACLE', 15, '*') FROM DUAL;
    ```

    Result: `ORACLE*********`

- **Ltrim:** Returns a character expression after removing leading blanks.
  - **Syntax:** `LTRIM(char, set)`
  - **Example:**

    ```
    SELECT LTRIM('SSMITHSS', 'S') FROM DUAL;
    ```

    Result: `MITHSS`

- **Rtrim:** Returns a character string after truncating all trailing blanks.
  - **Syntax:** `RTRIM(char, set)`
  - **Example:**

    ```
    SELECT RTRIM('SSMITHSS', 'S') FROM DUAL;
    ```

    Result: `SSMITH`

- **Lower:** Returns a character expression after converting uppercase character data to lowercase.
  - **Syntax:** `LOWER(char)`
  - **Example:**

    ```
    SELECT LOWER('DBMS') FROM DUAL;
    ```

    Result: `dbms`

- **Upper:** Returns a character expression with lowercase character data converted to uppercase.
  - **Syntax:** `UPPER(char)`
  - **Example:**

    ```
    SELECT UPPER('dbms') FROM DUAL;
    ```

    Result: `DBMS`

- **Length:** Returns the number of characters, rather than the number of bytes, of the given string expression, excluding trailing blanks.
    - **Syntax:** `LENGTH(char)`
    - **Example:**

    ```
    SELECT LENGTH('DATABASE') FROM DUAL;
    ```

    Result: `8`

- **Substr:** Returns part of a character, binary, text, or image expression.
    - **Syntax:** `SUBSTR(char, start_position, length)`
    - **Example:**

    ```
    SELECT SUBSTR('ABCDEFGHIJ', 3, 4) FROM DUAL;
    ```

    Result: `CDEF`

- **Instr:** The INSTR function searches string for substring and returns an integer indicating the position of the character in string that is the first character of this occurrence.
    - **Syntax:** `INSTR(string, substring, start_position, occurrence)`
    - **Example:**

    ```
    SELECT INSTR('CORPORATE FLOOR', 'OR', 3, 2) FROM DUAL;
    ```

    Result: `14`

COMPARISON OPERATORS

- **(=):** Checks if the values of two operands are equal.
- **(!=):** Checks if the values of two operands are not equal.
- **(< >):** Checks if the values of two operands are not equal.
- **(>):** Checks if the value of the left operand is greater than the value of the right operand.

- **(<):** Checks if the value of the left operand is less than the value of the right operand.
- **(>=):** Checks if the value of the left operand is greater than or equal to the value of the right operand.
- **(<=):** Checks if the value of the left operand is less than or equal to the value of the right operand.

SET OPERATORS

- **Union:** Returns all distinct rows selected by both queries.
- **Union All:** Returns all rows selected by either query, including duplicates.
- **Intersect:** Returns rows that are common to both queries.
- **Minus:** Returns all distinct rows selected by the first query but not by the second.

---

conclusion: hence we successfully

Executed of string, comparison, and set operations.