Postgres-BDR: Advanced HA Clustering & Scaling

Simon Riggs
CTO, 2ndQuadrant

17 Oct 2019
Postgres-BDR3 Advanced Clustering & Scaling
BDR History

Largest single contribution project to PostgreSQL

- 2009  Logical replication design
- 2012  BDR prototype
- 2014  BDR1 in production

- 2012+  Many BDR features contributed to PostgreSQL
BDR Editions and Versions

PostgreSQL 11, with Advanced Features

- BDR 3.6.9 current version
  - BDR-SE Standard Edition
    - All features in Extension
  - BDR-EE Enterprise Edition
    - Various advanced features

- BDR 3.7 available October 2019
BDR3 Fundamentals

MultiMaster Database for PostgreSQL

- Multiple Master nodes
- Fully automatic DML replication
- Fully automatic DDL replication
- Replication options
  - Efficient (Async)
  - Eager Replication
A small cluster of 2-3 nodes is called a **Group** (or Group), multiple groups form the building block for advanced clusters:

- Integrated backup
- Integrated routing for fast switchover to alternate nodes
- Repair lost nodes
BDR AlwaysOn

Very High Availability Clustering

- Active-Active
  - One Group on each Site
  - 2-3 DB nodes per Group
  - One main node, switching to other nodes should node, site or network failures occur
BDR Worldwide

- Multiple Sites
  - Up to 32 Sites
  - No distance limitation
  - Option to store data only on local site
  - Suitable for IoT, Monitoring and TimeSeries
BDR AutoScale

Massively Parallel Database & Elastic Scaling

- AutoScale offers Sharding solution
  - Elastically scalable cluster of 2+ Groups
  - Optional Read/Write Coordinator Groups(s)
  - Optional Disaster Recovery site
BDR Use Cases

Advanced Clustering & Scaling

- BDR Worldwide
  - Geographically Remote Databases

- BDR AlwaysOn (3.6+)
  - Very High Availability PostgreSQL

- BDR AutoScale (3.7+)
  - PostgreSQL MPP databases using BDR sharding
BDR Fast Switchover

- Execute on **node1** until failure, fast failover **node2**
- Compare 30-90s for single master failover
  Against <100ms for AlwaysOn failover
BDR Data Loss Protection

- **Commit At Most Once** ensures that any in-flight transactions with unknown state are fully resolved
  - No transactions are duplicated or skipped
  - Works for Session and Transaction mode pooling

- Data in other sites for Disaster Recovery protection can be read and used for reporting/additional uses, since they are active they can use temp tables etc.
Rolling System Upgrades

- Rolling upgrades start with least used node and roll across all nodes slowly

- System Upgrades can upgrade BDR and/or main PostgreSQL releases
  - e.g. PG10 to PG11
    - E.g. BDR3.5.5 to BDR3.6.2
  - Nodes re-negotiate their protocols to ensure compatibility
Rolling Database Schema Upgrades

- Rolling upgrades start with least used node and roll across all nodes slowly, managed under DevOps control
- Update application’s database schema
  - BDR tolerates mismatched schemas such as additional/missing columns, different datatypes, differing indexes
  - Application stays online during upgrade
  - Bad situations can be backed out
BDR Performance

Real-World Production Performance

- Massive partitioning performance gains
- Efficient logical replication
- Streaming of large transactions
- Efficient distributed sequences
- Choice of options for selecting appropriate robustness and performance trade-offs
- Replication performance analysis, Lock wait times and I/O timing
Writing to Postgres-BDR

Distributed database options

- **Post**-Commit Synchronization
  - Resolve issues *after* COMMIT
    - Row-level Conflict Handling by default
    - Column-level Conflict Handling option
    - Conflict-Free Custom Datatypes (CRDTs)
    - Logging and resolution of issues, Conflict Triggers

- Eventual Consistency
  - Fast: Low latency, suitable for wide distribution
Writing to Postgres-BDR

Distributed database options

- **Pre-Commit Synchronization**
  - Eager Replication *avoids* conflicts
    - All Nodes
    - ( Majority nodes: roadmap feature )

- Avoids issues at COMMIT
  - Additional latency not desirable in many cases
  - Some transaction aborts in conflict cases
  - Suitable for high value data/hi latency tolerance
BDR Application Requirements

Advisory, not Mandatory

- Unique identifiers for rows (INSERTs, UPDATEs)
- Don’t change identifiers (UPDATEs)
- Don’t reuse identifiers (quickly) (DELETEs)

- If you don’t follow these you may get conflicts/issues
- BDR Assessment offers tools to identify these
- BDR LiveCompare offers data verification/correction to assure production systems
AutoScale

Shard data for OLTP and BI

- OLTP on Coordinator group
- BI on array of Shard Groups
- We can add optional Read Coordinator nodes
- Easily upgrade array of groups, without moving existing data
AutoScale Read/Write

Data Node only configuration

- Coordinators have Foreign Tables to BDR Server
- BDR performs Data Routing
- All query access happens via Postgres FDW mechanisms
AutoScale HA

- Each Group has multiple nodes with redundant copies
- BDR performs Data Routing **dynamically** in case of down nodes
- Built-In HA
AutoScale Large Query

Data Node only configuration

- Multi-partition queries access multiple Foreign Tables
- Parallel query occurs because access is spread across multiple nodes
Example Query - SSB Q3.2

SELECT c_city, s_city, d_year, sum(lo_revenue) as revenue
FROM  lineorder
    JOIN customer ON lo_custkey = c_custkey
    JOIN supplier ON lo_suppkey = s_suppkey
    JOIN date ON lo_orderdate = d_datekey
WHERE c_nation = 'UNITED STATES'
    and s_nation = 'UNITED STATES'
    and d_year >= 1992 and d_year <= 1997
GROUP BY c_city, s_city, d_year
ORDER BY d_year asc, revenue desc;
AutoScale Join Query

- Join queries access multiple Foreign Tables
- Join is pushed down to shards

- Star Schema joins only, covers most performant case
AutoScale Supported Data Models

Snowflake Schema

- Multiple Fact Tables
  - Range Partitioned only using matching partitions
  - 1 to Many Relationships between Fact tables
  - Spliced across shard groups
- Multiple Dimension tables
  - Copies on all groups
  - Normalized
BDR Multi-node Query

Consistency and Performance

- Timestamp-based consistency (ClockSI)
- Allow consistent queries **across** nodes even with real-time replication of data
- Data verification between nodes

- Multi-node parallel query (MPP) across
  - Local clusters with remote DR nodes
  - Geo-distributed clusters
BDR3 Enterprise Edition (BDR-EE)
PostgreSQL, with Advanced Features

- Very High Availability
- Maximum Data Protection
- Rolling System Upgrades
- Rolling Application Upgrades
- AutoPartition
- AutoScale
- Performance & Security
- Robustness from Production Experience
Postgres-BDR Plugin for OmniDB

Visual Administration

Postgres-BDR3 Advanced Clustering & Scaling
Cloud Native Integration

*Working together in the Cloud*

- 2ndQuadrant is a Silver Member of the CNCF
- Kubernetes operators for BDR and PostgreSQL
- OpenTracing built into BDR3.7 for end-end observability
- Prometheus storage plugin for BDR AutoScale
- Fluentd integration via syslog input
- TPAexec Cloud/On-Premise Orchestration
- Postgres Cloud Manager for Pure/Hybrid Own-Management