Immunity to over-deployment

Over-deployment is what some proprietary database vendors regard as their #1 licence compliance problem. With PostgreSQL, no-one can sue you for breaking licensing agreements, as there is no associated licensing cost for the software.

Better support than the proprietary vendors

In addition to our strong support offerings, we have a vibrant community of PostgreSQL professionals and enthusiasts that your staff can draw upon and contribute to.

Extensible

The source code is available to all at no charge. If your staff members have a need to customise or extend PostgreSQL in any way then they are able to do so with a minimum of effort, and with no attached costs. This is complemented by the community of PostgreSQL professionals and enthusiasts around the globe that also actively extend PostgreSQL on a daily basis.

Designed for high volume environments

We use a multiple row data storage strategy called MVCC to make PostgreSQL extremely responsive in high volume environments. The leading proprietary database vendor uses this technology as well, for the same reasons.

Installation

- PostgreSQL runs on all modern Unix-like operating systems and on NT based Windows Systems.
- PostgreSQL uses standard configure and make options for building from source.
- Binary packages are available for most distributions and platforms.
- PostgreSQL can be installed and operated without root permissions.
- PostgreSQL is included in most popular Linux and BSD distributions.

Query Language Features

- **Foreign keys:** Supports standard CREATE TABLE ... FOREIGN KEY syntax for referential integrity. Supports different actions for updating and deleting, including cascading, restricting, and restoring to default or NULL values. These features are critical for data integrity in many applications. Foreign keys can be detected by database modeling tools, such as Computer Associates' ERwin, to ease database design and documentation.
- **Joins:** Implements all SQL99 join types: inner join, left, right, full outer join, natural join. Optimizer has several join algorithms available.
- **Views:** Views are stored SQL SELECT statements. Views can be used to encapsulate complex queries at the server level and to implement sophisticated access privilege granularity. This simplifies database design and maintenance. PostgreSQL also supports updatable views (writes to the view are pushed through to the underlying base tables).
- **Triggers:** Triggers are procedures that are called by the database on defined actions, in particular inserts or updates to particular tables. Typical applications include logging every time a record is inserted or deleted to or from a table, or to update a field every time another field has been changed. More generally, triggers can be used to implement side effects of any kind.
- ANSI SQL compliant.
- Additional types are available for geometric constructs (points, lines, etc.), TCP/IP network addresses, Ethernet card IDs, ISBN/ISSN, and more.
- New types can be defined, along with necessary supporting functions and operators.
- PostgreSQL supports storage of binary large objects, including pictures, sounds, or video. These objects can be retrieved in whole or part by client applications.
- Support for international character sets, multibyte character encodings, Unicode.
- Locale-aware for sorting, case-sensitivity, formatting.
- Support for standard SQL conditions, such as CASE ... WHEN, COALESCE, and NULLIF.
- Subqueries are nested queries that allow complex questions to be answered entirely through the database. Using subqueries can simplify and speed up database applications.
- Support for SELECT DISTINCT and SELECT DISTINCT ON (column), to show only unique rows, and matching data for unique values.
- Full support for GROUP BY and aggregate (domain) functions, including COUNT, SUM, AVG, MIN, MAX, STDDEV, and VARIANCE. New aggregate functions can be created in C or any procedural language PostgreSQL supports. Sub-selects in FROM clause, such as SELECT COUNT(x), AVG(x), SUM(x) FROM (SELECT SUM(weight) AS x FROM Items GROUP BY manufacturer) AS Items;
- Support for UNION and UNION ALL, INTERSECT and EXCEPT.
- Extensions for LIMIT and OFFSET to allow for the selection of any arbitrary number of records, e.g., SELECT * FROM Items ORDER BY cost LIMIT 5;
- Comprehensive library of functions and operators:
- Hundreds of built-in functions, including rich support for mathematical, date/time, and string manipulation.
- Support functions for full-text indexing
- ODBC functions, Oracle compatibility functions
- Standard SQL LIKE matching and case-insensitive LIKE matching and full POSIX regular expression matching and case-insensitive regular expression matching
- Loadable extension functions: soundex, cryptographic hash functions (e.g, SHA1 and MD5), user-handled long-term cooperative locking
- Users can add new functions and operators. User-defined functions allow database designer to encapsulate business logic in the database, rather than in the front-end.
- Server-side user defined functions can be written in several languages
- **Indexes:** B-tree, R-tree, Hash, and Gist indexes. User-definable index methods. Functional indexes. Indexes can be added and removed at any time.
- **Rules:** Rules are a special feature of PostgreSQL that allows a database designer to substitute any SQL table operation (SELECT, INSERT, DELETE, etc.) for another operation. Rules can be used to return different sets based on permissions, turn an attempted update into a delete, etc.
- Temporary tables, which are automatically dropped at the end of a database connection.
- LISTEN and NOTIFY can be used to pass messages or notify different clients of an event in the database. This

can be used to coordinate different front-end clients (even across different front-end systems).

- Several **Replication** solutions allowing the duplication of the master database to multiple slave machines.
- Cross-database compatibility functions for easing the transition from other, less SQL-compliant RDBMS.
- Functional and Partial indexes.
- Loadable extensions offering SHA1, MD5, XML, and other functionality.
- **Roles** simplify the management of large numbers of users with complex overlapping database rights.
- **IN/OUT Parameters** substantially improve support of complex business logic for J2EE and .NET applications.

Transactions, Concurrency

- Transactions allow all data modifications to be encapsulated in atomic blocks. This allows the database to rollback and restore all original data if there is an error in the transaction or if the database server crashes. Transactions are a critical feature for ensuring data integrity in database-backed applications.
- ACID compliant
- Support for rollback
- Serializable transaction isolation
- **Two-Phase Commit** allow ACID-compliant transactions across widely separated servers.
- Works transparently with transaction features of interface programs, such as Perl's DBI, Zope, JDBC and ODBC transactions, etc.
- Multi-Version Concurrency Control (MVCC) for highly scalable concurrent applications:
- Readers do not block writers and writers do not block readers.
- "Better than row-level locking."
- Shared Row Locking supports higher levels of concurrency through the addition of shared row locks for foreign keys. Shared locks will improve insert and update performance on many high-volume OLTP applications.
- Various row and table level locks are available as well.

Server Administration

- Hot backup, Point in Time Recovery with Write Ahead Logs and hot stand-by solutions are available for high availability
- User/group security model for database objects
- Access to the database server itself can be restricted based on host, user name, database.
- Supports Kerberos authentication.
- Encrypted database connections with SSL or SSH.
- TCP/IP network connections or local Unix domain sockets.
- Virtual hosting capability
- Virtually unlimited size for databases, tables, rows. Unlimited number of rows and indexes per table.

Client Programming Languages and Interfaces

Perl (through standard DBI/DBD or native interface) Python (through PyGreSql or PoPy) PHP (support built into PHP) Tcl ODBC, and therefore many ODBC client programs, including Microsoft Access, StarOffice, ApplixWare, etc. JDBC (Java Database Connectivity) Standard SQL Embedded C native C and C++ API's Emacs LISP Scheme (Guile) R (Open Source statistical package)

psql, the featured-filled text console-based interactive query monitor, includes full support for history and command line editing, customization, local/remote database access, importing and exporting of data, and has options to interface to shell scripts.

Several graphical interactive database interfaces are available: pgaccess, a platform-independent Tcl/Tk program for database maintenance and report-writing; pgAdmin, a comprehensive PostgreSQL administration program for Windows; phpPgAdmin for web-based administration; in addition, graphic query monitors are available for many operating systems and desktops, including Windows, KDE, and GNOME.

Documentation and Support

- Includes comprehensive manual set, including tutorial and reference, available in HTML or print format.
- Active mailing lists for support by the developers and user community.
- Online resource listing, user contributed extensions, and tips at http://techdocs.postgresql.org
- Commercial contracts are available from several companies, and consulting from hundreds of companies and independent consultants.

Created 8/2004 by Stefan Schumacher <stefan@net-tex.de>
Last updated to 1.10 in 4/2006 by Stefan Schumacher <stefan@net-tex.de>

PostgreSQL

"The most advanced Open Source database system in the world"

What is PostgreSQL?

PostgreSQL is an enhancement of the POSTGRES database management system (and is still sometimes referred to as simply "Postgres"), a next-generation DBMS research prototype. While PostgreSQL retains the powerful data model and rich data types of POSTGRES, it replaces the PostQuel query language with an extended subset of SQL. PostgreSQL is free and the complete source is available. PostgreSQL development is performed by a team of developers who all subscribe to the PostgreSQL development mailing list. This team is now responsible for all development of PostgreSQL. It is a community project and is not controlled by any company. The original name of the software at Berkeley was Postgres. When SQL functionality was added in 1995, its name was changed to Postgres95. The name was changed at the end of 1996 to PostgreSQL.

History, Developers, License

- Descended from INGRES and POSTGRES, cutting-edge academic database projects at the University of California at Berkeley
- Maintained, improved, and supported by a stable Global Development Group, and a large number of contributors
- Source code available for download, and anonymous CVS access is available
- Licensed under BSD license, allowing truly free modifications and redistribution of binaries

Legendary reliability and stability

Unlike many proprietary databases, it is extremely common for companies to report that PostgreSQL has never, ever crashed for them in several years of high activity operation. Not even once. It just works.