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 of 20× 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: GPU, SQL, RAPIDSDM, cuDF, cuML, Data Frame, Arrow, Big Data, Data Lakes, Machine Learning.

Context
The increasing availability of data has created a necessity to develop better techniques and methods to handle these massive volumes of complex data. For these challenges, CPU impose limits on performance to deliver these kind of solutions. Re-sorting to GPU programming is one approach to overcome these performance limitations.

GPUs in Machine Learning
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 [3].

However data preparation still happens on CPUs, and can’t keep up with the growth of data being processed [1].

1 RAPIDS AI
RAPIDS AI [4] 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 data preparation, cuML for machine learning, and cu-GRAPH for graph analytics, all under the standard specification of Apache Arrow [1] in GPU memory.

BlazingSQL and RAPIDS AI Ecosystem
RAPIDS [4] 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 drive 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) [5].

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.

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 XGBoost phase of this workload 20× faster than Apache Spark

Netflow Analysis: ETL + Visualization
BlazingSQL, the GPU SQL engine built on RAPIDS AI, worked with our partners at Graphistry to show how you can analyze log data over 100× faster than using Apache Spark at price parity.

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

What's Next
• Better string support (V0.3). Look for a release and blog early next week.
• Full distributed version (V0.4) in progress. We should have something for you (including a demo) very soon. We will also share our benchmarks at this point.
• Data Skipping (V0.5). Time to start optimizing the engine to bring massive performance benefits.

Acknowledgments

References