# Apache Beam: TPC-DS Benchmark on Beam SQL




**Name:** Kai Jiang
**Email:** jiangkai@gmail.com
**Github:** @vectorijk
**Prospective Mentor:** Kenneth Knowles
**Time Zone:** UTC -8:00


## Motivation

Beam has a number of classic streaming SQL benchmarks known as "Nexmark" coded up in both raw Java and also Beam SQL.

Currently, expanding functionality has been the focus of Beam SQL so there is little known about performance - we know only that it is a pretty straightforward mapping from SQL to Beam that should work OK a lot of the time. It would be interesting to see where the bottlenecks are when these SQL benchmarks are translated via Beam SQL into a Beam pipeline and then again translated to the native capabilities of e.g. Spark and Flink.

## About me

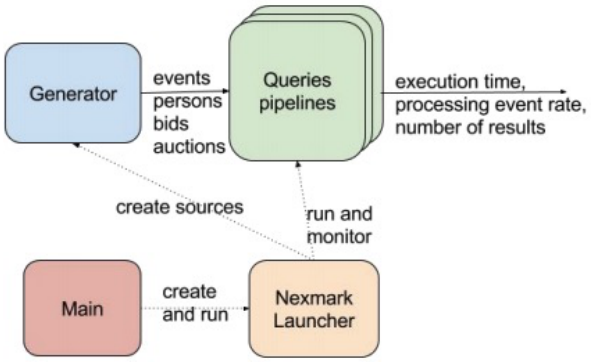
I am Kai Jiang, a master student in Computer Science from Portland State University, interested in database and distributed system. I have been working on Apache Beam for a few months. I

have contributed a number of pull requests related to SQL and Nexmark to Apache Beam project, which made me familiar with the Beam codebase and workflow.

Here are PRs I opened:
***Merged***
[BEAM-3476] [SQL] covariance aggregation functions #4466
[BEAM-2779] PipelineOptionsFactory should prevent non PipelineOptions interfaces from being constructed #3949
[BEAM-2612] Variance builtin aggregation functions #3568
[BEAM-560] Use ThreadLocal to cache Marshaller/Unmarshaller #1795
[BEAM-708] Using AutoValue in BoundedReadFromUnboundedSource #1794
[BEAM-1248] Combine with side inputs API should match ParDo #1755
[BEAM-2852] Nexmark Kafka source sink #3937

## Description of work
JIRA ticket: BEAM-3783

The below diagram shows architecture of existing Nexmark benchmark for streaming SQL.




Main parts:
1. Generate TPC-DS data either via nexmark generate or kafka streaming data connector.
2. Integrate TPC-DS query into Queries pipelines.
   SQL instead of raw java
   Data model
3. Run queries against different runner and scale out
4. Find the benchmark

Extract the generator

## Timeline

This timeline is tentative, given the fast-paced development in Beam.

Pre-GSoC Period and Community Bonding Period

- Set up an initial meeting with mentor and talk about more details
- Open tickets on JIRA
- Work on as many issues as possible to increase familiarity with the and the code or tickets on JIRA related to this GSoC project.

Week 1, 2, 3, 4

Run TPC-DS queries in Beam via the SQL query server
Prepare (Generate) TPC-DS dataset
https://github.com/gregrahn/tpcds-kit The purpose of this tool is used for pre-built TPC-DS dataset
https://github.com/databricks/spark-sql-perf

Benchmark on all runners (Spark / Flink cluster, etc)
Run with all 100 queries for generating performance meterics.
Batch job first, steam sql with fixed window

Week 13

Writing up blog and conclusion for the process of GSoC

- Wrap up the project

## Notes

1. I have no internship and school class during summer and GSoC project will be my major priority during this 12 week period.
2. I haven't submitted any other proposals to GSoC. This is my only consideration.
3. After Google Summer of Code, I would be very willing to keep contributing to Apache Beam

### References

[1] TPC-H for Hive by Yuntao Jia (https://issues.apache.org/jira/browse/hive-600)
[2] Running TPC-H on Apache Hive https://github.com/rxin/TPC-H-Hive
[3] TPC-DS on Spark: TPC-DS business case with Spark SQL
https://github.com/databricks/spark-sql-perf