EnterpriseDB®
The Enterprise Postgres Company

Developments in PostgreSQL 9.0

A look at the development cycle and some of the new features coming in PostgreSQL 9.0

Dave Page
PostgreSQL Core Team
Senior Software Architect, EnterpriseDB UK

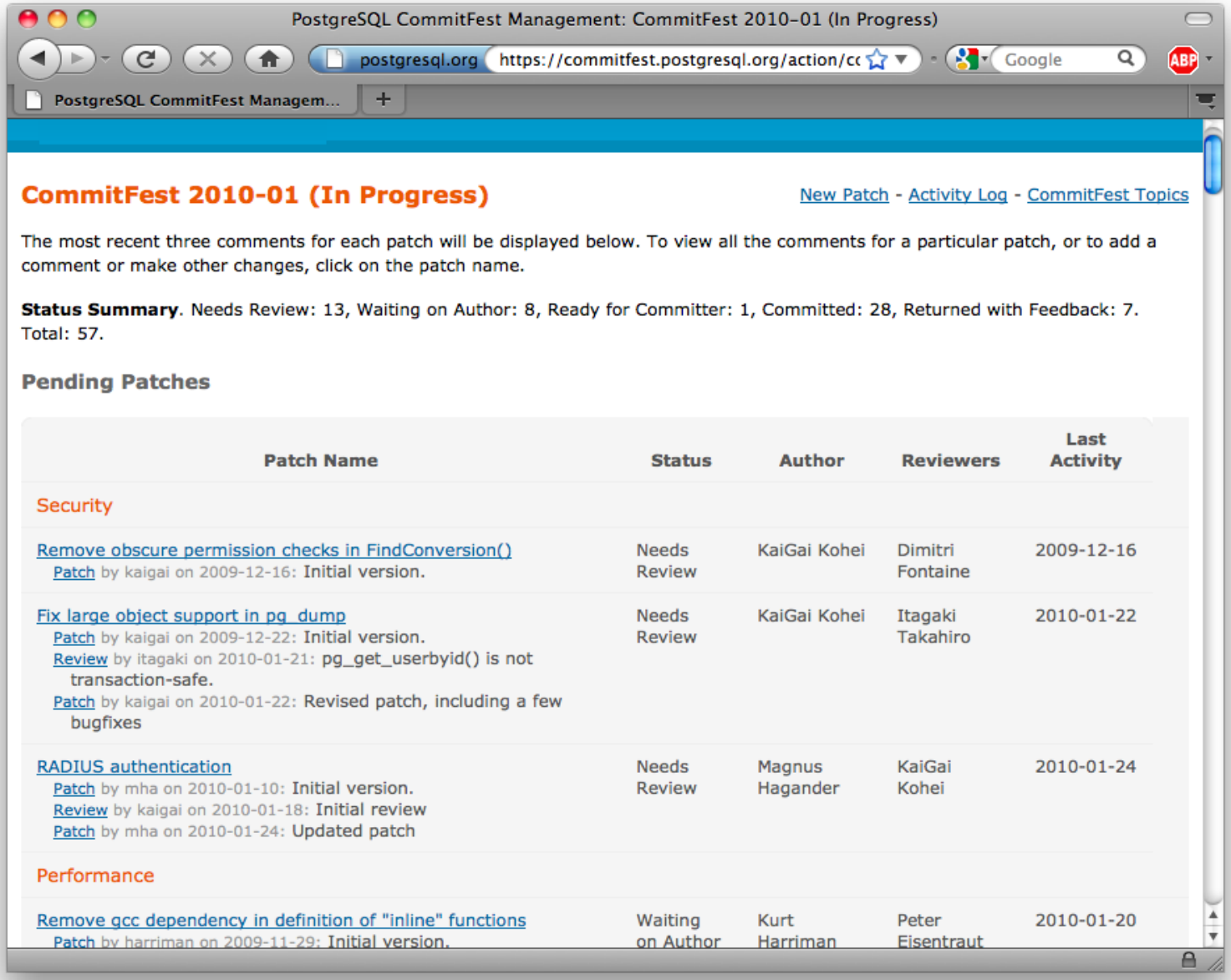
Commitfests

EnterpriseDB®
The Enterprise Postgres Company

- Introduced by the core team for PostgreSQL 8.4.
- Primary objective: to ensure timely review of patches.
- Managed on wiki.postgresql.org for PostgreSQL 8.4.
- Dedicated web app for 9.0 development cycle:
  - http://commitfest.postgresql.org
  - Written by Robert Haas

commitfest.postgresql.org

EnterpriseDB®
The Enterprise Postgres Company



Advantages

EnterpriseDB®
The Enterprise Postgres Company

- All patches can be properly tracked.
- Patch discussion remains on the mailing lists.
- Contributors get timely feedback.
- Easy to see “where we're at”.
- App works with our existing processes.

Alpha releases

EnterpriseDB®
The Enterprise Postgres Company

- New alpha releases following each commitfest:
  - Source, RPM and one-click installer builds.
  - Allow users to test features as soon as possible.
  - Early feedback helps reduce future beta time which can be very long.
  - Allow developers more time to incrementally update their apps to support the upcoming release.

9.0 Development cycle stats

EnterpriseDB®
The Enterprise Postgres Company

Commitfest	Committed patches	Returned patches	Rejected patches	Total
2009-07	37	24	5	66
2009-09	20	14	6	49
2009-11	27	11	0	38
2010-01	40	7	0	60 *

As of 6th February 2010

* Includes 13 outstanding patches

9.0 Development cycle stats

EnterpriseDB®
The Enterprise Postgres Company

- Total number of patches: 204 *
- Total number of patch submitters: 82
- Total number of patch committers: 14

* Not including bug fixes, or other patches directly applied by committers

A look at 9.0

EnterpriseDB®
The Enterprise Postgres Company

- The following patches are a tiny selection of the changes in 9.0.
- They are a pseudo-random selection of my choosing.
- They are in no particular order (except the last four for dramatic effect!).
- There are far more changes - see:
  - http://commitfest.postgresql.org/
  - PostgreSQL 9.0 release notes (when released).
- I don't know most of them intimately!

- Join removal
  - Patch by Robert Haas
  - Submitted 2009-07-21
  - Committed 2009-09-17
  - Query optimisation improvement to remove redundant joins from query plans

“The theoretical underpinning of this patch is as follows: Joins can affect the result of a query in one of two ways: either they change the set of rows returned (by either increasing it or decreasing it), or they provide attributes. To remove a join, we need to prove that neither of these things is possible. It's pretty easy to test that the join isn't providing any attributes above the level of the join, and you'll see that rel_attrs_needed_above_join() handles that here. Verifying that the number of rows returned isn't changed by the join is harder.”

Performance

EnterpriseDB®
The Enterprise Postgres Company

- New VACUUM FULL
  - Patch by Itagaki Takahiro
  - Submitted 2009-10-26
  - Committed 2010-01-06
  - Rewritten VACUUM FULL command

"The new version works like as CLUSTER USING ctid or rewriting in ALTER TABLE. It must be faster than them if we have many dead tuples and are not interested in physical order of tuples."

Procedural Languages

EnterpriseDB®
The Enterprise Postgres Company

- Anonymous code blocks
  - Patch by Petr Jelinek
  - Submitted 2009-08-30
  - Committed 2009-09-22
  - Add support for anonymous code blocks written in procedural languages

For example, grant all privileges on all views in schema public to role webuser:

```
DO $$DECLARE r record;
BEGIN
    FOR r IN SELECT table_schema, table_name FROM information_schema.tables
             WHERE table_type = 'VIEW' AND table_schema = 'public'
    LOOP
        EXECUTE 'GRANT ALL ON ' || quote_ident(r.table_schema) || '.' ||
                quote_ident(r.table_name) || ' TO webuser';
    END LOOP;
END$$;
```

Procedural Languages

EnterpriseDB®
The Enterprise Postgres Company

- pl/python improvements
  - Patches from multiple contributors:
    - Caleb welton
    - Peter Eisentraut
    - Hannu Valtonen
  - Python 3.1 support.
  - Anonymous block support.
  - Improved datatype conversions.

Procedural Languages

EnterpriseDB®
The Enterprise Postgres Company

- pl/perl improvements
  - Patches from multiple contributors:
    - Alexey Klyukin
    - Joshua Tolley
    - Tim Bunce
  - Error context support.
  - Anonymous block support.
  - Refactoring of plperl.c.
  - Add utility functions (quote_literal, encode_bytea, etc.).

Management/Security

EnterpriseDB®
The Enterprise Postgres Company

- Application Name
  - Patch by Dave Page
  - Submitted 2009-10-16
  - Committed 2009-11-28
  - Add an application name label to connections

This patch allows an application to set its name as a property of the connection. The name can then be reported in log messages and monitoring views such as *pg_stat_activity*.

- Application Name can be set in various ways:
  - libpq:
    - *PGAPPNAME* environment variable.
    - *application_name* connection string option.
  - SQL:
    - SET application_name = “My cool app”

- *fallback_application_name* libpq connection option:
  - Useful for generic utilities, e.g. a backup tool.
  - Value overridden by *application_name*.
  - Allows the backup tool to have a default name of ”Backup Tool” which may be overridden in a shell script, for example:

    PGAPPNAME=”Nightly backup” pg_dump dbname

- application_name can be seen:
  - In the *application_name* GUC variable
  - In CSV log output
  - In regular log output, if '%a' is included in log_line_prefix
  - In pg_stat_activity, e.g:

```
gator:~ dpage$ PGAPPNAME=Foo psql -p 5434 -U postgres postgres
Password for user postgres:
psql (9.0dev)
Type "help" for help.
postgres=# SELECT datname, procpid, usename, application_name FROM pg_stat_activity;
 datname  | procpid | usename  | application_name
----------+---------+----------+-----------------
 postgres |   40812 | postgres | Foo
(1 row)
```

Management/Security

EnterpriseDB®
The Enterprise Postgres Company

- Per user, per database GUCs
  - Patch by Alvaro Herrera
  - Submitted 2009-08-25
  - Committed 2009-10-07
  - Allow more complex user/database default GUC settings

8.4 allowed:

```
ALTER DATABASE <database> SET <guc> TO <value>;
ALTER ROLE <role> SET <guc> TO <value>;
```

9.0 adds:

```
ALTER ROLE <role> IN DATABASE <database> SET <guc> TO <value>;
```

Management/Security

EnterpriseDB®
The Enterprise Postgres Company

- GRANT ON ALL IN <schema>
  - Patch by Petr Jelinek
  - Submitted 2009-06-17
  - Committed 2009-10-12
  - Allow privileges to be set on the entire schema

Can greatly simplify the configuration of privileges:

```
GRANT <privileges> ON ALL <object type>S IN SCHEMA <schema> TO <role>;

REVOKE <privileges> ON ALL <object type>S IN SCHEMA <schema> FROM <role>;
```

Management/Security

EnterpriseDB®
The Enterprise Postgres Company

- Hook and module for checking password complexity
  - Patch by Laurenz Albe
  - Submitted 2009-10-09
  - Committed 2009-11-18
  - Allow a native method to enforce password complexity
- Often a security 'checkbox' item for evaluators.
- Recommended previous solutions revolve around use of external authentication methods, such as PAM or LDAP.

- Password check plugin function:

```
static void check_password(const char *username,
                           const char *password,
                           int password_type,
                           Datum validuntil_time,
                           bool validuntil_null)
{
    if (password_type == PASSWORD_TYPE_PLAINTEXT &&
        strlen(password) < 6)
    {
        ereport(ERROR,
                (errcode(ERRCODE_INVALID_PARAMETER_VALUE),
                errmsg("password is too short")));
    }
}
```

Management/Security

EnterpriseDB®
The Enterprise Postgres Company

- Caveats:
  - Cannot easily test MD5 passwords.
  - Should be used in conjunction with SSL.
- But:
  - Infinite flexibility to meet corporate security requirements.
  - Checks the box!

Triggers

EnterpriseDB®
The Enterprise Postgres Company

- Triggers on columns
  - Patch by Itagaki Takahiro
  - Submitted 2009-09-92
  - Committed 2009-10-14
  - Allow triggers to fire on actions against specific columns

- Avoids tedious checks for column value changes in procedure code:

```
CREATE TRIGGER <name>
    BEFORE UPDATE OF <col1>, <col12>, ...
    ON <table>
    FOR EACH ROW
    EXECUTE PROCEDURE <procedure>;
```

Triggers

- Triggers with WHEN clauses
  - Patch by Itagaki Takahiro
  - Submitted 2009-10-21
  - Committed 2009-11-20
  - Allow triggers to fire under specific criteria

- Avoids tedious checks for conditions in procedure code:

```
CREATE TRIGGER <trigger>
    BEFORE UPDATE ON <table>
    FOR EACH ROW
    WHEN (OLD.* IS DISTINCT FROM NEW.*)
    EXECUTE PROCEDURE <function>;
```

Useful stuff

EnterpriseDB®
The Enterprise Postgres Company

- Machine readable EXPLAIN output
  - Patch by Robert Haas and others
  - Submitted as multiple patches
  - Committed over multiple commitfests
  - Offers alternate formats for query plans
- Enables applications to easily read and interpret EXPLAIN output for application such as:
  - Graphical plan visualisation.
  - Monitoring systems.
- Can include much more information than the human readable output.

Useful stuff

EnterpriseDB®
The Enterprise Postgres Company

- Text (human readable) EXPLAIN output:

```
postgres=# EXPLAIN (FORMAT TEXT) SELECT * FROM pg_class;
                            QUERY PLAN
-------------------------------------------------------------------
 Seq Scan on pg_class  (cost=0.00..8.55 rows=255 width=186)
(1 row)
```

- XML EXPLAIN output:

```
postgres=# EXPLAIN (FORMAT XML) SELECT * FROM pg_class;
                          QUERY PLAN
------------------------------------------------------------
 <explain xmlns="http://www.postgresql.org/2009/explain">+
   <Query>                                                +
     <Plan>                                               +
       <Node-Type>Seq Scan</Node-Type>                    +
       <Relation-Name>pg_class</Relation-Name>            +
       <Alias>pg_class</Alias>                            +
       <Startup-Cost>0.00</Startup-Cost>                  +
       <Total-Cost>8.55</Total-Cost>                      +
       <Plan-Rows>255</Plan-Rows>                         +
       <Plan-Width>186</Plan-Width>                       +
     </Plan>                                              +
   </Query>                                               +
 </explain>
(1 row)
```

- YAML EXPLAIN output:

```
postgres=# EXPLAIN (FORMAT YAML) SELECT * FROM pg_class;
         QUERY PLAN
-----------------------------
 - Plan:                     +
     Node Type: Seq Scan     +
     Relation Name: pg_class+
     Alias: pg_class         +
     Startup Cost: 0.00      +
     Total Cost: 8.55        +
     Plan Rows: 255          +
     Plan Width: 186
(1 row)
```

Useful stuff

EnterpriseDB®
The Enterprise Postgres Company

- JSON EXPLAIN output:

```
postgres=# EXPLAIN (FORMAT JSON) SELECT * FROM pg_class;
              QUERY PLAN
-------------------------------------
 [                                  +
   {                                +
     "Plan": {                      +
       "Node Type": "Seq Scan",     +
       "Relation Name": "pg_class",+
       "Alias": "pg_class",         +
       "Startup Cost": 0.00,        +
       "Total Cost": 8.55,          +
       "Plan Rows": 255,            +
       "Plan Width": 186            +
     }                              +
   }                                +
 ]
(1 row)
```

Useful stuff

EnterpriseDB®
The Enterprise Postgres Company

- Deferrable UNIQUE constraints
  - Patch by Dean Rasheed
  - Submitted 2009-07-12
  - Committed 2009-07-29
  - Allows UNIQUE constraint checking to be deferred until commit

```
CREATE TABLE distributors (
    did     integer,
    name    varchar(40) UNIQUE DEFERRABLE
);
```

So what makes it 9.0?

Headline features

EnterpriseDB®
The Enterprise Postgres Company

- Operator Exclusion Constraints
  - Patch by Jeff Davis
  - Submitted 2009-10-25
  - Committed 2009-12-06

- Generalize the concept of uniqueness to support any indexable commutative operator, not just equality.

- Two rows violate the exclusion constraint if "row1.col OP row2.col" is TRUE for each of the columns in the constraint.

Headline features

EnterpriseDB®
The Enterprise Postgres Company

- Operator Exclusion Constraint pointless example:

```
postgres=# CREATE TABLE foo (
postgres(#     col INT4,
postgres(#     EXCLUDE (col WITH =)
postgres(# );
NOTICE:  CREATE TABLE / EXCLUDE will create implicit index "foo_col_exclusion" for
table "foo"
CREATE TABLE
postgres=# INSERT INTO foo (col) VALUES (1);
INSERT 0 1
postgres=# INSERT INTO foo (col) VALUES (2);
INSERT 0 1
postgres=# INSERT INTO foo (col) VALUES (1);
ERROR:  conflicting key value violates exclusion constraint "foo_col_exclusion"
DETAIL:  Key (col)=(1) conflicts with existing key (col)=(1).
```

Headline features

EnterpriseDB®
The Enterprise Postgres Company

- Operator Exclusion Constraint useful example:

```
postgres=# CREATE TABLE circles (
postgres(#     c circle,
postgres(#     EXCLUDE USING gist (c WITH &&)
postgres(# );
NOTICE:  CREATE TABLE / EXCLUDE will create implicit index "circles_c_exclusion" for
table "circles"
CREATE TABLE                            ^                               ^
postgres=# INSERT INTO circles (c) VALUES ('<(1, 1), 10>');
INSERT 0 1
postgres=# INSERT INTO circles (c) VALUES ('<(15, 15), 10>');
ERROR:  conflicting key value violates exclusion constraint "circles_c_exclusion"
DETAIL:  Key (c)=(<(15,15),10>) conflicts with existing key (c)=(<(1,1),10>).
postgres=# INSERT INTO circles (c) VALUES ('<(20, 20), 10>');
INSERT 0 1
```

Headline features

EnterpriseDB®
The Enterprise Postgres Company

- Operator Exclusion Constraint presentation:

Beyond UNIQUE:
Exclusion constraints in PostgreSQL 9.0

- Magnus Hagander

16:15 – 17:00 today

Headline features

EnterpriseDB®
The Enterprise Postgres Company

- Win64 Support
  - Patch by Tsutomu Yamada
  - Submitted 2009-12-02
  - Committed 2010-01-14
- Adds native support for a 64 bit Windows® port.
- Largely an exercise in getting the data types right.
- No user-visible changes.

Headline features

EnterpriseDB®
The Enterprise Postgres Company

- Win64 Support – why didn't we bother earlier?
  - There's no real advantage to using large amounts of shared memory on Windows.
  - PostgreSQL uses a multi-process architecture, so individual backends can **each** use more than 1GB of RAM if needed.
  - 64-bit pointers and integers may actually introduce additional overhead that isn't necessary.

Headline features

EnterpriseDB®
The Enterprise Postgres Company

- Win64 Support – beware....
  - Lack of Win64 support in some supporting libraries means fewer features enabled – e.g. GNU Gettext, MIT Kerberos.
  - Add-on applications will likely lag behind PostgreSQL, adding Win64 support in future releases – e.g. Slony, PostGIS.
  - Other applications may remain 32-bit only (which doesn't necessarily affect the user).

Headline features

EnterpriseDB®
The Enterprise Postgres Company

- Streaming replication
  - Patch by Fujii Masao
  - Submitted 2009-09-14
  - Committed 2010-01-15
- Adds streaming log-based replication.
- Replicates entire database cluster to one or more slaves.
- Works in conjunction with existing log shipping to provide reliable, near-realtime replication.

Headline features

EnterpriseDB®
The Enterprise Postgres Company

- Streaming replication – in a nutshell:
  - Create one or more servers with the same architecture, version etc. as the master and setup log shipping.
  - Take a base backup of the master, and load it on the slave.
  - Setup the slave to connect to the master to receive streamed logs (in *recovery.conf*).
  - Create a trigger file (when required) to cause the slave to exit recovery mode and startup.

Headline features

EnterpriseDB®
The Enterprise Postgres Company

- Streaming replication presentation:

Streaming replication under the hood

- Heikki Linnakangas

14:15 – 15:00 today

Headline features

EnterpriseDB®
The Enterprise Postgres Company

- Hot standby
  - Patch by Simon Riggs
  - Submitted 2009-09-15
  - Committed 2009-12-19
- Allows read-only queries on standby servers, utilising otherwise idle hardware.
- Works with record based log shipping or streaming replication.
- Feature developed over multiple releases.

Headline features

EnterpriseDB®
The Enterprise Postgres Company

- Hot standby presentation:

Hot Standby Live

- Simon Riggs

15:15 – 16:00 today

That's all folks

EnterpriseDB®
The Enterprise Postgres Company

Questions?

That's all folks

EnterpriseDB®
The Enterprise Postgres Company

Thank you!

http://www.postgresql.org/

http://commitfest.postgresql.org/