Announcements

Please sign up for the presentation slots following the email
Discussion Highlights

How far away is Snowflake from the “optimal design”?

- Auto-scaling
- Better optimized storage layer (like Aurora)
- Security and reliability
- Code compilation
- Caching can be improved (e.g., workload specific)
- Data sharing across virtual warehouses
- Opportunities to extend into providing HTAP solutions
- Cloud service layer might be a bottleneck

Combine data warehousing and OLTP in cloud?

- Master and slave nodes within a VW to support writes as well
- Build snapshot isolation into storage (concurrency control)
- Transaction log -> (intermedia storage) -> S3 -> data warehouse every Y hours
- VW per transaction?
Starling: A Scalable Query Engine on Cloud Function Services

Matthew Perron
MIT CSAIL
mperron@csail.mit.edu

David DeWitt
MIT CSAIL
david.dewitt@outlook.com

Raul Castro Fernandez
MIT CSAIL
raulcf@csail.mit.edu

Samuel Madden
MIT CSAIL
madden@csail.mit.edu

ABSTRACT
Much like on-premises systems, the natural choice for running database analytics workloads in the cloud is to provision a cluster of nodes to run a database instance. However, analytics workloads are often bursty or low volume, leaving clusters idle much of the time, meaning customers pay for

SIGMOD 2020

that allow compute nodes to be added or removed dynamically, this scaling often takes minutes, making it impractical on a per query basis. Further, many cloud database systems require data to be explicitly loaded into proprietary formats. For workloads that touch data a limited number of times, such as one-off queries or ETL queries, loading data results
What is Serverless Computing?

Serverless computing is a cloud computing execution model in which the cloud provider runs the server, and dynamically manages the allocation of machine resources. Pricing is based on the actual amount of resources consumed by an application, rather than on pre-purchased units of capacity.

According to a Berkeley TechReport [1], the core of serverless today is:

Serverless computing = FaaS + BaaS

Function-as-a-Service  Backend-as-a-Service

Function-as-a-Service

FaaS offerings

• **AWS Lambda**
• Google Cloud Functions
• Microsoft Azure Functions
• IBM/Apache's OpenWhisk (open source)
• Oracle Cloud Fn (open source)
AWS Lambda

Features

• Function starts execution (within a container) within sub-second
• Charged at 100ms granularity that the container runs
• Can run thousands/millions of small invocations in parallel

Limitations

• Limited runtime: 15 min
• Limited resources: 1 core, 3 GB main memory
• No direct communication between functions
Opinion from a CIDR’19 Paper [2]

- Cloud storage is 1—2 orders of magnitude slower than SSD
- No inter-function communication
- Paper gave suggestions for future work

However in our final example, Serverless SQLite, we identify a use case that maps so poorly to FaaS that we conclude that databases and other state-heavy applications will remain as BaaS”

Database: FaaS or BaaS?

**FaaS: Today’s paper**

**BaaS: Athena, Snowflake, Aurora, etc.**
## Cloud Analytics Databases

<table>
<thead>
<tr>
<th>System</th>
<th>Does not require loading</th>
<th>Pay by query</th>
<th>Tunable performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Athena</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>Snowflake</td>
<td>✗</td>
<td>✓*</td>
<td>✓</td>
</tr>
<tr>
<td>Presto</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Amazon Redshift</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Redshift Spectrum</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Google BigQuery</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>Azure SQL DW</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td><strong>Starling</strong></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 1: Comparison of cloud analytics databases
Starling Architecture

Coordinator
• Query compilation
• Initiate workers

Workers
• Query execution

Storage
• Input data
• Communication
Example Query Execution (TPC-H Q12)

### Query 12

```
SELECT l_shipmode, 
     CASE 
         WHEN o_orderpriority = '1-URGENT' 
             THEN 1 
         WHEN o_orderpriority = '2-HIGH' 
             THEN 0 
         ELSE 0 
     END AS high_line_count, 
     CASE 
         WHEN o_orderpriority <> '1-URGENT' 
             AND o_orderpriority <> '2-HIGH' 
             THEN 1 
         ELSE 0 
     END AS low_line_count 
FROM orders, lineitem 
WHERE 
    o_orderkey = l_orderkey 
    AND l_shipmode in ('MAIL', 'SHIP') 
    AND l_commitdate < l_receiptdate 
    AND l_shipdate < l_commitdate 
    AND l_receiptdate >= date '1994-01-01' 
    AND l_receiptdate < date '1994-01-01' + interval '1' year 
GROUP BY l_shipmode 
ORDER BY l_shipmode;
```
Optimizations

Parallel reads

![Bar chart showing effective throughput (MBps) vs. number of parallel reads (1 to 64)]
Optimizations

Parallel reads

Read straggler mitigation (RSM)
  • If a read request times out, send duplicate request
Optimizations

Parallel reads

Read straggler mitigation (RSM)

Write straggler mitigation (WSM)

• If a write request times out, send duplicate request
• Single Timer: allow only single time out
Optimizations

Parallel reads
Read straggler mitigation (RSM)
Write straggler mitigation (WSM)
Doublewrite
  • Producer writes two copies of an object; consumer reads the one ready first
Optimizations

Parallel reads
Read straggler mitigation (RSM)
Write straggler mitigation (WSM)
Doublewrite
Pipelining
  • Start the following stage before the previous stage finishes
Optimizations

Parallel reads
Read straggler mitigation (RSM)
Write straggler mitigation (WSM)
Doublewrite
Pipelining
Combining to reduce cost of shuffle
Evaluation

Starling can be faster than other S3-based cloud data warehouses
Starling can be cheaper than other cloud data warehouses
Easy to tune performance by changing the number of tasks
Starling vs. Snowflake

Control layer vs. Coordinator

Compute layer vs. Workers

Storage layer
Future of Serverless Computing

Opinion from Berkeley Report [1]

• Challenges: Abstraction, System, Networking, Security, Architecture
• Predictions: new BaaS, heterogeneous hardware, easy to program securely, cheaper, DB in BaaS, serverless replacing serverful

Opinion from a CIDR’19 Paper [2]

• Fluid Code and Data Placement
• Heterogeneous Hardware Support
• Long-Running, Addressable Virtual Agents
• Disorderly programming
• Flexible Programming, Common IR
• Service-level objectives & guarantees
• Security concerns


Replace S3 with other storage system?
What about sorting?
Is doublewrite an optimization?
Poor tail latency a common problem in a distributed system?
OLTP on serverless?
Lambda + Starling vs. Hadoop?
Starling bank based on Starling?
Starling relying on AWS specifics (e.g., S3, pricing model, etc.)
Cloud fosters the growth of small-scale data analytic needs?
Indexing?
Starling and Snowflake represent the FaaS and BaaS approaches of implementing a database, respectively. What are the relative advantages and disadvantages of both approaches?

What ideas can a BaaS implementation like Snowflake borrow from FaaS?

How can OLTP benefit from serverless computing? Are there major limiting factors in today’s cloud?