Multicore Programming
Project 3
-Faster initialization of a database buffer pool-

Due Date : 11:59 pm November 29 (Sunday)

Specifications

- We use MySQL-5.6.24 with the InnoDB storage engine as a baseline.
- InnoDB buffer pool may consist of multiple buffer instances (chunks), each of which is contiguous memory.
- The goal of this project is to make the buffer pool initialization faster by using concurrent programming techniques.
  1. Create chunks based on available cores using:

```
#include <unistd.h>
sysconf(_SC_NPROCESSORS_ONLN);
```

  2. Create multiple buffer pool init threads to make each buffer chunk initialized in parallel.
  3. For each buffer frame, InnoDB creates two mutexes and one rw_lock, and register them to the global {mutex|rw_lock} lists. Inserting {mutex|rw_lock} objects to the list is currently protected by a global mutex which will incur heavy contention, if you run multiple threads, doing the same thing at the same time → **You are required to change this list operation lock-free or something similar and faster**.

Before you start the project …

- You have to set up a few things for this project.
- Install “cscope” using “apt-get install cscope”
- Add key mapping content in “cscope_vim.pdf” in “.vimrc” located in your home directory.
- Create “mkcscope.sh”, make it executable “chmod +x mkscope.sh”, and put it in your local “bin” directory, which is included in $PATH that needs to be updated “.bashrc” by adding “export PATH=$PATH:~/bin”

```
------- mkcscope.sh ---------
#!/bin/sh
rm -rf cscope.files
find . \( -name '*.ic' -name '*.hp' -name '*.c' -o -name '*.cpp' -o -name
'*.cc' -o -name '*.h' -o -name '*.s' -o -name '*.S' \) -print >
cscope.files
cscope -b -q -k
```

Before you start the project …

- Build cscope index against mysql-5.6.24 source.
- $bash>mkcscope.sh
- Then you can use cscope in vim.
- Build and install MySQL
  - Build using cmake:
    - cmake -DCMAKE_INSTALL_PREFIX=/usr/local/mysql-5.6.24 -DWITH_INNOBASE_STORAGE_ENGINE=1 -DDEFAULT_CHARSET=utf8 -DWITH_EXTRA_CHARSETS=all -DDEFAULT_COLLATION=utf8_general_ci -DMYSQL_TCP_PORT=3306 -DMYSQL_UNIX_ADDR=/var/run/mysqld/mysqld.socket .
  - Once you build, you need to bootstrap the first DB:
    - bash:/usr/local/mysql-5.6.24$ ./scripts/mysql_install_db

- The {files|functions} you may have to look and change are :
  - mysql-5.6.24/storage/innobase/srv/srv0start.{cc|h} → buf_pool_init()
  - mysql-source/mysql-5.6.24/storage/innobase/buf/buf0buf.{cc|h}
    - buf_pool_init_instance(), buf_chunk_init(), buf_block_init()
  - mysql-source/mysql-5.6.24/storage/innobase/sync/sync0sync.{cc|h}
    - mutex_create_func()
  - mysql-source/mysql-5.6.24/storage/innobase/sync/sync0rw.{cc|h}
    - rw_lock_create_func()

What to submit ?

- Submit a patch file and a nicely written document that explains your design to TA.
  - How to create a patch file : http://www.thegeekstuff.com/2014/12/patch-command-examples/
  - The patch file **must be** created at the mysql-5.6.24 root directory, not somewhere else.
- TA will measure the buffer pool initialization time and run “sysbench” to ensure the correctness of the changed database.

- Please post questions to the class board.