MySQL

GUIA DE REFERÊNCIA RÁPIDA

Guia de Referência Rápida

MySQL

Comandos SQL

ALTER TABLE

Altera a estrutura de uma tabela.

ALTER [IGNORE] TABLE tabela ação [,...]

Parâmetro	Significado
IGNORE	Utiliza somente a primeira linha quando encontrar linhas com duplicidade em uma chave UNIQUE, as outras linhas serão eliminadas. Se a opção não for especificada, será emitida uma mensagem de erro e nenhuma alteração será feita na tabela.
tabela	Nome da tabela a ser alterada.
ação	Ação ou ações a serem efetuadas na tabela.

Ações do comando ALTER TABLE

ADD COLUMN

Adiciona uma coluna à tabela.

ALTER [IGNORE] TABLE tabela
    ADD [COLUMN] decl_coluna [local_coluna]

Parâmetro	Significado
decl_coluna	Declaração da coluna; mesmo formato usado no comando CREATE TABLE.
local_coluna	Localização da coluna na tabela. Se não especificada, a coluna será a última da tabela.
FIRST	Adiciona a nova coluna na primeira posição da tabela.
AFTER	Adiciona a nova coluna após a coluna especificada.

ADD INDEX

Adiciona um índice à tabela.

ALTER [IGNORE] TABLE tabela
    ADD INDEX [índice] (coluna,...)

ADD PRIMARY KEY

Adiciona uma chave primária à tabela.

ALTER [IGNORE] TABLE tabela
    ADD PRIMARY KEY (coluna,...)

ADD UNIQUE

Adiciona um índice sem duplicidade à tabela.

ALTER [IGNORE] TABLE tabela
    ADD UNIQUE [índice] (coluna,...)

ALTER COLUMN

Altera (SET) ou elimina (DROP) o valor default de uma coluna.

```
ALTER [ IGNORE ] TABLE tabela
    ALTER [ COLUMN ] coluna
    { SET | DROP } DEFAULT valor
```

CHANGE COLUMN

Altera o nome e/ou a declaração de uma coluna.

```
ALTER [ IGNORE ] TABLE tabela
    CHANGE [ COLUMN ] coluna  decl_coluna
```

Parâmetro	Significado
coluna	Nome atual da coluna.
decl_coluna	Declaração da coluna; mesmo formato usado no comando CREATE TABLE.

DROP COLUMN

Remove uma coluna da tabela.

```
ALTER [ IGNORE ] TABLE tabela
    DROP [ COLUMN ] coluna
```

DROP INDEX

Remove um índice da tabela.

```
ALTER [ IGNORE ] TABLE tabela
    DROP INDEX índice
```

DROP PRIMARY KEY

Remove a chave primária da tabela.

```
ALTER [IGNORE] TABLE tabela
    DROP PRIMARY KEY
```

MODIFY COLUMN [3.22.16]

Altera a declaração de uma coluna. Similar a CHANGE COLUMN, exceto que não permite alterar o nome da coluna.

```
ALTER [IGNORE] TABLE tabela
    MODIFY [ COLUMN ] decl_coluna
```

Parâmetro	Significado
decl_coluna	Declaração da coluna; mesmo formato usado no comando CREATE TABLE.

RENAME

Altera o nome de uma tabela.

```
ALTER [ IGNORE ] TABLE tabela
    RENAME [ AS ] novo_nome
```

CHECK TABLE

Verifica a existência de erros na estrutura das tabelas. Somente com tabelas MyISAM. Similar a myisamchk -m tabela. Retorna uma tabela com as seguintes colunas:

Coluna	Significado
Table	Nome da tabela.
Op	Sempre "check".
Msg_type	Um dos status: error, info ou warning.
Msg_text	Mensagem de erro.

```
CHECK TABLE tabela [, tabela...]
```

CREATE DATABASE

Cria um banco de dados. No MySQL os bancos de dados são implementados como diretórios contendo arquivos que correspondem a tabelas do banco de dados. Como ainda não existem tabelas no banco de dados criado, este comando somente cria um subdiretório no diretório de dados do MySQL. Ocorrerá um erro se o banco de dados já existir ou se você não tiver o privilégio apropriado.

```
CREATE DATABASE nomebd
```

CREATE FUNCTION

Adiciona ao MySQL (na tabela func do banco de dados mysql) uma função definida pelo usuário, que estende o MySQL e funciona da mesma forma que uma função nativa (built-in) do MySQL, tais como ABS() e CONCAT(). Para mais detalhes consulte a ajuda on-line do MySQL.

```
CREATE [ AGGREGATE ] FUNCTION nome
    RETURNS { STRING | REAL | INTEGER }
    SONAME nome_library
```

Parâmetro	Descrição
AGGREGATE	Indica que trata-se de uma função AGGREGATE. Essas funções funcionam exatamente como uma função de agregação nativa do MySQL, tais como SUM() ou COUNT().
FUNCTION	Especifica o nome pela qual será feita referência à função.
RETURNS	Indica o tipo de dado retornado pela função.
SONAME	Path da biblioteca que contém o código executável ou a função.

CREATE INDEX

Adiciona um índice a uma tabela.

```
CREATE [ UNIQUE ] INDEX índice ON tabela
    (coluna,...)
```

Parâmetro	Significado
índice	Nome do índice a ser criado.
UNIQUE	Não permite duplicidade.
ON	Especifica o nome da tabela.
coluna	Nome da coluna ou colunas usadas para formar o índice. No caso de várias colunas, os valores do índice serão formados pela concatenação das colunas.

CREATE TABLE

Cria uma tabela no banco de dados corrente.

```
CREATE [ TEMPORARY ] TABLE
    [ IF NOT EXISTS ] tabela [(definição,...)]
    [ opções ]
    [ { IGNORE | REPLACE } comando_select ]
```

definição:

```
{
    declaração_coluna
    | PRIMARY KEY (coluna_índice,...)
    | KEY [índice] (coluna_índice,...)
    | INDEX [índice] (coluna_índice,...)
    | UNIQUE [ INDEX ] [índice] (coluna_índice,...)
    }
```

declaração_coluna:

```
coluna  tipo
    [ NOT NULL | NULL]
    [ DEFAULT valor ]
    [ AUTO_INCREMENT [ = valorinicial ] ]
    [ PRIMARY KEY ]
```

Parâmetro	Significado
TEMPORARY	Cria uma tabela temporária, visível somente para a conexão corrente, a qual será removida quando a conexão for finalizada.
IF NOT EXISTS	Cria a tabela somente se não existir uma tabela com o mesmo nome.
tabela	Nome da tabela a ser criada. A partir da versão 3.22 o nome da tabela pode ser especificado como: nomebd.tabela.
definição	Lista das colunas e índices a serem criados.

declaração_coluna	Declaração da coluna. É opcional se a tabela for criada com o resultado de um comando SELECT.
coluna	Nome da coluna.
tipo	Tipo de dados da coluna. Ver Tipos de Dados na página 72.
NOT NULL	Não permite valores NULL na coluna.
NULL	Permite valores NULL na coluna (default).
DEFAULT	Atribui o valor default à coluna, quando uma nova linha for criada sem especificar o valor da coluna.
AUTO_INCREMENT	Tipo especial de coluna numérica que pode ser atualizada automaticamente. Quando um valor NULL (ou 0) for inserido na coluna, o MySQL o substituirá automaticamente pelo próximo número na seqüência da coluna. O primeiro valor da coluna pode ser atribuído na opção AUTO_INCREMENT (página 10), sendo que o default é 1. É permitida apenas uma coluna AUTO_INCREMENT por tabela e a coluna deve ser UNIQUE ou PRIMARY KEY. Veja também o comando SET INSERT_ID na página 22. Para mais detalhes consulte o manual on-line do MySQL.
PRIMARY KEY	Indica que a coluna é a PRIMARY KEY (chave primária da tabela).
PRIMARY KEY	Identifica a PRIMARY KEY da tabela.
coluna_índice	Coluna a ser utilizada no índice. Para colunas CHAR e VARCHAR pode ser especificado o número de caracteres da coluna, usando a sintaxe: coluna(tamanho). Para colunas BLOB e TEXT o tamanho deve ser sempre especificado.
KEY	Sinônimo de INDEX.
INDEX	Cria um índice. Podem existir até 16 índices por tabela.
UNIQUE	Cria um índice sem duplicidade. Ocorrerá um erro ao se adicionar uma nova linha que coincida com uma linha existente.
IGNORE | REPLACE	Define se os valores duplicados em um índice UNIQUE devem ser ignorados (IGNORE) ou substituídos (REPLACE). Se esta opção não for especificada, ocorreráum erro e os registros remanescentes serão ignorados.
comando_select	Cria a tabela a partir dos registros retornados pelo comando SELECT.

opções

AUTO_INCREMENT = valor	Especifica o valor inicial de AUTO_INCREMENT para a tabela (somente para tabelas MyISAM).
AVG_ROW_LENGTH = n	Tamanho médio aproximado das linhas na tabela. Usado para determinar o tamanho máximo de arquivo para tabelas com colunas BLOB ou TEXT que podem ultrapassar o limite de 4GB.
CHECKSUM = {0 | 1}	Mantém (1) ou não (0) um checksum para cada linha da tabela. Torna a atualização da tabela um pouco mais lenta, mas facilita a procura por erros na tabela (somente para MyISAM).
COMMENT = "string"	Comentário da tabela (até 60 caracteres).
DELAY_KEY_WRITE = {0 | 1}	Atualiza o cache do índice da tabela ocasionalmente (1) ou a cada inserção (0) (somente para tabelas MyISAM).
MAX_ROWS = n	Número máximo de linhas previsto para a tabela.
MIN_ROWS = n	Número mínimo de linhas previsto para a tabela.
PACK_KEYS = {0 | 1}	Se definida como 1, compacta os blocos de índices numa porcentagem maior que a usual, tornando as operações de atualização mais demoradas e as de leitura mais rápidas. Se definida como 0, somente os índices com chaves CHAR e VARCHAR, com 8 ou mais caracteres, são compactados (somente para tabelas MyISAM e ISAM).
PASSWORD = "senha"	Especifica uma senha para a encriptação do arquivo de descrição da tabela (.frm). Somente para versões especiais do MySQL; não tem efeito na versão padrão.
ROW_FORMAT= { default | dynamic | static | compressed}	Define como as linhas deverão ser armazenadas (no futuro).
TYPE = formato	Formato de armazenamento da tabela.
ISAM	Formato original usado pelo MySQL.
MYISAM	Formato melhor que ISAM e deve substitui-lo no futuro (default).
HEAP	Formato para tabelas temporárias armazenadas na memória.

DELETE

Elimina linhas de uma tabela.

DELETE [LOW_PRIORITY] FROM tabela
[WHERE condição] [LIMIT n]

Parâmetro	Descrição
LOW_PRIORITY	Retarda a execução do comando até que nenhum outro programa cliente esteja lendo a tabela.
WHERE	Especifica a condição que determina quais linhas devem ser eliminadas. Se a cláusula WHERE não for especificada, todas as linhas da tabela serão eliminadas.
LIMIT[3.22.7]	Limita o número de linhas a serem eliminadas.

DESCRIBE

Fornece informações sobre colunas de uma tabela. Similar ao comando SHOW COLUMNS FROM tabela.

DESC[RIBE] tabela coluna

Parâmetro	Descrição
tabela	Nome da tabela.
coluna	Nome da coluna. Podem ser especificados os caracteres curinga '%' e '_'.

DROP DATABASE

Elimina um banco de dados. Um banco de dados é representado por um diretório. Se existirem arquivos nesse diretório não relacionados com o banco de dados, eles não serão eliminados, nem o diretório será eliminado.

DROP DATABASE [IF EXISTS] nomebd

Parâmetro	Descrição
IF EXISTS	Evita a ocorrência de erro se o banco de dados não existir ou se você não tiver os privilégios necessários.

DROP FUNCTION

Remove uma função previamente adicionada com o comando CREATE FUNCTION.

DROP FUNCTION nome

DROP INDEX

Elimina um índice de uma tabela. Equivale ao comando ALTER TABLE DROP INDEX.

DROP INDEX índice ON tabela

DROP TABLE

Elimina uma ou mais tabelas. Todos os dados e definições das tabelas serão eliminados.

DROP TABLE [IF EXISTS] tabela [, ...]

Parâmetro	Descrição
IF EXISTS	Evita a ocorrência de erro devido a tabelas inexistentes.

EXPLAIN

EXPLAIN tabela

Lista as colunas de uma tabela. Similar ao comando SHOW COLUMNS FROM tabela.

EXPLAIN SELECT comandoSELECT

Exibe informações detalhadas sobre como o MySQL executará um comando SELECT. Permite visualizar onde as chaves não estão sendo usadas de maneira eficiente.

FLUSH

Limpa ou reinicia processos internos utilizados pelo MySQL.

FLUSH opção [,...]

Opção	Descrição
HOSTS	Limpa o cache que armazena informações de hostnames para clientes.
LOGS	Fecha e reabre os arquivos padrão de log.
PRIVILEGES	Recarrega os privilégios das tabelas de permissões do MySQL.
STATUS	Reinicializa as variáveis de status que indicam o estado corrente do servidor.
TABLES	Fecha todas as tabelas abertas e grava em disco os dados do cache.

GRANT

Atribui privilégios de acesso a usuários do MySQL.

```
GRANT privilégio [(colunas)]
    [, privilégio [(colunas)] ...]
    ON { *.* | * | bd.* | bd.tabela | tabela }
    TO usuário@host [ IDENTIFIED BY 'senha' ] [, ...]
    [ WITH GRANT OPTION ]
```

Parâmetro	Descrição
privilégio	Privilégio a ser atribuído.
ALL [PRIVILEGES]	Todos os privilégios.
ALTER	Alterar tabelas e índices.
CREATE	Criar bancos de dados e tabelas.
DELETE	Eliminar linhas de tabelas.
DROP	Eliminar bancos de dados e tabelas.
FILE	Ler e gravar arquivos no servidor.
INDEX	Criar e eliminar índices.
INSERT	Inserir linhas em tabelas.
PROCESS	Visualizar e eliminar quaisquer threads.
RELOAD	Recarregar as tabelas internas do MySQL (Ver comando FLUSH).
SELECT	Ler dados de tabelas.
SHUTDOWN	Desativar o servidor.
UPDATE	Alterar linhas em tabelas.
USAGE	Sem privilégios.
colunas	Nomes das colunas às quais os privilégios devem ser aplicados.
ON	Especifica o nível de privilégio.
.	Atribui os privilégios a todas as tabelas em todos os bancos de dados do servidor.
*	Atribui os privilégios a todas as tabelas do banco de dados corrente.
bd.*	Atribui os privilégios a todas as tabelas do banco de dados especificado.
bd.tabela	Atribui os privilégios a todas as colunas da tabela especificada.
tabela	Atribui os privilégios a todas as colunas da tabela especificada no banco de dados default.
TO	Especifica o(s) nome(s) do(s) usuário(s) a ter(em) privilégios atribuídos. Formato: usuário@host.
usuário	Nome do usuário, ou um string vazio para especificar um usuário anônimo.
host	Nome do host. Pode ser: localhost, um nome de host, ou um endereço IP. Os caracteres '%' e '_' podem ser utilizados.
IDENTIFIED BY	Atribui uma senha ao usuário, ou altera a senha de um usuário existente.
WITH GRANT OPTION	Permite ao usuário atribuir a outros usuários qualquer um dos seus privilégios no nível de privilégio especificado.

INSERT

Insere linhas em uma tabela.

```
INSERT [ LOW_PRIORITY | DELAYED ] [ IGNORE ]
    [ INTO ] tabela [(coluna,...)]
    VALUES (expr [,...]) [,(...)] ...

INSERT [ LOW_PRIORITY | DELAYED ] [ IGNORE ]
    [ INTO ] tabela [(coluna,...)]
    SELECT ...

INSERT [ LOW_PRIORITY | DELAYED] [ IGNORE ]
    [ INTO ] tabela
    SET coluna=expr [,coluna=expr] ...
```

Parâmetro	Descrição
LOW_PRIORITY	Retarda a execução do comando até que nenhum outro programa cliente esteja lendo a tabela.
DELAYED[3.22.15]	Coloca as linhas em uma fila para inserção posterior, quando nenhum outro cliente estiver acessando a tabela.
IGNORE[3.22.10]	Descarta linhas com valores duplicados para chaves UNIQUE. Se esta opção não for especificada, ocorrerá um erro se forem encontrados valores duplicados e nenhuma outra linha será inserida.
[INTO] tabela	Nome da tabela. INTO é opcional.
VALUES	Insere linhas baseado nos valores especificados. A partir da versão 3.22.5 múltiplas listas de valores podem ser especificadas, permitindo a inserção de múltiplas linhas em um único INSERT.
SELECT	Insere as linhas retornadas pelo comando SELECT.
SET[3.22.10]	Atribui os valores das expressões às colunas. Colunas não especificadas terão seus valores default.

KILL

Elimina um thread. Cada conexão com o mysqld é executada em um thread separado. Utilize SHOW PROCESSLIST para ver quais threads estão sendo executados.

```
KILL id_thread
```

LOAD DATA

Insere linhas de um arquivo em uma tabela.

```
LOAD DATA [ LOW_PRIORITY ] [ LOCAL ]
    INFILE 'nomearq.txt' [ IGNORE | REPLACE ]
    INTO TABLE tabela
    delimitadores
    [ IGNORE n LINES ] (coluna,...)]
```

delimitadores:

```
[ FIELDS
      [ TERMINATED BY 'string' ]
      [ OPTIONALLY ] ENCLOSED BY 'char' ]
      [ ESCAPED BY 'char' ] ]
    [ LINES TERMINATED BY 'string' ]
```

Parâmetro	Descrição
LOW_PRIORITY	Retarda a execução do comando até que nenhum outro programa cliente esteja lendo a tabela.
LOCAL[3.22.15]	Indica que o arquivo será lido do host cliente e enviado pela rede para o servidor. Se não especificado, o arquivo será lido do servidor.
INFILE	Especifica o nome do arquivo texto.
IGNORE | REPLACE	Indicam qual procedimento deve ser adotado no caso dos dados da chave primária a serem importados já existirem no banco de dados. A opção REPLACE faz com que o dados antigos sejam removidos e os novos inseridos. A opção IGNORE faz com que o mysql ignore os dados. Caso nenhuma das opções tenha sido escolhida, exibe um erro e ignora o resto do arquivo.
INTO TABLE	Nome da tabela que receberá os dados.
delimitadores	
FIELDS	Faz com que as opções TERMINATED BY, [OPTIONALLY] ENCLOSED BY e ESCAPED BY sejam facultativas, mas pelo menos uma deve ser declarada.
FIELDS TERMINATED BY	Especifica o caractere ou caracteres utilizados para delimitar valores na linha (default=TERMINATED BY '\t').
OPTIONALLY	Somente colunas dos tipos CHAR e VARCHAR recebem o caractere especificado por ENCLOSED BY.
FIELDS ENCLOSED BY	Especifica o caractere utilizado para delimitar a string de caracteres (default=ENCLOSED BY '').
FIELDS ESCAPED BY	Especifica o caractere utilizado para indicar que o próximo caractere não é um caractere especial (default=ESCAPED BY '\\').
LINES TERMINATED BY	Especifica o caractere ou caracteres que indicam o final de uma linha (default='\n').
IGNORE n LINES	Ignora as primeiras n linhas do arquivo.

LOCK TABLES

Bloqueia uma ou mais tabelas. Espera até conseguir bloquear todas as tabelas especificadas.

```
LOCK TABLES tabela [ AS alias ]
    { READ | [ LOW_PRIORITY ] WRITE }
    [, tabela { READ | [ LOW_PRIORITY ] WRITE } ...]
```

Parâmetro	Descrição
AS	Alias que deverá ser utilizado nas consultas subseqüentes que dependam das tabelas bloqueadas.
READ	Permite aos outros clientes somente a leitura da tabela.
LOW_PRIORITY [3.22.5]	Bloqueios WRITE normalmente possuem prioridade maior que bloqueios READ, para garantir que as atualizações sejam processadas o mais rapidamente possível. LOW_PRIORITY funciona no sentido de outros threads poderem receber READ locks enquanto o thread espera que a sua requisição de WRITE lock seja aceita, a qual será aceita quando não existir nenhum READ LOCK na tabela.
WRITE	Não permite aos outros clientes ler ou gravar na tabela.

OPTIMIZE TABLE

Otimiza o espaço ocupado por uma tabela. Uma tabela poderá ficar fragmentada quando ocorrerem muitas eliminações, principalmente com registros de tamanhos diferentes. Recomenda-se, nesse caso, executar periodicamente o comando OPTIMIZE TABLE, para manter o desempenho. Durante o processo de otimização o acesso à tabela é feito normalmente, exceto que as gravações na tabela são direcionadas para uma tabela temporária.

```
OPTIMIZE TABLE tabela
```

REPAIR TABLE

Repara tabelas danificadas. Somente com tabelas MyISAM. Similar a myisamchk -r tabela. Similar a myisamchk -m tabela. Retorna uma tabela com as seguintes colunas:

Coluna	Significado
Table	Nome da tabela.
Op	Sempre "repair".
Msg_type	Um dos status: error, info ou warning.
Msg_text	Mensagem de erro.

```
REPAIR TABLE tabela [,tabela...]
```

REPLACE

Similar ao comando INSERT, exceto que se a linha a ser inserida tiver um valor para um índice UNIQUE igual ao valor em uma linha já existente na tabela, esta será eliminada antes da nova linha ser inserida.

```
REPLACE [ LOW_PRIORITY | DELAYED ]
    [ INTO ] tabela [(coluna,...)]
    VALUES (expressão,...)
REPLACE [ LOW_PRIORITY | DELAYED ]
    [ INTO ] tabela [(coluna,...)] SELECT ...
REPLACE [ LOW_PRIORITY | DELAYED ]
    [ INTO ] tabela
    SET coluna=expressão, coluna=expressão,...
```

Parâmetro	Descrição
LOW_PRIORITY	Retarda a execução do comando até que nenhum outro programa cliente esteja lendo a tabela.
DELAYED[3.22.15]	Coloca as linhas em uma fila para inserção posterior, quando nenhum outro programa cliente estiver acessando a tabela.
[INTO] tabela	Nome da tabela. INTO é opcional.
VALUES	Insere linhas baseado nos valores especificados. A partir da versão 3.22.5, múltiplas listas de valores podem ser especificadas, permitindo a inserção de múltiplas linhas em um único INSERT.
SELECT	Insere as linhas retornadas pelo comando SELECT.
SET[3.22.10]	Atribui os valores das expressões às colunas. Colunas não especificadas terão valores default.

REVOKE

Retira os privilégios atribuídos a usuários do MySQL através do comando GRANT.

```
REVOKE privilégio [(colunas)]
    [, privilégio [(colunas)] ...]
    ON {tabela | * | *.* | nomebd.*}
    FROM usuário [, ...]
```

Parâmetro	Descrição
privilégio	Privilégio a ser retirado.
ALL [PRIVILEGES]	Todos os privilégios.
ALTER	Alterar tabelas e índices.
CREATE	Criar bancos de dados e tabelas.
DELETE	Eliminar linhas de tabelas.
DROP	Eliminar bancos de dados e tabelas.
FILE	Ler e gravar arquivos no servidor.
INDEX	Criar e eliminar índices.
INSERT	Inserir linhas em tabelas.

PROCESS	Visualizar e eliminar quaisquer threads.
RELOAD	Recarregar as tabelas internas do MySQL (ver comando FLUSH).
SELECT	Ler dados de tabelas.
SHUTDOWN	Desativar o servidor.
UPDATE	Alterar linhas em tabelas.
USAGE	Sem privilégios.
colunas	Nomes das colunas das quais os privilégios devem ser retirados.
ON	Especifica o nível de privilégio.
.	Retira os privilégios de todas as tabelas em todos os bancos de dados do servidor.
*	Retira os privilégios de todas as tabelas do banco de dados corrente.
bd.*	Retira os privilégios de todas as tabelas do banco de dados especificado.
bd.tabela	Retira os privilégios de todas as colunas da tabela especificada.
tabela	Retira os privilégios de todas as colunas da tabela especificada do banco de dados default.
FROM	Especifica o(s) nome(s) do(s) usuário(s) a ter(em) privilégios retirados. Formato: usuário@host.
usuário	Nome do usuário ou um string vazio para especificar um usuário anônimo.
host	Nome do host. Pode ser: localhost, um nome de host ou um endereço IP. Os caracteres curinga '%' e '_' podem ser utilizados.

SELECT

Seleciona linhas de uma ou mais tabelas.

```
SELECT [ predicado ] [ opções_select ]
      coluna [,... ]
      [ INTO {OUTFILE | DUMPFILE} 'nomearq'
      delimitadores ]
      [ FROM ref_tabelas
      [ WHERE condição ]
      [ GROUP BY coluna,...] [ HAVING condição ]
      [ ORDER BY coluna [ ASC | DESC ] ,...]
      [ LIMIT [ linha_inicial, ] num_linhas ]
      [ PROCEDURE nome ]
      ]
opções_select:
      [ STRAIGHT_JOIN ]
      [ SQL_SMALL_RESULT | SQL_BIG_RESULT]
      [ HIGH_PRIORITY ]
```

delimitadores:

[FIELDS
[TERMINATED BY 'string']
[OPTIONALLY] ENCLOSED BY 'char']
[ESCAPED BY 'char']]
[LINES TERMINATED BY 'string']

Parâmetro	Descrição
predicado	Restringe as linhas a serem retornadas.
DISTINCT	Omite registros que contenham dados duplicados nos campos selecionados.
DISTINCTROW	Similar a DISTINCT, mas seleciona somente quando todos os campos das linhas forem idênticos.
ALL	Seleciona todas as linhas que atendam às condições especificadas.
opções_select	
STRAIGHT_JOIN	Força a junção das tabelas na ordem em que elas estão especificadas na cláusula FROM. Pode ser usada caso você ache que o MySQL não está fazendo a junção da melhor forma.
SQL_SMALL_RESULT [3.22.12]	Pode ser usado com GROUP BY ou DISTINCT para informar ao MySQL que o resultado do SELECT será pequeno. Nesse caso, o MySQL usa tabelas temporárias para armazenar os resultados em vez de ordená-los.
SQL_BIG_RESULT	Pode ser usado com GROUP BY ou DISTINCT para informar ao MySQL que o resultado do SELECT será grande. Nesse caso, o MySQL ordena os resultados em vez de usar uma tabela temporária. No caso de falta de memória, tabelas temporárias em disco podem ser utilizadas pelo MySQL.
HIGH_PRIORITY	Atribui ao comando SELECT prioridade maior que aos comandos que atualizam tabelas. Deve ser usada somente para consultas que são muito rápidas e que devam ser feitas uma única vez.
coluna	Nome da coluna ou colunas a serem retornadas pela consulta. Pode-se atribuir um alias a uma coluna, usando a sintaxe: AS alias. O alias pode ser utilizado no lugar do nome nas cláusulas GROUP BY, ORDER BY e HAVING.

INTO	Redireciona a saída do comando SELECT para um arquivo. É necessário ter o privilégio FILE e o arquivo de saída não deve existir.
OUTFILE	Redireciona para um arquivo que é criado no servidor.
DUMPFILE	Escreve apenas uma linha no arquivo sem mudança de linha e sem modificar o conteúdo. Útil para armazenar o conteúdo de um campo BLOB em um arquivo.
delimitadores	
FIELDS	Faz com que as opções TERMINATED BY, [OPTIONALLY] ENCLOSED BY e ESCAPED BY sejam facultativas, mas pelo menos uma deve ser declarada.
FIELDS TERMINATED BY	Especifica o caractere ou caracteres utilizados para delimitar valores na linha (default= '\t').
OPTIONALLY	Somente colunas dos tipos CHAR e VARCHAR recebemo caractere especificado por ENCLOSED BY.
FIELDS ENCLOSED BY	Especifica o caractere utilizado para delimitar o string de caracteres (default='').
FIELDS ESCAPED BY	Especifica o caractere utilizado para indicar que o próximo caractere não é um caractere especial (default= '\\').
LINES TERMINATED BY	Especifica o caractere ou caracteres que indicam o final de uma linha (default='\n').
FROM ref_tabelas	Especifica o nome das tabelas dos campos a serem utilizados no comando SELECT. Ver cláusulaJOIN na página 21.
WHERE	Especifica a condição que determina quais linhas serão retornadas.
GROUP BY	Retorna um valor para cada grupo de registros. Cria um registro de resumo para cada registro definido.
HAVING	Similar à cláusula WHERE, mas restringe somente os registros a serem retornados com a cláusula GROUP BY.
ORDER BY	Especifica as colunas a serem usadas como chave para ordenar os registros retornados. A partir da versão 3.23.2 pode ser especificada uma fórmula.
DESC	Ordem descendente.
ASC	Ordem ascendente (default).

LIMIT	Limita o número de linhas a serem retornadas pelo comando SELECT.
linha_inicial	Especifica o número da primeira linha a ser retornada (inicia em 0).
num_linhas	Especifica o número de linhas a serem retornadas.
PROCEDURE	Especifica uma procedure para a qual os dados retornados serão enviados antes de serem retornados ao cliente. A partir da versão 3.23, pode-se usar PROCEDURE ANALYSE() para obter informações sobre as características dos dados nas colunas.

Cláusula JOIN

As seguintes sintaxes JOIN são permitidas em comandos SELECT:

ref_tabela1, ref_tabela2

ref_tabela1 [CROSS] JOIN ref_tabela2 [3.21.18]

ref_tabela1 INNER JOIN ref_tabela2 [3.23]

ref_tabela1 STRAIGHT_JOIN ref_tabela2

ref_tabela1 LEFT [OUTER] JOIN ref_tabela2 ON condição

ref_tabela1 LEFT [OUTER] JOIN ref_tabela2 USING (colunas)

ref_tabela1 NATURAL LEFT [OUTER] JOIN ref_tabela2

oj ref_tabela1 LEFT OUTER JOIN ref_tabela2 ON condição

ref_tabela:

tabela [[AS] alias] [USE INDEX (chaves)] [IGNORE INDEX (chaves)]

Tipo de junção	Descrição
, (vírgula)	Retorna todas as combinações possíveis de linhas das tabelas. Normalmente a condição de junção é especificada na cláusula WHERE.
JOIN	Equivale à utilização da vírgula.
CROSS JOIN	O mesmo que JOIN.
INNER JOIN	O mesmo que JOIN.
STRAIGHT_JOIN	Similar a JOIN, exceto que a tabela da esquerda é sempre lida antes da tabela da direita. Utilizado quando o otimizador efetua a junção de forma errada.

LEFT JOIN	Retorna todas as linhas da tabela da esquerdamesmoque não exista equivalência na tabela da direita (nesse caso, preenche com NULL). A condição para a junção é determinada pelas cláusulas ON ou USING.
ON	Especifica a condição para a junção das tabelas.
USING	Especifica a lista de colunas que devem existir em ambas as tabelas. Exemplo: A LEFT B JOIN USING (C1, C2, C3, ...).
LEFT OUTER JOIN	O mesmo que LEFT JOIN. Somente para compatibilidade com ODBC.
NATURAL LEFT JOIN	Equivalente a LEFT JOIN USING.

SET [OPTION]

Define opções para a sessão corrente do MySQL.

SET [OPTION] opção= valor, ...

Opções	Significado
CHARACTER SET { character_set | DEFAULT }	Especifica o character_set utilizado pelo programa cliente. Atualmente, a única opção é cp1251_koi8 (alfabeto Russo). A opção DEFAULT retorna o character_set original.
INSERT_ID = n	Especifica o valor a ser usado pelo próximo comando INSERT quando estiver inserindo em uma coluna AUTO_INCREMENT.
LAST_INSERT_ID = n	Especifica o valor a ser retornado pela função LAST_INSERT_ID().
PASSWORD = PASSWORD('senha')	Define a senha para o usuário corrente.
PASSWORD [FOR usuário@host] = PASSWORD('senha')	Define a senha para um usuário específico no servidor host corrente.
SQL_AUTO_IS_NULL = { 0 | 1 }[3,23.5]	Se definida como 1, a última linha inserida contendo um valor AUTO_INCREMENT pode ser selecionada usando a cláusula WHERE auto_inc_col IS NULL. Usada somente por alguns programas ODBC, tal como o Microsoft Access (default=1).
SQL_BIG_SELECTS = { 0 | 1 }	Determina se o MySQL deve ser abortado (0) ou não (1) quando SELECT retornar mais que max_join_size linhas (default=1).
SQL_BIG_TABLES = { 0 | 1 }	Determina se as tabelas temporárias serão armazenadas em disco (1) ou na memória RAM (0) (default=0).

SQL_LOG_OFF = { 0 | 1 }
Determina se o arquivo de log padrão deve ser atualizado (0) ou não (1). O arquivo de log padrão registra quando o servidor iniciou, terminou, ou quando aconteceu algum erro. Esta opção não afeta o arquivo de update log.

SQL_LOG_UPDATE = { 0 | 1 } [3.23.5]
Determina se o arquivo de update-log deve ser atualizado (1) ou não (0). O arquivo de update log registra todos os comandos que alteram as tabelas, sendo muito utilizado para backup. Esta opção não afeta o arquivo de log padrão.

SQL_LOW_PRIORITY_UPDATES = { 0 | 1 }
Determina se os comandos que alteram tabelas (DELETE, INSERT, REPLACE, UPDATE) devem aguardar (1) ou não (0) até que não exista nenhum comando SELECT ativo ou pendente na tabela afetada.

SQL_SELECT_LIMIT = { n | DEFAULT }
Especifica o número máximo de registros a serem retornados por um comando SELECT. O valor DEFAULT retorna todos os registros. A cláusula LIMIT do comando SELECT tem precedência sobre esta opção.

SQL_WARNINGS = { 0 | 1 }
Se definida como 1, o MySQL deve informar a contagem de linhas, mesmo que seja para a inserção de uma única linha. Normalmente a contagem é informada somente para comandos INSERT, que inserem múltiplas linhas (default=0).

TIMESTAMP = { valor | DEFAULT }
Especifica o valor TIMESTAMP (data e hora) da sessão. O valor DEFAULT retorna a data e a hora do sistema.

SHOW

SHOW COLUMNS

Exibe informações sobre as colunas de uma tabela. Se os tipos de colunas forem diferentes do especificado no comando CREATE TABLE, lembre-se que o MySQL às vezes altera automaticamente os tipos das colunas. SHOW FIELDS é um sinônimo de SHOW COLUMNS.

SHOW COLUMNS FROM tabela
[FROM nomebd] [LIKE curinga]

Parâmetro	Descrição
LIKE	Limita as informações a serem exibidas. Ver operador LIKE na página 27.

SHOW DATABASES

Lista os bancos de dados no servidor MySQL.

SHOW DATABASES [LIKE curinga]

Parâmetro	Descrição
LIKE	Limita as informações a serem exibidas. Ver operador LIKE na página 27.

SHOW GRANTS [3.23.4]

Exibe informações sobre as permissões de um usuário.

SHOW GRANTS FOR usuário@host

SHOW INDEX

Exibe informações sobre os índices de uma tabela. SHOW KEYS é um sinônimo de SHOW INDEX.

SHOW INDEX FROM tabela [FROM nomebd]

SHOW PROCESSLIST

Exibe informações sobre os threads que estão sendo executados no servidor.

SHOW [FULL] PROCESSLIST

Parâmetro	Descrição
FULL[3.23.7]	Exibe todos os caracteres de cada consulta (default=100 primeiros).

SHOW STATUS [3.21.21]

Exibe as variáveis de status do servidor e seus valores.

SHOW STATUS

SHOW TABLE STATUS [3.23]

Exibe informações sobre as tabelas em um banco de dados.

SHOW TABLE STATUS
[FROM nomebd] [LIKE curinga]

Parâmetro	Descrição
LIKE	Limita as informações a serem exibidas. Ver operador LIKE na página 27.

SHOW TABLES

Lista as tabelas em um banco de dados.

SHOW TABLES [FROM nomebd] [LIKE curinga]

Parâmetro	Descrição
LIKE	Limita as informações a serem exibidas. Ver operador LIKE na página 27.

SHOW VARIABLES [3.21.22]

Exibe uma lista das variáveis do servidor e seus valores. Veja na página 54 a descrição dessas variáveis.

```
SHOW VARIABLES [ LIKE curinga ]
```

Parâmetro	Descrição
LIKE	Limita as informações a serem exibidas. Ver operador LIKE na página 27.

UNLOCK TABLES

Desbloqueia as tabelas que foram bloqueadas pelo usuário corrente.

```
UNLOCK TABLES
```

UPDATE

Altera o conteúdo de linhas em uma tabela.

```
UPDATE [ LOW_PRIORITY ] tabela
    SET coluna=expr,coluna=expr,...
    [ WHERE condição ] [ LIMIT n ]
```

Parâmetro	Descrição
LOW_PRIORITY [3.22.5]	Retarda a execução do comando até que nenhum outro programa cliente esteja lendo a tabela.
tabela	Nome da tabela a ser atualizada.
SET	Especifica as colunas a serem atualizadas e os respectivos valores a serem atribuídos.
WHERE	Especifica a condição que determina quais linhas serão atualizadas. Se não especificada, todas as linhas serão atualizadas.
LIMIT [3.23.3]	Limita o número de linhas a serem atualizadas.

USE

Identifica o banco de dados default, o qual será utilizado quando as referências a uma tabela não especificarem explicitamente o nome do banco de dados.

```
USE nomebd
```

Operadores

Operadores Aritméticos

Operador	Sintaxe	Significado
+	x + y	Adiciona dois valores numéricos.
-	x - y	Subtrai um valor numérico de outro.
*	x * y	Multiplica dois valores numéricos.
/	x / y	Divide um valor numérico por outro. A divisão por zero produz um resultado NULL.
%	x % y	Retorna o módulo aritmético de dois valores numéricos.

Operadores Bitwise

Operador	Sintaxe	Significado
|	x | y	Executa um "bitwise OR" em dois valores inteiros.
&	x & y	Executa um "bitwise AND" em dois valores inteiros.
<<	x << y	Desloca à esquerda, em x, o número de bits definido por y.
>>	x >> y	Desloca à direita, em x, o número de bits definido por y.
~	~x	Inverte todos os bits. Converte bit 1 em bit 0 e bit 0 em bit 1.

Operadores Lógicos

Operador	Sintaxe	Significado
NOT ou !	NOT x	Retornará 1 (True) se o argumento for 0 (False); caso contrário, retornará 0 (False). Exceção: NOT NULL retorna NULL.
OR ou ||	x OR y	Retornará 1 (True) se um dos argumentos não for 0 (False) e nem NULL.
AND ou &&	x AND y	Retornará 0 (False) se um dos argumentos for 0 (False) ou NULL; caso contrário, retornará 1 (True).

Operadores de Comparação

Operador	Sintaxe	Significado
=	x = y	Igualdade.
<> ou !=	x <> y	Diferente.
<=	x <= y	Menor que ou igual.
<	x < y	Menor que.
>=	x >= y	Maior que ou igual.
>	x > y	Maior que.
<=>[3,23]	x <=> y	True se os dois operandos forem iguais, mesmo para comparações com NULL.

IS [NOT] NULL		Retornará 1 (True) se um valor for NULL; caso contrário, retornará 0 (False). Se NOT for especificado, retornará 1 (True) se o valor não for NULL.

expr BETWEEN min AND max

Determina se o valor de uma expressão está dentro de um intervalo de valores. A cláusula BETWEEN é inclusiva, isso significa que tanto min como max serão incluídos na comparação. Se expr for maior ou igual a min e expr for menor ou igual a max, BETWEEN retornará 1; caso contrário, retornará 0.

expr [NOT] IN (valor,...)

Retornará 1 se expr for um dos valores na lista IN; caso contrário, retornará 0. Se NOT for especificado, retornará 1 se não estiver na lista.

Operador Cast (BINARY)[3.23]

O operador BINARY converte (casts) o string que o segue para string binário. Essa é uma maneira de forçar uma comparação de coluna ser case sensitive, mesmo que a coluna não esteja definida como BINARY ou BLOB. Somente com tipos CHAR e VARCHAR.

Exemplos:

"novatec" = "NOVATEC" -> 1

"novatec" = BINARY "NOVATEC" -> 0

BINARY "novatec" = "NOVATEC" -> 0

Operador LIKE

Retornará 1 (True) se uma expressão corresponder a um padrão. Permite a utilização de expressões numéricas (isto é, uma extensão ao padrão ANSI SQL). Se NOT for especificado, retornará 1 se não possuir tiver correspondência.

expr [NOT] LIKE padrão [ESCAPE 'escape-char']

Parâmetro	Descrição
padrão	Seqüência de caracteres em relação a qual expr é comparada. Não é feita distinção entre minúsculas e maiúsculas. Os caracteres '%' e '_' dentro do padrão têm a seguinte interpretação:
%	Especifica qualquer seqüência de zero ou mais caracteres.
_	Especifica qualquer caractere.
ESCAPE	Especifica o caractere escape (default='\'). Um caractere escape remove o significado especial do caractere % ou \ que o segue.

Operador REGEXP

Executa a procura de um padrão dentro da string expr. Retornará 1 no caso de sucesso e 0 em caso de falha.Todo caractere '\' deve ser colocado como '\\' e, a partir da versão 3.23.4 é “case insentive” para strings normais (não binárias). REGEXP usa o character set corrente (ISO-8859-1 Latin1 por default) quando tem que decidir o tipo de um caractere.

expr REGEXP padrão

expr NOT REGEXP padrão

O mesmo que NOT (expr REGEXP padrão).

expr NOT RLIKE padrão

O mesmo que NOT REGEXP.

expr RLIKE padrão

O mesmo que REGEXP.

Caracteres Especiais usados em REGEXP

Permite que diversos formatos de palavras possam ser encontrados utilizando expressões regulares.

Seqüência	Significado
^	Somente combina o início de um string.
$	Somente combina o final de um string.
.	Combina qualquer caractere (incluindo newline).
[...]	Combina qualquer caractere entre os colchetes. Dentro dos colchetes pode ser especificado um intervalo, como por exemplo: [0-9] para selecionar os dígitos de 0 a 9.
[^...]	Combina qualquer caractere que não esteja entre os colchetes.
e*	Combina zero ou mais ocorrências do elemento e.
e+	Combina uma ou mais ocorrências do elemento e.
e?	Combina zero ou uma ocorrência do elemento e.
e1 | e2	Combina com o elemento e1 ou e2.
e {m}	Combina m ocorrências do elemento e.
e {m,}	Combina m ou mais ocorrências do elemento e.
e {,n}	Combina zero ou n ocorrências do elemento e.
e {m,n}	Combina de m a n ocorrências do elemento e.
(...)	Agrupa elementos em um simples elemento.
outro	Caracteres normais combinam com eles mesmos.

Para mais informações sobre expressões regulares, consulte o manual on-line do MySQL.

Funções de Agregação

`AVG(expr )`

Retorna a média de expr para valores não-NULL nas linhas selecionadas.

`BIT_AND(expr )`[3.21.11]

Retorna o resultado da operação lógica AND em todos os bits de expr.

`BIT_OR(expr )`[3.21.11]

Retorna o resultado da operação lógica OR em todos os bits de expr.

`COUNT(expr )`

Retorna o número de valores não-NULL nas linhas retornadas pelo comando SELECT.

`COUNT(*)`

Retorna o número de valores nas linhas retornadas pelo comando SELECT (inclusive valores NULL).

`COUNT(DISTINCT expr [,...])`[3.23.2]

Retorna o número de valores não-NULL nas linhas retornadas pelo comando SELECT excluindo os valores duplicados.

`MAX(expr )`

Retorna o valor máximo de expr para valores não-NULL nas linhas selecionadas.

`MIN(expr )`

Retorna o valor mínimo de expr para valores não-NULL nas linhas selecionadas.

`STD(expr )`

Retorna o desvio-padrão de expr para valores não-NULL nas linhas selecionadas.

`STDDEV(expr )`

O mesmo que STD().

`SUM(expr )`

Retorna a soma de expr para valores não-NULL nas linhas selecionadas.

Funções de Comparação

GREATEST(expr1, expr2,...)

Retorna o maior valor dentre os argumentos fornecidos. A função GREATEST substituiu a função MAX a partir da versão 3.22.5.

IF(expr1,expr2,expr3)

Retornará expr2 se expr1 for True (não 0 ou NULL); caso contrário, retornará expr3. IF() retornará um valor numérico ou string, dependendo do contexto em que for utilizado.

IFNULL(expr1, expr2)

Retornará expr2 se expr1 = NULL; caso contrário, retornará expr1.

INTERVAL(n,n1,n2,...)

Retornará 0 se n < n1, 1 se n < n2 e assim por diante, ou -1 se n for NULL. Os valores n1, n2, ... devem estar em ordem crescente (n1 < n2 < ...) porque é feita uma pesquisa binária.

ISNULL(expr)

Retornará 1 (True) se a expr for NULL; caso contrário, retornará 0 (False). Observe que uma comparação de valores NULL usando o caractere "=" será sempre False.

LEAST(expr1, expr2,...)

Retorna o menor valor dentre os argumentos fornecidos. A função LEAST substituiu a função MIN a partir da versão 3.22.5.

STRCMP(str1, str2)

Retornará 0 se os strings forem iguais, -1 se o primeiro argumento for menor que o segundo de acordo com a ordem de classificação corrente; e 1 caso contrário.

Função CASE

CASE

Formato 1:

Compara uma expressão a um conjunto de expressões simples para determinar o resultado. Retorna o resultado onde valor = valor_compara.

```
CASE valor
    WHEN [ valor_compara ] THEN resultado
    [ WHEN [ valor_compara ] THEN resultado...]
    [ ELSE resultado]
END
```

Argumento	Significado
WHEN	Especifica o valor a ser comparado.
THEN	Especifica o resultado se houver correspondência com o valor especificado em WHEN.
ELSE	Se não for encontrada nenhuma correspondência de valores, então será retornado o resultado após ELSE. Se ELSE não for especificado, então será retornado NULL.

Formato 2:

Avalia um conjunto de expressões booleanas para determinar o resultado. Retorna o resultado da primeira condição verdadeira (TRUE).

```
CASE
    WHEN [ condição ] THEN resultado
    [ WHEN [ condição ] THEN resultado...]
    [ ELSE resultado ]
END
```

Argumento	Significado
WHEN	Especifica a condição a ser verificada.
THEN	Especifica o resultado caso a condição WHEN for satisfeita.
ELSE	Se nenhuma condição for satisfeita, então será retornado o resultado após ELSE. Se ELSE não for especificado, então será retornado NULL.

Exemplo:

```
mysql> SELECT CASE WHEN 1>0 THEN “true” ELSE “false” END;
    -> “true”
```

Funções Numéricas

`ABS(x)`

Retorna o valor absoluto de x.

`ACOS(x)`[3.21.8]

Retorna o arco co-seno de x, ou NULL se x não estiver entre -1 e 1.

`ASIN(x)`[3.21.8]

Retorna o arco seno de x, ou NULL se x não estiver entre -1 e 1.

`ATAN(x)`[3.21.8]

Retorna o arco tangente de x.

`ATAN2(x,x)`

Retorna o arco tangente de duas variáveis x e y. Similar a ATAN2(x,y), exceto que usa os sinais dos argumentos para determinar o quadrante do valor retornado.

`CEILING(x)`

Retorna o menor número inteiro maior ou igual a x.

`COS(x)`[3.21.8]

Retorna o co-seno de x, onde x é fornecido em radianos.

`COT(x)`[3.21.16]

Retorna a cotangente de x, onde x é fornecido em radianos.

`DEGREES(x)`[3.21.16]

Retorna o valor de x convertido de radianos para graus.

`EXP(x)`

Retorna e^x (e=2.711828).

`FLOOR(x)`

Retorna o maior número inteiro menor ou igual a x.

`LOG(x)`

Retorna o logaritmo natural (base e) de x.

`LOG10(x)`

Retorna o logaritmo (base 10) de x.

MOD(m,n)

Retorna o resto da divisão de m por n. O mesmo que m % n.

PI() [3.21.8]

Retorna o valor de PI.

POW(x,y)

Retorna o valor de x^y

POWER(x,y)[3.21.16]

O mesmo que POW(x,y).

RADIANS(x)[3.21.16]

Retorna o valor de x convertido de graus para radianos.

RAND([n])

Retorna um valor aleatório de ponto-flutuante no intervalo de 0.0 a 1.0.

Argumento	Descrição
n	Valor da semente para a geração de números aleatórios. Chamadas com o mesmo valor de n sempre retornam o mesmo resultado.

ROUND(x)

Retorna o valor de x arredondado para um inteiro.

ROUND(x,d)

Retorna o valor de x arredondado para um número com d casas decimais. Se d = 0, o resultado não terá ponto decimal ou parte fracionária.

SIGN(x)

Retorna -1, 0 ou 1, dependendo de x ser negativo, zero ou positivo.

SIN(x)[3.21.8]

Retorna o seno de x, onde x é fornecido em radianos.

SQRT(x)

Retorna a raiz quadrada de x.

TAN(x)[3.21.8]

Retorna a tangente de x, onde x é fornecido em radianos.

TRUNCATE(x,d)[3.21.16]

Retorna o valor de x truncado para d casas decimais.

Funções de String

ASCII(str)[3.21.2]

Retorna o código ASCII do caractere mais à esquerda de str. Retornará 0 se str for um string vazio ou NULL se str for NULL.

BIN(n)[3.22.4]

Retorna um string com a representação binária do valor de n, onde n é um número Longlong (BIGINT). Equivale à função CONV(n,10,2).

CHAR(n1,n2,...)[3.21.0]

Retorna os caracteres correspondentes ao códigos ASCII especificado.

CHAR_LENGTH(str)

O mesmo que LENGTH().

CHARACTER_LENGTH(str)

O mesmo que LENGTH().

COALESCE(expr1, expr2, ...)[3.23.3]

Retorna o primeiro elemento não-NULL da lista.

CONCAT(str1,str2,...)

Retorna o string resultante da concatenação dos strings especificados. Retornará NULL se qualquer um dos argumentos for NULL.

CONV(n,base1,base2)[3.22.4]

Retorna a representação string do número n convertido da base base1 para a base base2.

ELT(n,str1,str2,...)

Retorna o n-ésimo string da lista str1,str2,... Retornará NULL se não existir o n-ésimo elemento ou se o string em n for NULL. ELT() é o complemento de FIELD().

EXPORT_SET(n, on, off,[separador,[numbits]])

Retorna um string consistindo dos strings on e off, separados pelo string separador.

Parâmetro	Significado
on	É usado para cada bit ligado a n.
off	É usado para bit desligado de n.
numbits	Número máximo de bits a serem verificados em n (default=64).
separador	String utilizado para separar os valores on e off (default=",").

FIELD(str,str1,str2,...)

Retorna o índice de str na lista de strings str1, str2,... Retornará 0 se str não for encontrado ou se str for NULL. FIELD() é o complemento de ELT().

FIND_IN_SET(str, lista_str)

Retorna o índice de str no string lista-str. lista_str é um string que consiste de substrings separados por vírgulas.

FORMAT(x,d)

Retorna o número x formatado ('#,###,###.##'), arredondado para d casas decimais.

HEX(n)

Retorna um string com a representação hexadecimal do valor n. Equivale a CONV(n,10,16).

INSERT(str,pos,tam,novostr)

Retorna o string str, com a parte iniciando na posição pos e com o tamanho tam sendo substituída pelo string novostr.

INSTR(str,substr)

Retorna a posição da primeira ocorrência do substring substr no string str. Similar à função LOCATE(substr,str), exceto que os argumentos são invertidos.

LCASE(str)

Retorna o string str com os caracteres convertidos para minúsculas.

LEFT(str,tam)

Retorna tam caracteres iniciais do string str.

LENGTH(str)

Retorna o número de caracteres do string str.

LOCATE(substr,str,pos)

Retorna a posição da primeira ocorrência do substring substr no string str, iniciando a pesquisa a partir da posição pos. Retornará 0 se substr não estiver contido em str.

LOCATE(substr,str)

Retorna a posição da primeira ocorrência do substring substr no string str. Retornará 0 se substr não estiver contido em str.

LOWER(str)

O mesmo que LCASE().

LPAD(str, tam, padstr)

Retorna um string consistindo do valor do string str, preenchido à esquerda com o string padstr até o tamanho tam.

LTRIM(str)

Retorna o string str sem os espaços em branco iniciais.

MAKE_SET(n,str_bit0,str_bit1,...)

Retorna o valor SET (um string contendo substrings separados por vírgulas). Para cada bit ligado a n, o string correspondente ao bit (str_bit0, str_bit1,...) será incluído no resultado.

MID(str,pos,tam)

Retorna um substring do string str, a partir da posição pos e tamanho tam.

OCT(n)

Retorna um string com a representação octal do valor n. Equivale a CONV(n,10,8).

OCTET_LENGTH(str)

O mesmo que LENGTH().

POSITION(substr IN str)

O mesmo que LOCATE(substr, str).

REPEAT(str,n)

Retorna um string consistindo do string str repetido n vezes.

REPLACE(str,de_str,para_str)

Retorna um string consistindo do string str com todas as ocorrências do string de_str substituídas pelo string para_str.

REVERSE(str)

Retorna o string str com a ordem dos caracteres invertida.

RIGHT(str,tam)

Retorna tam caracteres finais do string str.

RPAD(str,tam,padstr)

Retorna um string consistindo do valor do string str, preenchido à direita com o string padstr até o tamanho tam.

RTRIM(str)

Retorna o string str sem os espaços em branco finais.

SOUNDEX(str)

Retorna o código fonético de um string. Utilizada em aplicações para pesquisar nomes com sons parecidos.

SPACE(n)

Retorna um string com n espaços.

SUBSTRING(str,pos,tam)

O mesmo que MID(str, pos, tam).

SUBSTRING(str FROM pos FOR tam)

O mesmo que MID(str, pos, tam).

SUBSTRING(str,pos) ou SUBSTRING(str FROM pos)

Retorna um substring do string str iniciando na posição pos.

SUBSTRING_INDEX(str,delim,n)

Retorna um substring do string str após n ocorrências de delim. Se n for positivo, retornará os caracteres à esquerda do delimitador delim encontrado; se negativo, retornará os caracteres à direita do delimitador delim. Se delim não for encontrado retornará o string inteiro.

TRIM([[opção] [remstr] FROM] str)[1.2.1][12]

Retorna o string str com os caracteres remstr iniciais e/ou finais removidos.

Parâmetro	Descrição
opção	Indica a localização dos caracteres a serem removidos.
BOTH	Remove os caracteres remstr iniciais e finais de str (default).
LEADING	Remove os caracteres remstr iniciais de str.
TRAILING	Remove os caracteres remstr finais de str.
remstr	String com os caracteres a serem removidos de str. Se não especificado removerá os caracteres em banco.
str	String fonte.

UCASE(str)

Retorna o string str com os caracteres convertidos para maiúsculas.

UPPER(str)

O mesmo que UCASE().

Funções de Data e Hora

ADDDATE(datainicial, INTERVAL expr tipo)

O mesmo que DATE_ADD().

CURDATE()

Retorna a data corrente como um string no formato "AAAA-MM-DD", ou como um número no formato AAAAMMDD, dependendo do contexto utilizado.

CURRENT_DATE

O mesmo que CURDATE(), exceto que não utiliza parênteses ().

CURRENT_TIME

O mesmo que CURTIME(), exceto que não utiliza parênteses ().

CURRENT_TIMESTAMP

O mesmo que NOW(), exceto que não utiliza parênteses ().

CURTIME()

Retorna o horário corrente como um string no formato "HH:MM:SS", ou como um número no formato HHMMSS, dependendo do contexto utilizado.

DATE_ADD(datainicial, INTERVAL expr intervalo)

Retorna o resultado de uma data (ou de uma data e hora) adicionada de um intervalo especificado.

Argumento	Descrição
datainicial	Data inicial no formato DATETIME ou DATE.
expr	Valor do intervalo a ser adicionado à data inicial. Pode ser especificado como um número ou um string. Por exemplo: INTERVAL "2:3" YEAR_MONTH.
intervalo	Especifica como interpretar o intervalo.

Intervalo	Significado	Formato do valor
SECOND	Segundos	ss
MINUTE	Minutos	mm
HOUR	Horas	hh
DAY	Dias	DD
MONTH	Meses	MM
YEAR	Anos	YY
MINUTE_SECOND	Minutos e segundos	"mm:ss"
HOUR_MINUTE	Horas e minutos	"hh:mm"
HOUR_SECOND	Horas, minutos e segundos	"hh:mm:ss"
DAY_HOUR	Dias e horas	"DD hh"
DAY_MINUTE	Dias, horas e minutos	"DD hh:mm"
DAY_SECOND	Dias, horas, minutos e segundos	"DD hh:mm:ss"
YEAR_MONTH	Anos e meses	"YY-MM"

DATE_FORMAT(data,formato)

Retorna uma data formatada. Os seguintes especificadores podem ser utilizados no string de formatação.

Especificador	Significado
%S, %s	Segundos (00..59)
%i	Minutos (00..59)
%H	Hora, no formato numérico - 24 horas (00..23)
%h, %I	Hora, no formato numérico - 12 horas (01..12)
%k	Hora, no formato numérico - 24 horas (0..23)
%l	Hora, no formato numérico - 12 horas (1..12)
%T	Horário no formato 24 horas (hh:mm:ss)
%r	Horário no formato 12 horas (hh:mm:ss AM/PM)
%p	AM ou PM
%W	Dia da semana por extenso (Sunday..Saturday)
%a	Dia da semana abreviado (Sun..Sat)
%d	Dia do mês (00..31)
%e	Dia do mês (0..31)
%D	Dia do mês com sufixo inglês (1st, 2nd, 3rd, ...)
%w	Dia da semana (0=Domingo..6=Sábado)
%j	Dia do ano (001..366)
%U	Semana (0..52), onde Domingo é o primeiro dia da semana
%u	Semana (0..52), onde Segunda é o primeiro dia da semana
%M	Nome do mês por extenso (January..December)
%b	Nome do mês abreviado (Jan..Dec)
%m	Mês (01..12)
%c	Mês (1..12)
%Y	Ano (4 dígitos)
%y	Ano (2 dígitos)
%%	Um literal '%'.

Todos os outros caracteres são apenas copiados para o resultado sem nenhuma interpretação.

DATE_SUB(datainicial, INTERVAL expr tipo)

Retorna o resultado de uma data (ou de uma data e hora) subtraído um intervalo de tempo especificado.

Argumento	Descrição
datainicial	Data inicial no formato DATETIME ou DATE.
expr	Valor do intervalo a ser subtraído da data inicial. Pode ser especificado como um número ou um string. Por exemplo: INTERVAL "2:3" YEAR_MONTH.
intervalo	Especifica como interpretar o intervalo. Veja a tabela na função DATE_ADD().

DAYNAME(data)

Retorna o dia da semana por extenso de uma data.

DAYOFMONTH(data)

Retorna o dia do mês (1-31) de uma data.

DAYOFWEEK(data)

Retorna o valor numérico do dia da semana de uma data (1=Domingo,...,7=Sábado).

DAYOFYEAR(data)

Retorna o dia do ano (1-366) de uma data.

EXTRACT(intervalo FROM data)

Retorna a parte da data indicada por intervalo. Veja na função DATE_ADD() os especificadores para intervalo.

FROM_DAYS(numdias)

Retorna a data correspondente a um número de dias desde o ano 0. Somente para datas do calendário Gregoriano (1582 em diante).

FROM_UNIXTIME(unix_timestamp)

Retorna a representação do argumento unix_timestamp, como uma data e hora no formato "AAAA-MM-DD HH:MM:SS", ou como um número no formato AAAAMMDDHHMMSS, dependendo do contexto utilizado.

FROM_UNIXTIME(unix_timestamp,formato)

Retorna a representação string do Unix timestamp de acordo com o formato especificado.

Argumento	Descrição
formato	Utiliza o mesmos especificadores da função DATE_FORMAT().

HOUR(horário)

Retorna as horas (0-23) de um horário.

MINUTE(horário)

Retorna os minutos (0-59) de um horário.

MONTH(data)

Retorna o mês (1-12) de uma data.

MONTHNAME(data)

Retorna o mês por extenso de uma data.

NOW()

Retorna a data e hora corrente como um string no formato "AAAA-MM-DD HH:MM:SS", ou como um número no formato AAAAMMDDHHMMSS, dependendo do contexto utilizado.

PERIOD_ADD(período, n)

Retorna o resultado da adição de n meses ao período especificado. O formato do resultado é AAAAMM. O formato do período pode ser AAAAMM ou AAMM.

PERIOD_DIFF(per1,per2)

Retorna o número de meses entre os períodos per1 e per2. O formato dos argumentos per1 e per2 pode ser AAAAMM ou AAMM.

QUARTER(data)

Retorna o trimestre (1-4) do ano de uma data.

SEC_TO_TIME(segundos)

Retorna o argumento segundos, convertido para horas, minutos e segundos, como um string no formato "HH:MM:SS", ou como um número no formato HHMMSS, dependendo do contexto utilizado.

SECOND(horário)

Retorna os segundos (0-59) de um horário.

SUBDATE(data_inicial, INTERVAL expr tipo)

O mesmo que DATE_SUB().

SYSDATE()

O mesmo que NOW().

TIME_FORMAT(horário,formato)[3][21.3]

Similar à função DATE_FORMAT(), mas o string formato pode conter somente especificadores de formato que manipulem horas, minutos e segundos. Outros especificadores retornam um valor NULL ou 0.

TIME_TO_SEC(horário)

Retorna um horário ("hh:mm:ss") convertido para o número de segundos.

TO_DAYS(data)

Retorna o número de dias desde a data especificada até o ano 0.

UNIX_TIMESTAMP([data])

Retorna o número de segundos desde a data especificada, ou desde "1970-01-01 00:00:00 GMT" se a data não for especificada.

Argumento	Descrição
data	Pode ser um string DATE, um string DATETIME, um TIMESTAMP, ou um número no formato AAMMDD ou AAAAMMDD na hora local.

WEEK(data)[3.21.22]

Retorna o número da semana (0-52) de uma data, sendo Domingo o primeiro dia da semana.

WEEK(data, primeiro)[3.21.1]

Retorna o número da semana (0-52) de uma data. O argumento primeiro especifica se a semana inicia no Domingo (0) ou na Segunda (1).

WEEKDAY(data)

Retorna o valor numérico do dia da semana de uma data (0=Segunda,...,6=Domingo).

YEAR(data)[3.21.22]

Retorna o ano de uma data, no intervalo 1000 a 9999.

Miscelânea

BENCHMARK(n,expr)

Executa n vezes uma expressão. Utilizada para medir o tempo necessário para o utilitário cliente mysql processar uma expressão para análise de desempenho.

BIT_COUNT(n)

Retorna o número de bits ligados ao argumento n (BIGINT de 64 bits).

DATABASE()

Retorna o nome do banco de dados corrente, ou um string vazio se não existir.

DECODE(str_cript,senha)

Descriptografa str_cript utilizando a senha especificada. Contrasta com ENCODE().

ENCODE(str,senha)

Retorna um string binário com o string str criptografado utilizando a senha especificada. Para descriptografar o resultado, utilize a função DECODE().

ENCRYPT(str)

Retorna o string str criptografado através da system call crypt() do Unix. Se crypt() não estiver disponível, ENCRYPT() retornará NULL.

ENCRYPT(str, salt)

Retorna o string str criptografado através da system call crypt() do Unix. Se crypt() não estiver disponível, ENCRYPT() retornará NULL.

Parâmetro	Significado
salt	String de dois caracteres nas versões anteriores a versão 3.22.16. A especificação de salt faz com que o resultado da criptografia do string str seja sempre o mesmo.

GET_LOCK(str,timeout)

Tenta efetuar um lock em str, no tempo especificado por timeout (em segundos). Retornará 1 se tiver sucesso, 0 se tiver ocorrido timeout, NULL se tiver ocorrido um erro. O lock é desbloqueado quando um RELEASE_LOCK() ou um novo GET_LOCK() é executado, ou quando o thread termina.

LAST_INSERT_ID()

Retorna o último valor AUTO_INCREMENT gerado na sessão corrente, ou 0 se tal valor não tiver sido gerado.

LAST_INSERT_ID(expr)

Retorna o valor de expr como se tivesse sido gerado automaticamente para AUTO_INCREMENT.

LOAD_FILE(nomearq)

Lê um arquivo no servidor e retorna o seu conteúdo em um string. Você deve ter o privilégio FILE. Retornará NULL se o arquivo não puder ser lido por alguma razão. O tamanho do arquivo deve ser menor que max_allowed_packet.

MD5(string)

Retorna um checksum (hexa 32 long) baseado no algoritmo MD5 (Message-Digest Algorithm da RSA) do string.

PASSWORD(str)

Retorna uma senha criptografada a partir do string str. Não utiliza o mesmo algoritmo para criptografia do Unix.

RELEASE_LOCK(str)

Desbloqueia o lock do nome str criado com a função GET_LOCK(). Retornará 1 se o lock tiver sido desbloqueado, 0 se o lock não tiver sido bloqueado por este thread (continua bloqueado) e NULL se o o lock str não existir.

SYSTEM_USER()

O mesmo que USER().

SESSION_USER()

O mesmo que USER().

USER()

Retorna o nome do usuário corrente do MySQL, no formato "usuário@host".

VERSION()

Retorna um string indicando a versão do servidor MySQL e um dos seguintes sufixos:

Sufixo	Significado
-log	O logging está ativado.
-debug	servidor executado em modo debug.
-demo	O servidor está sendo executado no modo demo.
-shareware	A versão do servidor é shareware.

Comentários no código SQL

Os caracteres #, -- e /* */ são utilizados para indicar comentários no código SQL:

```
#   comentário para o fim da linha
--   comentário para o fim da linha. Requer pelo menos um espaço após --
/* comentário de uma única linha */
/*
este comentário ocupa
mais de uma linha
*/
```

Existem algumas limitações no uso de /*...*/:

- Caracteres como apóstrofo e aspas indicam o início ou o fim de um string, mesmo dentro do comentário.
- Ponto e vírgula indicam o final do comando. Tudo que estiver após o ponto e vírgula é interpretado como um novo comando.

Tipo especial de comentário

A partir do MySQL 3.23 é possível desabilitar recursos específicos do MySQL, utilizando a forma especial de comentário /*! ... */ dentro de um comando. O sinal de ! indica que o MySQL deve interpretar e executar o código dentro dos comentários, enquanto outros servidores ignoram o que estiver entre /* e */. Por exemplo:

```
SELECT /*! STRAIGHT_JOIN */ coluna
    FROM tabela1, tabela2 WHERE ...
```

No exemplo acima somente o MySQL considerará a opção STRAIGHT_JOIN.

Também é possível desabilitar recursos de versões antigas do MySQL. Se você adicionar o número da versão após o sinal '!', a sintaxe será executada somente se a versão do MySQL for igual ou mais recente que o número da versão utilizada:

```
CREATE /*!32302 TEMPORARY */ TABLE (a int);
```

O comando acima significa que o MySQL utilizará a palavra-chave TEMPORARY se a versão for 3.23.02 ou mais recente.

Tipos de Colunas

As seguintes convenções são utilizadas na descrição dos tipos de colunas:

Convenção	Significado
M	Representa o número de caracteres usados para exibir os valores da coluna (1 a 255 caracteres).
D	Número de casas decimais em campos no formato ponto flutuante (0 a 30). Não pode ser maior que M-2; caso contrário, o valor de M será ajustado para D+2.
L	Indica o tamanho da informação armazenada no campo STRING.
NATIONAL[3.23.5]	Especificação ANSI SQL que define o grupo de caracteres padrão a ser utilizado.
UNSIGNED	Modificador que faz com que uma coluna aceite apenas os números positivos (sem sinal). Somente para tipos Integer.
ZEROFILL	Adiciona zeros de preenchimento à esquerda de cada número. Por exemplo: um campo INT(5) ZEROFILL exibirá o número 235 como 00235. Observe que se for especificado ZEROFILL para uma coluna, MySQL automaticamente adicionará o atributo UNSIGNED à coluna.
AUTO_INCREMENT	Tipo especial de coluna numérica que pode ser atualizada automaticamente. Quando um valor NULL (ou 0) for inserido na coluna, o MySQL o substituirá automaticamente pelo próximo número na seqüência da coluna.

Tipos de Dados do MySQL

Numéricos

TINYINT[(M)]

Um inteiro de tamanho muito pequeno.

Atributos: AUTO_INCREMENT, UNSIGNED, ZEROFILL.

Intervalo: Entre -128 e 127 com sinal,
ou entre 0 e 255 sem sinal.

Espaço req: 1 byte.

SMALLINT[(M)]

Um inteiro pequeno.

Atributos: AUTO_INCREMENT, UNSIGNED, ZEROFILL.

Intervalo: Entre -32768 e 32767 com sinal,
ou entre 0 e 65535 sem sinal.

Espaço req: 2 bytes.

MEDIUMINT[(M)]

Um inteiro de tamanho médio.

Atributos: AUTO_INCREMENT, UNSIGNED, ZEROFILL.

Intervalo: Entre -8388608 e 8388607 com sinal, ou entre 0 e 16777215 sem sinal.

Espaço req: 3 bytes

INT[(M)]

Um inteiro de tamanho normal.

Atributos: AUTO_INCREMENT, UNSIGNED, ZEROFILL.

Intervalo: Entre -2147483648 e 2147483647 com sinal, ou entre 0 e 4294967295 sem sinal.

Espaço req: 4 bytes.

INTEGER[(M)]

Sinônimo de INT.

BIGINT[(M)]

Um inteiro grande.

Atributos: AUTO_INCREMENT, UNSIGNED, ZEROFILL.

Intervalo: Entre -9223372036854775808 e 9223372036854775807 com sinal, ou entre 0 e 18446744073709551615 sem sinal.

Espaço req: 8 bytes.

FLOAT[(M,D)]

Um número de ponto-flutuante com precisão simples.

Atributos: ZEROFILL.

Intervalo: Entre -3.402823466E+38 e -1.175494351E-38, 0 (zero) e entre 1.175494351E-38 e 3.402823466E+38.

Espaço req: 4 bytes

DOUBLE[(M,D)]

Um número de ponto-flutuante com precisão dupla.

Atributos: ZEROFILL.

Intervalo: Entre -1.7976931348623157E+308 e -2.2250738585072014E-308, 0 (zero) e entre 2.2250738585072014E-308 e 1.7976931348623157E+308.

Espaço req: 8 bytes.

DOUBLE PRECISION[(M,D)]

Sinônimo de DOUBLE.

REAL[(M,D)]

Sinônimo de DOUBLE.

DECIMAL[(M [,D])]

Um número de ponto-flutuante armazenado com um string (1 byte por dígito, ponto decimal, ou sinal "-").

Atributos: ZEROFILL

Intervalo: O intervalo máximo é o mesmo de DOUBLE.

Espaço req: M+2 bytes

NUMERIC(M,D)

Sinônimo de DECIMAL.

Data e Hora

DATE

Uma data no formato "YYYY-MM-DD".

Intervalo: Entre '1000-01-01' e '9999-12-31'.

Espaço req: 3 bytes.

DATETIME

Uma combinação de data e hora no formato "YYYY-MM-DD HH:MM:SS".

Espaço req: 8 bytes.

Intervalo: Entre "1000-01-01 00:00:00" e "9999-12-31 23:59:59".

TIMESTAMP[(M)]

Um timestamp (combinação da data e hora corrente) no formato YYYYMMDDhhmmss.

Intervalo: Entre 19700101000000 e algum dia do ano 2037.

Espaço req: 4 bytes.

Valor M	Formato
TIMESTAMP(14)	YYYYMMDDHHMMSS
TIMESTAMP(12)	YYMMDDHHMMSS
TIMESTAMP(10)	YYMMDDHHMM
TIMESTAMP(8)	YYYYMMDD
TIMESTAMP(6)	YYMMDD
TIMESTAMP(4)	YYMM
TIMESTAMP(2)	YY

TIME

Um horário no formato "HH:MM:SS".

Intervalo: Entre '-838:59:59' e '838:59:59'.

Espaço req: 3 bytes.

YEAR [3.22]

Um ano no formato YYYY.

Intervalo: 1900 a 2155.

Espaço req: 1 byte.

String

Os tipos CHAR, VARCHAR e TEXT não distinguem minúsculas de maiúsculas em comparações, exceto se o atributo BINARY for especificado. Os tipos BLOB distinguem minúsculas de maiúsculas.

CHAR(M)

Um string de caracteres de tamanho fixo (0 a 255 caracteres).

Atributos: BINARY.

Espaço req: M bytes.

VARCHAR(M)

Um string de caracteres de tamanho variável (0 a 255 caracteres).

Atributos: BINARY.

Espaço req: Tamanho do valor + 1 byte para registrar o tamanho.

TINYBLOB

Um valor BLOB pequeno (0 a 255 bytes).

Espaço req: Tamanho do valor + 1 byte para registrar o tamanho.

BLOB

Um valor BLOB de tamanho normal (0 a 65535 bytes).

Espaço req: Tamanho do valor + 2 bytes para registrar o tamanho.

MEDIUMBLOB

Um valor BLOB de tamanho médio (0 a 16777215 bytes).

Espaço req: Tamanho do valor + 3 bytes para registrar o tamanho.

LONGBLOB

Um valor BLOB grande (0 a 4294967295 bytes).

Espaço req: Tamanho do valor + 4 bytes para registrar o tamanho.

TINYTEXT

Um valor TEXT pequeno (0 a 255 bytes).

Espaço req: Tamanho do valor + 1 byte para registrar o tamanho.

TEXT

Um valor TEXT de tamanho normal (0 a 65535 bytes).

Espaço req: Tamanho do valor +21 bytes para registrar o tamanho.

MEDIUMTEXT

Um valor TEXT de tamanho médio (0 a 16777215 bytes).

Espaço req: Tamanho do valor + 3 bytes para registrar o tamanho.

LONGTEXT

Um valor TEXT grande (0 a 4294967295 bytes).

Espaço req: Tamanho do valor + 4 bytes para registrar o tamanho.

ENUM('valor1','valor2',...)

Uma enumeração. Permite especificar uma lista de valores, dos quais um pode ser atribuído à coluna.

Espaço req: 1 byte para enum de 1 a 255 valores;

2 bytes para enum de 256 a 65535 valores.

SET('valor1','valor2',...)

Um grupo. Permite especificar uma lista de valores dos quais um ou mais podem ser atribuídos a uma coluna.

Espaço req: 1 byte para setS de 1 a 8 membros;

2 bytes para setS de 9 a 16 membros;

3 bytes para setS de 17 a 24 membros;

4 bytes para setS de 25 a 32 membros;

8 bytes para setS de 33 a 64 membros.

Literais

Strings

Um string é uma seqüência de caracteres delimitados por apóstrofe ou aspas. Exemplos:

'um string'

“outro string”

Dentro de um string, certas seqüências têm um significado especial. Cada uma dessas seqüências começa com uma barra invertida (\) conhecida como caractere escape.

Caractere	Significado
\0	Um caractere ASCII 0 (NUL).
\n	Um caractere Newline.
\t	Um caractere Tab.
\r	Um caractere Carriage Return.
\b	Um caractere Backspace.
\'	Um caractere apóstrofe.
\”	Um caractere aspas.
\\	Um caractere barra invertida ('\').
\%	Um caractere '%' . O caractere '%' é utilizado na cláusula WHERE .
_	Um caractere '_' . O caractere '_' é utilizado na cláusula WHERE .

Existem várias maneiras de incluir aspas dentro de um string:

- Um caractere apóstrofe dentro de um string entre apóstrofes pode ser escrito como ''. Exemplo (1).
- Um caractere aspas dentro de um string entre aspas pode ser escrito como “”. Exemplo (2).
- Preceder o caractere aspas com um caractere escape ('\'). Exemplo (3).
- Um caractere apóstrofe dentro de um string entre aspas ou um caractere aspas dentro de um string entre apóstrofes não precisam de tratamento diferenciado.

Na inserção de dados binários em uma coluna BLOB, os seguintes caracteres devem ser representados por seqüências escape:

Caractere	Significado
NUL	ASCII 0. Inserido através de '\0'.
\	ASCII 92. Barra invertida, inserido através de '\\'.
'	ASCII 39. Apóstrofe, inserido através de '\''.
“	ASCII 34. Aspas, inserido através de '\”'.

Números

Inteiros são representados como uma seqüência de dígitos. Floats usam '.' como separador decimal.

Exemplos de números inteiros válidos:

1560

0

-11

Exemplos de números de ponto-flutuante válidos:

15.34

-1203.7381e+10

142.00

Um inteiro pode ser utilizado em um contexto de ponto-flutuante, mas será interpretado com um número de ponto-flutuante.

Valores Hexadecimais

O MySQL suporta valores hexadecimais. Em contexto de números funciona como um integer (precisão de 64 bits). Em contexto de strings funciona como um string binário, onde cada par de dígito hexa é convertido para um caractere.

```
mysql> SELECT 0xb+0
    > 11
mysql> select 0x64656667;
    > defg
```

Valores NULL

O valor NULL significa “sem dados” e é diferente de valores tais como: 0 para tipos numéricos ou string vazio para tipos string.

NULL pode ser representado por \N em formatos de arquivo texto (LOAD DATA INFILE, SELECT ... INTO OUTFILE).

Palavras Reservadas do MySQL

action
add
aggregate
all
alter
after
and
as
asc
avg
avg_row_length
auto_increment
between
bigint
bit
binary
blob
bool
both
by
cascade
case
char
character
change
check
checksum
column
columns
comment
constraint
create
cross
current_date
current_time
current_timestamp
data
database
databases
date
datetime
day
day_hour
day_minute
day_second
dayofmonth
dayofweek
dayofyear
dec
decimal
default
delayed
delay_key_write
delete
desc
describe
distinct
distinctrow
double
drop
end
else
escape
escaped
enclosed
enum
explain
exists
fields
file
first
float
float4
float8
flush
foreign
from
for
full
function
global
grant
grants
group
having
heap
high_priority
hour
hour_minute
hour_second
hosts
identified
ignore
in
index
infile
inner
insert
insert_id
int
integer
interval
int1
int2
int3
int4
int8
into
if
is
isam
join
key
keys
kill
last_insert_id
leading
left
length
like
lines
limit
load
local
lock
logs
long
longblob
longtext
low_priority
max
max_rows
match
mediumblob
mediumtext
mediumint
middleint
min_rows
minute
minute_second
modify
month
monthname
myisam
natural
numeric
no
not
null
on
optimize
option
optionally
or
order
outer
outfile
pack_keys
partial
password
precision
primary
procedure
process
processlist
privileges
read
real
references
reload
regexp
rename
replace
restrict
returns
revoke
rlike
row
rows
second
select
set
show
shutdown
smallint
soname
sql_big_tables
sql_big_selects
sql_low_priority_updates
sql_log_off
sql_log_update
sql_select_limit
sql_small_result
sql_big_result
sql_warnings
straight_join
starting
status
string
table
tables
temporary
terminated
text
then
time
timestamp
tinyblob
tinytext
tinyint
trailing
to
type
use
using
unique
unlock
unsigned
update
usage
values
varchar
variables
varying
varbinary
with
write
when
where
year
year_month
zerofill