Conditional expressions in Standard SQL

Conditional expressions impose constraints on the evaluation order of their inputs. In essence, they are evaluated left to right, with short-circuiting, and only evaluate the output value that was chosen. In contrast, all inputs to regular functions are evaluated before calling the function. Short-circuiting in conditional expressions can be exploited for error handling or performance tuning.

CASE expr

```
CASE expr
  WHEN expr_to_match THEN result
  [ ... ]
  [ ELSE else_result ]
END
```

Description

Compares `expr` to `expr_to_match` of each successive `WHEN` clause and returns the first result where this comparison returns true. The remaining `WHEN` clauses and `else_result` are not evaluated. If the `expr = expr_to_match` comparison returns false or NULL for all `WHEN` clauses, returns `else_result` if present; if not present, returns NULL.

`expr` and `expr_to_match` can be any type. They must be implicitly coercible to a common supertype; equality comparisons are done on coerced values. There may be multiple `result` types. `result` and `else_result` expressions must be coercible to a common supertype.

Return Data Type

Supertype of `result`[, ...] and `else_result`.

Example

```
WITH Numbers AS
 (SELECT 90 as A, 2 as B UNION ALL
  SELECT 50, 8 UNION ALL
  SELECT 60, 6 UNION ALL
  SELECT 50, 10)
SELECT A, B,
  CASE A
    WHEN 90 THEN 'red'
    WHEN 50 THEN 'blue'
    ELSE 'green'
  END
  AS result
FROM Numbers

+------------------+
| A  | B  | result |
+------------------+
| 90 | 2  | red    |
| 50 | 8  | blue   |
| 60 | 6  | green  |
| 50 | 10 | blue   |
+------------------+
```

CASE

```
CASE
  WHEN condition THEN result
  [ ... ]
  [ ELSE else_result ]
  END
```

Description

Evaluates the condition of each successive `WHEN` clause and returns the first result where the condition is true; any remaining `WHEN` clauses and `else_result` are not evaluated. If all conditions are false or NULL, returns `else_result` if present; if not present, returns NULL.

`condition` must be a boolean expression. There may be multiple `result` types. `result` and `else_result` expressions must be implicitly coercible to a common supertype.

Return Data Type

Supertype of `result`[, ...] and `else_result`.

Example

```
WITH Numbers AS
 (SELECT 90 as A, 2 as B UNION ALL
  SELECT 50, 6 UNION ALL
  SELECT 20, 10)
SELECT A, B,
  CASE
    WHEN A > 60 THEN 'red'
    WHEN A > 30 THEN 'blue'
    ELSE 'green'
  END
  AS result
FROM Numbers

+------------------+
| A  | B  | result |
+------------------+
| 90 | 2  | red    |
| 50 | 6  | blue   |
| 20 | 10 | green  |
+------------------+
```

COALESCE

```
COALESCE(expr[, ...])
```

Description

Returns the value of the first non-null expression. The remaining expressions are not evaluated. An input expression can be any type. There may be multiple input expression types. All input expressions must be implicitly coercible to a common supertype.

Return Data Type

Supertype of `expr`[, ...].

Examples

```
SELECT COALESCE('A', 'B', 'C') as result

+--------+
| result |
+--------+
| A      |
+--------+
```

```
SELECT COALESCE(NULL, 'B', 'C') as result

+--------+
| result |
+--------+
| B      |
+--------+
```

IF

```
IF(expr, true_result, else_result)
```

Description

If `expr` is true, returns `true_result`, else returns `else_result`. `else_result` is not evaluated if `expr` is true. `true_result` is not evaluated if `expr` is false or NULL.

`expr` must be a boolean expression. `true_result` and `else_result` must be coercible to a common supertype.

Return Data Type

Supertype of `true_result` and `else_result`.

Example

```
WITH Numbers AS
 (SELECT 10 as A, 20 as B UNION ALL
  SELECT 50, 30 UNION ALL
  SELECT 60, 60)
SELECT
  A, B,
  IF( A<B, 'true', 'false') as result
FROM Numbers

+------------------+
| A  | B  | result |
+------------------+
| 10 | 20 | true   |
| 50 | 30 | false  |
| 60 | 60 | false  |
+------------------+
```

IFNULL

```
IFNULL(expr, null_result)
```

Description

If `expr` is NULL, return `null_result`. Otherwise, return `expr`. If `expr` is not NULL, `null_result` is not evaluated.

`expr` and `null_result` can be any type and must be implicitly coercible to a common supertype. Synonym for `COALESCE(expr, null_result)`.

Return Data Type

Supertype of `expr` or `null_result`.

Examples

```
SELECT IFNULL(NULL, 0) as result

+--------+
| result |
+--------+
| 0      |
+--------+
```

```
SELECT IFNULL(10, 0) as result

+--------+
| result |
+--------+
| 10     |
+--------+
```

NULLIF

```
NULLIF(expr, expr_to_match)
```

Description

Returns NULL if `expr = expr_to_match` is true, otherwise returns `expr`.

`expr` and `expr_to_match` must be implicitly coercible to a common supertype, and must be comparable.

Return Data Type

Supertype of `expr` and `expr_to_match`.

Example

```
SELECT NULLIF(0, 0) as result

+--------+
| result |
+--------+
| NULL   |
+--------+
SELECT NULLIF(10, 0) as result

+--------+
| result |
+--------+
| 10     |
+--------+
```