Bagger: Massive Log Management on PostgreSQL

Chris Travers and Felix Wischke

February 2, 2020

About Adjust

Adjust is an mobile advertisement attribution company, focusing on transparency and fairness. Our attribution service is the basis of our analytic offerings. (Offer to give a 30 sec intro to what we do.)

We are also a big user of PostgreSQL, mostly on analytic workloads.

About The Authors

Chris Travers (Speaking) serves as Head of Database at Adjust. He has been a software developer and PostgreSQL user for over twenty years.

Felix Wischke is a platform engineer and one of the maintainers of Bagger. He is an expert on Linux, databases, and system programming.

Section 1

Why Bagger?

Our Needs

- Debug log for operations and support
- Reconcile data customers ask about
- Locate bad data sent to us
- Generalized application debug log
- Short-term storage

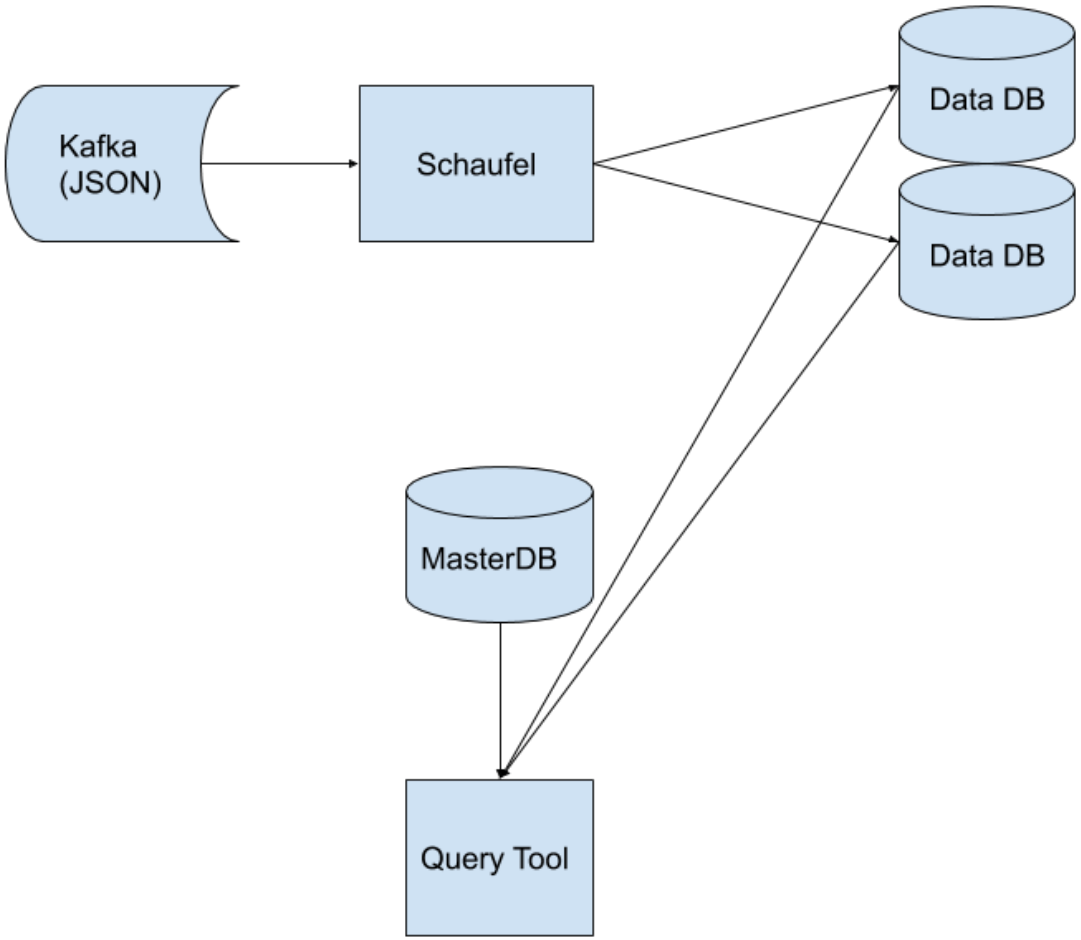
Previous System

- Elastic Search Cluster
- Over 1PB data
- Hard to manage
- Easy to crash

Section 2

Bagger's Structure

Overall Architecture



Master Databases Role

- Tracks data nodes
- Tracks usable partitions on each node
- Used to build queries by the query tool
- Used to configure Schaufel

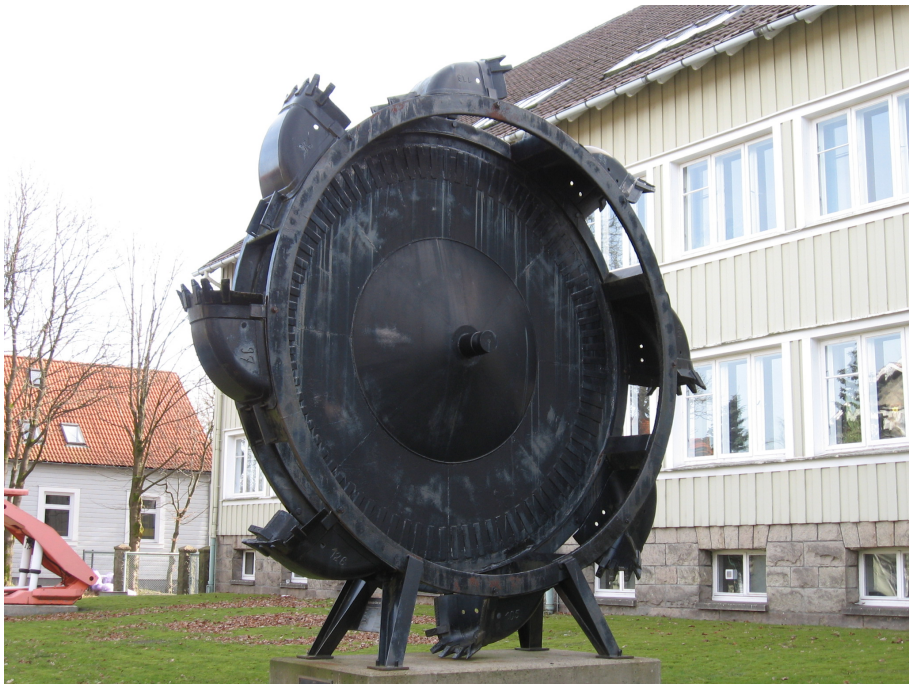
Structure of Master Database

- Backends (id, host, port, state, prefix/namespace)
- Partitions (id, name, master_id, slave_id, stats)
- Generations (id, map)

Structure of Data Nodes

- Each data node has 8-10 PG instances, each with own disks
- Using ZFS for data storage in order to optimize insert throughput
- Each schema contains a table with a single "data" JSONB column
- Each PG instance has two data schemas, a primary and a mirror
- Schaufel writes two instances, one primary and one mirror

Section 3



Schaufel

Image credit user NetNet on Wikipedia

Overview and Purpose

- High performance, parallel "shovel" for data
- Intended as a way to move data in/out of Kafka and other queues.
- Basically a giant swiss-army knife for data transfer
- `https://www.github.com/adjust/schaufel`

Basic Capabilities

- Parallel, multi-threaded
- Supports reading from Kafka, Redis queues, files
- Supports writing to Kafka, Redis queues, and PostgreSQL
- But schema, table, column names are hardcoded for Pg

Current Development

- Support configurable database schemas by "unwrapping" JSON
- Want other features? Get involved!
- `https://www.github.com/adjust/schaufel`

Section 4

PostgreSQL Issues

Partitioning/Inheritance Limits

- Currently have around 200k tables per data instance
- Initially were children of two parent tables
- Querying parent table took 15 minutes to **plan**
- Solution: Precalculate child tables and query them.

Indexing Limits

- Don't want to index whole jsonb for space reasons
- Want to index more than 32 sub-fields
- Have to patch PostgreSQL to use (change MAX_INDEX_KEYS)
- This has to be fixed before all of our tooling can be fully released.

COW File System Caveats

- Fragmentation from autovacuum
- Turn off autovacuum globally but what about catalogs
- Excels in linear writes but most reads become random

Section 5

Towards a Full Release

Of Server Orchestration Tools

Production Prototype Problem

- Requires patched PostgreSQL
- Not super-easy to run yet
- Not very efficient in space yet
- Autovacuum turned off globally instead of on data tables
- Catalog access is SLOW

Generalization Problem

- Too much dependence on our internal use case
- Need configuration for things like indexes
- Question of backwards compatibility

Section 6

Stats and Conclusions

Current Stats

- 80 servers
- Four full racks of servers
- 740x8TB HDD
- Approx 17.5PB raw data.
- 30 days of data

Lessons Learned

- Don't turn of autovacuum globally
- Be careful with ZFS and PostgreSQL
- PostgreSQL is awesome.

Thanks for Coming

Thanks for coming! Feel free to reach out after in the hallways etc.

Chris Travers
mailto:chris.travers@adjust.com

Felix Wischke
mailto:felix.wischke@adjust.com