Operating PostgreSQL as a Data Source for Analytics Pipelines

data egret

Your remote PostgreSQL DBA team

Ilya Kosmodemiansky

ik@dataegret.com

Securing your PostgreSQL database availability and high performance

Contributing sponsor to PostgreSQL

MIGRATION
POSTGRESQL SUPPORT
CLOUD COST MANAGEMENT

data egret

Your PostgreSQL DBA team

EXPERTISE

- DBA with 10+ years of experience in PostgreSQL administration.
- Significant Contributors

EDUCATION

Co-founders of Open Alliance for PostgreSQL Education

Open Alliance For PostgreSQL Education

OPEN SOURCE ADVOCATES

Accompanying you on your journey to Open Source PostgreSQL

OSBA Mitglied

COMMUNITY

Recognised as Significant Contributing Sponsor to PostgreSQL.

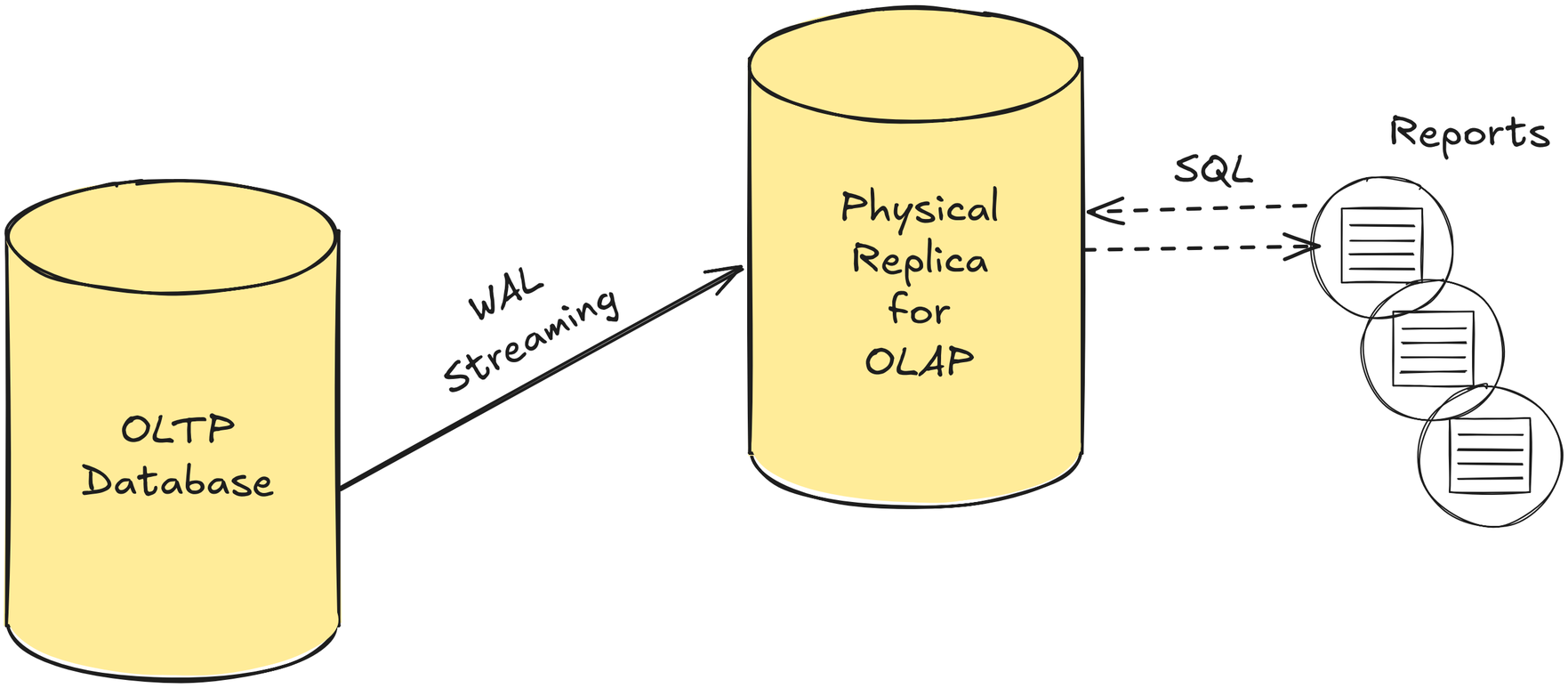
COURSES
FOR DBA
AND DEVELOPERS

Scan to explore curriculum →

DWH, OLAP, Reporting

Analytics as it used to be

Postgres does it good



Physical Replication (Good):

- Physical replication is easy to operate
- We have a physical copy of production OLTP database
- We can use all power of relational databases to create our reports
- Data scientists can’t ruin your production environment*
- Good performance**

Physical Replication (Bad):

- We have a physical copy of production OLTP database
- …and it is read only
- Lack of useful features on the replica (materialized views, temporary tables…)
- OLTP and OLAP workloads are best friends
- Nature of data and workload can affect performance
- Data scientists can ruin everything!
- Are reports the only analytics we want to have in 2025?

Logical replication

Solves the problem slightly better

Logical Replication (Good):

- We can have different schema, more OLAP optimized one
- Our DWH can be a subscriber for different sources
- We can retain data which is removed from primary
- No conflict with recovery problem

Logical Replication (Bad):

- We end up with a big RDBMS (which is not always bad, but…)
- DDL (and sometimes DML)
- Failover (especially before Postgres 17)
- Every team *must* be aware of logical replication

Operating logical replication

Is a topic for a separate talk or rather a training

Christophe made a good one

And before touching performance

we concentrate on something else

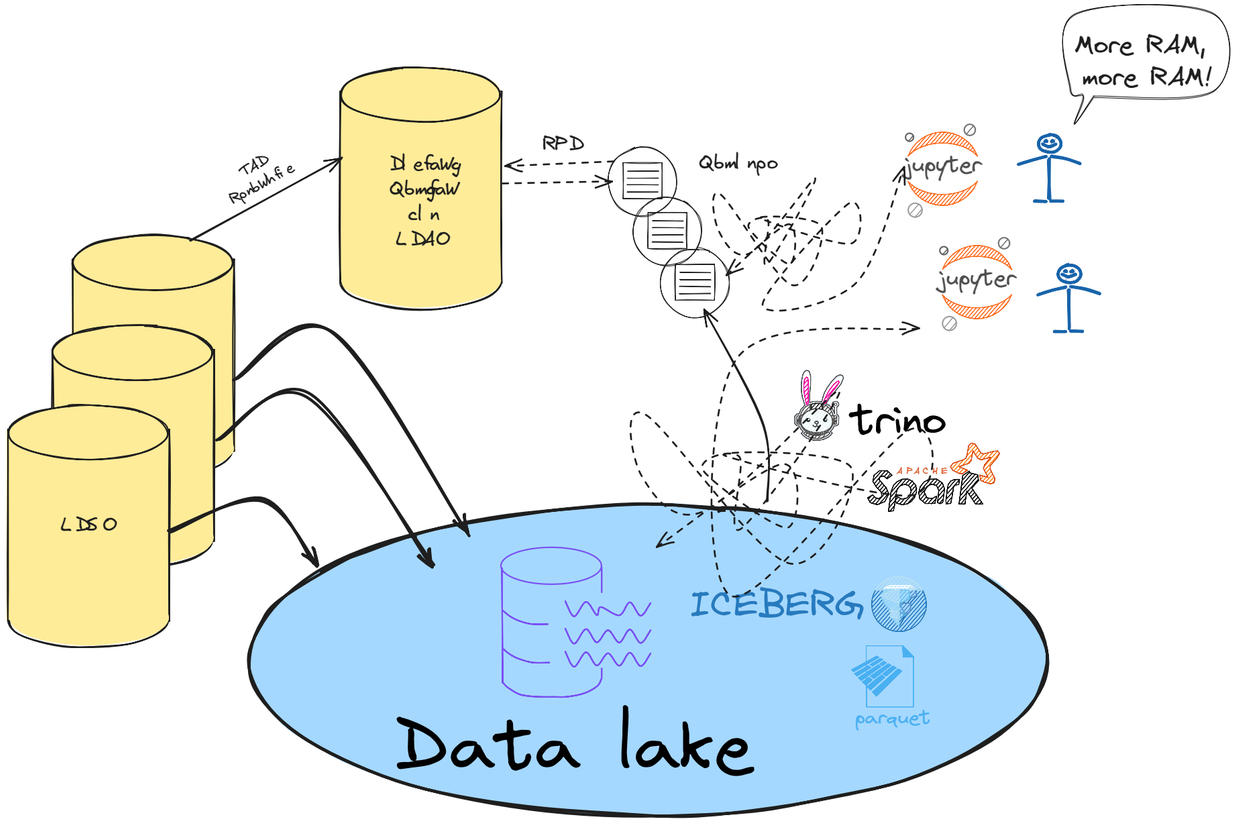
We end up with a big RDBMS

and to figure out if it is good or bad thing, lets make a step back

What is modern Data Analytics?

- Reports are still there
  - And old good Postgres still can SQL better than many
- Machine learning
  - Brave new world of Data Lakes and Lakehouses
  - Completely different lifecycle of data
  - Sometimes realtime and mission critical (anti-fraud, recommendations)
  - Businesses are learning how to earn their money on
  - Engineers learn DataOps job (and FinOps)

Brave new world



ML infrastructure requirements

- Data delivery from operational Postgres
- ML-specific workflows: feature management, DAG storage, enrichment, ingestion, snapshots
- Cost efficiency
- Reliability
- Regulations: for example GDPR and mandatory data retention

What can we use as a Data Lake?

All in one solution

Amazon Redshift, GCP Big Query, Snowflake

- Very attractive
- Data delivery is still a question
- Could easily became *extremely* expensive and barely manageable

Postgres

- Easy to manage
- Data delivery is less a problem
- ML-specific workflows? Big if.

Postgres: ML-specific workflow

- pgvector allows to store features in Postgres
  - but it’s job is rather to provide quick vector search
  - some enrichment capabilities
  - despite all improvements, indexes are not particularly small
- Enrichment, ingestion and generally ETL could easily lead to severe performance issues
- Do we really need a DBMS for a data which fits into S3/Iceberg/Parquet?
- Postgres can be a perfect source of truth however

Better Data Lake

- is usually a combination of more specific technologies (Iceberg, Spark, Hive, Trino, Clickhouse…)
- We still need to deliver operational data

Change Data Capture and Postgres

- ctid based, logical replication based
- Could include some enrichment
- There are ether low code or efficient solutions

Before you start

- Postgres instance must be well tuned
  - Especially WAL and autovacuum
  - Good guide from us on that
- Plan carefully what to CDC
- Workbooks:
  - What to do if CDC stuck?
  - How to resume/resync after an incident
  - Upgrades and failover/switchover

Airbyte

- A typical example of easy low code CDC solution for Postgres to many destinations
- Postgres connector is based on Debezium and we can’t configure it a lot because of no-code approach
- Lets convert everything to JSON approach is not particularly efficient
- Not all destinations are available in open source version

PeerDB and ClickPipes

- PeerDB is open under Elastic license, ClickPipes are cloud-first
- Combination of ctid and logical decoding can significantly speed up initial load

Debezium

- Most matured CDC tool
- Can be used as standalone server, but it is not the best idea
  - Scalability is a problem
  - Much better tools like Kafka Connect/Kafka Streams and Apache Flink use embedded Debezium as a library

Kafka Connect

- Brings us all power of Kafka
- We need to know how to operate Kafka
- It is easy until is not (or until it is just CDC, not stream processing)
- Kafka Streams can add stream processing to CDC, which is good for ML infrastructure
  - But it is rather a framework to write your own software (and **to operate it!**)

Apache Flink

- Client/Server architecture but you still need to write your jobs in Java (you can use Python or SQL as well, but there are limitations)
- Scales well
- A lot of options for performance tuning
- Extremely performant and handles backpressure
- Swissknife in terms of stream processing, so enrichment could be very complex and still fast

What are the options to use Flink?

- Self-hosted FOSS Community Edition
- Aiven for Apache Flink
- AWS Managed Service for Apache Flink
- Confluent Cloud for Apache Flink
- Ververica VERA

Self-hosted FOSS Community Edition

- Most attractive in terms of features
- Standalone or k8s (with operator or without)
- DataStream for low-level stateful streaming, Table API / SQL for relational work, and PyFlink for Python users.
- Connectors for Kafka, Pulsar, Kinesis, JDBC, filesystems, Hive, Iceberg, HBase, MongoDB, Elasticsearch, and more, plus the separately-released Flink CDC project for database CDC (Not only Postgres).

Aiven for Apache Flink

- Was initially limited to SQL first and Aiven infrastructure first Flink
- Currently not publicly available: Aiven for Apache Flink® service creation is limited

AWS Managed Service for Apache Flink

- Most compatible with FOSS Flink in terms of features
- AWS Manages high availability
- IAM, VPC, and KMS

Confluent and Ververica

- Rich in features but most far away from FOSS Flink
- Quite high end in terms of price

Main Takeaways

- Postgres is a perfect source for your analytical data platform in 2026
- Logical replication brings enorm flexibility
- There are plenty of tools based on logical replication
- You can pick up something very simple or something very performant:
  - Airbite for simplicity
  - Kafka if you used to it
  - Flink if you need ultimate performance for complex streaming

Questions?

BTW we have a CDC Whitepaper