What is the Future of PostgreSQL?

- Robert Haas

PostgreSQL Popularity By The Numbers

Date	Rating	Increase vs. Prior Year	% Increase
January 2016	282.401	+27.913	+11%
January 2017	330.37	+47.969	+16%
January 2018	386.183	+55.813	+16%
January 2019	466.114	+79.931	+20%

Source: db-engines.com

- PostgreSQL is the #4 system, still far behind Oracle, MySQL, and SQL Server – but gaining quickly.
- Overall, from January 2016 – January 2019, PostgreSQL's rating increased by 65%; the ratings of all three of the top systems *decreased* by 10 – 16% during the same period.

PostgreSQL Popularity Qualitatively

- AWS
  - Platinum Sponsor, PGCONF.IN 2019
  - Platinum Sponsor, PGCONF.ASIA 2018
  - Gold Sponsor, PostgresOpen SV 2018
  - Acquired OpenSCG, March 2018
- Microsoft
  - Platinum Sponsor, PGCONF.IN 2019
  - Platinum Sponsor, PGCONF.ASIA 2018
  - Diamond Sponsor, PostgresOpen SV 2018
  - Acquired Citus Data, January 2019
- Fujitsu
  - Platinum Sponsor, PGCONF.ASIA 2018
  - Platinum Sponsor, PGDay Down Under 2018

What Is Driving This Popularity?

- There are some non-technical factors, such as:
  - PostgreSQL's reputation for quality,
  - a permissive license,
  - a healthy community not controlled by any single company, and
  - Oracle's acquisition of MySQL.

- However, a big part of the reason for the gains PostgreSQL has made has to do with new features.

What Is Driving This Popularity?

- There are some non-technical factors, such as:
  - PostgreSQL's reputation for quality,
  - a permissive license,
  - a healthy community not controlled by any single company, and
  - Oracle's acquisition of MySQL.

- However, a big part of the reason for the gains PostgreSQL has made has to do with new features.

Major New Features: v9.0 – v9.5

- *Streaming Replication*. Standby lag measured in seconds or milliseconds, instead of minutes.
- *Hot Standby*. Run queries on the standby, instead of using it only for HA.
- *Foreign Tables*. Access heterogeneous data.
- *JSONB*. Store and index unstructured data.
- *INSERT .. ON CONFLICT UPDATE*. Also known as UPSERT.

- Many of these features have been through multiple iterations, adding new capabilities and fixing shortcomings.

Major New Features: v9.6 – v11

- *Logical Replication*. An amazing new feature! Allows for partial replication and cross-version replication, and doesn't replicate database bloat or cause query cancellations.
- *Parallelism*. For the first time, PostgreSQL can use more than one CPU for the same task.
- *Partitioning*. Easier to set up than table inheritance, and much faster, especially in v11.
- *Stored Procedures*. Lets you put more logic in the database and more easily migrate code from other systems.
- *Just-in-Time Compilation*. Speeds up analytic queries, especially those with complex expressions.

Caveat

- These are just some of the major new features.
- There are another 10+ major features listed in the release notes just for v9.6 - v11.
- These are my picks, but somebody else might have a different list.
- I picked these five by looking at –
  - what the feature does today, and
  - where it could go in the future.

Refinement Needed!

- Even though these are already great features, there is room for further improvement.
- Other database products will continue to innovate, and in order to be successful, PostgreSQL must continue to innovate as well – and preferably, to an even greater degree.
- “Good” is a moving target.
- So what’s next?

Major New Features: v9.6 - v11

- *Logical Replication*. An amazing new feature! Allows for partial replication and cross-version replication, and doesn't bloat or cause query cancellations.
  - Doesn't handle failover to standbys.
  - Difficult to avoid downtime when performing major version upgrades.
  - Multi-master replication requires third-party tools.
- *Parallelism*. For the first time, PostgreSQL can use more than one CPU for the same task.
  - Some simple queries run much faster, but more complex queries often see little or no benefit.
  - The only maintenance command supported in current releases is CREATE INDEX – for btree indexes only.

Major New Features: v9.6 - v11

- *Partitioning*. Easier to set up than table inheritance, and much faster, especially in v11.
  - Still can easily be slower than no partitioning.
  - Some SQL features still don't work on a partitioned table the way they do on a plain table (e.g. can't be referenced by a foreign key).
- *Stored Procedures*. Lets you put more logic in the database and more easily migrate code from other systems.
  - Still missing some related features that are present in other systems, e.g. schema variables.
- *Just-in-Time Compilation*. Speeds up analytic queries, especially those with complex expressions.
  - Could be applied more widely and optimized better.

Work is in Progress!

- Improvements in most of these areas – and many others which I didn't have time to mention – are already in progress.
- It's too early to say exactly how these developments will affect PostgreSQL 12, 13, and beyond, but expect progress on many of them.

New Development Drivers

- *Large-Scale Analytics*. PostgreSQL's core strength is often considered to be OLTP, but many users turn to PostgreSQL for analytic workloads, and data sets are growing.
- *Cloud*. The traditional way to deploy software – including PostgreSQL – is on a dedicated server in the data center, but users increasingly want to deploy virtual machines, containers, or in the cloud.
- *Database Migration*. The PostgreSQL community has sometimes been skeptical of compatibility features, but there is new emphasis on helping users move away from proprietary database engines.

Large-Scale Analytics: It All Takes Longer

- *Slower Queries*. Parallel query and just-in-time compilation help, but they are not sufficient.
- *Slower Maintenance Operations*. B-tree index creation runs 2-3x faster with parallelism, but other maintenance operations such as backup can't use parallelism yet.
- *Slower Backup and Restore*. Copying a lot of data with pg_dump takes a very long time. Even with hot backup, it's still slow if your database is big enough.

Cloud: Non-Stop Chaos

- *Unattended Operation*. There is no DBA.
- *Flash Mobs*. Load can increase and decrease very quickly, and may exceed what a single server can bear.
- *Zero Downtime*. Even at night, it's daytime somewhere.
- *Global Reach*. Must be able to run transactions quickly from both Tokyo and New York.

Database Migration: Old Idea, New Trend

- There are still some good reasons to run Oracle.
- But for a large and growing percentage of use cases, PostgreSQL is an option.
- Big companies like Amazon and Microsoft are getting involved.

Current Work: Pluggable Storage

- Hope to see pluggable storage in PostgreSQL 12.
- Allows for innovation at the storage layer.
  - Try new things without breaking the existing heap.
  - Add special-purpose storage formats such as a WORM (write-once read-many), columnar, in-memory, non-transactional.
- Hope to see first version of zheap in PostgreSQL 13.
  - In-place update with undo means that bloat goes away automatically in most cases.
  - So, better for unattended operation + large data sets.
  - Smaller on disk and fewer writes, too.

Current Work: Sharding/Clustering

- Two-phase commit for FDWs.
  - If a transaction uses FDWs, and if it makes data changes on multiple nodes, we should ensure that the transaction commits on all nodes or rolls back on all nodes.
- Cluster-wide MVCC.
  - In fact, we would like it to appear as though the commit or rollback happens simultaneously on all nodes.
- Asynchronous execution.
  - If we can't make any more progress on one part of the query plan because we have to wait for some reason, do something else meanwhile.
  - For example, if the plan involves query multiple remote servers via the foreign data wrapper interface, it would be *very* useful to be able to send all the queries at once and wait for whichever one responds first.

Current Work: Compatibility

- MERGE.
  - Allows INSERT, UPDATE, and DELETE to be rolled into a single, SQL-standard command.
- Schema variables.
  - One function or procedure can store data, and another function or procedure running in the same session can later read it.

Future: Cluster Management

- Graceful failover without risk of transaction loss.
- Easy cluster reconfiguration, including automatically demoting a master to a standby.
- Automatic topology discovery.
- Built-in load-balancing.

Future: Processes → Threads

- Scalability to thousands of connections, especially when most are idle.
- Better and faster parallel query.
- Better for some procedural languages – avoids a separate interpreter per backend.

Commercialization

- Microsoft Azure
- Amazon Aurora with PostgreSQL Compatibility
- EDB Postgres Advanced Server
- Greenplum
- CitusDB
- Many others...

Internationalization

- To be really successful, PostgreSQL needs to affect the best developers from all over the world.
- PostgreSQL has always had an international community, but it's now stronger than ever with the latest round of committer promotions:
  - Alexander Korotkov (Russia)
  - Amit Kapila (India)
  - Tomas Vondra (Czech Republic)
  - Michael Paquier (Japan)
  - Thomas Munro (New Zealand)
  - Peter Geoghegan (US)
  - Etsuro Fujita (Japan)
- PostgreSQL also needs non-committer developers, translators, user group organizers, and press contacts for every part of the world. It will not be better than you help make it.

Thanks

- Please continue to use, support, and contribute to PostgreSQL!