# 编写一个简单的mysql存储引擎

## 目标

ha\_example是mysql提供的存储引擎实现例子，实现了最简单的存储引擎接口，通过改造这个例子，完成一个名为cedb的存储引擎。

c表示ctrip，e表示example

## ha\_example

我们先把mysql例子运行起来

### 编译

默认ha\_example是不进行编译的，需要通过指定编译选项-DWITH\_EXAMPLE\_STORAGE\_ENGINE=1，生成ha\_example.so的存储引擎插件，该插件位于mysql的plugin目录中

1、下载mysql源码

2、在源码目录创建编译链接目录

mkdir bld

3、生成编译文件

cd bld

cmake .. -DCMAKE\_BUILD\_TYPE=Debug -DCMAKE\_INSTALL\_PREFIX=/home/jiangyx/work/temp/mysql/runnable/ -DWITH\_EXAMPLE\_STORAGE\_ENGINE=1

4、编译

make

5、安装

make install

### 安装

install plugin example soname "ha\_example.so";

### 总结

至此，mysql自带的例子已经完成

## cedb

### 编译

在源码目录的storage目录下创建我们的引擎目录



修改CMakeList.txt

分别指明编译后的模块名和编译模块所需的源文件



说明

a）在MYSQL\_ADD\_PLUGIN中可指定插件是否默认加载。指定为MODULE\_ONLY，插件被编译成so模块不作为mysql的内嵌模块，启动时不加载；指定为MANDATORY，插件被编译为内嵌模块，启动时加载。

b）内嵌模块。在指定为MANDATORY时，mysql会在生成内嵌模块的源码中将该模块加入（编译时生成的源代码文件为sql\_builtin.cc，由cmake读取mysql的配置生成）；在指定为内嵌模块时，需要实现builtin\_xxx\_plugin函数，通过该函数mysqld加载内嵌的模块。

### 安装

install plugin cedb soname “ha\_cedb.so”;

### 功能设计

1、创建/删除表

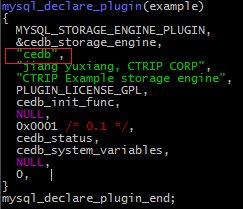
2、打开/关闭表

3、插入数据

4、定义变量/状态

### 功能实现

#### 定义插件



红框为插件名，在“install plugin xxx”语句中指定的插件名

cedb\_init\_func：用来初始化插件，插件运行期间只被调用一次。

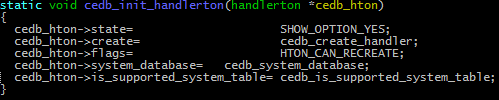
cedb\_system\_variables：插件的系统变量

cedb\_status：插件状态变量

#### 定义handlerton

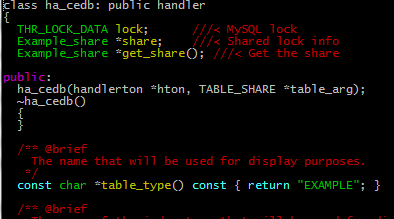
handlerton是handler singleton的缩写，mysql在初始化每个存储引擎需要时，将传入handlerton的对象，由存储引擎进行初始化，提供该存储引擎的元信息。

mysql通过handlerton对存储引擎进行管理，每个存储引擎在运行时只有一个handler

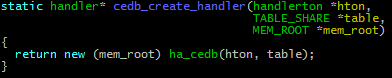


#### 表实例

mysql存储引擎通过handler进行表操作，定义存储引擎的handler

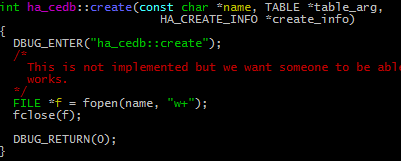


mysql通过hanlderton的create函数进行handler的创建，每个handler对应一个表

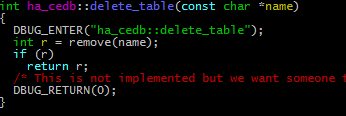


#### 创建/删除表

创建表



删除表



说明：

1、在创建/删除表时，name部分包含schema和table的名字，并以符号“/”分割，如语句

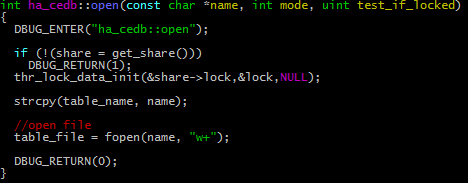
use test；

create table t\_cedb(f1 int) engine=cedb;

得到name为”test/t\_cedb”

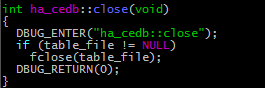
#### 打开/关闭表

打开表



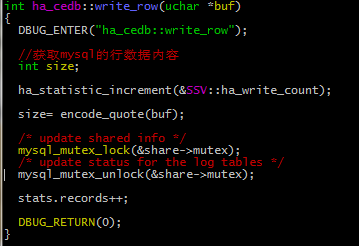
在做任何DML操作前，首先要进行open操作

关闭表

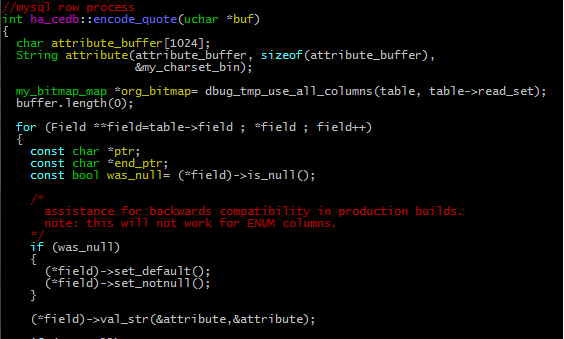


#### 插入

插入数据需要实现handler的write\_row接口



在encode\_quote中对传入的数据进行解析



#### 定义变量和状态

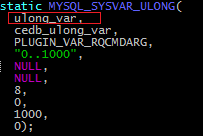
##### 变量

如下是定义cedb\_ulong\_var变量的代码，其他变量参考例子代码

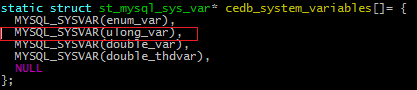
1、首先定义c中存储数据的变量



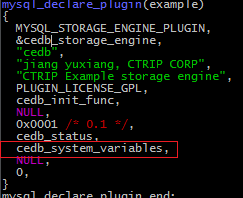
然后定义mysql的变量，红框部分为变量的名字（注2）



2、然后定义变量的数组，使用NULL作为数组的结尾。数组中红框的名字必须与步骤1中红框内定义的名字保持一致

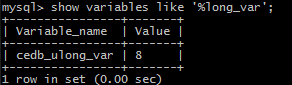


3、最后在定义存储引擎时，红框部分使用步骤2定义的变量数组，通过“show variables”语句可查看



4、查看变量情况

其中数值8为步骤1中定义变量时指定的默认值



注

1、其他类型变量参考代码

2、变量前会自动添加插件名字作为前缀，这个名字如步骤4中查看的

##### 状态

如下是定义cedb\_state状态的实现

1、定义变量

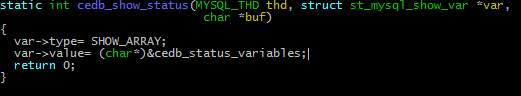
定义c中保存数据的变量



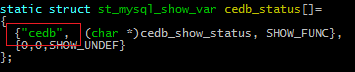
定义mysql的状态变量数组，红框内为



2、定义返回状态变量的函数，在show status时，最终将调用该函数，获得状态

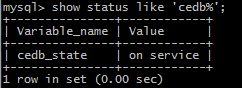


3、定义show status所需的数组



红框部分为status的前缀，和步骤1

4、查看状态信息



### 代码

在git仓库上<http://git.dev.sh.ctripcorp.com/ops-mysql/mysql-5-6-21-ctrip>，分支test\_storage\_engine是此文档的代码例子

## 参考

http://dev.mysql.com/doc/internals/en/custom-engine.html