EXTRA CREDIT

OVERVIEW

We are offering an extra credit assignment for those students that are interested in benchmarking a DBMS. This assignment is **optional**.

The CMU Database Group has created an open-source database benchmarking framework called OLTP-Bench. It is designed to be a single framework that supports multiple benchmarks on multiple DBMSs. The framework is written in Java. All of the benchmarks use SQL with JDBC to load databases and execute queries.

Your assignment is to deploy a DBMS on Amazon EC2 and get one benchmark to work on it. You will need to configure and tune the system to the best of your abilities to maximize its performance. This means that you should not just run the system with the default configuration.

You are allowed to work in groups of three on a particular DBMS. You can work together to setup and deloy the system. Each team member is responsible for one benchmark.

Release Date: Sep 01, 2017
Due Date: Dec 14, 2017 @ 11:59pm

INFORMATION

Although there are 15 different benchmarks available in the framework, your team should focus on the following ones:

1. TPC-C (OLTP)
2. Wikipedia (OLTP)
3. TPC-H (OLAP)

We are choosing these benchmarks so that we can compare results between groups. If you are unable to get your selected benchmark running due to some unfixable reason (e.g., unsuppported SQL features), please contact Andy.

All of the benchmarks in the OLTP-Bench framework are able to generate their own databases **except** for TPC-H. You will need to use the dbgen tool to load the database.

Important: It is likely that some systems will not be able to support all of the features in TPC-H. If you are unable to support at least 15 out of 22 TPC-H queries, please contact Andy and you will be assigned an alternative benchmark.

FRAMEWORK CONFIGURATION

In order for OLTP-Bench to use a new DBMS, you have to add it to DatabaseType enum.

You have the define the following settings for each new entry:

- `includeColNames` : Whether the benchmark loader should include the column names in the SQL statement that inserts tuples.

Most systems will likely just need to use `escapeNames=true` and `includeColNames=false` . Contact Andy if you are stuck on this.

SQL DIALECTS

Each benchmark implementation includes a `dialects` directory that contains XML files that allow you to override its transactions' default SQL statements. All of the benchmark are designed run on MySQL by default. Each transaction contains pre-defined `SQLStmt` objects. The name of the `SQLStmt` variable is the identifable handle that you can override in the dialects file.

For example, the TPC-C Delivery transaction contains the delivGetOrderIdSQL query. The oracle-dialects.xml file contains a version of the same query that is designed to work on Oracle. The framework will automatically substitute the default SQL statement with the DBMS-specific one at runtime.

BENCHMARK SETTINGS

You should use the following settings in the config file for each benchmark. Do not change the `weights` settings.

TPC-C

Sample Config File: config/sample_tpcc_config.xml

`isolation` : TRANSACTION_READ_COMMITTED
`terminals` : 64
`scalefactor` : 16
`rate` : unlimited
`time` : 300

WIKIPEDIA

Sample Config File: config/sample_wikipedia_config.xml

`isolation` : TRANSACTION_READ_COMMITTED
`terminals` : 64
`scalefactor` : 100
`rate` : unlimited
`time` : 300

TPC-H

Sample Config File: config/sample_tpch_config.xml

`isolation` : TRANSACTION_READ_COMMITTED
`terminals` : 1
`scalefactor` : 20
`rate` : unlimited
`time` : Do not set!

INSTRUCTIONS

1. Download the extra credit submission file. You will need to document all of the steps that you took to get the benchmark running on your DBMS in this file and submit it at the end to get credit.

~~has to approve your DBMS first before you are allowed to proceed.~~

3. Setup OLTP-Bench and your target DBMS on local development machine. This will allow you work on getting the OLTP-Bench framework to run your selected benchmark without costing you AWS credits.
4. Deploy your DBMS on EC2. You can use a cheaper instance type (e.g., `t2.2xlarge`) to get started with the EC2 environment. You will need to document all of the steps that you did to install and configure the DBMS to get it to work. You will then deploy OLTP-Bench on another instance:
   1. If you are executing an OLTP benchmark, then you will want to use an instance type with more compute cores (`c4.4xlarge`).
   2. If you are executing an OLAP benchmark, then you can use a less powerful instance type (`t2.2xlarge`).

   For the final benchmarking, you must switch your DBMs instance type to use the `c4.4xlarge` instance type.
5. Run each benchmark three times using the following command:

```
$ ant build execute -Dbenchmark=$(BENCHMARK) \
        -Dconfig=config/$(CONFIG).xml \
        -Dcreate=false \
        -Dload=false \
        -Dexecute=false \
        -Dextra="-s 10"
```

   Note that the "-s 10" command option tells the framework to summarize the performance of the DBMS every 10 seconds.
   The framework will print the DBMS's throughput to the terminal. It should look something like this:

```
Rate limited reqs/s: Results(nanoSeconds=10099645360, measuredRequests=101) = 10.00035 request
```

   The **requests/sec** is the number that you want to report on the spreadsheet. Each benchmark trial will also generate two files:
   1. `results/oltpbench.$(NUM).csv` (Raw Execution Stats)
   2. `results/oltpbench.$(NUM).res` (Aggregated Throughput)

   Save these files and include them in your final submission.

HINTS

You can reach out to the DBMS vendors to ask for help in setting up your DBMS. Look for tuning guides.

There are some rare cases where the OLTP-Bench framework gets stuck at the end of the benchmark when it is trying to collect the results. You can just kill the benchmark (`CTRL+C`). If this happens repeatedly for you, please email Andy.

If your system supports distributed deployments, you only need to benchmark it with a single node configuration.

LATE POLICY

No late submissions are allowed.

SUBMISSION

Your team should submit the following items:

1. Update the class result spreadsheet with the results (see Canvas announcement for link).

3. Send Andy an email with a tarball that contains the following files from your experiments:
   - Submission File
   - TPC-C Results:
     - `tpcc/oltpbench.$(TRIAL1).csv`
     - `tpcc/oltpbench.$(TRIAL1).res`
     - `tpcc/oltpbench.$(TRIAL2).csv`
     - `tpcc/oltpbench.$(TRIAL2).res`
     - `tpcc/oltpbench.$(TRIAL3).csv`
     - `tpcc/oltpbench.$(TRIAL3).res`
   - Wikipedia Results:
     - `wikipedia/oltpbench.$(TRIAL1).csv`
     - `wikipedia/oltpbench.$(TRIAL1).res`
     - `wikipedia/oltpbench.$(TRIAL2).csv`
     - `wikipedia/oltpbench.$(TRIAL2).res`
     - `wikipedia/oltpbench.$(TRIAL3).csv`
     - `wikipedia/oltpbench.$(TRIAL3).res`
   - TPC-H Results:
     - `tpch/oltpbench.$(TRIAL1).csv`
     - `tpch/oltpbench.$(TRIAL1).res`
     - `tpch/oltpbench.$(TRIAL2).csv`
     - `tpch/oltpbench.$(TRIAL2).res`
     - `tpch/oltpbench.$(TRIAL3).csv`
     - `tpch/oltpbench.$(TRIAL3).res`

   You can use the following command to create your tarball:

```
$ tar zfvc extracredit-$(SYSTEM).tar.gz extracredit-submission.txt tpcc/* wikipedia/* tpch/*
```

Last Updated: Dec 10, 2017