Migrating to PostgreSQL
Hello!

At Sharper Informatics Solutions, we believe that smart informatics can drive good decision-making and open doors to new opportunities. We specialize in creating custom data-driven systems for clients in energy efficiency, data science and engineering.

Jennifer Scheuerell

CTO and Database Software Chameleon at Sharper Informatics Solutions
Comparing RDBMS

Key factors

- Database administration overhead
- Development expense
- Software cost
- Performance
- Integration
- Tools
Why choose PostgreSQL

- Security
- Performance
- Reliability
- Integration Tools
- Automation
- Cost
Migrating to PostgreSQL

Advanced Features / Reliability / Performance

Expense

PostgreSQL

MySQL

SQLite

ORACLE

Microsoft SQL Server
Increasingly popular

PostgreSQL is the DBMS of the Year 2020
by Paul Andlinger, Matthias Gelbmann, 4 January 2021
Tags: DBMS of the year, Microsoft Azure SQL Database, MongoDB, PostgreSQL

PostgreSQL is the database management system that gained more popularity in our DB-Engines Ranking within the last year than any of the other 360 monitored systems.
We thus declare PostgreSQL as the DBMS of the Year 2020.

For determining the DBMS of the year, we subtracted the popularity scores of January 2020 from the latest scores of January 2021. We use the difference of these numbers, rather than a percentage, because that would favor systems with a tiny popularity at the beginning of the year. The result is a list of DBMSs sorted by how much they managed to increase their popularity in 2020, or in other words, how many additional people started to communicate about it in one of the ways we measure in our methodology, for example job offers, professional profile entries and citations on the web.

DBMS of the Year: PostgreSQL

PostgreSQL already won our DBMS of the Year award in 2017 and 2018, and now becomes the first system to win this title three times.
Continuous Migration

The migration method proposed here is called Continuous Migration. Continuous Migration makes it easy to make incremental progress over a period of time, and also to pause and resume the migration work later on, should you need to do that. The method is pretty simple — just follow those steps:

1. Setup your target PostgreSQL architecture
2. Fork a Continuous Integration environment that uses PostgreSQL
3. Migrate the data over and over again every night, from your current production RDBMS
4. As soon as the CI is all green using PostgreSQL, schedule the D-day
5. Migrate without any surprises... and enjoy!

This method makes it possible to break down a huge migration effort into smaller chunks, and also to pause and resume the project if need be. It also ensures that your migration process is well understood and handled by your team, drastically limiting the number of surprises you may otherwise encounter on migration D-day.
Case Studies

CEDARS
Advice Letters
Energy Data Web

Migrating to PostgreSQL

01
Data problem
What motivated the migration? How was it approached?

02
Migration
Handy tools and techniques

03
Lessons
What we’ve learned
CEDARS securely manages California Energy Efficiency Program data reported to the Commission by Investor Owned Utilities, Regional Energy Networks (REns), and certain Community Choice Aggregators (CCAs.)
Advice Letters

CPUC Advice Letters database for Energy Utilities
Energy Data Web

CPUC public-facing document and project sites
Migrating to PostgreSQL

Generic Migration Steps

1. Agree to undertake the migration
2. Set aside time and budget
3. Plan new servers
4. Build out new servers
5. Set up and start continuous migration
6. Set up backups, automate recovery, test
7. Schedule cut over
8. Run final migration
9. Retire old servers
Deciding to migrate

Do you wait for "the incident" or migrate before it happens?
Migrating to PostgreSQL

Dedicating Resources

- Build new system
- Maintain old system
- Maintain new system

(time →)

(↑ $↑)
Migrating to PostgreSQL

Generic Migration Steps

1. Agree to undertake the migration
2. Set aside time and budget
3. Plan new servers
4. Build out new servers
5. Set up and run continuous migration
6. Set up backups, script recovery, test
7. Schedule cut over
8. Run final migration
9. Retire old servers
Server expense

Is Oracle DB expensive?

With one of its products licensed at $300,000 per Oracle Processor. Oracle Business Intelligence Suite Foundation, licensing a relatively small 4-processor quad-core server would cost $2.4 million (not including support and maintenance), making it one of the world's most expensive software products to license. Jan 31, 2017

https://www.linkedin.com › pulse › oracle-most-expensive...

Oracle, the most expensive software on the planet? - LinkedIn
Server expense

We run three PostgreSQL servers on Linux for less cost than one MS SQL Server database on Windows using the same hardware.
Reliability

We saw significant site performance degradation over time on SQLite.

The Windows MS SQL Server crashed and took five days to restore.
Migrating to PostgreSQL

Integration Tools

Foreign data wrapper

```
CREATE EXTENSION IF NOT EXISTS postgres_fdw;

CREATE SERVER etrm FOREIGN DATA WRAPPER postgres_fdw OPTIONS (  
  port '5432',  
  servername 'publicdb.caetrm.com'  
);

CREATE USER MAPPING FOR postgres
  SERVER etrm
  OPTIONS (username 'readonly', password '3PIAFrEdKV');

CREATE FOREIGN TABLE etrm_permutations (  
  MeasDetailID text
)  
SERVER etrm
OPTIONS(query  
  'SELECT distinct "MeasDetailID" as measdetailid  
    FROM public.permutations');

create materialized view main_measdetailid as
  SELECT measdetailid  
  FROM etrm_permutations;
```
Performance

Overall better performance

Materialized views
Migrating to PostgreSQL

Powerful tool

https://pgloader.io/
https://github.com/dimitri/pgloader

pgloader loads data into PostgreSQL and allows you to implement Continuous Migration from your current database to PostgreSQL.

$ apt-get install pgloader
Why choose PostgreSQL

- Security
- Performance
- Reliability
- Integration Tools
- Automation
- Cost
Migrating to PostgreSQL

Hurdles for management

1. Agree to vision; dedicate people and time to the effort
2. Maintain current data processing while new tools are built.
3. Schedule and execute switch over
Conclusion

PostgreSQL for the win

- Happy clients
- Lighter database administration overhead
- Increased reliability
- Better performance
- Powerful integrations
- Much lower cost
THANK YOU!