```
+------+------+----------+
```

Use of `GROUPING()` is subject to these limitations:

- Do not use subquery `GROUP BY` expressions as `GROUPING()` arguments because matching might fail. For example, matching fails for this query:

```
mysql> SELECT GROUPING((SELECT MAX(name) FROM t1))
       FROM t1
       GROUP BY (SELECT MAX(name) FROM t1) WITH ROLLUP;
ERROR 3580 (HY000): Argument #1 of GROUPING function is not in GROUP BY
```

- `GROUP BY` literal expressions should not be used within a `HAVING` clause as `GROUPING()` arguments. Due to differences between when the optimizer evaluates `GROUP BY` and `HAVING`, matching may succeed but `GROUPING()` evaluation does not produce the expected result. Consider this query:

```
SELECT a AS f1, 'w' AS f2
FROM t
GROUP BY f1, f2 WITH ROLLUP
HAVING GROUPING(f2) = 1;
```

  `GROUPING()` is evaluated earlier for the literal constant expression than for the `HAVING` clause as a whole and returns 0. To check whether a query such as this is affected, use `EXPLAIN` and look for `Impossible having` in the `Extra` column.

For more information about `WITH ROLLUP` and `GROUPING()`, see Section 12.20.2, “GROUP BY Modifiers”.

- `INET_ATON(expr)`

Given the dotted-quad representation of an IPv4 network address as a string, returns an integer that represents the numeric value of the address in network byte order (big endian). `INET_ATON()` returns `NULL` if it does not understand its argument.

```
mysql> SELECT INET_ATON('10.0.5.9');
        -> 167773449
```

For this example, the return value is calculated as $10\times256^3 + 0\times256^2 + 5\times256 + 9$.

`INET_ATON()` may or may not return a non-`NULL` result for short-form IP addresses (such as `'127.1'` as a representation of `'127.0.0.1'`). Because of this, `INET_ATON()`a should not be used for such addresses.

Note

To store values generated by `INET_ATON()`, use an `INT UNSIGNED` column rather than `INT`, which is signed. If you use a signed column, values corresponding to IP addresses for which the first octet is greater than 127 cannot be stored correctly. See Section 11.1.7, “Out-of-Range and Overflow Handling”.

- `INET_NTOA(expr)`

Given a numeric IPv4 network address in network byte order, returns the dotted-quad string representation of the address as a string in the connection character set. `INET_NTOA()` returns `NULL` if it does not understand its argument.

```
mysql> SELECT INET_NTOA(167773449);
        -> '10.0.5.9'
```