`[INTERVAL] '+00:00'` or `'UTC'`. The precision of the `DATETIME` value returned by the cast can be specified up to 6 decimal places, if desired. The `ARRAY` keyword is not supported with this construct.

`TIMESTAMP` values inserted into a table using a timezone offset are also supported. Use of `AT TIME ZONE` is not supported for `CONVERT()` or any other MySQL function or construct.

For further information and examples, see the description of the `CAST()` function.

- **Dump file output synchronization.** MySQL 8.0.22 and later supports periodic synchronization when writing to files by `SELECT INTO DUMPFILE` and `SELECT INTO OUTFILE` statements. This can be enabled by setting the `select_into_disk_sync` system variable to `ON`; the size of the write buffer is determined by the value set for `select_into_buffer_size`; the default is 131072 (2^{17}) bytes.

  In addition, an optional delay following synchronization to disk can be set using `select_into_disk_sync_delay`; the default is no delay (0 milliseconds).

  For more information, see the descriptions of the variables referenced previously in this item.

- **Single preparation of statements.** As of MySQL 8.0.22, a prepared statement is prepared a single time, rather than once each time it is executed. This is done when executing `PREPARE`. This is also true for any statement inside a stored procedure; the statement is prepared once, when the stored procedure is first executed.

  One result of this change is that the fashion in which dynamic parameters used in prepared statements are resolved is also changed in the ways listed here:

  - A prepared statement parameter is assigned a data type when the statement is prepared; the type persists for each subsequent execution of the statement (unless the statement is reprepared; see following).

    Using a different data type for a given parameter or user variable within a prepared statement for executions of the statement subsequent to the first execution may cause the statement to be reprepared; for this reason, it is advisable to use the same data type for a given parameter when re-executing a prepared statement.

  - The following constructs employing window functions are no longer accepted, in order to align with the SQL standard:

    - `NTILE(NULL)`
    - `NTH_VALUE(expr, NULL)`
    - `LEAD(expr, nn)` and `LAG(expr, nn)`, where *nn* is a negative number

    This facilitates greater compliance with the SQL standard. See the individual function descriptions for further details.

  - A user variable referenced within a prepared statement now has its data type determined when the statement is prepared; the type persists for each subsequent execution of the statement.

  - A user variable referenced by a statement occurring within a stored procedure now has its data type determined the first time the statement is executed; the type persists for any subsequent invocation of the containing stored procedure.

  - When executing a prepared statement of the form `SELECT expr1, expr2, ... FROM table ORDER BY ?`, passing an integer value *N* for the parameter no longer causes ordering of the results by