PostgreSQL

Who, What, When, Where, Why, How?

QUIS?

Who's involved with PostgreSQL?

- Core team: https://www.postgresql.org/community/contributors/
- Large Users: https://www.postgresql.org/about/users/
- Case Studies: https://www.postgresql.org/about/casestudies/
- Open Source Edition: https://www.postgresql.org/
- Commercial/Supported Editions
  - EnterpriseDB Postgres (and EDB PgSQL Advanced Server)
  - 2ndQPostgres
  - Crunchy PostgreSQL
  - Postgres by BigSQL
  - Postgres PRO Standard & Enterprise
  - And more: https://www.postgresql.org/download/products/8-postgresql-derived-servers/

QUID?

What is this stuff?

- PostgreSQL started as the Postgres, the successor to Ingres (it's the "Post"-Ingres database, ha ha ha)
- Originally not SQL, then added POSTSQL, then finally replaced with new ANSI-SQL engine
- Renamed to PostgreSQL at v6.0 (Commonly abbreviated "PgSQL")
- Object-relational, hybrid operation is a built-in feature
- MVCC, PIT recovery, async replication, nested transactions, hot backups, WAL, i18n charsets, l10n collations, full UNICODE, GIS, FTS.
- Multi-petabyte scalability (using clusters, otherwise multi-terabyte)

QUANDO?

When should you use or switch to PgSQL?

- MySQL database workload is no longer embarrassingly read-only or write-only
- Database engine needs to fit into less memory
- Existing database query optimizer isn't sophisticated enough to handle increasingly-complex reporting
- GIS functions require expensive add-on
- User licensing requirements require expensive upgrade
- Can't distribute GPL source code with your closed-source product
- New projects: why would you use anything else?

UBI?

Where to use PgSQL?

- Embedded systems - PgSQL uses the OS buffer cache instead of reserving its own memory, and can provide nearly-deterministic performance.
- Application-backing database - PgSQL requires zero maintenance out of the box. Nothing grows without bound unexpectedly and nondeterministically. (Lookin' at you, `IBDATA1.LOG`...) Defaults are sensible for a wide range of applications. Only local UNIX socket connections are enabled by default.
- Large databases - the query optimizer is extremely intelligent.
- Hybrid systems - PgSQL is natively both an object database and an RDBMS.
- GIS systems - PostGIS rivals or betters all its commercial competitors.
- ORACLE replacement - EnterpriseDB Advanced Server is a compile-time replacement for ORACLE RDBMS.

CUR?

Why should I use PgSQL?

- You need an RDBMS. Period. That's good enough, really.
- You need an open-source RDBMS
- You need a non-GPL RDBMS
- You need a mostly-<whatever>-compatible RDBMS
- You need a ANSI-SQL:2008 conforming RDBMS
- You need a highly-extensible RDBMS
- You need a highly-scalable RDBMS

QUEM AD MODUM?

OK, now what?

- PostgreSQL is available in the package repository of - as far as I know - every Linux and *BSD distribution.
  - `[apt-get|yum|dnf] install postgresql`, or some variant on that, will install a reasonably-recent version of PostgreSQL
  - PostgreSQL project maintains dpkg and yum repositories for every supported version.
- Download either binary installers or tarballs/zipfiles for macOS, Windows, and other UNIXes.
- Consider a commercially-supported version if you expect to move into to production with it.
- If migrating from another database, just use a commercial distribution with support for source-DB-specific migration tools.