链滴

MySQL 配置文件 my.cnf / my.ini 逐行详解

作者：HiJiangChuan

原文链接：https://ld246.com/article/1638717185397

来源网站：链滴

本文首发：MySQL 配置文件 my.cnf / my.ini 逐行详解 - 卡拉云

MySQL 配置文件的意义

充分理解 MySQL 配置文件中各个变量的意义对我们有针对性的优化 MySQL 数据库性能有非常大的义。我们需要根据不同的数据量级，不同的生产环境情况对 MySQL 配置文件进行优化。

Windows 和 Linux 下的 MySQL 配置文件的名字和存放位置都是不同的，WIndows 下 MySQL 配文件是 my.ini 存放在 MySQL 安装目录的根目录下；Linux 下 MySQL 配置文件是 my.cnf 存放在 /et /my.cnf、/etc/mysql/my.cnf。我们也可以通过 find 命令进行查找。

另外要注意的是，通过 rpm 命令安装的 MySQL 是没有 /etc/my.cnf 文件的，如果需要配置 MySQL 可以在/etc/my.cnf新建配置文件，然后把本文的配置信息复制到文件中即可。

本教程将带领大家逐条解析最新的 MySQL 8.0 的配置文件，争取搞懂每一条变量。当然，我们理解变量的意义外，更重要的是针对自己的数据库外部环境，在实践中进行微调，以达到优化性能的目的。

提示：你可以使用 Ctrl+F 快速定位。 MySQL 配置文件详解

- [client]
- [mysqld_safe]
- [mysqld]
- Query Cache MySQL 错误日志设置 慢查询记录 全局查询日志
- 从属线程变量
- 安全变量
- MyISAM 变量

- MEMORY 变量
- InnoDB 变量 WSREP 配置

MySQL 配置文件详解

文件位置: Windows、Linux、Mac 有细微区别，Windows 配置文件是 .ini，Mac/linux 是 .cnf

[Windows]
MySQL\MySQL Server 5.7\my.ini

[Linux / Mac]
/etc/my.cnf
/etc/mysql/my.cnf

当然我们也可以使用命令来查看 MySQL 默认配置文件位置

mysql --help|grep 'cnf'

[client]

客户端设置。当前为客户端默认参数

port = 3306

默认连接端口为 3306

socket = /tmp/mysql.sock

本地连接的 socket 套接字

default_character_set = utf8

设置字符集，通常使用 uft8 即可

[mysqld_safe]

mysqld_safe 是服务器端工具，用于启动 mysqld，也是 mysqld 的守护进程。当 mysql 被 kill 时，ysqld_safe 负责重启启动它。

open_files_limit = 8192

此为 MySQL 打开的文件描述符限制，它是 MySQL 中的一个全局变量且不可动态修改。它控制着 my

qld 进程能使用的最大文件描述符数量。默认最小值为 1024

需要注意的是这个变量的值并不一定是你在这里设置的值，mysqld 会在系统允许的情况下尽量取最值。

当 open_files_limit 没有被配置时，比较 max_connections*5 和 ulimit -n 的值，取最大值

当 open_file_limit 被配置时，比较 open_files_limit 和 max_connections*5 的值，取最大值

user = mysql

用户名

log-error = error.log

错误 log 记录文件

[mysqld]

服务端基本配置

port = 3306

mysqld 服务端监听端口

socket = /tmp/mysql.sock

MySQL 客户端程序和服务器之间的本地通讯指定一个套接字文件

max_allowed_packet = 16M

允许最大接收数据包的大小，防止服务器发送过大的数据包。

当发出长查询或 mysqld 返回较大结果时，mysqld 才会分配内存，所以增大这个值风险不大，默认 1 M，也可以根据需求改大，但太大会有溢出风险。取较小值是一种安全措施，避免偶然出现但大数据导致内存溢出。

default_storage_engine = InnoDB

创建数据表时，默认使用的存储引擎。这个变量还可以通过 –default-table-type 进行设置

max_connections = 512

最大连接数，当前服务器允许多少并发连接。默认为 100，一般设置为小于 1000 即可。太高会导致存占用过多，MySQL 服务器会卡死。作为参考，小型站设置 100 - 300

max_user_connections = 50

用户最大的连接数，默认值为 50 一般使用默认即可。

thread_cache_size = 64

线程缓存，用于缓存空闲的线程。这个数表示可重新使用保存在缓存中的线程数，当对方断开连接时如果缓存还有空间，那么客户端的线程就会被放到缓存中，以便提高系统性能。我们可根据物理内存对这个值进行设置，对应规则 1G 为 8；2G 为 16；3G 为 32；4G 为 64 等。

Query Cache

query_cache_type = 1

设置为 0 时，则禁用查询缓存（尽管仍分配query_cache_size个字节的缓冲区）。

设置为 1 时 ，除非指定SQL_NO_CACHE，否则所有SELECT查询都将被缓存。

设置为 2 时，则仅缓存带有SQL CACHE子句的查询。

请注意，如果在禁用查询缓存的情况下启动服务器，则无法在运行时启用服务器。

query_cache_size = 64M

缓存select语句和结果集大小的参数。

查询缓存会存储一个select查询的文本与被传送到客户端的相应结果。

如果之后接收到一个相同的查询，服务器会从查询缓存中检索结果，而不是再次分析和执行这个同样查询。

如果你的环境中写操作很少，读操作频繁，那么打开query_cache_type=1，会对性能有明显提升。果写操作频繁，则应该关闭它（query_cache_type=0）。

Session variables

sort_buffer_size = 2M

MySQL 执行排序时，使用的缓存大小。增大这个缓存，提高 group by，order by 的执行速度。

tmp_table_size = 32M

HEAP 临时数据表的最大长度，超过这个长度的临时数据表 MySQL 可根据需求自动将基于内存的 HEP 临时表改为基于硬盘的 MyISAM 表。我们可通过调整 tmp_table_size 的参数达到提高连接查询速的效果。

read_buffer_size = 128k

MySQL 读入缓存的大小。如果对表对顺序请求比较频繁对话，可通过增加该变量值以提高性能。

read_rnd_buffer_size = 256k

用于表的随机读取，读取时每个线程分配的缓存区大小。默认为 256k ，一般在 128 - 256k之间。在 order by 排序操作时，会用到 read_rnd_buffer_size 空间来暂做缓冲空间。

join_buffer_size = 128k

程序中经常会出现一些两表或多表 Join （联表查询）的操作。为了减少参与 Join 连表的读取次数以高性能，需要用到 Join Buffer 来协助 Join 完成操作。当 Join Buffer 太小时，MySQL 不会将它写磁盘文件。和 sort_buffer_size 一样，此参数的内存分配也是每个连接独享。

table_definition_cache = 400

限制不使用文件描述符存储在缓存中的表定义的数量。

```
table_open_cache   = 400
```

限制为所有线程在内存中打开的表数量。

扩展阅读：《在 MySQL 中 DATETIME 和 TIMESTAMP 时间类型的区别及使用场景》

MySQL 错误日志设置

```
log_error = error.log
log_warnings = 2
```

- log_warnings 为0， 表示不记录告警信息。
- log_warnings 为1， 表示告警信息写入错误日志。
- log_warnings 大于1， 表示各类告警信息，例如有关网络故障的信息和重新连接信息写入错误日
。

扩展阅读：《MySQL「 Every derived table must have its own alias」错误 ERROR 1248 修复方法》

MySQL 慢查询记录

```
slow_query_log_file = slow.log
slow_query_log  = 0
log_queries_not_using_indexes  = 1
long_query_time = 0.5
min_examined_row_limit = 100
```

slow_query_log ： 全局开启慢查询功能。

slow_query_log_file ： 指定慢查询日志存储文件的地址和文件名。

log_queries_not_using_indexes：无论是否超时，未被索引的记录也会记录下来。

long_query_time：慢查询阈值（秒），SQL 执行超过这个阈值将被记录在日志中。

min_examined_row_limit：慢查询仅记录扫描行数大于此参数的 SQL。

关于 MySQL 慢查询日志更多扩展内容，请看《如何使用慢查询日志对 MySQL 进行性能优化 - Profiling、mysqldumpslow 实例详解》

MySQL 全局查询日志

```
general_log_file = general.log
general_log = 0
```

这一段比较好理解，存放文件名，是否开启日志记录

```
Binary logging and Replication

server_id  = 42
log_bin  = mysql-bin
binlog_cache_size  = 1M
```

控制二进制日志缓存大小，增加其值可改善处理大事务的系统的性能。在具有大量数据库连接的环境应限制该值。

binlog_stmt_cache_size = 1M

如果二进制日志处于活动状态，则此变量确定在每次事务中保存二进制日志更改记录的缓存的每个连的字节大小。单独的变量binlog_stmt_cache_size设置了语句缓存的上限。

该binlog_cache_disk_use 和 binlog_cache_use 服务器状态变量将显示这个变量是否需要增加。

max_binlog_size = 128M

如果二进制日志在写入后超出此大小，则服务器会通过关闭它并打开新的二进制日志来旋转它。

单个事务将始终存储在同一二进制日志中，因此服务器将等待未完成的事务在轮换之前完成。

如果将 max_relay_log_size 设置为 0，此图也适用于中继日志的大小。

sync_binlog = 0

控制 binlog 写磁盘频率

expire_logs_days = 5

自动二进制日志文件删除的天数。默认值为 0，table 示“不自动删除”。在启动时和清除二进制日时，可能会删除它们。

binlog_format = ROW

此变量设置二进制日志记录格式，并且可以是 STATEMENT，ROW 或 MIXED 三选一

binlog_row_image = MINIMAL

对于 MySQL 基于行的复制，此变量确定如何将行图像写入二进制日志。

从属线程变量

log_slave_updates = 1

如果设置为 0 （默认值），则复制期间从主服务器接收到的从服务器上的更新不会记录在从服务器的进制日志中。如果设置为 1 ，则为。需要启用从站的二进制日志才能生效。

read_only = 0
skip_slave_start = 0

安全变量

local_infile = 0

此变量控制LOAD DATA语句的服务器端LOCAL功能。根据local_infile设置，服务器会拒绝或允许 Clint 端启用LOCAL的 Client 端加载本地数据。

#secure_auth = 1
#sql_mode = TRADITIONAL,ANSI,ONLY_FULL_GROUP_BY

#skip_name_resolve = 0

检查 Client 端连接时是否解析主机名。如果此变量是 0 ， 则 mysqld 在检查 Client 端连接时解析主名。

如果是 1 ， 则 mysqld 仅使用 IPNumbers；在这种情况下，授权 table 中的所有Host列值都必须是 I 地址

MyISAM 变量

key_buffer_size = 8M

MyISAM table 的索引块被缓冲并由所有线程共享。

key_buffer_size 是用于索引块的缓冲区的大小。密钥缓冲区也称为密钥缓存。

myisam_recover = BACKUP,FORCE

设置MyISAM存储引擎恢复模式。变量值是 OFF，DEFAULT，BACKUP，FORCE或QUICK的值的任组合。

如果指定多个值，请用逗号分隔。在服务器启动时指定没有值的变量与指定DEFAULT相同，指定显值""会禁用恢复(与OFF的值相同)。

如果启用了恢复，则每次 mysqld 打开 MyISAMtable 时，它都会检查该 table 是否标记为已崩溃或正确关闭。

(只有在禁用外部锁定的情况下运行，最后一个选项才起作用.)在这种情况下，mysqld在 table 上运行查。如果 table 已损坏，mysqld尝试修复它。

MEMORY 变量

max_heap_table_size = 64M

此变量设置允许用户创建的 MEMORY table 增长的最大大小。

变量的值用于计算MEMORYtableMAX_ROWS的值。除非使用诸如CREATE TABLE之类的语句重新建该 table 或使用ALTER TABLE或TRUNCATE TABLE对其进行更改，否则设置此变量对任何现有的MMORYtable 均无效。

服务器重新启动还会将现有 MEMORYtable 的最大大小设置为全局max_heap_table_size值。

InnoDB 变量

innodb_buffer_pool_size = 128M

控制缓存表和索引数据的 InnoDB 缓冲池的内存大小

innodb_file_per_table = 1

此为独立表空间模式，每个数据库的每个表都会生成一个数据空间。当删除或截断一个数据库表时，也可以回收未使用的空间。这样配置的另一个好处是你可以将某些数据库表放在一个单独的存储设备这可以大大提升你磁盘的I/O负载。

独立表空间优点： 每个表都有自已独立的表空间。 每个表的数据和索引都会存在自已的表空间中。 以实现单表在不同的数据库中移动。 空间可以回收（除drop table操作处，表空不能自已回收）

缺点：单表增加过大，如超过100G

结论：共享表空间在Insert操作上少有优势。其它都没独立表空间表现好。当启用独立表空间时，请理调整：innodb_open_files

```
#innodb_buffer_pool_instances   = n
#innodb_write_io_threads        = 8
#innodb_read_io_threads         = 8
```

Innodb使用后台线程处理数据页上的读写 I/O(输入输出)请求,根据你的 CPU 核数来更改,默认是4 #注这两个参数不支持动态改变,需要把该参数加入到my.cnf里，修改完后重启MySQL服务,允许值的范围1-64

```
#innodb_io_capacity = 1000
```

innodb_io_capacity变量定义InnoDB可用的总体 I/O 容量。应该将其设置为大约系统每秒可以执行的I/O 操作数(IOPS)。设置 innodb_io_capacity时，InnoDB根据设置的值估计可用于后台任务的 I/O宽。

您可以将innodb_io_capacity设置为 100 或更大的值。默认值为200。通常，大约 100 的值适用于 Cnsumer 级别的存储设备，例如最高 7200 RPM 的硬盘驱动器。

```
innodb_flush_log_at_trx_commit = 2
```

这个选项决定着什么时候把日志信息写入日志文件以及什么时候把这些文件物理地写(术语称为”同步)到硬盘上。

当设为 0 ,log buffer每秒就会被刷写日志文件到磁盘，提交事务的时候不做任何操作（执行是由mysq的master thread线程来执行的。

当设为 1 时，每次提交事务的时候，都会将log buffer刷写到日志。

当设为 2 ,每次提交事务都会写日志，但并不会执行刷的操作。每秒定时会刷到日志文件。要注意的，并不能保证100%每秒一定都会刷到磁盘，这要取决于进程的调度。

```
innodb_log_buffer_size          = 8M
```

此参数确定些日志文件所用的内存大小，以M为单位。缓冲区更大能提高性能，但意外的故障将会丢数据。事务日志所使用的缓存区。InnoDB在写事务日志的时候为了提高性能，先将信息写入Innodb og Buffer中，当满足innodb_flush_log_trx_commit参数所设置的相应条件（或者日志缓冲区写满时，再将日志写到文件（或者同步到磁盘）中。可以通过innodb_log_buffer_size参数设置其可以使的最大内存空间。默认是8MB，一般为16～64MB即可。

```
innodb_log_file_size            = 256M
```

事务日志文件写操作缓存区的最大长度。更大的设置可以提高性能，但也会增加恢复故障数据库所需时间 Galera specific MySQL parameter default_storage_engine = InnoDB 服务器启动时必须启默认存储引擎，否则服务器将无法启动。默认设置是 MyISAM。 这项设置还可以通过–default-table-ype选项来设置。

```
#innodb_flush_log_at_trx_commit = 0
```

当值为 1 时，默认情况下，日志缓冲区被写入InnoDB重做日志文件，并在每次事务处理后刷新到磁。要完全符合ACID。 当值为 0 时，提交时不做任何事情；而是将日志缓冲区每秒写入一次并刷新到InnoDB重做日志中。这样可以提供更好的性能，但是服务器崩溃可以清除事务的最后一秒。 当值为 2，每次提交后，日志缓冲区都会写入InnoDB重做日志，但刷新每秒发生一次。性能稍好一些，但是作系统或断电可能导致最后一秒的事务丢失。

#innodb_autoinc_lock_mode = 2

为InnoDB表生成 AUTO_INCREMENT 值时使用的锁定模式。 有效值为:

0 是传统锁定模式。 1 是连续锁定模式。 2 是交错锁定模式。

#binlog_format = row

#query_cache_type = 0

- 设置为 0 时，则禁用查询缓存（尽管仍分配 query_cache_size个字节的缓冲区）。
- 设置为 1 时,除非指定 SQL_NO_CACHE，否则所有SELECT查询都将被缓存。
- 设置为 2 时，则仅缓存带有 SQL CACHE子句的查询。请注意，如果在禁用查询缓存的情况下启服务器，则无法在运行时启用服务器。

扩展阅读：《如何在 MySQL / MariaDB 中查找和删除重复记录？ - 4 种 MySQL 数据去重法》

WSREP 配置

#wsrep_provider = /usr/lib/libgalera_smm.so

wsrep 库的位置

通常不同版本的 Linux 位置

- Debian 和 Ubuntu 在 /usr/lib/libgalera_smm.so
- Red Hat / CentOS 在 /usr/lib64/libgalera_smm.so

#wsrep_cluster_name = "kalacloud.com Galera Cluster"

集群的名称。节点无法连接到名称不同的集群，因此在同一集群中的所有节点上都必须相同。

#wsrep_cluster_address = "gcomm://"
#wsrep_cluster_address = "gcomm://192.168.0.2,192.168.0.3"

启动时要连接的群集节点的地址

#wsrep_node_name = "Node A"

此节点的名称。此名称可以在wsrep_sst_donor中用作首选供体。请注意，集群中的多个节点可以具相同的名称。

#wsrep_node_address = 192.168.0.1

以 ip address[:port] 格式指定节点的网络地址。

wsrep_node_incoming_address   = 10.0.0.1

这是节点用来侦听客户端连接的地址。如果未指定地址或将其设置为 AUTO （默认），则mysqld使用-bind-address或--wsrep-node-address，或尝试以相同顺序从可用网络接口列表中获取一个地址。

#wsrep_causal_reads = 0

设置为 1 时（默认为 0 ），则在整个集群中强制执行读取提交的特征。

如果主机比从机更快地应用事件，则两者可能会短暂地不同步。

在将此变量设置为 1 的情况下，从属将等待事件应用，然后再处理其他查询。

#wsrep_sst_method = mysqldump

用于进行状态快照传输（SST）的方法。可选方法有 rsync，mysqldump ， xtrabackup ， xtraback p-v2， mariabackup

#wsrep_sst_auth   = sst:sst

用于复制的用户名和密码。如果wsrep_sst_method设置为 rsync ，则不使用，而对于其他方法，它格式应为 <user>:<password>。当使用SHOW VARIABLES查询值时，内容在日志中被屏蔽。

#wsrep_sst_receive_address   = 192.168.0.1

这是集群中其他节点（供体）连接到的地址，用于发送状态转移更新。如果未指定地址或将其设置为UTO （默认），则mysqld使用--wsrep_node_address的值作为接收地址。但是，如果未设置--wsre_node_address，则它将使用--bind-address中的地址，或尝试以相同顺序从可用网络接口列表中获一个地址。

[mysql]
no_auto_rehash

#关闭自动补全 SQL 命令功能

max_allowed_packet = 16M

数据包或生成的/中间的字符串的最大大小（以字节为单位）。

数据包消息缓冲区使用net_buffer_length中的值进行初始化，但可以增长到max_allowed_packet个节。设置为最大BLOB的最大值（1024的倍数）。

如果更改此值，则也应该在客户端更改它。

prompt = '\u@\h [\d]> '

此为修改 mysql 提示符内容的变量。我们自定义提示符信息。通过配置可以显示登入的主机地址，陆用户名，当前时间，当前数据库等信息。

[mysqldump]

max_allowed_packet = 16M

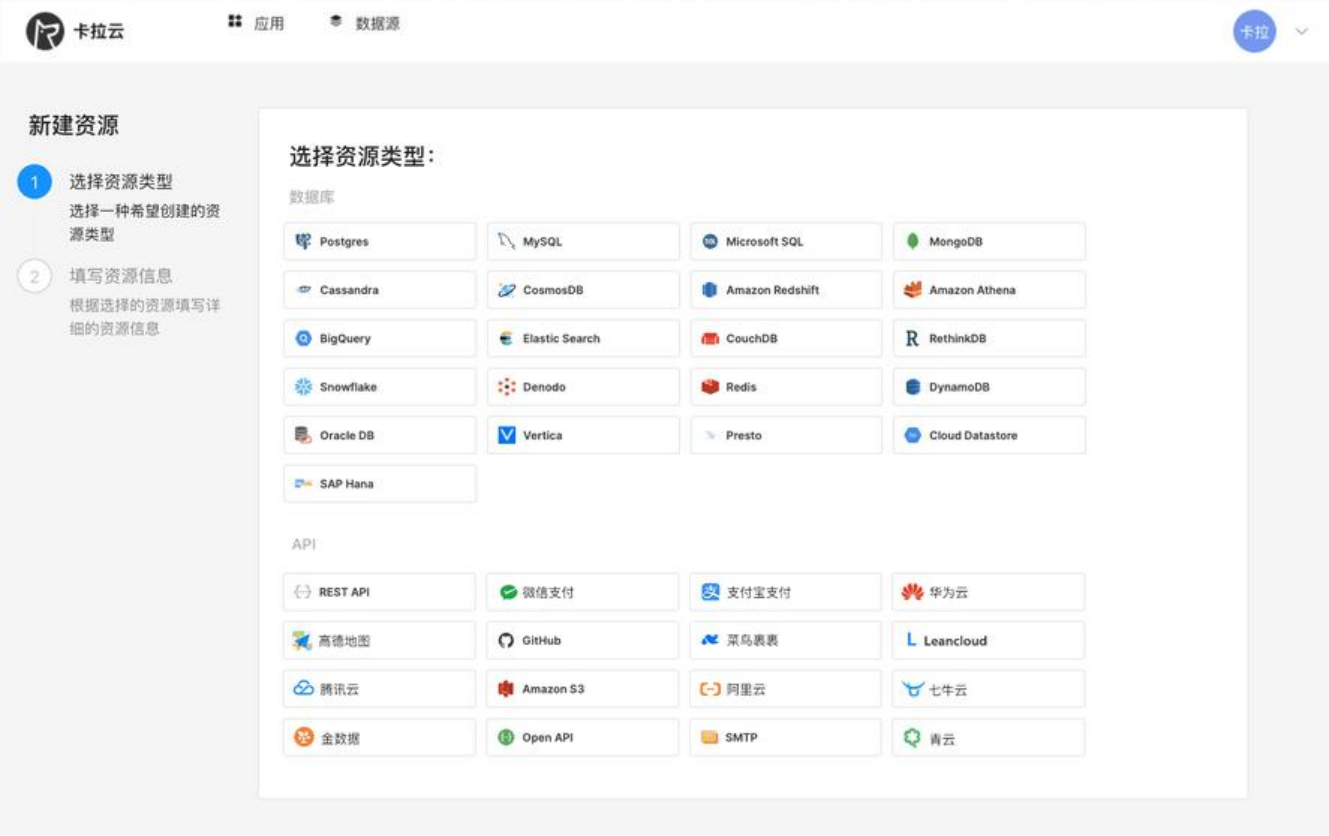
#限制接受的数据包大小，这里的值为 MySQL 服务器端和客户端在一次传送数据包的过程当中数据的大小

扩展阅读：《最好用的 10 款 MySQL GUI 管理工具横向测评》

卡拉云 - 新一代低代码开发工具

MySQL 配置文件对于优化数据库性能有着极大的意义，我们不仅要搞懂每一行代码的意义，更要结实际情况，在实践中边改边测，最终达到性能最大化的目标。

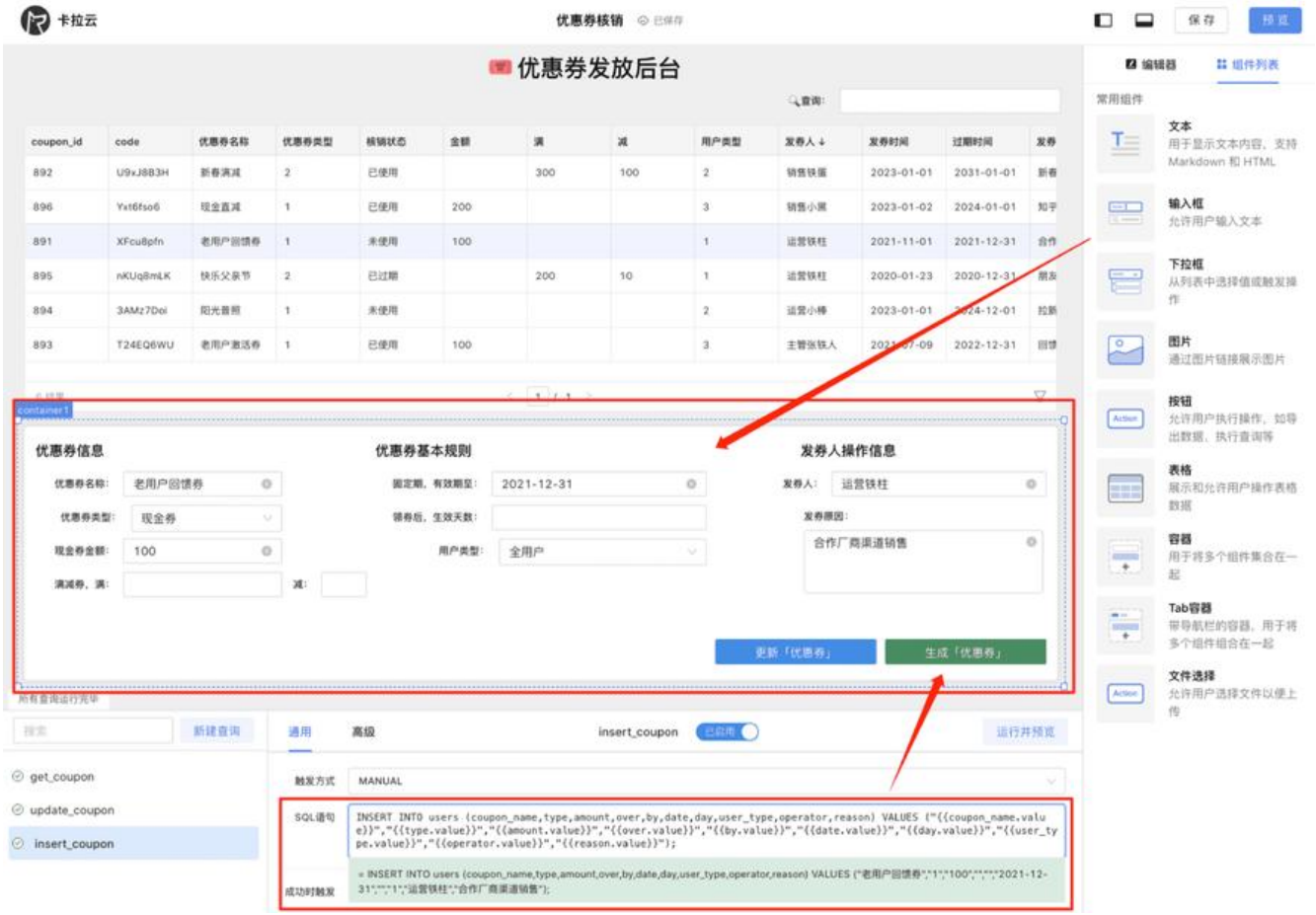
如果你想在你的 MySQL 数据库上构建应用工具，可以试试卡拉云，卡拉云是新一代低代码开发工具免安装部署，可一键接入包括 MySQL 在内的常见数据库及 API。不仅可以完成 Workbench 所有功，还可根据自己的工作流，定制开发。无需繁琐的前端开发，只需要简单拖拽，即可快速搭建企业内工具。**数月的开发工作量，使用卡拉云后可缩减至数天，。**



卡拉云可一键接入常见的数据库及 API

卡拉云可根据公司工作流需求，轻松搭建数据看板，并且可分享给组内的小伙伴共享数据

下图为使用卡拉云在 5 分钟内搭建的「优惠券发放核销」后台，仅需要简单拖拽即可快速生成前端组，只要会写 SQL，便可搭建一套趁手的数据库工具。**欢迎试用卡拉云。**



原文链接：MySQL 配置文件 my.cnf / my.ini 逐行详解

有关 MySQL 教程，可继续拓展学习：

- 如何远程连接 MySQL 数据库，阿里云腾讯云外网连接教程
- 如何在 MySQL / MariaDB 中导入导出数据，导入导出数据库文件、Excel、CSV
- 如何在两台服务器之间迁移 MySQL 数据库 阿里云腾讯云迁移案例
- MySQL 中如何实现 BLOB 数据类型的存取，BLOB 有哪些应用场景？
- MySQL 触发器的创建、使用、查看、删除教程及应用场景实战案例