Microsoft

skype™

PostgreSQL database platform architecture at Skype

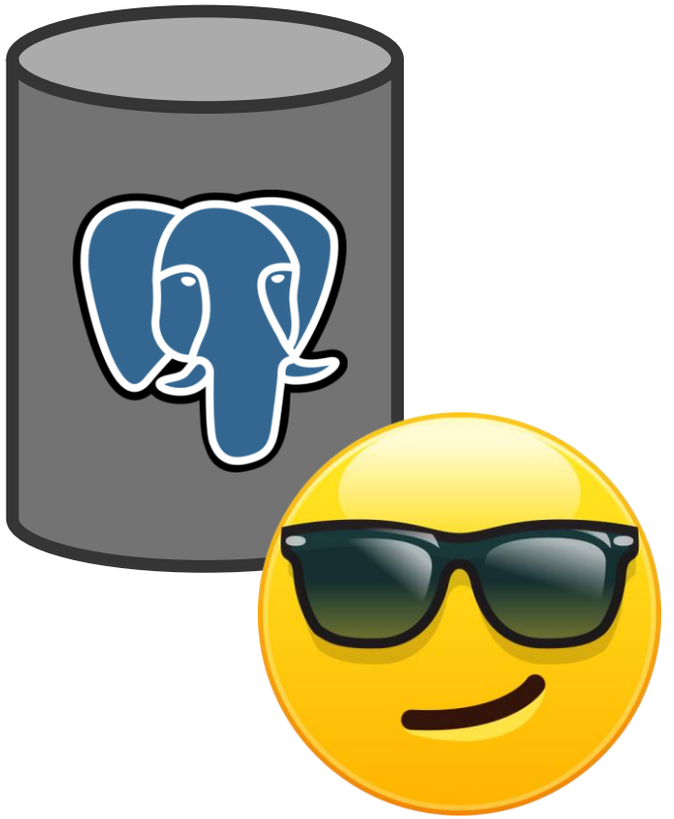
Aleksei Plotnikov
Postgres User Group Estonia
30 May 2017

Who?

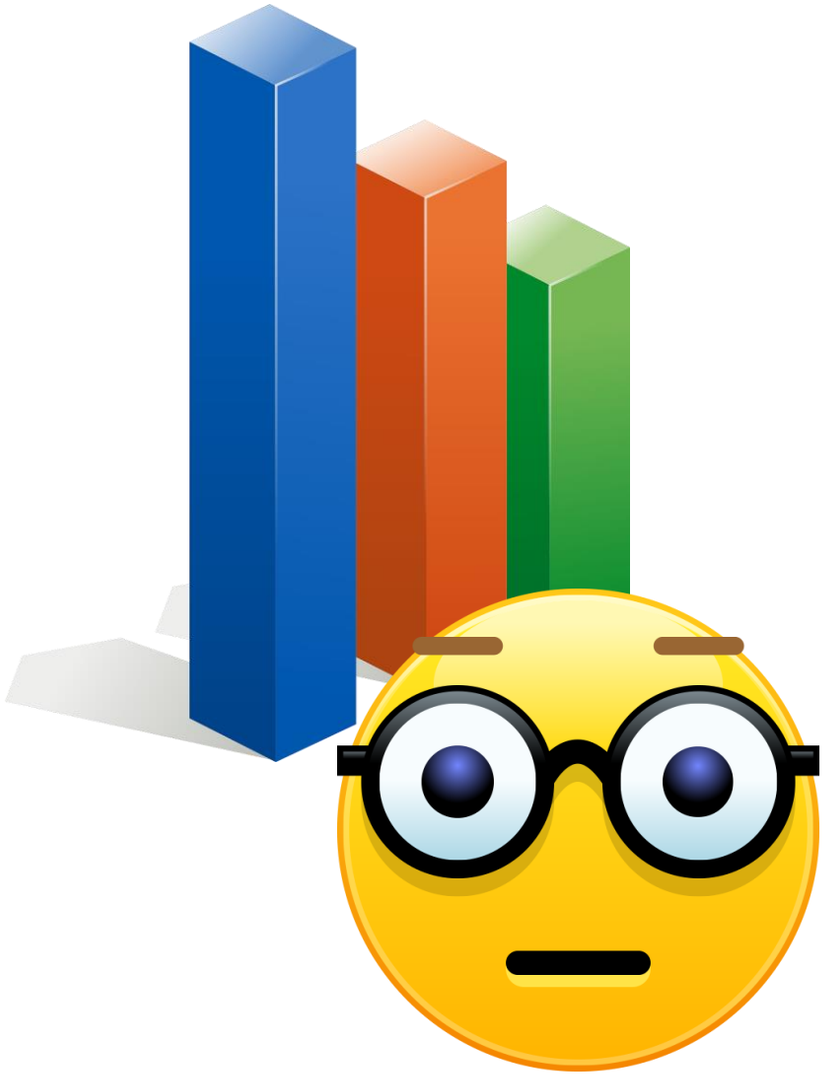
- PostgreSQL active user since v8.2
- Senior Service Engineer @ Skype
- Skype Database Platform team

PostgreSQL at Skype

- Skype for Consumer
- History
  - PostgreSQL choice
  - Quick growth
  - Microsoft
- When PostgreSQL is used?
- Hundreds of millions of active users



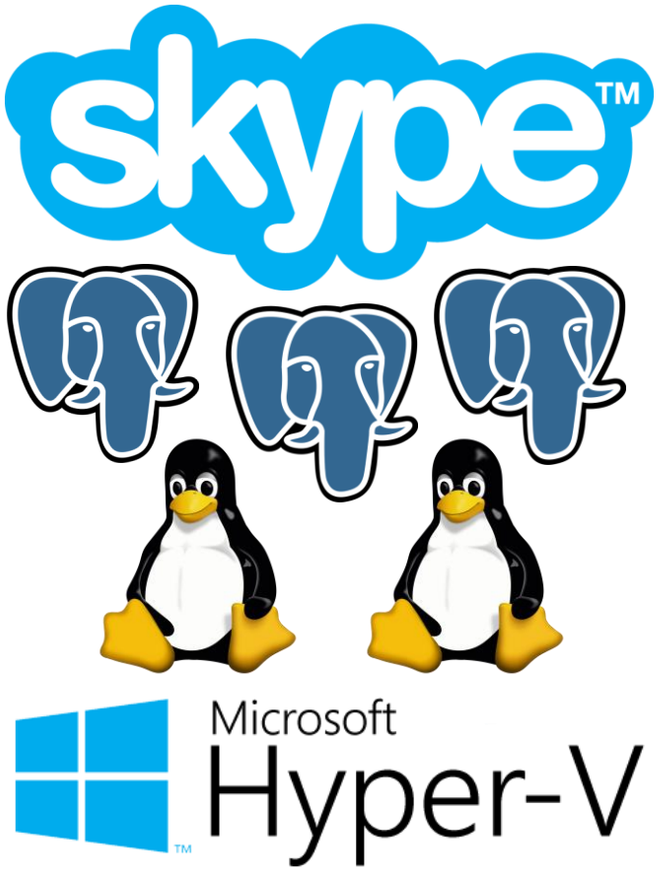
Statistics



- 160 logical databases
- 2000+ physical instances
- ca 1000 servers
- > 200k transactions per second
- Almost 500 TB data volume

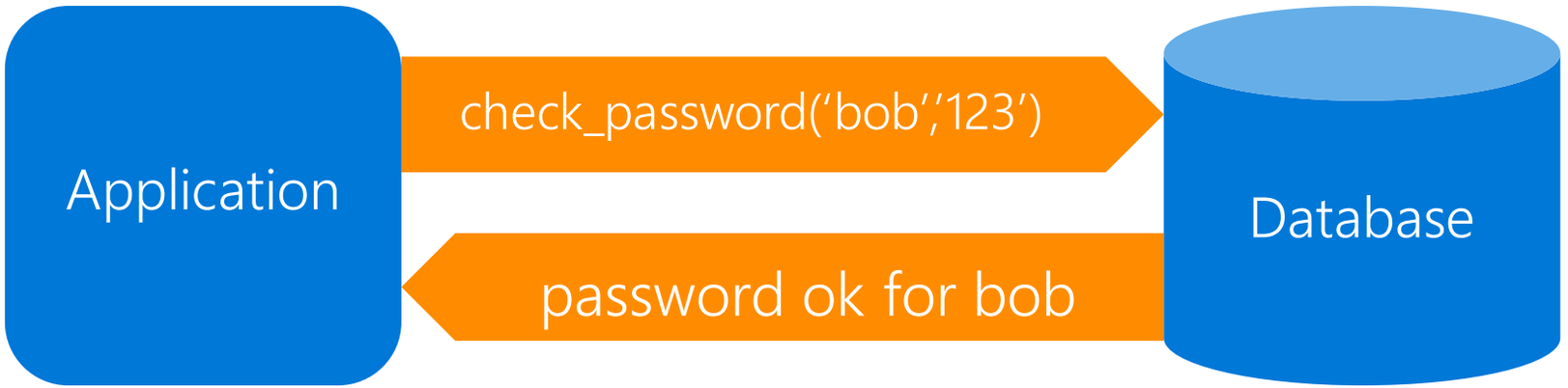
Infrastructure

- 2 datacenters
- Hyper-V virtualization
- SAN
- Debian Linux
- PostgreSQL 9.4



Logical architecture

- Database as a Service
- Stored Procedure API
- Logical access endpoints for applications

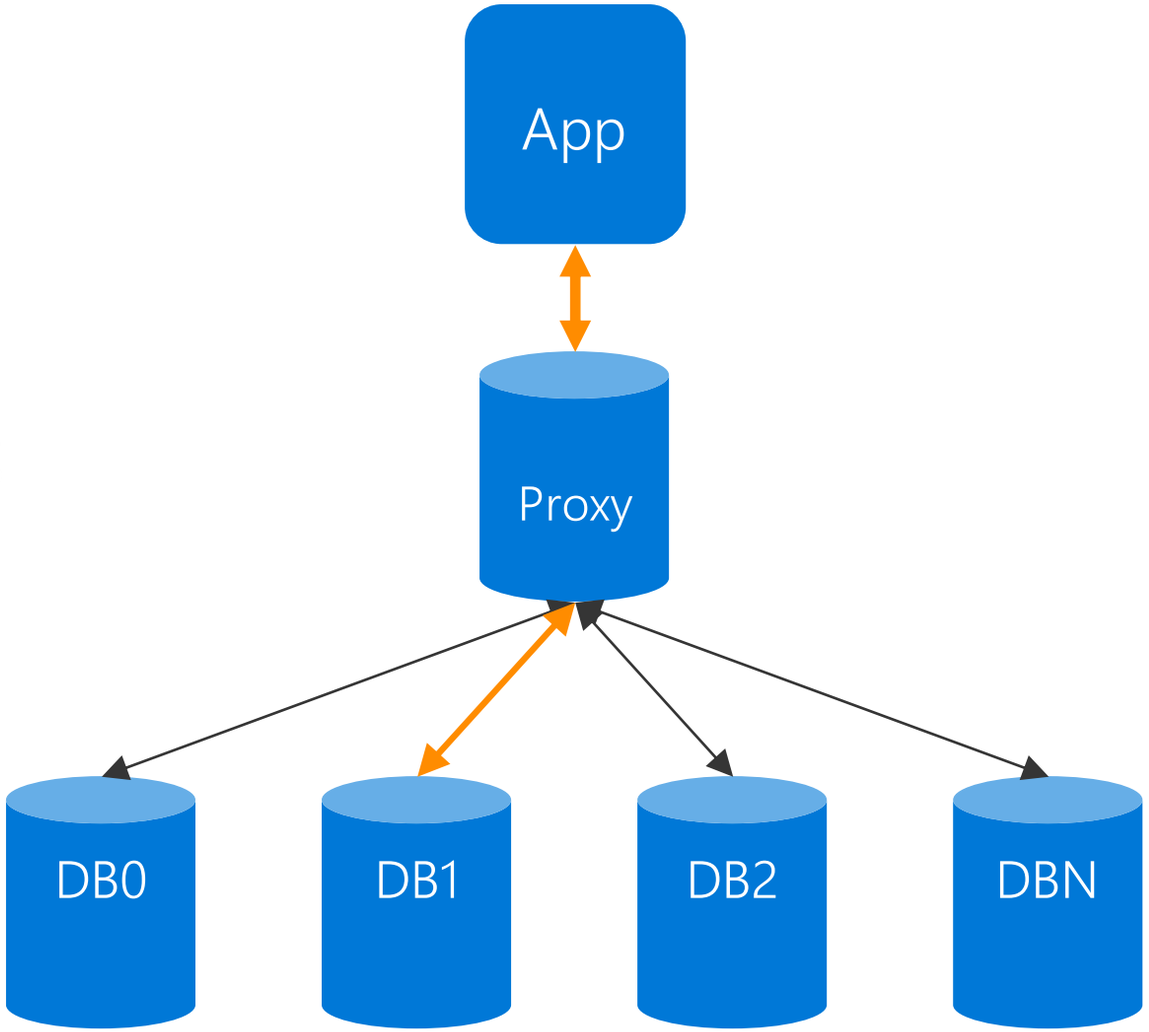


Stored procedures

- Significant limitation, but has many advantages
  - Data processing business logic is run in databases
  - Simplified security model
  - Transparent development and maintenance
- Statistics
  - ~ 20k logical functions
  - More than a million rows of code

Database cluster

- Two levels
  - Data shards
  - Proxy databases
- Remote procedure calls

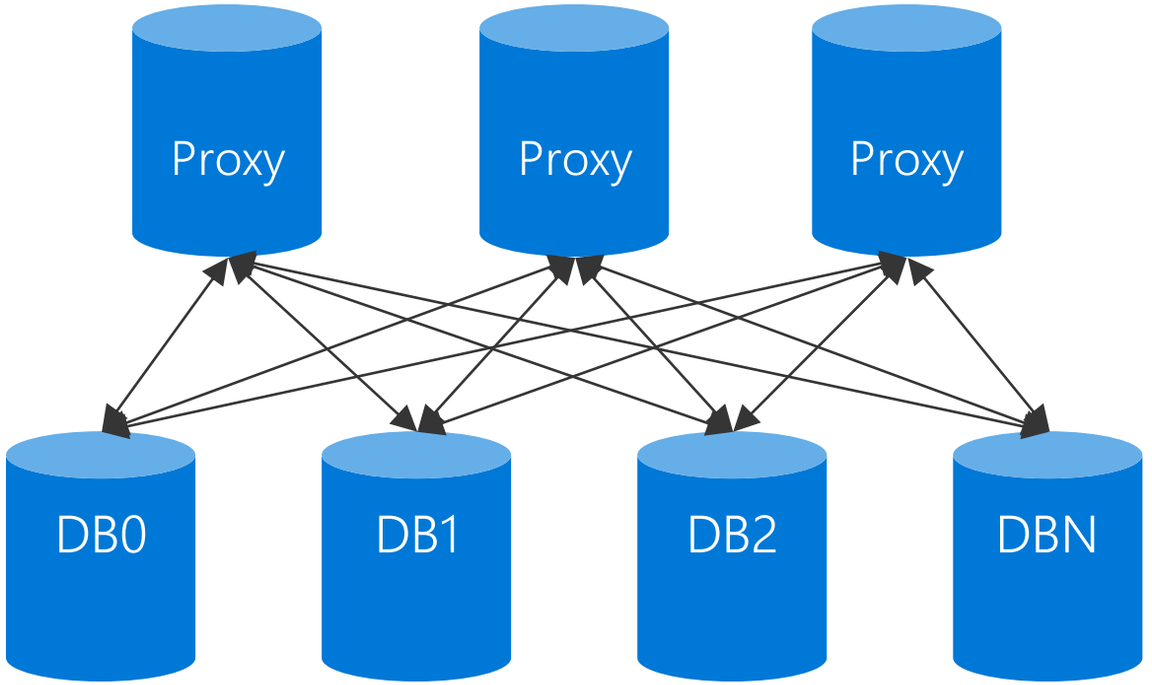


PL/Proxy

- Procedural language for RPC
  - Open source https://plproxy.github.io/
- Simple to configure and use
  - Proxy functions
  - Remote database selection rules
  - Remote procedure with same signature
- Sharding support
  - Dynamic remote DB selection
  - Hashing for equal distribution of data
  - Unlimited scalability
  - Number of shards – power of 2
  - Largest cluster – 256 shards
  - Resharding
- Horizontal RPC

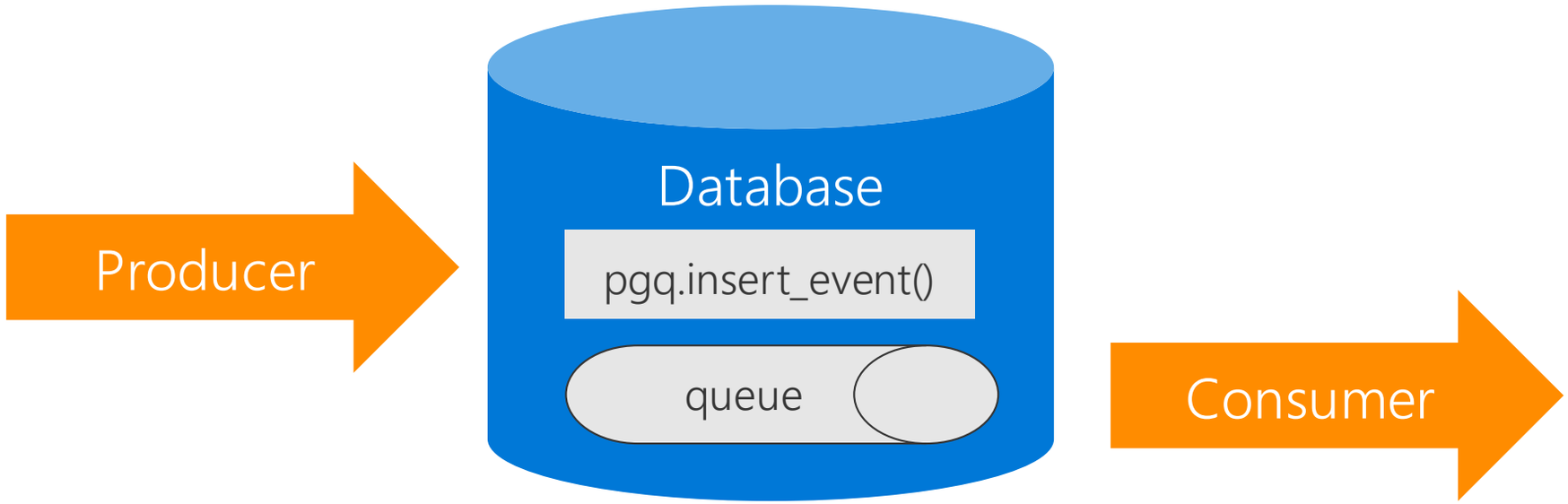
Scaling proxy databases

- Easy to scale
  - No data, only functions and configuration
  - Identical copies
- Load balancing
  - DNS Round-Robin
  - Automatic pool management



PgQ

- Queueing system inside PostgreSQL, written in PL/pgSQL and C
- Asynchronous transactional snapshot based event processing

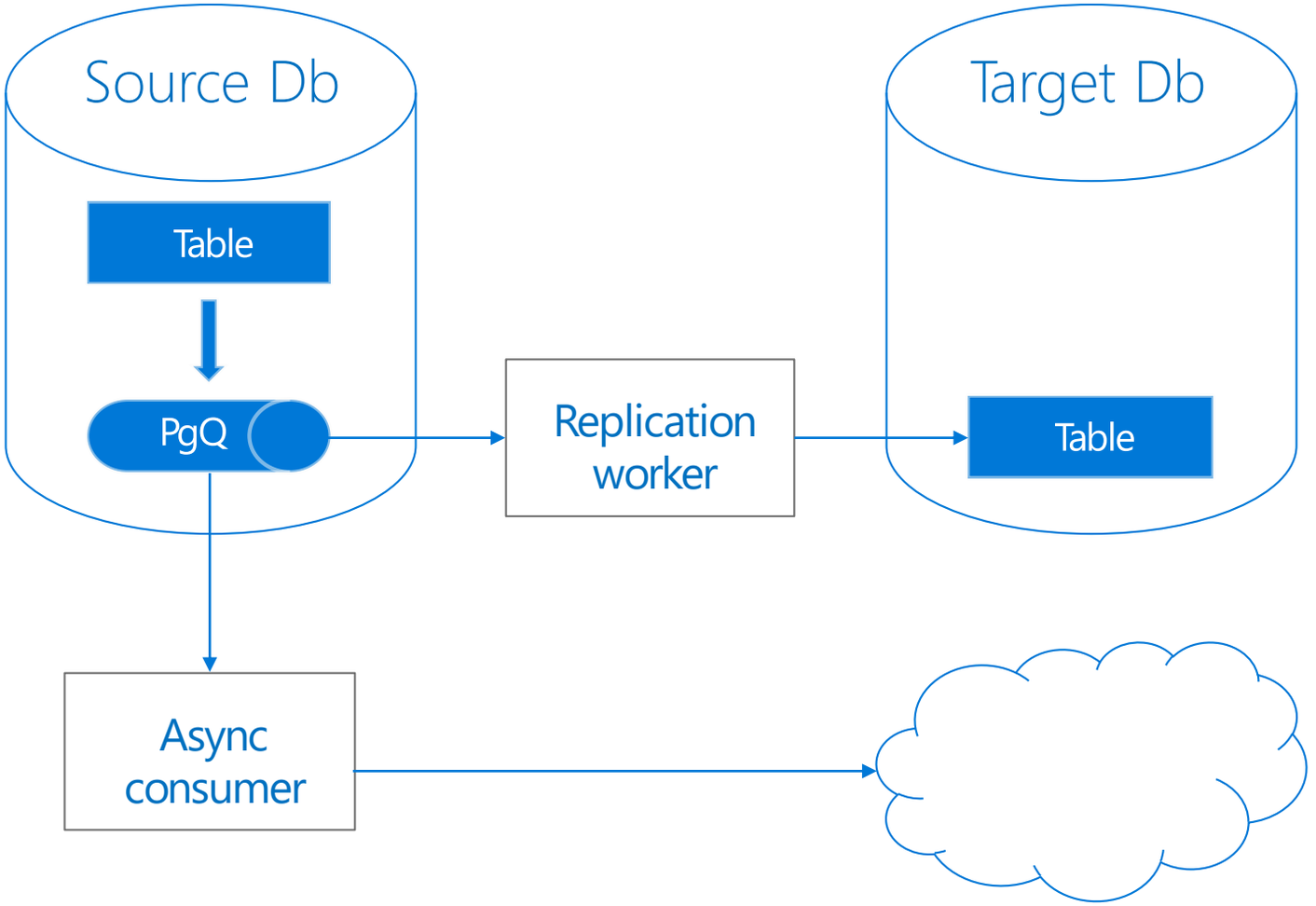


PgQ components

- Producers
  - Application directly via API function call
  - Insert/Update/Delete/Truncate table triggers
- Consumers
  - Every consumer will see every event at least once
  - Processed event tracking or retrying using different methods
  - Cooperative consumers and cascade consumers
- Pgqd ticker
  - External daemon
  - Generate ticks to cut groups of events – “batches”, for improved performance
  - PgQ queues regular maintenance
- Queue tables

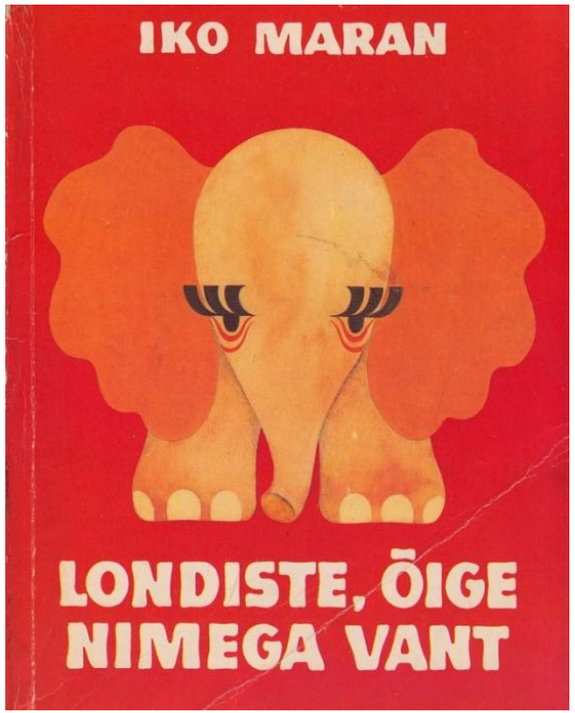
PgQ use cases

- Asynchronous batch data processing
- Replication



Londiste3

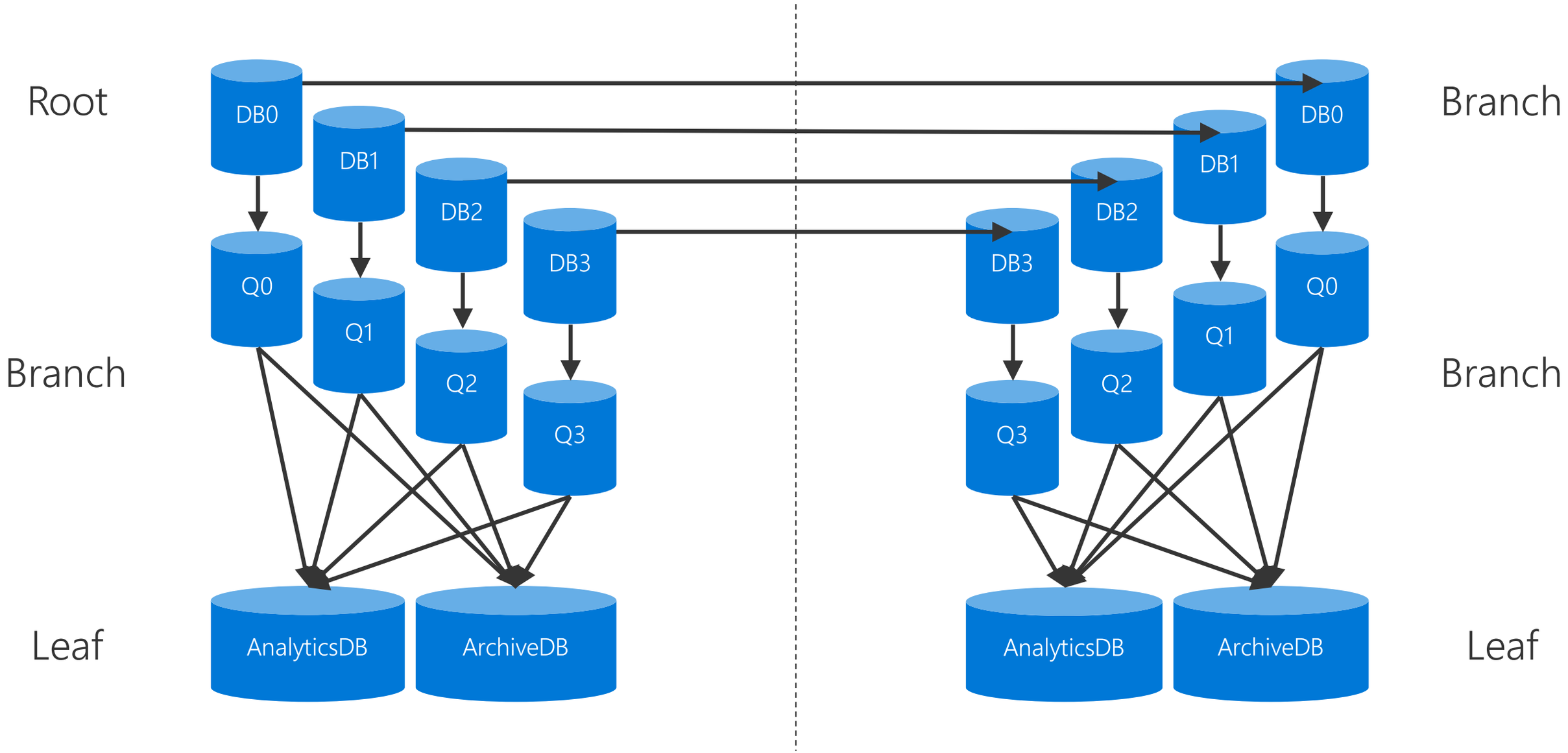
- Trigger based replication system
- Asynchronous logical replication
- Uses PgQ as transport engine
- Complex PgQ Python consumer
- Tables and sequences
  - Primary key
  - Automatic creation
  - No structure synchronization



Cascaded replication

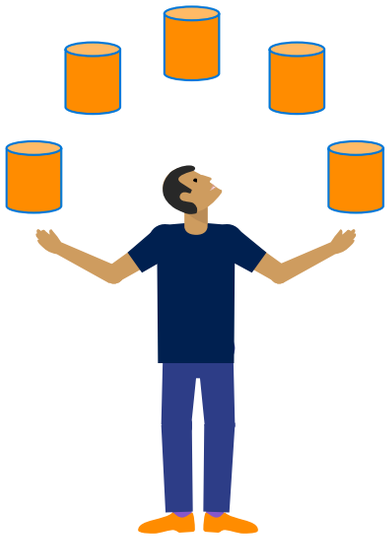
- "Chains" of replicas
- PgQ allows to replicate queue structure and data
- Cascade node types
  - Root
  - Branch
  - Leaf
- Queue nodes

Cluster cascades topology example



Cascade topology management

- Online reconfiguration
- Provider change
- Takeover
- Switchover
  - Deny triggers
  - Traffic redirection
- Failover
  - Dead root
  - Possible data loss
- Resurrect
  - Return old root into cascade as branch
  - Creates JSON file with not replicated events



Additional features

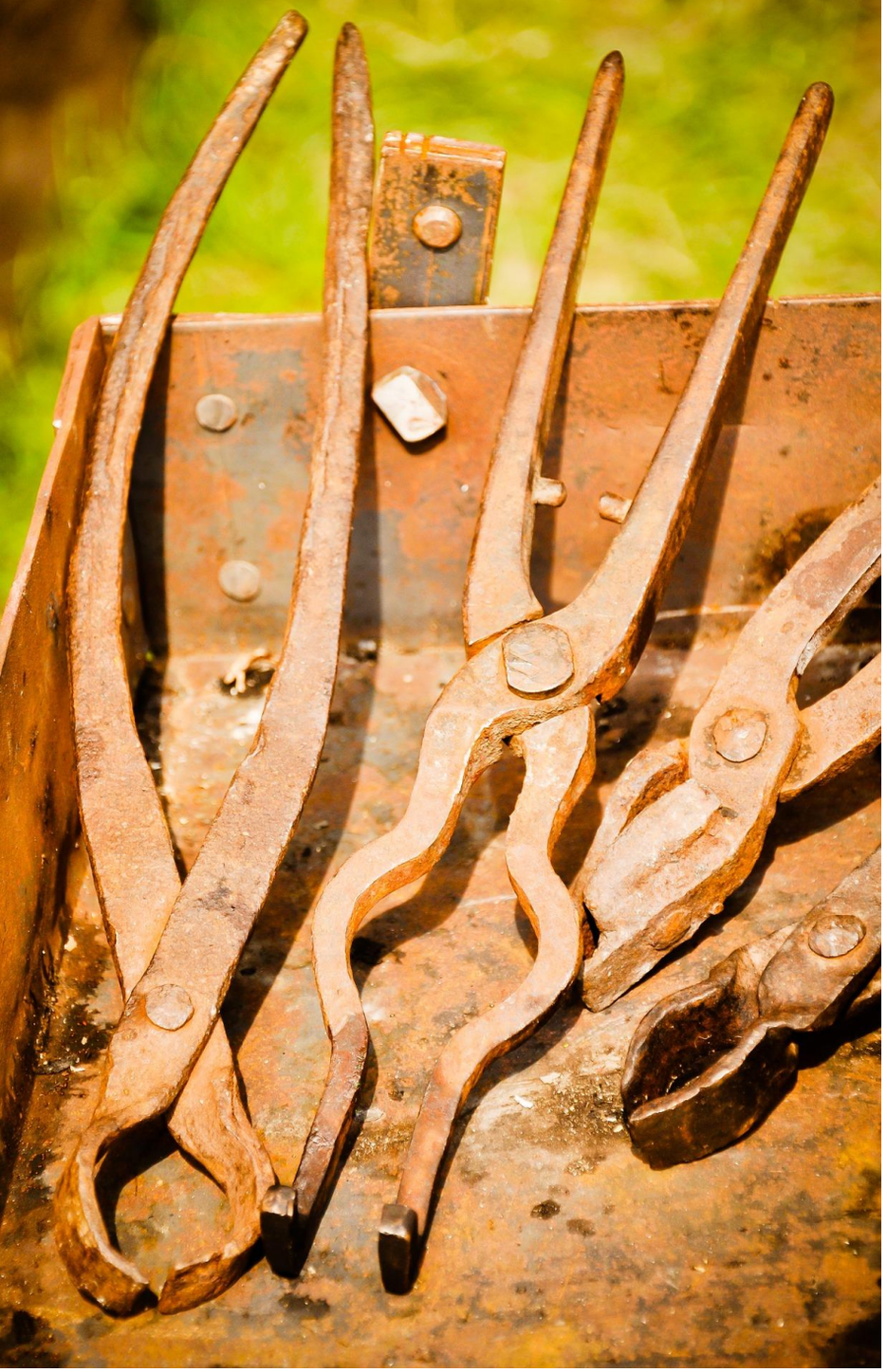
- Handlers – data processing methods
  - Data partitioning
  - Sharded data merging and splitting
  - Skip columns
  - Fix incorrect UTF8 data
- Check and synchronize data
- Cascades merging

Londiste usage examples

- Data copying from online databases to internal for analysis and archiving
- And vice versa
- Read-only databases for load-balancing
- Database copies in other sites
  - Disaster recovery
  - Heavyweight DDL
  - DB, OS, hardware maintenance and upgrades

Londiste and PgQ pitfalls

- Needs attention
- Incorrect data processing
- Lag
- Database structure differences
- Large batch processing

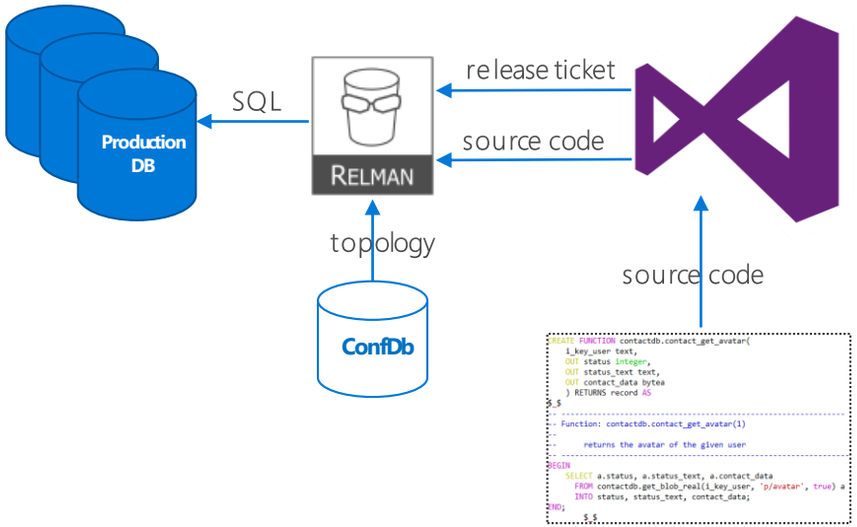


Skytools3

- Package with PgQ, Londiste and other technologies for PostgreSQL
- Open source project
- Python framework

Database code change management

- Up to 200 database releases monthly
- Relman
  - Database code deployment automation system
  - Declarative objects description
  - Support for all DB platform components
  - Visual Studio Team Services intergration
- Automatic deployment
  - Sanity checks
  - > 90% of release items
  - Others need DBA assistance
- DBA team impact

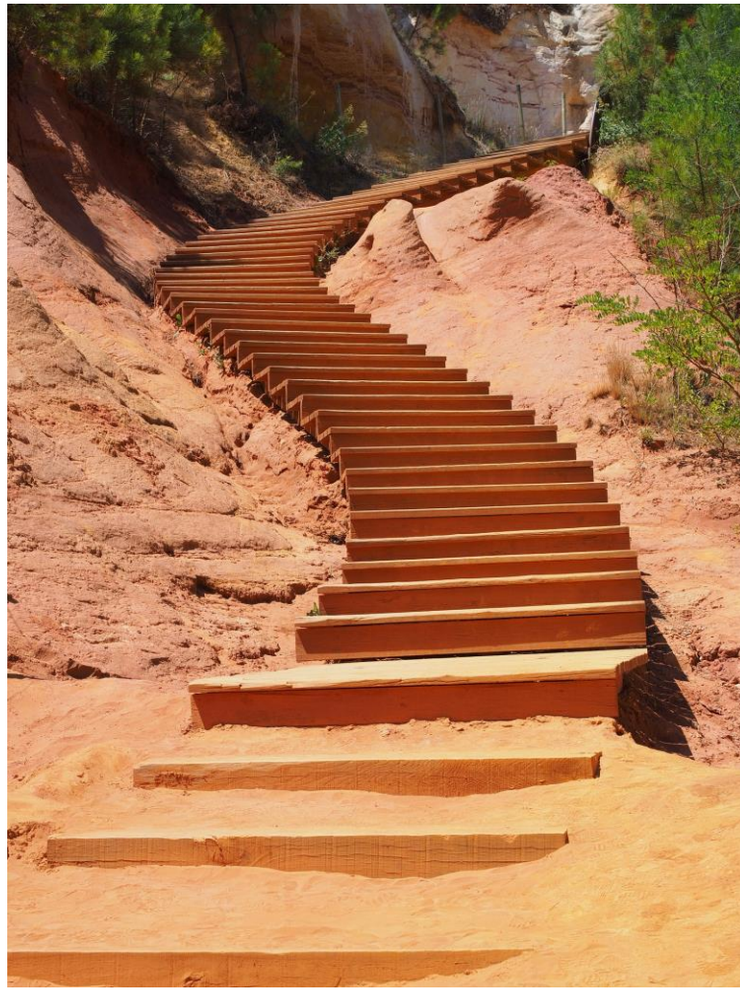


My contacts

- Skype: agent_persik
- E-mail: aleksei.plotnikov@skype.net

PostgreSQL major version upgrade

- Large online database
- Used 24h
- No data loss
- Zero downtime
- Roll back option



PostgreSQL major version upgrade

- Preliminary testing
- Two copies of DB, primary and secondary
- Londiste replication between them
- Upgrade secondary
  - pg_upgrade --link
- Switch over to new version
  - londiste3 takeover
  - Applications traffic redirection
  - Check
- In the case of any problems – switch back
- Primary database upgrade and switchover using same methods