INITIAL_SIZE (described in the following item) is 256 MB and whose EXTENT_SIZE is 128M has just two extents, and so can be used to store data from at most two different disk data table partitions.

You can see how many extents remain free in a given data file by querying the INFORMATION_SCHEMA.FILES table, and so derive an estimate for how much space remains free in the file. For further discussion and examples, see Section 26.3.15, "The INFORMATION_SCHEMA FILES Table".

- INITIAL_SIZE: This option is specific to NDB, and is not supported by InnoDB, where it fails with an error.

  The INITIAL_SIZE parameter sets the total size in bytes of the data file that was specific using ADD DATAFILE. Once this file has been created, its size cannot be changed; however, you can add more data files to the tablespace using ALTER TABLESPACE ... ADD DATAFILE.

  INITIAL_SIZE is optional; its default value is 134217728 (128 MB).

  On 32-bit systems, the maximum supported value for INITIAL_SIZE is 4294967296 (4 GB).

- AUTOEXTEND_SIZE: Ignored by MySQL prior to MySQL 8.0.23; From MySQL 8.0.23, defines the amount by which InnoDB extends the size of the tablespace when it becomes full. The setting must be a multiple of 4MB. The default setting is 0, which causes the tablespace to be extended according to the implicit default behavior. For more information, see Section 15.6.3.9, "Tablespace AUTOEXTEND_SIZE Configuration".

  Has no effect in any release of MySQL NDB Cluster 8.0, regardless of the storage engine used.

- MAX_SIZE: Currently ignored by MySQL; reserved for possible future use. Has no effect in any release of MySQL 8.0 or MySQL NDB Cluster 8.0, regardless of the storage engine used.

- NODEGROUP: Currently ignored by MySQL; reserved for possible future use. Has no effect in any release of MySQL 8.0 or MySQL NDB Cluster 8.0, regardless of the storage engine used.

- WAIT: Currently ignored by MySQL; reserved for possible future use. Has no effect in any release of MySQL 8.0 or MySQL NDB Cluster 8.0, regardless of the storage engine used.

- COMMENT: Currently ignored by MySQL; reserved for possible future use. Has no effect in any release of MySQL 8.0 or MySQL NDB Cluster 8.0, regardless of the storage engine used.

- The ENCRYPTION clause enables or disables page-level data encryption for an InnoDB general tablespace. Encryption support for general tablespaces was introduced in MySQL 8.0.13.

  As of MySQL 8.0.16, if the ENCRYPTION clause is not specified, the default_table_encryption setting controls whether encryption is enabled. The ENCRYPTION clause overrides the default_table_encryption setting. However, if the table_encryption_privilege_check variable is enabled, the TABLE_ENCRYPTION_ADMIN privilege is required to use an ENCRYPTION clause setting that differs from the default_table_encryption setting.

  A keyring plugin must be installed and configured before an encryption-enabled tablespace can be created.

  When a general tablespace is encrypted, all tables residing in the tablespace are encrypted. Likewise, a table created in an encrypted tablespace is encrypted.

  For more information, see Section 15.13, "InnoDB Data-at-Rest Encryption"

- ENGINE: Defines the storage engine which uses the tablespace, where *engine_name* is the name of the storage engine. Currently, only the InnoDB storage engine is supported by standard MySQL