HAL
open science

# SQL for GPU Data Frames in RAPIDS Accelerating end-to-end data science workflows using GPUs

Alexander Ocsa

## To cite this version:

Alexander Ocsa. SQL for GPU Data Frames in RAPIDS Accelerating end-to-end data science workflows using GPUs. LatinX in AI Research at ICML 2019, Jun 2019, Long Beach, United States. 2019. hal-02264776

**HAL Id: hal-02264776**

**https://hal.science/hal-02264776v1**

Submitted on 7 Aug 2019

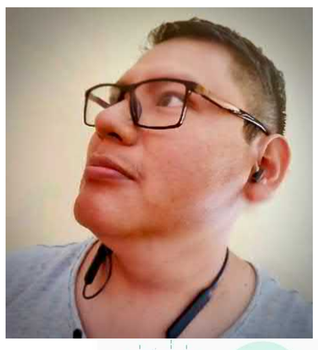
**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Public Domain

# SQL for GPU Data Frames in RAPIDS

## Accelerating end-to-end data science workflows using GPUs




**Alexander Ocsa[1*], *BlazingDB member and CUDF contributor**

[1]BlazingDB Inc, US;

Correspondence to: alexander@blazingdb.com github.com/aocsa — @aocsa



In this work, we present BlazingSQL [2] a SQL engine build on RAPIDS open-source software, which allows us to query enterprise data lakes lightning fast with full interoperability with the RAPIDs stack. BlazingSQL makes it simple for data scientists to SQL query raw files directly into GPU memory. RAPIDS can then take these results to continue machine learning, deep learning, and visualization workloads. We present two demo workflows using BlazingSQL and RAPIDS. Moreover, our solution presents an average from 20-100x faster than an identical query on Spark Cluster at price parity. This significant gain in speed allows us to evaluate the solution on a large, realistic, and challenging set of database use cases.



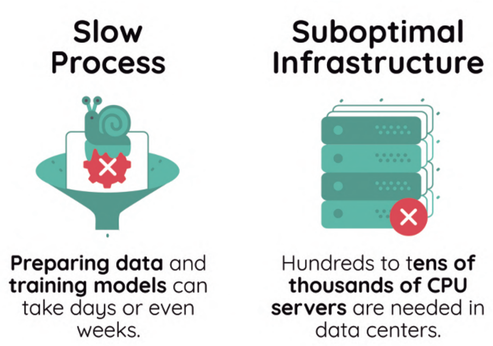
***Keywords:*** *BlazingSQL, GPU, SQL, RAPIDS, cuDF, cuML, Data Frame, Apache Arrow, Big Data, Machine Learning.*


## Context

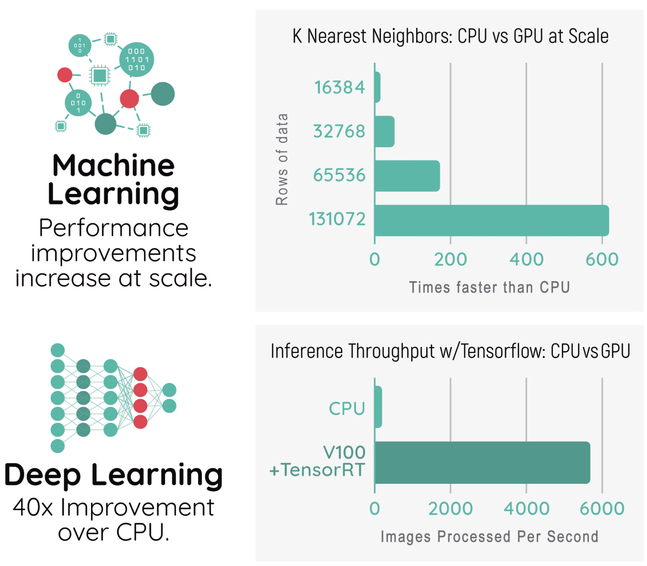
The increasing availability of data has created a necessity to develop better techniques and methods in order to discover knowledge from massive volumes of complex data. For these challenges, CPU impose limits on performance to deliver these kind of solutions. Resorting to GPU programming is one approach to overcome these performance limitations.

### GPUs in Machine Learning

CPUs can no longer handle the growing data. AI/ML is unable to keep up with the growth of data being processed [3].




GPUs are well known for accelerating the training. GPUs are able to scale to the new data demands. The bigger the dataset is, the higher the training performance difference is between CPU and GPU [4].



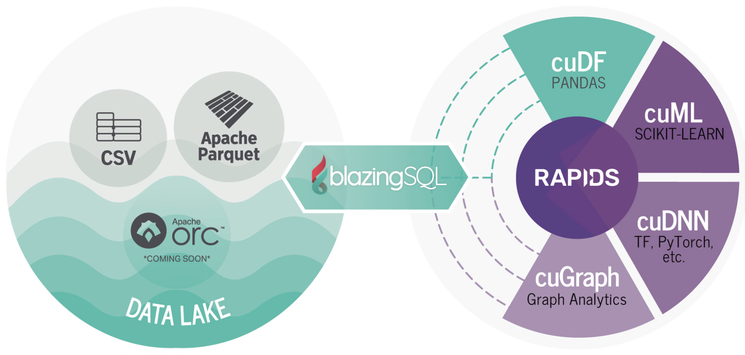

However data preparation still happens on CPUs, and can't keep up with GPU accelererated machine learning.

## RAPIDS

RAPIDS [5] is an end-to-end analytics solution on GPUs. More extensively, RAPIDS is a set of open source libraries for GPU accelerating data preparation and machine learning built by multiple contributors like NVIDIA, Anaconda, BlazingDB, etc. It covers all the steps of the most common data science pipelines. It is composed of cuDF for data preparation, cuML for machine learning, and cuGRAPH for graph analytics all under the standard specification of Apache Arrow [1] in GPU memory.

## BlazingSQL and RAPIDS Ecosystem

RAPIDS [5] allows data scientists to accelerate end-to-end data analytics solution on GPUs. Part fundamental of RAPIDS is the GPU DataFrame (GDF) which has the goal to support interoperability between GPU applications and define a common GPU in-memory data layer. In this context, CUDA DataFrame (cuDF) from RAPIDS covers the GPU Data Processing for GDFs (formed by GPU compute kernels and a pandas-like API) [6].

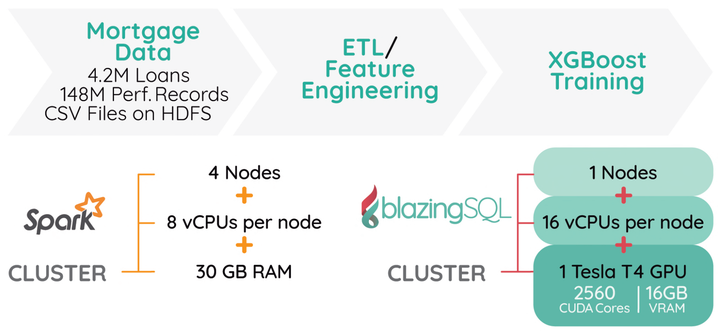



BlazingSQL [2] provides a simple SQL interface to ETL massive datasets into GPU memory for AI and Deep Learning workloads. Furthermore, BlazingSQL can directly query files, such as CSV and Apache Parquet, on data lakes, like HDFS and AWS S3, all these processes directly into GPU memory

## End-to-end workflows

### Mortgage: Load Risk end-to-end processing

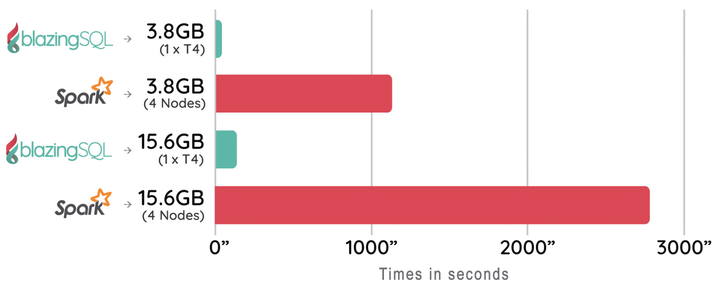
Train a model to assess risk of new mortgage loans based on Fannie Mae loan performance data.




BlazingSQL + XGBoost Loan Risk Demo

The end to end analytics workload:

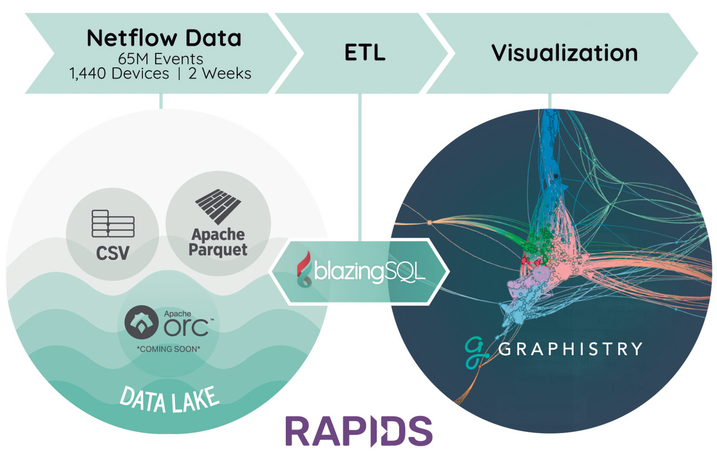
- Data Lake → ETL/Feature Engineering → XGBoost Training
- We built two price equivalent clusters on GCP, one for Apache Spark and another for BlazingSQL
- BlazingSQL ran the ETL phase of this workload **20x** faster than Apache Spark




RAPIDS + BlazingSQL outperforms traditional CPU pipelines

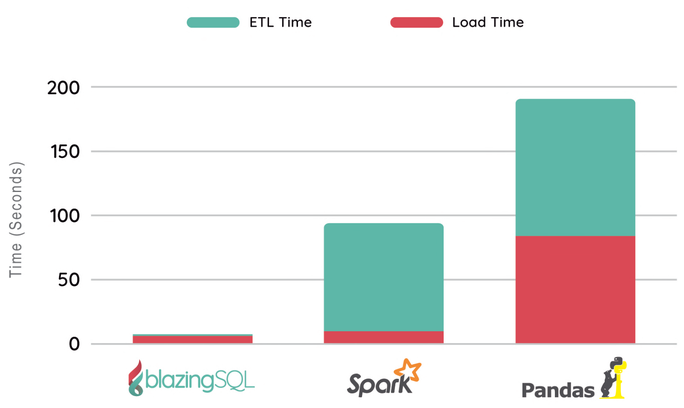
### Netflow Analysis: ETL + Visualization

BlazingSQL, the GPU SQL engine built on RAPIDS, worked with our partners at Graphistry to show how you can analyze log data over **100x** faster than using Apache Spark at price parity.




Visually analyze the VAST netflow data set inside Graphistry in order to quickly detect anomalous events

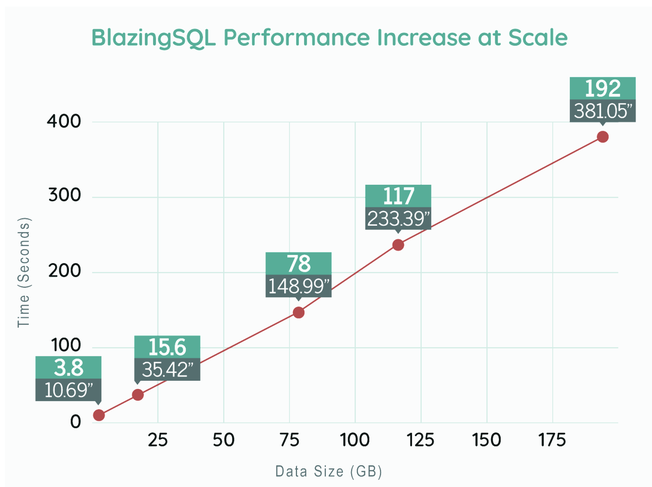
We took 65M rows of netflow data in Apache Parquet, and in less than a second our query built a table of nodes and edges to render a visual graph.




Netflow Demo Timings (Load & ETL)

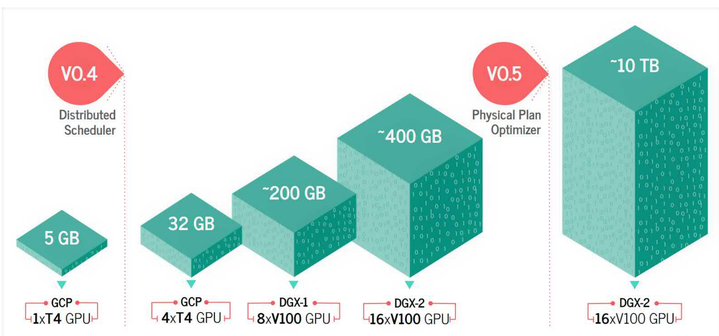
## Scalability

We ran an end to end analytics workload using the same configuration of mortgage workflow at different data scale.




These results were achieved on a single DGX-1 with 4 V100 GPUs:

## Whats Next




Whats Next

- Full distributed version (V0.4) in progress. We should have something for you (including a demo) very soon. We will also update our benchmarks at this point.
- Data Skipping (V0.5). Time to start optimizing the engine to bring massive performance benefits.

## Acknowledge

## References


[1] Apache. A cross-language development platform for in-memory data. https://arrow.apache.org, February 2016.

[2] Rodrigo Aramburu. Announcing BlazingSQL, A GPU SQL Engine for RAPIDS Open-Source Software from NVIDIA. https://blazingdb.com, October 2018.

[3] William Malpica, Rodrigo Aramburu, and Felipe Aramburu. BlazingSQL on RAPIDS: SQL for Apache Arrow in GPU Memory. Connect Data Lakes to RAPIDS. https://goo.gl/GCh3MU, March 2019.

[4] NVIDIA. TensorRT 3: Faster TensorFlow Inference and Volta Support. https://devblogs.nvidia.com/, December 2017.

[5] Nvidia, Anaconda, and BlazingDB. RAPIDS, Open GPU Data Science. https://rapids.ai/, October 2018.

[6] Nvidia, BlazingDB, Anaconda, and Apache Arrow. cuDF - GPU DataFrame Library. https://github.com/rapidsai/cudf, October 2018.