Major Features: Postgres 10

Bruce Momjian

PostgreSQL is an open-source, full-featured relational database. This presentation gives an overview of the Postgres 10 release.

 http://momjian.us/presentations

Last updated: July, 2018

Postgres 10 Feature Outline

1. Logical replication
2. Partitioning syntax
3. Crash-safe, faster, and replicated hash indexes
4. ICU library
5. Quorum commit
6. Progress on parallelism
7. Multi-column statistics
8. pg_stat_activity improvements
9. SCRAM-SHA-256 authentication
10. FDW aggregate pushdown
11. More

Full item list at https://www.postgresql.org/docs/devel/static/release-10.html

1. Logical Replication

Cluster 1, port 5432, database 'test'	Cluster 2, port 5433, database 'test'
`$ psql -p 5432 -c ''ALTER SYSTEM SET wal_level = 'logical';'' test`	
`$ pg_ctl -p 5432 restart`	
`$ psql -p 5432 test`	`$ psql -p 5433 test`
`CREATE TABLE test (x INT PRIMARY KEY);`	`CREATE TABLE test (x INT PRIMARY KEY);`
`INSERT INTO test VALUES (1);`	
`CREATE PUBLICATION mypub FOR TABLE test;`	
	`CREATE SUBSCRIPTION mysub CONNECTION 'dbname=test port=5432' PUBLICATION mypub;`

Logical Replication in Action

Cluster 1	Cluster 2
	`SELECT * FROM test;` `1`
`INSERT INTO test VALUES (2);`	

	`SELECT * FROM test;`
	`1`
	`2`

Benefits of Logical Replication

Logical replication allows:

- table-level granularity
- replication from multiple clusters to a single cluster (aggregation)
- replication of a single table to multiple clusters (broadcasting)
- replication and upgrades between major Postgres versions
- creation of local objects on subscribers, e.g. tables, indexes

2. Partitioning Syntax

```
CREATE TABLE numbers (x INTEGER) PARTITION BY RANGE (x);
CREATE TABLE negatives PARTITION OF numbers FOR VALUES FROM (UNBOUNDED) TO (0);
CREATE TABLE positives PARTITION OF numbers FOR VALUES FROM (0) TO (UNBOUNDED);
```

Partition Table Structure

```
\d+ numbers
                                    Table "public.numbers"
 Column |  Type   | Collation | Nullable | Default | …
--------+---------+-----------+----------+---------+…
 x      | integer |           | not null |         |…
Partition key: RANGE (x)
Partitions: negatives FOR VALUES FROM (UNBOUNDED) TO (0),
            positives FOR VALUES FROM (0) TO (UNBOUNDED)

\d negatives
             Table "public.negatives"
 Column |  Type   | Collation | Nullable | Default
--------+---------+-----------+----------+---------
 x      | integer |           | not null |
Partition of: numbers FOR VALUES FROM (UNBOUNDED) TO (0)

\d positives
             Table "public.positives"
 Column |  Type   | Collation | Nullable | Default
--------+---------+-----------+----------+---------
 x      | integer |           | not null |
Partition of: numbers FOR VALUES FROM (0) TO (UNBOUNDED)
```

Tuple Routing

```
INSERT INTO numbers VALUES (-4), (-1), (7), (12);

SELECT * FROM numbers;
 x
----
 -4
 -1
  7
 12

SELECT * FROM negatives;
 x
----
 -4
 -1

SELECT * FROM positives;
 x
----
  7
 12
```

Partitioning Benefits and Limitations

Partitioning does:

- Create proper child constraints
- Route parent INSERTs into child tables

Partitioning does not yet:

- Hash partitioning
- Create child tables for values not already covered (it errors instead)
- Move updated rows that no longer match the partition constraints (it errors instead)
- Prune child tables faster than PG 9.6
- Perform executor-stage partition pruning
- Perform parallel partition processing

3. Crash-Safe, Faster, and Replicated Hash Indexes

Hash indexes is now a first-class feature:

- Crash safe
- Replicated
- Reduced locking during bucket splits
- Faster lookups
- More even index growth
- Single-page pruning

4. ICU Library

- Uses ICU library instead of OS-supplied internationalization library
- Allows detection of collation changes that can affect index ordering
- Enabled via `configure --with-icu`

5. Quorum Commit

- `synchronous_standby_names = FIRST 1 (s1, s2)` continues when the first active standby replies (pre-10 behavior)
- Now `synchronous_standby_names = ANY 1 (s1, s2)` continues when the any server from the list replies
- `synchronous_standby_names = ANY 2 (s1, s2, s3)` is also possible
- Called quorum commit

6. Progress on Parallelism

Parallelism is now supported in:

- Btree index scans
- Bitmap heap scans
- Merge joins
- Procedural languages

7. Multi-Column Statistics

- Previously, `WHERE a=1 AND b=1` multiplied the probabilities of the two columns, assuming they were unrelated
- Now `CREATE STATISTICS … WITH (dependencies)` records multi-column correlation
- The correlation is used when combining single-column probabilities

8. pg_stat_activity Improvements

- Additional wait tracking
  - client reads, writes
  - server reads, writes, fsyncs
  - synchronous replication
- Additional process display
  - auxiliary processes
  - worker processes
  - WAL senders

pg_stat_activity Example

```
SELECT wait_event_type, wait_event, count(*)
FROM pg_stat_activity
WHERE backend_type = 'client backend'
GROUP BY wait_event_type, wait_event
ORDER BY 1, 2;
```

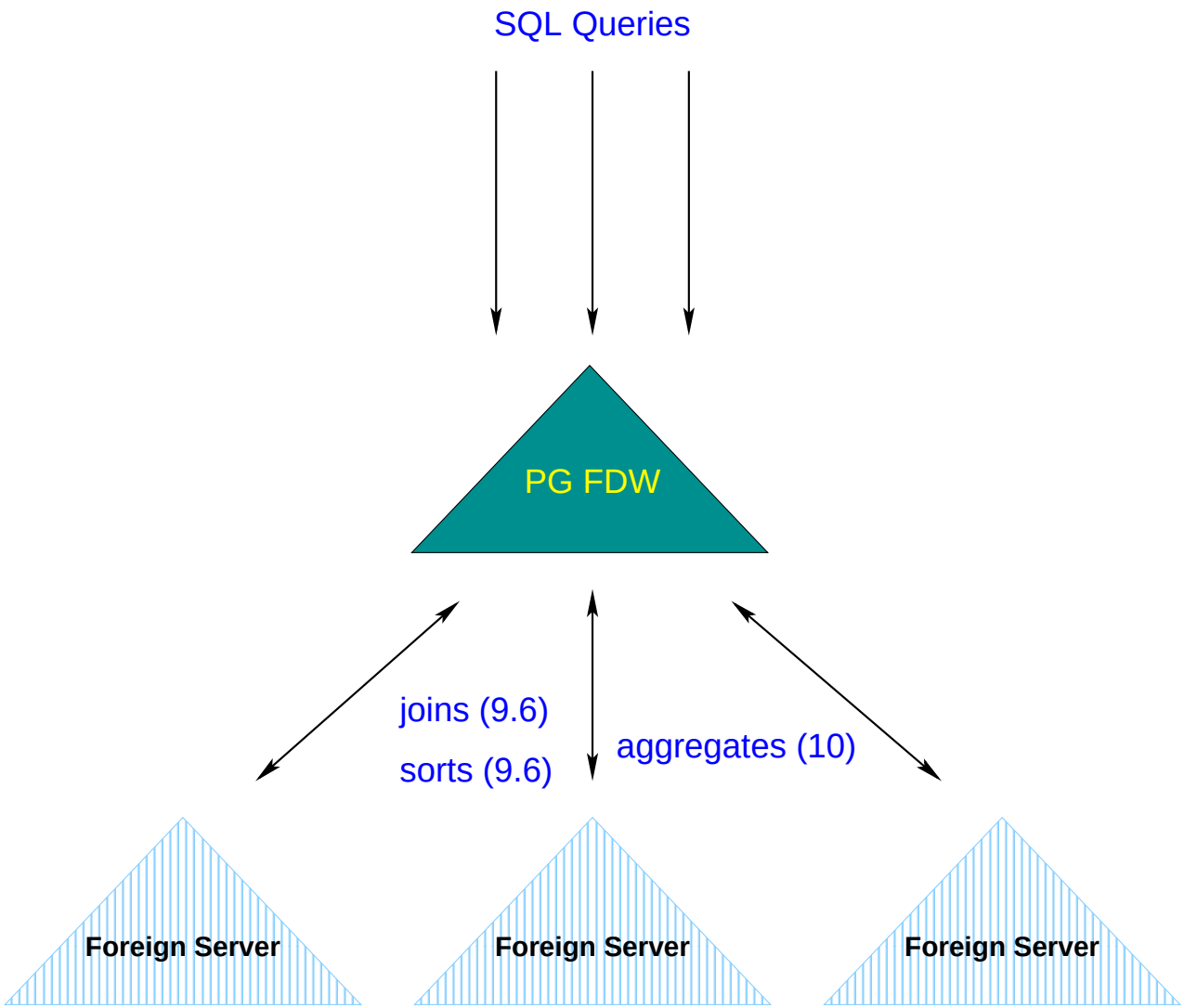
```
 wait_event_type |  wait_event    | count
-----------------+----------------+-------
 Client          | ClientRead     |     2
 IO              | DataFileWrite  |     1
 Lock            | transactionid  |     3
 LWLock          | WALWriteLock   |    19
                 |                |     8
```

9. SCRAM-SHA-256 Authentication

SCRAM-SHA-256 provides a more secure password authentication method than MD5:

- Make packet replay more difficult (MD5 has a 50% probability of repeating after 64k connections)
- Make stolen hashed password reuse more difficult
- Make brute-force attacks more difficult

10. FDW Aggregate Pushdown



11. More

- Restrictive row-level security policies, provides AND/required policies
- AFTER trigger transition tables
- Full text search support for JSON and JSONB
- Default permissions on schemas
- Multiple libpq-specified host names, plus write-mode filter

Pending Postgres 11 Features

- Procedures with transaction control
- JIT compilation of the executor
- Parallel query improvements, e.g. parallel CREATE INDEX
- Partitioning
  - hash partitioning
  - partition-wise join, benefits joins between identically-partitioned tables
  - executor-stage partitioning, benefits join pruning and prepared queries
  - partition key updates move affected rows to their new partition
  - default partitioning are committed
  - foreign keys pointing to a partitioned table

Current as of 2018-01-17

Conclusion

http://momjian.us/presentations

https://www.flickr.com/photos/thevlue/