If a number is inserted into an exact type column (`DECIMAL` or integer), it is inserted with its exact value if it is within the column range and precision.

If the value has too many digits in the fractional part, rounding occurs and a note is generated. Rounding is done as described in Section 12.25.4, “Rounding Behavior”. Truncation due to rounding of the fractional part is not an error, even in strict mode.

If the value has too many digits in the integer part, it is too large (out of range) and is handled as follows:

- If strict mode is not enabled, the value is truncated to the nearest legal value and a warning is generated.
- If strict mode is enabled, an overflow error occurs.

For `DECIMAL` literals, in addition to the precision limit of 65 digits, there is a limit on how long the text of the literal can be. If the value exceeds approximately 80 characters, unexpected results can occur. For example:

```
mysql> SELECT
       CAST(00000000000000000000000000000000000000000000000000000000000000000000000000000020.01 AS DECIMA
+------------------+
| val              |
+------------------+
| 9999999999999.99 |
+------------------+
1 row in set, 2 warnings (0.00 sec)

mysql> SHOW WARNINGS;
+---------+------+-----------------------------------------+
| Level   | Code | Message                                 |
+---------+------+-----------------------------------------+
| Warning | 1292 | Truncated incorrect DECIMAL value: '20' |
| Warning | 1264 | Out of range value for column 'val' at row 1 |
+---------+------+-----------------------------------------+
2 rows in set (0.00 sec)
```

Underflow is not detected, so underflow handling is undefined.

For inserts of strings into numeric columns, conversion from string to number is handled as follows if the string has nonnumeric contents:

- A string that does not begin with a number cannot be used as a number and produces an error in strict mode, or a warning otherwise. This includes the empty string.
- A string that begins with a number can be converted, but the trailing nonnumeric portion is truncated. If the truncated portion contains anything other than spaces, this produces an error in strict mode, or a warning otherwise.

By default, division by zero produces a result of `NULL` and no warning. By setting the SQL mode appropriately, division by zero can be restricted.

With the `ERROR_FOR_DIVISION_BY_ZERO` SQL mode enabled, MySQL handles division by zero differently:

- If strict mode is not enabled, a warning occurs.
- If strict mode is enabled, inserts and updates involving division by zero are prohibited, and an error occurs.

In other words, inserts and updates involving expressions that perform division by zero can be treated as errors, but this requires `ERROR_FOR_DIVISION_BY_ZERO` in addition to strict mode.