PostgreSQL users

+BIG RUSSIAN Enterprise!
PostgreSQL is #4!

<table>
<thead>
<tr>
<th>Rank</th>
<th>DBMS</th>
<th>Database Model</th>
<th>Score Sep 2017</th>
<th>Score Aug 2017</th>
<th>Score Sep 2016</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Oracle</td>
<td>Relational DBMS</td>
<td>1359.09</td>
<td>-8.78</td>
<td>-66.47</td>
</tr>
<tr>
<td>2.</td>
<td>MySQL</td>
<td>Relational DBMS</td>
<td>1312.61</td>
<td>-27.69</td>
<td>-41.41</td>
</tr>
<tr>
<td>3.</td>
<td>Microsoft SQL Server</td>
<td>Relational DBMS</td>
<td>1212.54</td>
<td>-12.93</td>
<td>+0.99</td>
</tr>
<tr>
<td>4.</td>
<td>PostgreSQL</td>
<td>Relational DBMS</td>
<td>372.36</td>
<td>+2.60</td>
<td>+56.01</td>
</tr>
<tr>
<td>5.</td>
<td>MongoDB</td>
<td>Document store</td>
<td>332.73</td>
<td>+2.24</td>
<td>+16.74</td>
</tr>
<tr>
<td>6.</td>
<td>DB2</td>
<td>Relational DBMS</td>
<td>198.34</td>
<td>+0.87</td>
<td>+17.15</td>
</tr>
<tr>
<td>7.</td>
<td>Microsoft Access</td>
<td>Relational DBMS</td>
<td>128.81</td>
<td>+1.78</td>
<td>+5.50</td>
</tr>
<tr>
<td>8.</td>
<td>Cassandra</td>
<td>Wide column store</td>
<td>126.20</td>
<td>-0.52</td>
<td>-4.29</td>
</tr>
<tr>
<td>9.</td>
<td>Redis</td>
<td>Key-value store</td>
<td>120.41</td>
<td>-1.49</td>
<td>+12.61</td>
</tr>
<tr>
<td>10.</td>
<td>Elasticsearch</td>
<td>Search engine</td>
<td>120.00</td>
<td>+2.35</td>
<td>+23.52</td>
</tr>
</tbody>
</table>

334 systems in ranking, September 2017

https://db-engines.com/en/ranking
PostgreSQL 10

- New version numbering
- DBA visible changes
- Logical Replication
- Native Table Partitioning
- Improved Query Parallelism
- Performance improvement
- Quorum Commit for Sync Replication
- Assorted improvements
New version numbering

• Postgres version numbering was always weird
  • Check https://www.postgresql.org/docs/10/static/release.html
    0.01, 0.02, 0.03, 1.0, 1.01,..,1.02.1, 1.09 - Postgres95
    6.0,..,6.5.3
    7.0,..,7.4.30
    8.0,..,8.4.22
    9.0,...9.6.5
  • Generally version looks like major1.major2.minor1, difficult to decide which major number to advance
    • 6.0 — PostgreSQL, Postgres95 was known as Postgres Release 5
      commit 9b41da6ce48e3bed6730f9a6347a5461175c4f83
      Author: Bruce Momjian <bruce@momjian.us>
      Date:   Wed Dec 11 00:28:15 1996 +0000
        Rename postgres95 to PostgreSQL. Add comment for SELECT NULL
    • 7.0 — really usable server (FK, SQL 92 JOIN, better optimizer)
    • 8.0 - Microsoft Windows Native Server
    • 9.0 — Built-in replication

• Now version numbering is simple: major.minor
• Expect 10.0 release Sep 25, 2017
• Next major release will be 11.0
• Fool-tolerance
  • Directories pg_xlog to pg_wal, pg_clog to pg_xact, all functions with reference to „xlog“
  • log_directory (for log files) from pg_log to log
• HASH indexes must be rebuilt after pg_upgrade
• ICU library (--with-icu, ICU4C needed), stable collation support
• Replication in pg_hba.conf
  • Allow replication connections from localhost by a user with the replication privilige (^^Gitlab)
• wal_level = replica - support pg_basebackup new default to include required WALs
  • max_wal_senders = 10, max_replication_slots = 10
  • wal_level can be on of {minimal, replica (replaced archive and hot_standby), or logical}
DBA visible changes 2/2

- password_encryption is md5 (on, default)
  - #password_encryption = md5, scram-sha-256
  - no plain, no UNENCRYPTED option in CREATE/ALTER USER, --unencrypted option removed from createuser command

- + ssl_dh_params_file (Diffie-Hellman parameters)

- - {create,drop}lang (create/drop extensions) contrib/tsearch2

- + idle_in_transaction_session_timeout = 0 # in milliseconds, 0 is disabled

- - sql_inheritance = on

- Better commenting importance of fsync = on (eat my data off)
  - # flush data to disk for crash safe (turning this off can cause unrecoverable data corruption)
Logical Replication

- 9.4: logical decoding of WAL records
- 10.0: functionality (some) pglogical extension was ported to the core, added SQL interface
- Big step to multimaster

- Partial replication (individual objects)
  - One-to-many, many-to-one
- Replication between different versions (starting from 10.0), different platforms (Linux to Windows)
- Upgrades (with minimal or no downtime)
- Write operations on secondary servers are possible
Logical Replication

• SQL interface:

ON MASTER: wal_level = logical

CREATE/ALTER/DROP PUBLICATION name
[ FOR TABLE [ ONLY ] table_name [ * ] [, ...]
[ FOR ALL TABLES ]
[ WITH ( publication_parameter [= value] [, ... ] ) ]

WITH (publish = 'insert, delete')

ON SECONDARY:

CREATE/ALTER/DROP SUBSCRIPTION subscription_name
CONNECTION 'conninfo'
  PUBLICATION publication_name [, ...]
  [ WITH ( subscription_parameter [= value] [, ... ] ) ]

WITH ( copy_data = false )
Logical Replication

• Limitations in 10.0
  • does not replicate schema/DDL
  • does not replicate sequences
  • does not replicate TRUNCATE
  • only supports replicating base (normal) table to base table
• Wait for the next releases!
Table partitioning

- Before 10.0: table inheritance + constraint exclusion
- Manual setup, slow for partitions pruning

- 10.0: still table inheritance + metadata
- Declarative syntax, still slow for partition pruning
- But, metadata makes possible to improve planner in future releases!

- pg_pathman for really fast partitioning
  - It doesn’t use table inheritance
  - It demonstrates how fast could be native partitioning - orders of magnitude faster (for 500 partitions)
  - https://github.com/postgrespro/pg_pathman
Table Partitioning

- Declarative Partitioning provides SQL syntax for:
  - range and list partitioning, Multi-level partitioning
  - Attach/detach partitions, creating partitions as foreign tables
  - Fast tuple routing

By range:

```
CREATE TABLE t1(created timestamp)
    PARTITION BY RANGE(EXTRACT(YEAR FROM created));
CREATE TABLE t1_2017
    PARTITION OF t1 FOR VALUES FROM (2017) TO (2018);
```

By list:

```
CREATE TABLE t2(category text)
    PARTITION BY LIST(category);
CREATE TABLE t2_books
    PARTITION OF t2 FOR VALUES IN ('books');
```
Table Partitioning

- Limitations:
  - Need to manually create indexes on partitioned tables
  - No automatic creation of partitions
  - No routing tuples to foreign partitions
  - No splitting or merging partitions

Future improvements:
- Add hash partitioning
- Global index
- Sharding
pg_shardman: sharding via pg_pathman, postgres_fdw and logical replication

3 nodes, 2 shards, one replica per shard
Improved Query Parallelism

- **9.3: Infrastructure**
  - background workers

- **9.6: Feature introduction**
  - Parallel sequential scans
  - Hash joins
  - Nested Loops

- **10.0: Improvement**
  - Bitmap heap scans
  - Index scans
  - Merge joins
  - Subqueries

- **>10.0: Even better**
  - Create index
  - Parallel Append
Performance improvements

- Faster expression evaluations in executor
  - Currently benefit is about 6-20%
  - But it made possible future JIT-ing (expected several times improvements)
Quorum Commit for Sync Replication

9.6: primary wait for commit confirmation from N of M
- Priority set of N nodes with M standbys (order of standbys is important)
- GUC variable synchronous_standby_names
- synchronous_standby_names = N(standby_1,…,standby_M)

10.0: Quorum Commit
- Quorum set of N nodes (order of standbys is not important)
- synchronous_standby_names = ANY N(standby_1,…,standby_M)
- Use FIRST instead of ANY to emulate 9.6 — this is default
Assorted Improvements

- XMLTABLE (better standard, infrastructure for json_table)
- Durable HASH indexes
- FDW aggregate pushdown
- Transaction traceability
  - `txid_status(BIGINT)` — useful to recover from indeterminate COMMIT.
  - https://blog.2ndquadrant.com/traceable-commit-postgresql-10/
- `pg_stat_activity`
  - More wait events: client reads/writes, server reads/writes and fsync ops, synchronous replication
  - Worker processes, WAL senders and more
- Extended Statistics - Functional Dependencies, Multivariate N-Distinct Counts
  - `CREATE STATISTICS stname (dependencies, ndistinct) ON col1, col2,... FROM tabname;`
- Security Technical Implementation Guide (STIG DoD), 1st OSS database
- RLS (permissive +restrictive) — policies can be AND-ed
- Better (than md5) authentication - SCRAM-SHA-256
- FTS for JSONB (https://obartunov.livejournal.com/194683.html)
- +many (>100) features
PostgreSQL Future

Several Postgres groups are working on

- Postgres Distributed
- Postgres Vectorized
- Postgres Parallel
- Postgres Asynchronous
- Postgres Extendable+

Postgres NoSQL — check SQL/JSON
http://sqlfiddle.postgrespro.ru:6081/#!21/

Postgres Scalable (Vertical & Horizontal)
Conclusions

- PostgreSQL is the universal database with clear roadmap
- Proven technology of developing major features
- Postgres 10 is a big step in product evolution
תודה רבה על תשומת לבכם.
JSON Roadmap

- 8.2 hstore extension
- 9.2 JSON datatype
- 9.4 JSONB datatype, indexes
- 10 FTS for JSON(B)
- >10 SQL/JSON standard
- >10 JSONB compression
References

- Documentation: Release Notes for version 10
- Postgres Wiki: New in Postgres 10
- Bruce Momjian: Major Features: Postgres 10
- Robert Haas: New Features Coming in PostgreSQL 10
- Michael Paquier: Postgres 10 highlight … series
- Hubert (depesz) Lubaczewski: Waiting for PostgreSQL 10 … series
- Robert Haas: Parallel Query v2
- Robert Haas: Partitioning plans for v11
- Simon Riggs: News and Roadmap for BDR
- Petr Jelinek: Logical Replication in PostgreSQL 10