WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

CS 839: Topics in Database Management Systems

Lecture 14: Serverless-3

Xiangyao Yu

10/23/2023

Group Discussion from Lecture #13

In Lecture 6, we discussed functions that could be pushed to a "middle layer" between compute and storage. Can the middle layer be built using lambda functions? What are the advantages and disadvantages of using lambda functions to build the layer?

- Examples of middle-layer functions: Filtering and aggregation, intermediate results store, log replay, Transactions, data shuffle, access control, caching, data format translation, real-time analytics, data virtualization, ETL functions

Can a transactional database benefit from lambda functions? If so, how?

Group Discussion from Lecture #13

What lambda function can do:

- Data format translation
- Filtering and aggregation
- Data virtualization
- Log replay
- Real-time analytics
- ETL?
- Compression, encryption?
- Data shuffle?

Things it cannot do:

- Caching
- Transactions (coordination)

Group Discussion from Lecture #13

Pros of using lambda for middle layer

- Elastic, easily scalable
- Stateless
- No payment for idle resources
- High aggregate throughput
- No need to provision infrastructure

Cons:

- Lambda functions are more expensive than VMs
- Limited compute power
- Execution time limit
- Network overhead
- Startup overhead

Group Discussion from Lecture #13

Can a transactional database benefit from lambda functions?

- Consistency is a challenge
- Workload patterns that can benefit from lambda functions: Bursty workloads, infrequent jobs, high-variance workloads
- With a reliable lock service, can leverage easy scalability of cloud functions
- Use MVCC + OCC to minimize coordination overhead with the storage
- Data backup for SQLite
- Implement stateless modules, e.g., proxies and coordinator in FoundationDB

Discussion Questions

What are the key limitations of cloud functions today? What would it take, in your opinion, to resolve each of these limitations? (e.g., better industry implementations or fundamental research breakthroughs?)

Ideally, a cloud database should be truly serverless. Existing systems fall short realizing this goal. What do you think are the primary obstacles that must be overcome to realize this vision?