- For numeric types, the default is 0, with the exception that for integer or floating-point types declared with the AUTO_INCREMENT attribute, the default is the next value in the sequence.
- For date and time types other than TIMESTAMP, the default is the appropriate “zero” value for the type. This is also true for TIMESTAMP if the explicit_defaults_for_timestamp system variable is enabled (see Section 5.1.8, “Server System Variables”). Otherwise, for the first TIMESTAMP column in a table, the default value is the current date and time. See Section 11.2, “Date and Time Data Types”.
- For string types other than ENUM, the default value is the empty string. For ENUM, the default is the first enumeration value.

11.7 Data Type Storage Requirements

The storage requirements for table data on disk depend on several factors. Different storage engines represent data types and store raw data differently. Table data might be compressed, either for a column or an entire row, complicating the calculation of storage requirements for a table or column.

Despite differences in storage layout on disk, the internal MySQL APIs that communicate and exchange information about table rows use a consistent data structure that applies across all storage engines.

This section includes guidelines and information for the storage requirements for each data type supported by MySQL, including the internal format and size for storage engines that use a fixed-size representation for data types. Information is listed by category or storage engine.

The internal representation of a table has a maximum row size of 65,535 bytes, even if the storage engine is capable of supporting larger rows. This figure excludes BLOB or TEXT columns, which contribute only 9 to 12 bytes toward this size. For BLOB and TEXT data, the information is stored internally in a different area of memory than the row buffer. Different storage engines handle the allocation and storage of this data in different ways, according to the method they use for handling the corresponding types. For more information, see Chapter 16, *Alternative Storage Engines*, and Section 8.4.7, “Limits on Table Column Count and Row Size”.

InnoDB Table Storage Requirements

See Section 15.10, “InnoDB Row Formats” for information about storage requirements for InnoDB tables.

NDB Table Storage Requirements

Important

NDB tables use *4-byte alignment*; all NDB data storage is done in multiples of 4 bytes. Thus, a column value that would typically take 15 bytes requires 16 bytes in an NDB table. For example, in NDB tables, the TINYINT, SMALLINT, MEDIUMINT,