Real time streaming in PostgreSQL

Kaushik Iyer

Agenda

- Initial approach
- Streaming techniques
- Updated pipeline
- Observations

Effective replication of data

All the necessary data is in postgresql

JDBC Connector to effectively pull the data

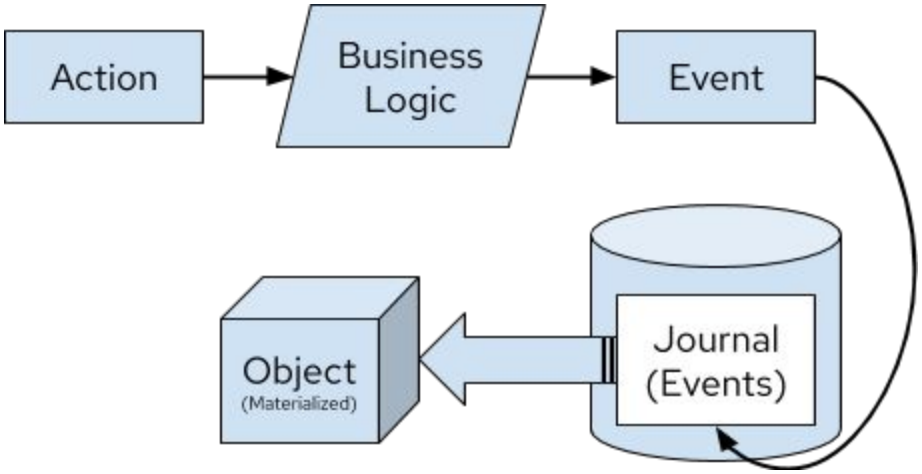
Perform ETL and push to ElasticSearch

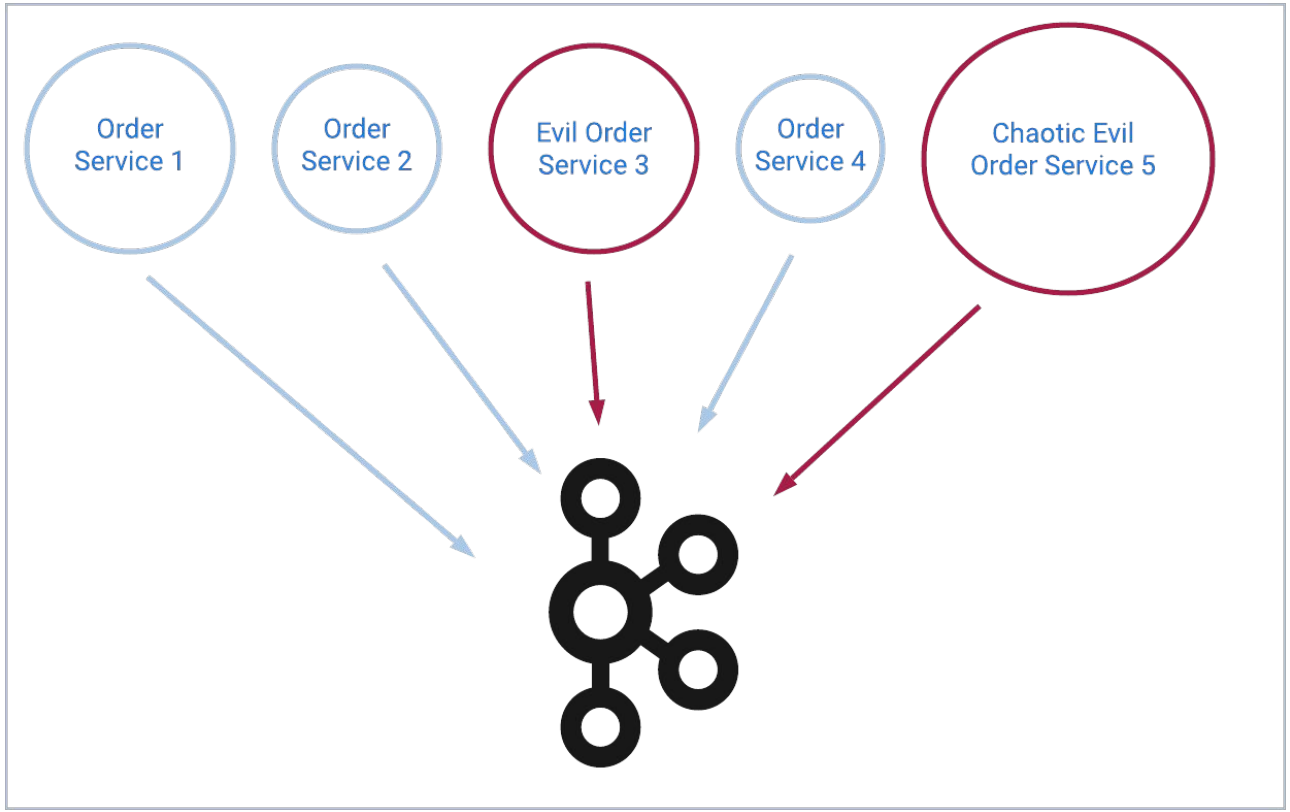
Drawbacks

- Snapshot mechanism involved a lot of trial and error.
- A pertinent lag of 5 minutes was present.
    - Inconsistency in user experience.
    - Non computable during bulk loads.
- We were storing relational data as such in a Document store.

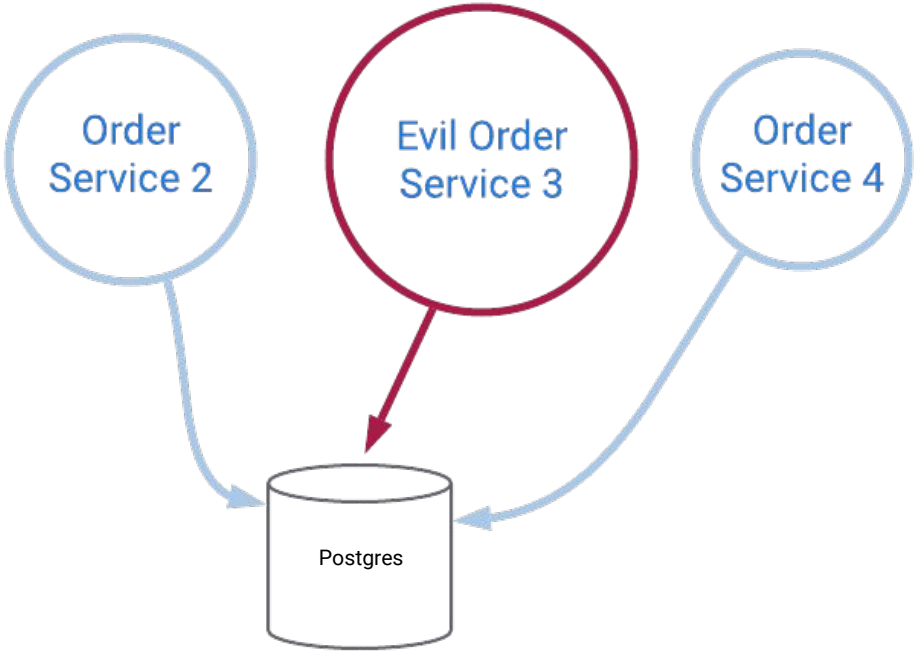
Event Sourcing

Components of a Event Sourcing pipeline

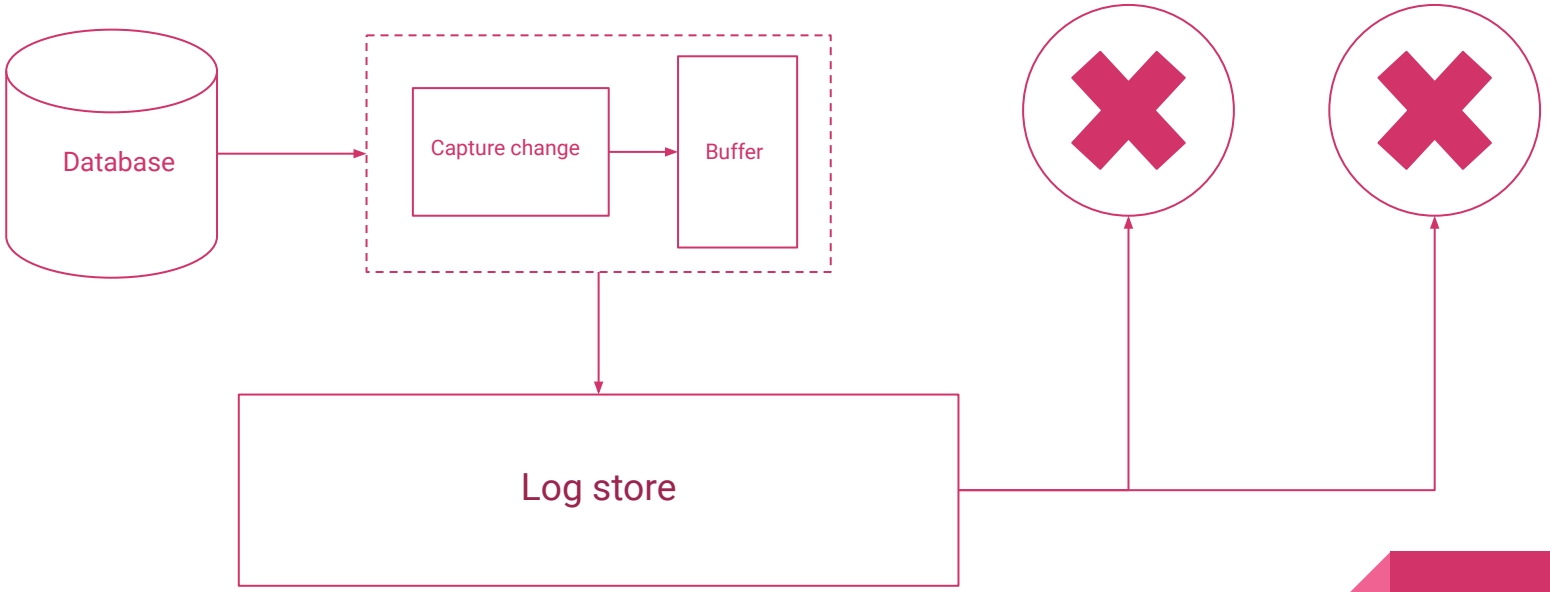



Order Service 1
Order Service 2
Evil Order Service 3
Order Service 4
Chaotic Evil Order Service 5

Change Data Capture


Order
Service 2
Evil Order
Service 3
Order
Service 4
Postgres

Components of a CDC pipeline

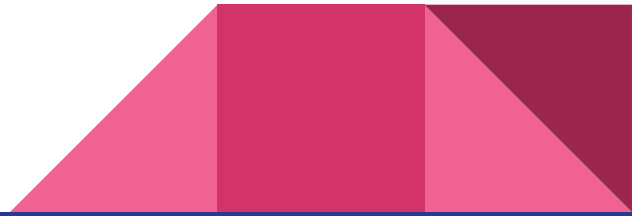


CDC vs Event Sourcing

- Change Data Capture is a form of derived event sourcing.
- We cannot add event generators throughout the platform.
- Change Data Capture is more flexible.

Available CDC software

- LinkedIn DataBus
- Stitch data
- Qlik data
- Oracle GoldenGate
- Netflix Delta



Debezium

Key features

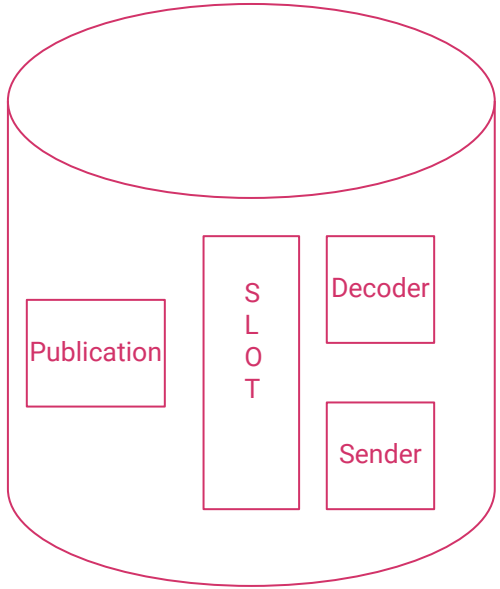
- Detailed message structure, with a plethora of metadata.
- Requires no change to the schema of tables.
- Robust snapshot mechanisms.
- Built in filters and masking options.
- Monitoring through JMX.
- Embedded variant also available.

Debezium sample Event

The CDC Pipeline

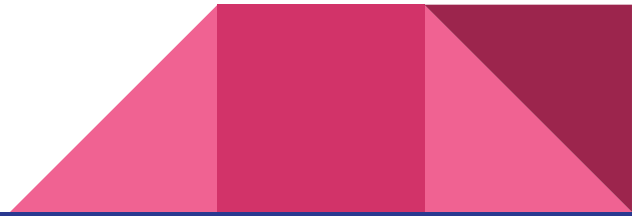
PostgreSQL

- Logical Replication
  - Publisher
  - Subscriber
- Log Sequence numbers
- Replication Slots
- Logical Decoding
  - Wal2json
  - Pgoutput



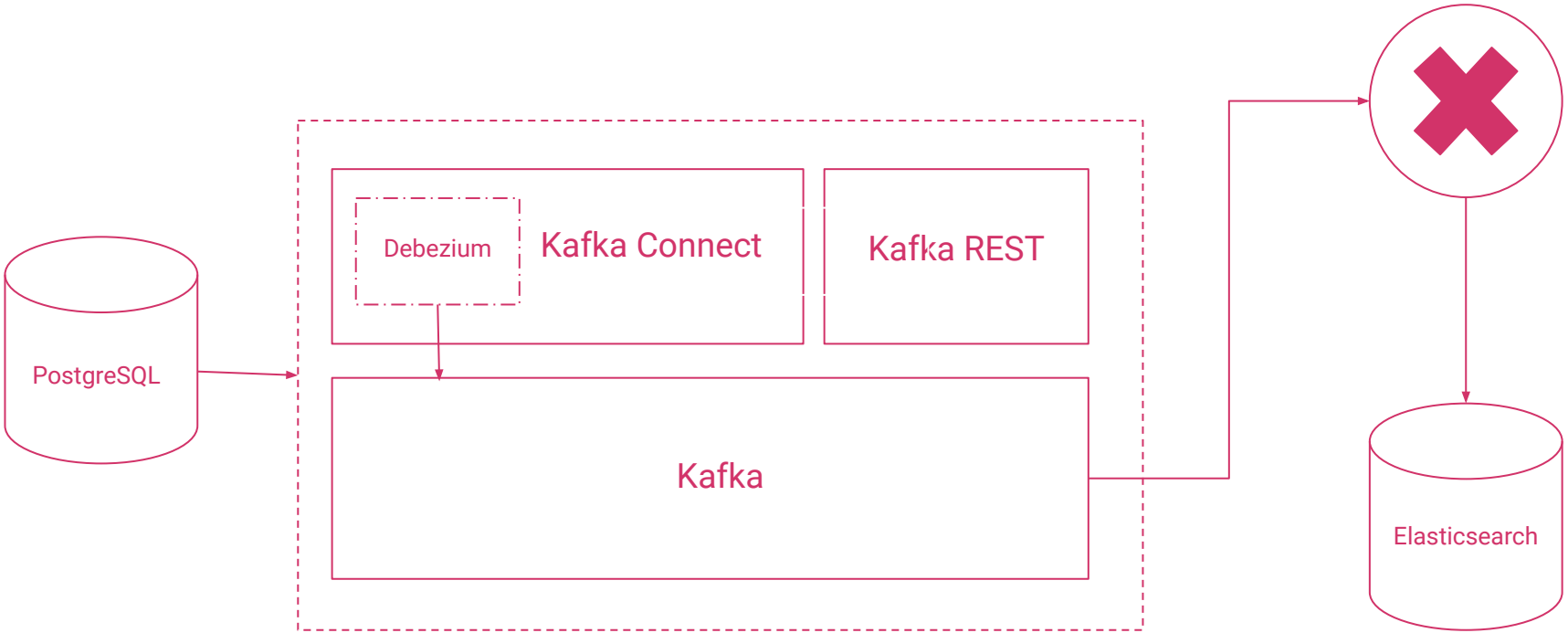
Transaction Log

- We are using Apache Kafka as our transaction log.
- Kafka Connect runs Debezium.
- Kafka REST to manage the lifecycle of connector.

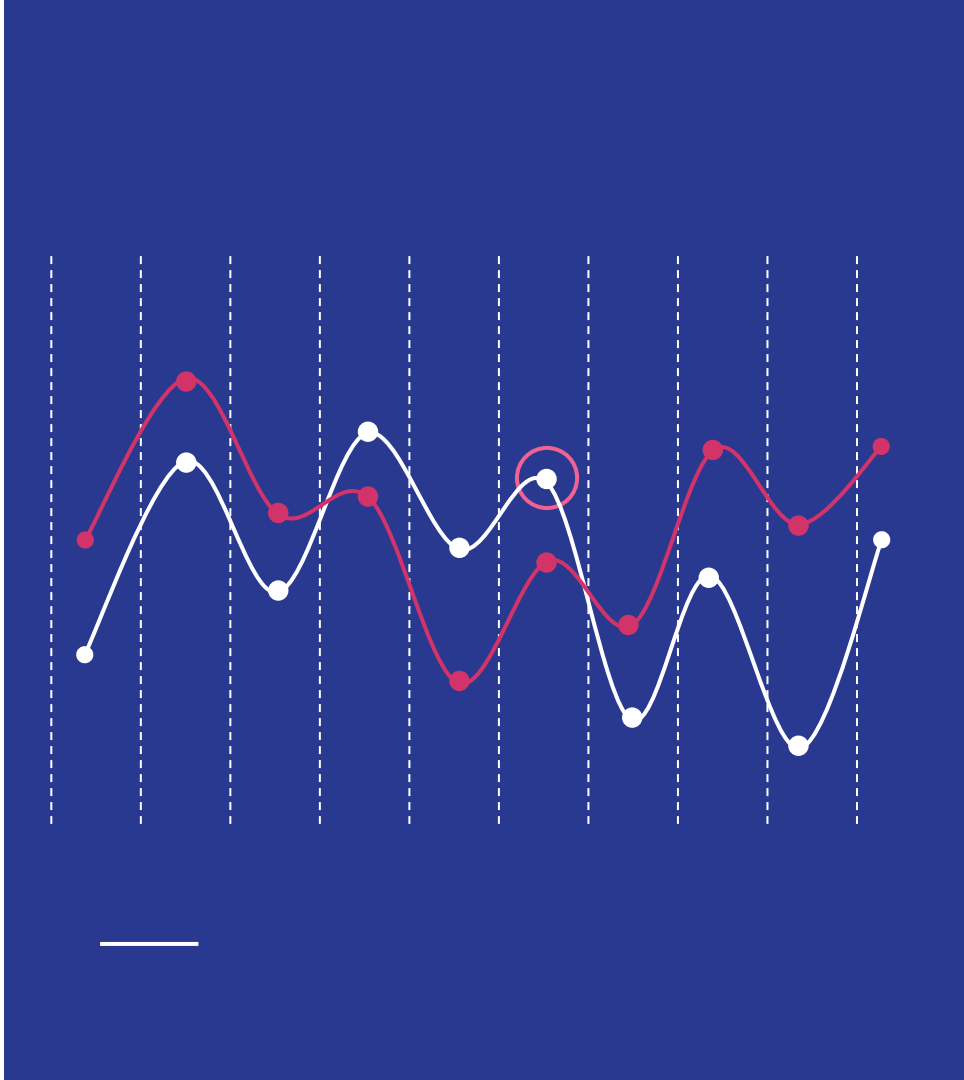


Sample Debezium configuration

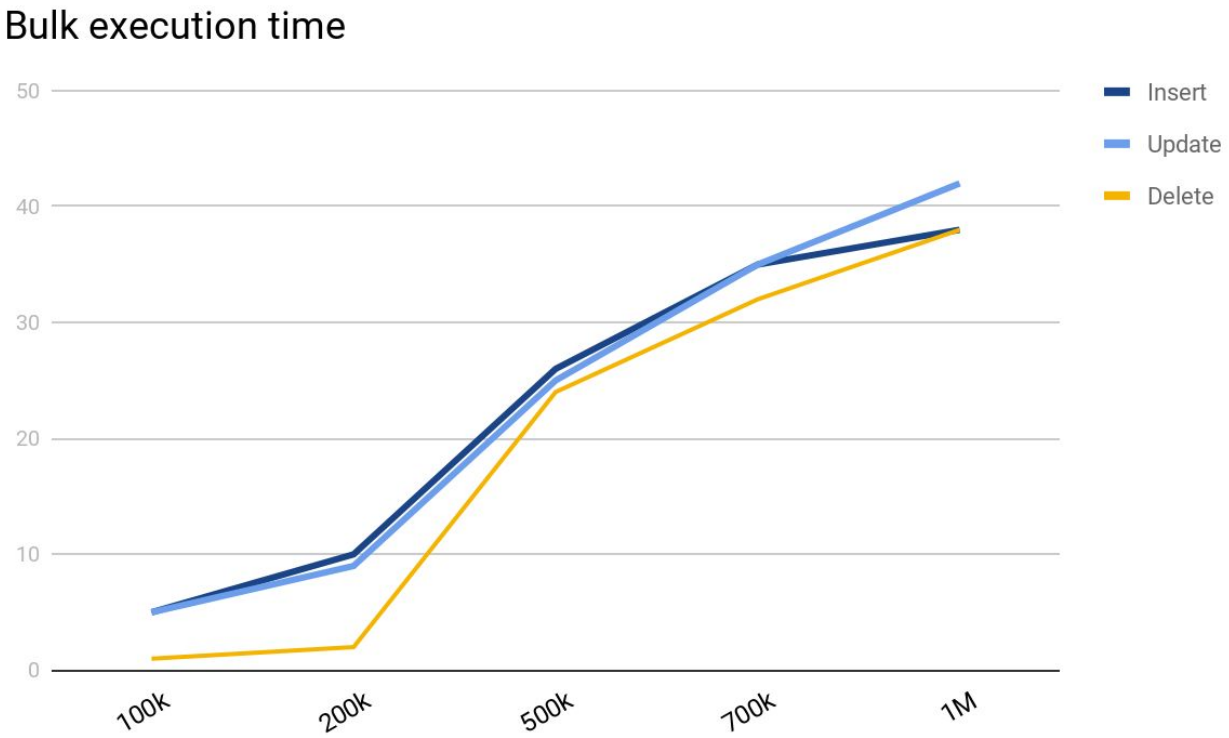
```
{
  "connector.class": "io.debezium.connector.postgresql.PostgresConnector",
  "database.user": "postgres",
  "database.dbname": "foundation",
  "slot.name": "pgoutputfoundationressler2",
  "tasks.max": "1",
  "database.server.name": "foundationtestingreseller2",
  "plugin.name": "pgoutput",
  "database.port": "5432",
  "table.whitelist": "public.reseller, public.domorder",
  "key.converter.schemas.enable": "false",
  "include.unknown.datatypes": "true",
  "database.hostname": "localhost",
  "database.password": "postgres",
  "value.converter.schemas.enable": "false",
  "name": "pgoutput-foundation-reseller2",
  "value.converter": "org.apache.kafka.connect.json.JsonConverter",
  "key.converter": "org.apache.kafka.connect.json.JsonConverter",
  "snapshot.mode": "exported"
}
```


PostgreSQL
Debezium
Kafka Connect
Kafka REST
Kafka
Elasticsearch

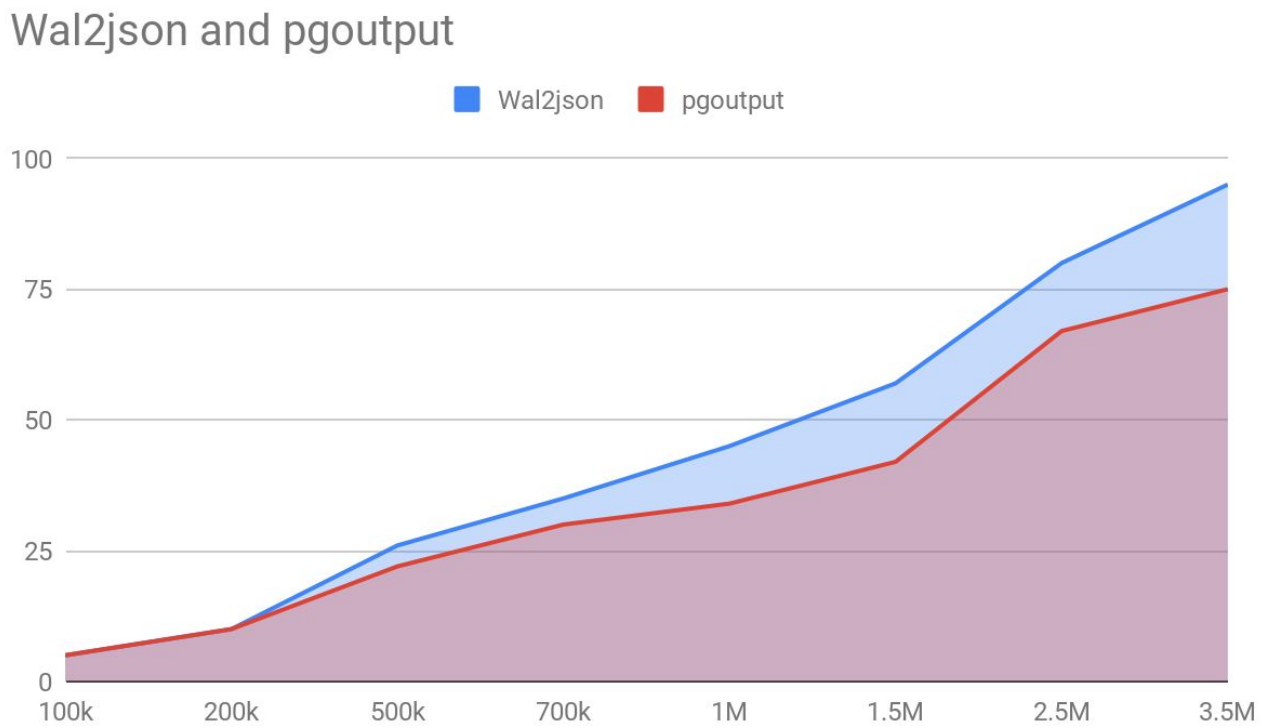
Results and observations



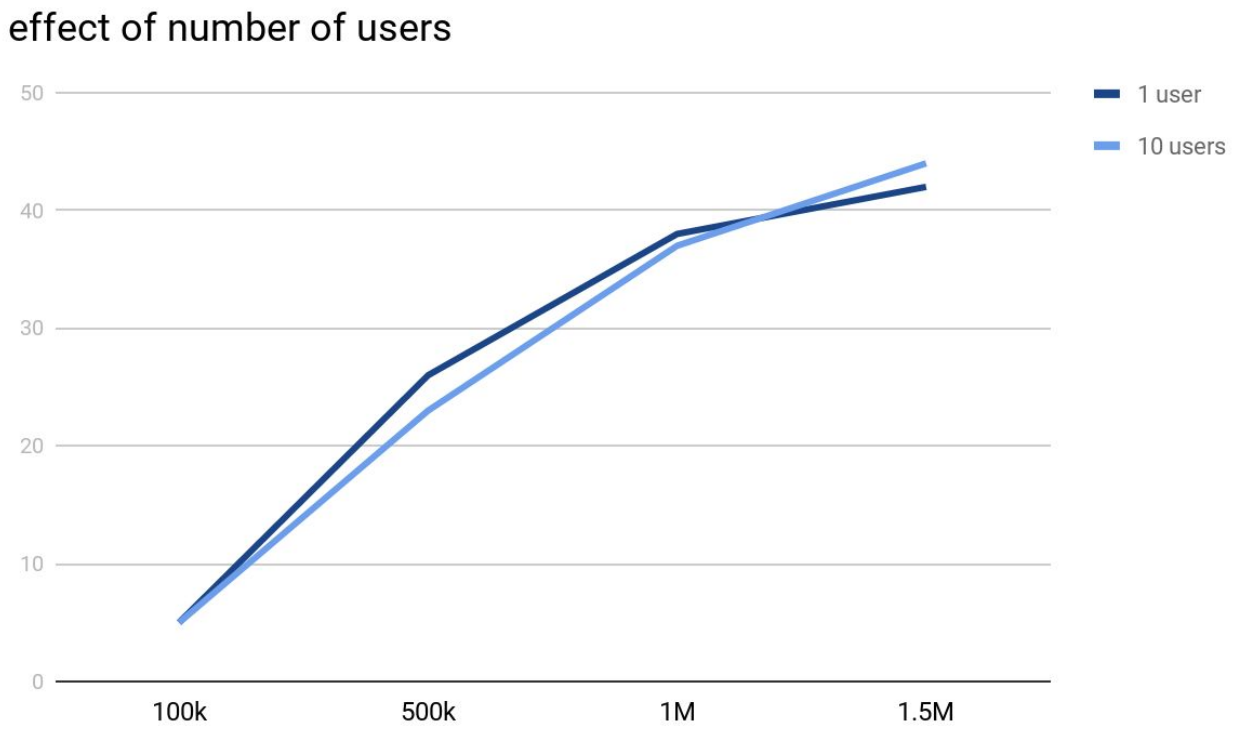
Performance in bulk loads



Performance of logical decoders

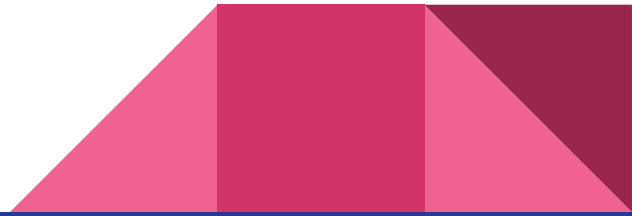


Dependency on number of users



Improvement

- The lag is now in the order of ms
- Effective decoupling.
- High fault tolerance.
- Setup details:
  - M4.xlarge container - ~ $0.2/hour



Thank you

Email: kaushik.i@endurance.com
Github: Kaushiklyer16
Twitter: @kaushiiyer
Linkedin: www.linkedin.com/in/kaushiiyer