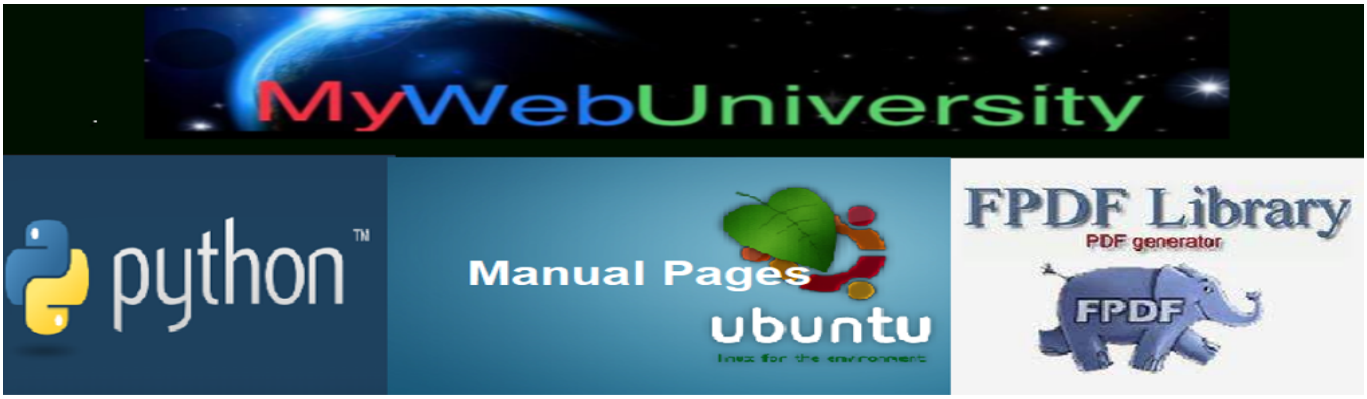

Rocky Enterprise Linux 9.2 Manual Pages on command 'myisamchk.1'

$ man myisamchk.1

MYISAMCHK(1) MySQL Database System MYISAMCHK(1)

NAME

myisamchk - MyISAM table-maintenance utility

SYNOPSIS

myisamchk [options] tbl_name ...

DESCRIPTION

The myisamchk utility gets information about your database tables or checks, repairs, or optimizes them. myisamchk works with MyISAM tables (tables that have .MYD and .MYI files for storing data and indexes).

You can also use the CHECK TABLE and REPAIR TABLE statements to check and repair MyISAM tables. See Section 13.7.3.2, ?CHECK TABLE Statement?, and Section 13.7.3.5, ?REPAIR TABLE Statement?.

The use of myisamchk with partitioned tables is not supported.

Caution

It is best to make a backup of a table before performing a table repair operation; under some circumstances the operation might cause data loss. Possible causes include but are not limited to file system errors.

Invoke myisamchk like this:

myisamchk [options] tbl_name ...

The options specify what you want myisamchk to do. They are described in the following sections. You can also get a list of options by invoking myisamchk --help.

With no options, myisamchk simply checks your table as the default operation. To get more information or to tell myisamchk to take corrective action, specify options as described

in the following discussion.

tbl_name is the database table you want to check or repair. If you run myisamchk somewhere other than in the database directory, you must specify the path to the database directory, because myisamchk has no idea where the database is located. In fact, myisamchk does not actually care whether the files you are working on are located in a database directory. You can copy the files that correspond to a database table into some other location and perform recovery operations on them there.

You can name several tables on the myisamchk command line if you wish. You can also specify a table by naming its index file (the file with the .MYI suffix). This enables you to specify all tables in a directory by using the pattern *.MYI. For example, if you are in a database directory, you can check all the MyISAM tables in that directory like this:

```
myisamchk *.MYI
```

If you are not in the database directory, you can check all the tables there by specifying the path to the directory:

```
myisamchk /path/to/database_dir/*.MYI
```

You can even check all tables in all databases by specifying a wildcard with the path to the MySQL data directory:

```
myisamchk /path/to/datadir/*/*.MYI
```

The recommended way to quickly check all MyISAM tables is:

```
myisamchk --silent --fast /path/to/datadir/*/*.MYI
```

If you want to check all MyISAM tables and repair any that are corrupted, you can use the following command:

```
myisamchk --silent --force --fast --update-state \
        --key_buffer_size=64M --myisam_sort_buffer_size=64M \
        --read_buffer_size=1M --write_buffer_size=1M \
        /path/to/datadir/*/*.MYI
```

This command assumes that you have more than 64MB free. For more information about memory allocation with myisamchk, see the section called ?MYISAMCHK MEMORY USAGE?.

For additional information about using myisamchk, see Section 7.6, ?MyISAM Table Maintenance and Crash Recovery?.

Important

You must ensure that no other program is using the tables while you are running myisamchk. The most effective means of doing so is to shut down the MySQL server while

running myisamchk, or to lock all tables that myisamchk is being used on.

Otherwise, when you run myisamchk, it may display the following error message:

warning: clients are using or haven't closed the table properly

This means that you are trying to check a table that has been updated by another program (such as the mysqld server) that hasn't yet closed the file or that has died without closing the file properly, which can sometimes lead to the corruption of one or more MyISAM tables.

If mysqld is running, you must force it to flush any table modifications that are still buffered in memory by using FLUSH TABLES. You should then ensure that no one is using the tables while you are running myisamchk

However, the easiest way to avoid this problem is to use CHECK TABLE instead of myisamchk to check tables. See Section 13.7.3.2, ?CHECK TABLE Statement?.

myisamchk supports the following options, which can be specified on the command line or in the [myisamchk] group of an option file. For information about option files used by MySQL programs, see Section 4.2.2.2, ?Using Option Files?.

MYISAMCHK GENERAL OPTIONS

The options described in this section can be used for any type of table maintenance operation performed by myisamchk. The sections following this one describe options that pertain only to specific operations, such as table checking or repairing.

? --help, -?

??????????????????????????????

?Command-Line Format ? --help ?

??????????????????????????????

Display a help message and exit. Options are grouped by type of operation.

? --HELP, -H

??????????????????????????????

?Command-Line Format ? --HELP ?

??????????????????????????????

Display a help message and exit. Options are presented in a single list.

? --debug=debug_options, -# debug_options

??

?Command-Line Format ? --debug[=debug_options] ?

??

?Type ? String ?

??

?Default Value ? d:t:o,/tmp/myisamchk.trace ?

??

Write a debugging log. A typical debug_options string is d:t:o,file_name. The default is d:t:o,/tmp/myisamchk.trace.

This option is available only if MySQL was built using WITH_DEBUG. MySQL release binaries provided by Oracle are not built using this option.

? --defaults-extra-file=file_name

??

?Command-Line Format ? --defaults-extra-file=file_name ?

??

?Type ? File name ?

??

Read this option file after the global option file but (on Unix) before the user option file. If the file does not exist or is otherwise inaccessible, an error occurs. If file_name is not an absolute path name, it is interpreted relative to the current directory.

For additional information about this and other option-file options, see Section 4.2.2.3, ?Command-Line Options that Affect Option-File Handling?.

? --defaults-file=file_name

??

?Command-Line Format ? --defaults-file=file_name ?

??

?Type ? File name ?

??

Use only the given option file. If the file does not exist or is otherwise inaccessible, an error occurs. If file_name is not an absolute path name, it is interpreted relative to the current directory.

For additional information about this and other option-file options, see Section 4.2.2.3, ?Command-Line Options that Affect Option-File Handling?.

? --defaults-group-suffix=str

??

?Command-Line Format ? --defaults-group-suffix=str ?

??

?Type ? String ?

??

Read not only the usual option groups, but also groups with the usual names and a suffix of str. For example, myisamchk normally reads the [myisamchk] group. If this option is given as --defaults-group-suffix=_other, myisamchk also reads the [myisamchk_other] group.

For additional information about this and other option-file options, see Section 4.2.2.3, ?Command-Line Options that Affect Option-File Handling?.

? --no-defaults

??????????????????????????????????????

?Command-Line Format ? --no-defaults ?

??????????????????????????????????????

Do not read any option files. If program startup fails due to reading unknown options from an option file, --no-defaults can be used to prevent them from being read.

The exception is that the .mylogin.cnf file is read in all cases, if it exists. This permits passwords to be specified in a safer way than on the command line even when --no-defaults is used. To create .mylogin.cnf, use the mysql_config_editor utility. See mysql_config_editor(1).

For additional information about this and other option-file options, see Section 4.2.2.3, ?Command-Line Options that Affect Option-File Handling?.

? --print-defaults

??????????????????????????????????????

?Command-Line Format ? --print-defaults ?

??????????????????????????????????????

Print the program name and all options that it gets from option files.

For additional information about this and other option-file options, see Section 4.2.2.3, ?Command-Line Options that Affect Option-File Handling?.

? --silent, -s

??????????????????????????????

?Command-Line Format ? --silent ?

??????????????????????????????

Silent mode. Write output only when errors occur. You can use -s twice (-ss) to make myisamchk very silent.

? --verbose, -v

??????????????????????????????????

?Command-Line Format ? --verbose ?

??????????????????????????????????

Verbose mode. Print more information about what the program does. This can be used with -d and -e. Use -v multiple times (-vv, -vvv) for even more output.

? --version, -V

??????????????????????????????????

?Command-Line Format ? --version ?

??????????????????????????????????

Display version information and exit.

? --wait, -w

??????????????????????????????????

?Command-Line Format ? --wait ?

??????????????????????????????????

?Type ? Boolean ?

??????????????????????????????????

?Default Value ? false ?

??????????????????????????????????

Instead of terminating with an error if the table is locked, wait until the table is unlocked before continuing. If you are running mysqld with external locking disabled, the table can be locked only by another myisamchk command.

You can also set the following variables by using --var_name=value syntax:

??

?Variable ? Default Value ?

??

?decode_bits ? 9 ?

??

?ft_max_word_len ? version-dependent ?

??

?ft_min_word_len ? 4 ?

??

?ft_stopword_file ? built-in list ?

??

?key_buffer_size ? 523264 ?

??

?myisam_block_size ? 1024 ?

??

?myisam_sort_key_blocks ? 16 ?

??

?read_buffer_size ? 262136 ?

??

?sort_buffer_size ? 2097144 ?

??

?sort_key_blocks ? 16 ?

??

?stats_method ? nulls_unequal ?

??

?write_buffer_size ? 262136 ?

??

The possible myisamchk variables and their default values can be examined with myisamchk --help:

myisam_sort_buffer_size is used when the keys are repaired by sorting keys, which is the normal case when you use --recover. sort_buffer_size is a deprecated synonym for myisam_sort_buffer_size.

key_buffer_size is used when you are checking the table with --extend-check or when the keys are repaired by inserting keys row by row into the table (like when doing normal inserts). Repairing through the key buffer is used in the following cases:

? You use --safe-recover.

? The temporary files needed to sort the keys would be more than twice as big as when creating the key file directly. This is often the case when you have large key values for CHAR, VARCHAR, or TEXT columns, because the sort operation needs to store the complete key values as it proceeds. If you have lots of temporary space and you can force myisamchk to repair by sorting, you can use the --sort-recover option.

Repairing through the key buffer takes much less disk space than using sorting, but is also much slower.

If you want a faster repair, set the key_buffer_size and myisam_sort_buffer_size variables to about 25% of your available memory. You can set both variables to large values, because only one of them is used at a time.

myisam_block_size is the size used for index blocks.

stats_method influences how NULL values are treated for index statistics collection when the --analyze option is given. It acts like the myisam_stats_method system variable. For more information, see the description of myisam_stats_method in Section 5.1.8, ?Server System Variables?, and Section 8.3.8, ?InnoDB and MyISAM Index Statistics Collection?.

ft_min_word_len and ft_max_word_len indicate the minimum and maximum word length for FULLTEXT indexes on MyISAM tables. ft_stopword_file names the stopword file. These need to be set under the following circumstances.

If you use myisamchk to perform an operation that modifies table indexes (such as repair or analyze), the FULLTEXT indexes are rebuilt using the default full-text parameter values for minimum and maximum word length and the stopword file unless you specify otherwise. This can result in queries failing.

The problem occurs because these parameters are known only by the server. They are not stored in MyISAM index files. To avoid the problem if you have modified the minimum or maximum word length or the stopword file in the server, specify the same ft_min_word_len, ft_max_word_len, and ft_stopword_file values to myisamchk that you use for mysqld. For example, if you have set the minimum word length to 3, you can repair a table with myisamchk like this:

```
myisamchk --recover --ft_min_word_len=3 tbl_name.MYI
```

To ensure that myisamchk and the server use the same values for full-text parameters, you can place each one in both the [mysqld] and [myisamchk] sections of an option file:

```
[mysqld]
ft_min_word_len=3
[myisamchk]
ft_min_word_len=3
```

An alternative to using myisamchk is to use the REPAIR TABLE, ANALYZE TABLE, OPTIMIZE TABLE, or ALTER TABLE. These statements are performed by the server, which knows the proper full-text parameter values to use.

MYISAMCHK CHECK OPTIONS

myisamchk supports the following options for table checking operations:

? --check, -c

??????????????????????????????

?Command-Line Format ? --check ?

??????????????????????????????

Check the table for errors. This is the default operation if you specify no option

that selects an operation type explicitly.

? --check-only-changed, -C

??

?Command-Line Format ? --check-only-changed ?

??

Check only tables that have changed since the last check.

? --extend-check, -e

??????????????????????????????????????

?Command-Line Format ? --extend-check ?

??????????????????????????????????????

Check the table very thoroughly. This is quite slow if the table has many indexes.

This option should only be used in extreme cases. Normally, myisamchk or myisamchk

--medium-check should be able to determine whether there are any errors in the table.

If you are using --extend-check and have plenty of memory, setting the key_buffer_size

variable to a large value helps the repair operation run faster.

See also the description of this option under table repair options.

For a description of the output format, see the section called ?OBTAINING TABLE

INFORMATION WITH MYISAMCHK?.

? --fast, -F

??????????????????????????????

?Command-Line Format ? --fast ?

??????????????????????????????

Check only tables that haven't been closed properly.

? --force, -f

??????????????????????????????

?Command-Line Format ? --force ?

??????????????????????????????

Do a repair operation automatically if myisamchk finds any errors in the table. The repair type is the same as that specified with the --recover or -r option.

? --information, -i

??????????????????????????????????????

?Command-Line Format ? --information ?

??????????????????????????????????????

Print informational statistics about the table that is checked.

? --medium-check, -m

??????????????????????????????????????

?Command-Line Format ? --medium-check ?

??????????????????????????????????????

Do a check that is faster than an --extend-check operation. This finds only 99.99% of all errors, which should be good enough in most cases.

? --read-only, -T

??????????????????????????????????

?Command-Line Format ? --read-only ?

??????????????????????????????????

Do not mark the table as checked. This is useful if you use myisamchk to check a table that is in use by some other application that does not use locking, such as mysqld when run with external locking disabled.

? --update-state, -U

??????????????????????????????????

?Command-Line Format ? --update-state ?

??????????????????????????????????

Store information in the .MYI file to indicate when the table was checked and whether the table crashed. This should be used to get full benefit of the --check-only-changed option, but you shouldn't use this option if the mysqld server is using the table and you are running it with external locking disabled.

MYISAMCHK REPAIR OPTIONS

myisamchk supports the following options for table repair operations (operations performed when an option such as --recover or --safe-recover is given):

? --backup, -B

??????????????????????????????????

?Command-Line Format ? --backup ?

??????????????????????????????????

Make a backup of the .MYD file as file_name-time.BAK

? --character-sets-dir=dir_name

??

?Command-Line Format ? --character-sets-dir=path ?

??

?Type ? String ?

??

?Default Value ? [none] ?

??

The directory where character sets are installed. See Section 10.15, ?Character Set Configuration?.

? --correct-checksum

??

?Command-Line Format ? --correct-checksum ?

??

Correct the checksum information for the table.

? --data-file-length=len, -D len

??

?Command-Line Format ? --data-file-length=len ?

??

?Type ? Numeric ?

??

The maximum length of the data file (when re-creating data file when it is ?full?).

? --extend-check, -e

????????????????????????????????????

?Command-Line Format ? --extend-check ?

????????????????????????????????????

Do a repair that tries to recover every possible row from the data file. Normally, this also finds a lot of garbage rows. Do not use this option unless you are desperate.

See also the description of this option under table checking options.

For a description of the output format, see the section called ?OBTAINING TABLE INFORMATION WITH MYISAMCHK?.

? --force, -f

??????????????????????????????

?Command-Line Format ? --force ?

??????????????????????????????

Overwrite old intermediate files (files with names like tbl_name.TMD) instead of aborting.

? --keys-used=val, -k val

??????????????????????????????????????

?Command-Line Format ? --keys-used=val ?

??????????????????????????????????????

?Type ? Numeric ?

??????????????????????????????????????

For myisamchk, the option value is a bit value that indicates which indexes to update. Each binary bit of the option value corresponds to a table index, where the first index is bit 0. An option value of 0 disables updates to all indexes, which can be used to get faster inserts. Deactivated indexes can be reactivated by using myisamchk -r.

? --max-record-length=len

??

?Command-Line Format ? --max-record-length=len ?

??

?Type ? Numeric ?

??

Skip rows larger than the given length if myisamchk cannot allocate memory to hold them.

? --parallel-recover, -p

??????????????????????????????????????

?Command-Line Format ? --parallel-recover ?

??????????????????????????????????????

Note

This option is deprecated in MySQL 8.0.28 and removed in MySQL 8.0.30.

Use the same technique as -r and -n, but create all the keys in parallel, using different threads. This is beta-quality code. Use at your own risk!

? --quick, -q

??????????????????????????????

?Command-Line Format ? --quick ?

??????????????????????????????

Achieve a faster repair by modifying only the index file, not the data file. You can specify this option twice to force myisamchk to modify the original data file in case of duplicate keys.

? --recover, -r

??????????????????????????????

?Command-Line Format ? --recover ?

??????????????????????????????

Do a repair that can fix almost any problem except unique keys that are not unique (which is an extremely unlikely error with MyISAM tables). If you want to recover a table, this is the option to try first. You should try --safe-recover only if myisamchk reports that the table cannot be recovered using --recover. (In the unlikely case that --recover fails, the data file remains intact.)

If you have lots of memory, you should increase the value of myisam_sort_buffer_size.

? --safe-recover, -o

??????????????????????????????????????

?Command-Line Format ? --safe-recover ?

??????????????????????????????????????

Do a repair using an old recovery method that reads through all rows in order and updates all index trees based on the rows found. This is an order of magnitude slower than --recover, but can handle a couple of very unlikely cases that --recover cannot. This recovery method also uses much less disk space than --recover. Normally, you should repair first using --recover, and then with --safe-recover only if --recover fails.

If you have lots of memory, you should increase the value of key_buffer_size.

? --set-collation=name

??????????????????????????????????????

?Command-Line Format ? --set-collation=name ?

??

?Type ? String ?

??

Specify the collation to use for sorting table indexes. The character set name is implied by the first part of the collation name.

? --sort-recover, -n

????????????????????????????????????

?Command-Line Format ? --sort-recover ?

????????????????????????????????????

Force myisamchk to use sorting to resolve the keys even if the temporary files would be very large.

? --tmpdir=dir_name, -t dir_name

????????????????????????????????????

?Command-Line Format ? --tmpdir=dir_name ?

????????????????????????????????????

?Type ? Directory name ?

????????????????????????????????????

The path of the directory to be used for storing temporary files. If this is not set, myisamchk uses the value of the TMPDIR environment variable. --tmpdir can be set to a list of directory paths that are used successively in round-robin fashion for creating temporary files. The separator character between directory names is the colon (:) on Unix and the semicolon (;) on Windows.

? --unpack, -u

????????????????????????????????

?Command-Line Format ? --unpack ?

????????????????????????????????

Unpack a table that was packed with myisampack.

OTHER MYISAMCHK OPTIONS

myisamchk supports the following options for actions other than table checks and repairs:

? --analyze, -a

????????????????????????????????

?Command-Line Format ? --analyze ?

????????????????????????????????

Analyze the distribution of key values. This improves join performance by enabling the join optimizer to better choose the order in which to join the tables and which indexes it should use. To obtain information about the key distribution, use a myisamchk --description --verbose tbl_name command or the SHOW INDEX FROM tbl_name statement.

? --block-search=offset, -b offset

??

?Command-Line Format ? --block-search=offset ?

??

?Type ? Numeric ?

??

Find the record that a block at the given offset belongs to.

? --description, -d

????????????????????????????????????

?Command-Line Format ? --description ?

????????????????????????????????????

Print some descriptive information about the table. Specifying the --verbose option once or twice produces additional information. See the section called ?OBTAINING TABLE INFORMATION WITH MYISAMCHK?.

? --set-auto-increment[=value], -A[value] Force AUTO_INCREMENT numbering for new records to start at the given value (or higher, if there are existing records with AUTO_INCREMENT values this large). If value is not specified, AUTO_INCREMENT numbers for new records begin with the largest value currently in the table, plus one.

? --sort-index, -S

????????????????????????????????????

?Command-Line Format ? --sort-index ?

????????????????????????????????????

Sort the index tree blocks in high-low order. This optimizes seeks and makes table scans that use indexes faster.

? --sort-records=N, -R N

??

?Command-Line Format ? --sort-records=# ?

??????????????????????????????????????

?Type ? Numeric ?

??????????????????????????????????????

Sort records according to a particular index. This makes your data much more localized and may speed up range-based SELECT and ORDER BY operations that use this index. (The first time you use this option to sort a table, it may be very slow.) To determine a table's index numbers, use SHOW INDEX, which displays a table's indexes in the same order that myisamchk sees them. Indexes are numbered beginning with 1.

If keys are not packed (PACK_KEYS=0), they have the same length, so when myisamchk sorts and moves records, it just overwrites record offsets in the index. If keys are packed (PACK_KEYS=1), myisamchk must unpack key blocks first, then re-create indexes and pack the key blocks again. (In this case, re-creating indexes is faster than updating offsets for each index.)

OBTAINING TABLE INFORMATION WITH MYISAMCHK

To obtain a description of a MyISAM table or statistics about it, use the commands shown here. The output from these commands is explained later in this section.

? myisamchk -d tbl_name

Runs myisamchk in ?describe mode? to produce a description of your table. If you start the MySQL server with external locking disabled, myisamchk may report an error for a table that is updated while it runs. However, because myisamchk does not change the table in describe mode, there is no risk of destroying data.

? myisamchk -dv tbl_name

Adding -v runs myisamchk in verbose mode so that it produces more information about the table. Adding -v a second time produces even more information.

? myisamchk -eis tbl_name

Shows only the most important information from a table. This operation is slow because it must read the entire table.

? myisamchk -eiv tbl_name

This is like -eis, but tells you what is being done.

The tbl_name argument can be either the name of a MyISAM table or the name of its index file, as described in myisamchk(1). Multiple tbl_name arguments can be given.

Suppose that a table named person has the following structure. (The MAX_ROWS table option is included so that in the example output from myisamchk shown later, some values are

smaller and fit the output format more easily.)

```
CREATE TABLE person
(
  id         INT NOT NULL AUTO_INCREMENT,
  last_name  VARCHAR(20) NOT NULL,
  first_name VARCHAR(20) NOT NULL,
  birth      DATE,
  death      DATE,
  PRIMARY KEY (id),
  INDEX (last_name, first_name),
  INDEX (birth)
) MAX_ROWS = 1000000 ENGINE=MYISAM;
```

Suppose also that the table has these data and index file sizes:

```
-rw-rw----  1 mysql  mysql  9347072 Aug 19 11:47 person.MYD
-rw-rw----  1 mysql  mysql  6066176 Aug 19 11:47 person.MYI
```

Example of myisamchk -dvv output:

```
MyISAM file:         person
Record format:       Packed
Character set:       utf8mb4_0900_ai_ci (255)
File-version:        1
Creation time:       2017-03-30 21:21:30
Status:              checked,analyzed,optimized keys,sorted index pages
Auto increment key:              1  Last value:              306688
Data records:             306688  Deleted blocks:                0
Datafile parts:           306688  Deleted data:                  0
Datafile pointer (bytes):        4  Keyfile pointer (bytes):       3
Datafile length:          9347072  Keyfile length:          6066176
Max datafile length:    4294967294  Max keyfile length:   17179868159
Recordlength:                   54
table description:
Key Start Len Index   Type                     Rec/key         Root  Blocksize
1   2     4   unique  long                     1               1024
2   6     80  multip. varchar prefix               0               1024
```

87 80 varchar 0

3 168 3 multip. uint24 NULL 0 1024

Field Start Length Nullpos Nullbit Type

1 1 1

2 2 4 no zeros

3 6 81 varchar

4 87 81 varchar

5 168 3 1 1 no zeros

6 171 3 1 2 no zeros

Explanations for the types of information myisamchk produces are given here. ?Keyfile? refers to the index file. ?Record? and ?row? are synonymous, as are ?field? and ?column.? The initial part of the table description contains these values:

? MyISAM file

Name of the MyISAM (index) file.

? Record format

The format used to store table rows. The preceding examples use Fixed length. Other possible values are Compressed and Packed. (Packed corresponds to what SHOW TABLE STATUS reports as Dynamic.)

? Chararacter set

The table default character set.

? File-version

Version of MyISAM format. Always 1.

? Creation time

When the data file was created.

? Recover time

When the index/data file was last reconstructed.

? Status

Table status flags. Possible values are crashed, open, changed, analyzed, optimized keys, and sorted index pages.

? Auto increment key, Last value

The key number associated the table's AUTO_INCREMENT column, and the most recently generated value for this column. These fields do not appear if there is no such column.

? Data records

The number of rows in the table.

? Deleted blocks

How many deleted blocks still have reserved space. You can optimize your table to minimize this space. See Section 7.6.4, ?MyISAM Table Optimization?.

? Datafile parts

For dynamic-row format, this indicates how many data blocks there are. For an optimized table without fragmented rows, this is the same as Data records.

? Deleted data

How many bytes of unreclaimed deleted data there are. You can optimize your table to minimize this space. See Section 7.6.4, ?MyISAM Table Optimization?.

? Datafile pointer

The size of the data file pointer, in bytes. It is usually 2, 3, 4, or 5 bytes. Most tables manage with 2 bytes, but this cannot be controlled from MySQL yet. For fixed tables, this is a row address. For dynamic tables, this is a byte address.

? Keyfile pointer

The size of the index file pointer, in bytes. It is usually 1, 2, or 3 bytes. Most tables manage with 2 bytes, but this is calculated automatically by MySQL. It is always a block address.

? Max datafile length

How long the table data file can become, in bytes.

? Max keyfile length

How long the table index file can become, in bytes.

? Recordlength

How much space each row takes, in bytes.

The table description part of the output includes a list of all keys in the table. For each key, myisamchk displays some low-level information:

? Key

This key's number. This value is shown only for the first column of the key. If this value is missing, the line corresponds to the second or later column of a multiple-column key. For the table shown in the example, there are two table description lines for the second index. This indicates that it is a multiple-part index with two parts.

? Start

Where in the row this portion of the index starts.

? Len

How long this portion of the index is. For packed numbers, this should always be the full length of the column. For strings, it may be shorter than the full length of the indexed column, because you can index a prefix of a string column. The total length of a multiple-part key is the sum of the Len values for all key parts.

? Index

Whether a key value can exist multiple times in the index. Possible values are unique or multip. (multiple).

? Type

What data type this portion of the index has. This is a MyISAM data type with the possible values packed, stripped, or empty.

? Root

Address of the root index block.

? Blocksize

The size of each index block. By default this is 1024, but the value may be changed at compile time when MySQL is built from source.

? Rec/key

This is a statistical value used by the optimizer. It tells how many rows there are per value for this index. A unique index always has a value of 1. This may be updated after a table is loaded (or greatly changed) with myisamchk -a. If this is not updated at all, a default value of 30 is given.

The last part of the output provides information about each column:

? Field

The column number.

? Start

The byte position of the column within table rows.

? Length

The length of the column in bytes.

? Nullpos, Nullbit

For columns that can be NULL, MyISAM stores NULL values as a flag in a byte. Depending on how many nullable columns there are, there can be one or more bytes used for this

purpose. The Nullpos and Nullbit values, if nonempty, indicate which byte and bit contains that flag indicating whether the column is NULL.

The position and number of bytes used to store NULL flags is shown in the line for field 1. This is why there are six Field lines for the person table even though it has only five columns.

? Type

The data type. The value may contain any of the following descriptors:

? constant

All rows have the same value.

? no endspace

Do not store endspace.

? no endspace, not_always

Do not store endspace and do not do endspace compression for all values.

? no endspace, no empty

Do not store endspace. Do not store empty values.

? table-lookup

The column was converted to an ENUM.

? zerofill(N)

The most significant N bytes in the value are always 0 and are not stored.

? no zeros

Do not store zeros.

? always zero

Zero values are stored using one bit.

? Huff tree

The number of the Huffman tree associated with the column.

? Bits

The number of bits used in the Huffman tree.

The Huff tree and Bits fields are displayed if the table has been compressed with myisampack. See myisampack(1), for an example of this information.

Example of myisamchk -eiv output:

```
Checking MyISAM file: person
Data records:  306688   Deleted blocks:       0
- check file-size
```

- check record delete-chain

No recordlinks

- check key delete-chain

block_size 1024:

- check index reference

- check data record references index: 1

Key: 1: Keyblocks used: 98% Packed: 0% Max levels: 3

- check data record references index: 2

Key: 2: Keyblocks used: 99% Packed: 97% Max levels: 3

- check data record references index: 3

Key: 3: Keyblocks used: 98% Packed: -14% Max levels: 3

Total: Keyblocks used: 98% Packed: 89%

- check records and index references

*** LOTS OF ROW NUMBERS DELETED ***

Records: 306688 M.recordlength: 25 Packed: 83%

Recordspace used: 97% Empty space: 2% Blocks/Record: 1.00

Record blocks: 306688 Delete blocks: 0

Record data: 7934464 Deleted data: 0

Lost space: 256512 Linkdata: 1156096

User time 43.08, System time 1.68

Maximum resident set size 0, Integral resident set size 0

Non-physical pagefaults 0, Physical pagefaults 0, Swaps 0

Blocks in 0 out 7, Messages in 0 out 0, Signals 0

Voluntary context switches 0, Involuntary context switches 0

Maximum memory usage: 1046926 bytes (1023k)

myisamchk -eiv output includes the following information:

? Data records

The number of rows in the table.

? Deleted blocks

How many deleted blocks still have reserved space. You can optimize your table to minimize this space. See Section 7.6.4, ?MyISAM Table Optimization?.

? Key

The key number.

? Keyblocks used

What percentage of the keyblocks are used. When a table has just been reorganized with myisamchk, the values are very high (very near theoretical maximum).

? Packed

MySQL tries to pack key values that have a common suffix. This can only be used for indexes on CHAR and VARCHAR columns. For long indexed strings that have similar leftmost parts, this can significantly reduce the space used. In the preceding example, the second key is 40 bytes long and a 97% reduction in space is achieved.

? Max levels

How deep the B-tree for this key is. Large tables with long key values get high values.

? Records

How many rows are in the table.

? M.recordlength

The average row length. This is the exact row length for tables with fixed-length rows, because all rows have the same length.

? Packed

MySQL strips spaces from the end of strings. The Packed value indicates the percentage of savings achieved by doing this.

? Recordspace used

What percentage of the data file is used.

? Empty space

What percentage of the data file is unused.

? Blocks/Record

Average number of blocks per row (that is, how many links a fragmented row is composed of). This is always 1.0 for fixed-format tables. This value should stay as close to 1.0 as possible. If it gets too large, you can reorganize the table. See Section 7.6.4, ?MyISAM Table Optimization?.

? Recordblocks

How many blocks (links) are used. For fixed-format tables, this is the same as the number of rows.

? Deleteblocks

How many blocks (links) are deleted.

? Recorddata

How many bytes in the data file are used.

? Deleted data

How many bytes in the data file are deleted (unused).

? Lost space

If a row is updated to a shorter length, some space is lost. This is the sum of all such losses, in bytes.

? Linkdata

When the dynamic table format is used, row fragments are linked with pointers (4 to 7 bytes each). Linkdata is the sum of the amount of storage used by all such pointers.

MYISAMCHK MEMORY USAGE

Memory allocation is important when you run myisamchk. myisamchk uses no more memory than its memory-related variables are set to. If you are going to use myisamchk on very large tables, you should first decide how much memory you want it to use. The default is to use only about 3MB to perform repairs. By using larger values, you can get myisamchk to operate faster. For example, if you have more than 512MB RAM available, you could use options such as these (in addition to any other options you might specify):

```
myisamchk --myisam_sort_buffer_size=256M \
        --key_buffer_size=512M \
        --read_buffer_size=64M \
        --write_buffer_size=64M ...
```

Using --myisam_sort_buffer_size=16M is probably enough for most cases.

Be aware that myisamchk uses temporary files in TMPDIR. If TMPDIR points to a memory file system, out of memory errors can easily occur. If this happens, run myisamchk with the --tmpdir=dir_name option to specify a directory located on a file system that has more space.

When performing repair operations, myisamchk also needs a lot of disk space:

? Twice the size of the data file (the original file and a copy). This space is not needed if you do a repair with --quick; in this case, only the index file is re-created. This space must be available on the same file system as the original data file, as the copy is created in the same directory as the original.

? Space for the new index file that replaces the old one. The old index file is truncated at the start of the repair operation, so you usually ignore this space. This

space must be available on the same file system as the original data file.

? When using --recover or --sort-recover (but not when using --safe-recover), you need space on disk for sorting. This space is allocated in the temporary directory (specified by TMPDIR or --tmpdir=dir_name). The following formula yields the amount of space required:

(largest_key + row_pointer_length) * number_of_rows * 2

You can check the length of the keys and the row_pointer_length with myisamchk -dv tbl_name (see the section called ?OBTAINING TABLE INFORMATION WITH MYISAMCHK?). The row_pointer_length and number_of_rows values are the Datafile pointer and Data records values in the table description. To determine the largest_key value, check the Key lines in the table description. The Len column indicates the number of bytes for each key part. For a multiple-column index, the key size is the sum of the Len values for all key parts.

If you have a problem with disk space during repair, you can try --safe-recover instead of --recover.

COPYRIGHT

SEE ALSO

For more information, please refer to the MySQL Reference Manual, which may already be installed locally and which is also available online at http://dev.mysql.com/doc/.

AUTHOR

Oracle Corporation (http://dev.mysql.com/).