PostgreSQL 9.4

FOSDEM 2014
Brussels, Belgium

Magnus Hagander
magnus@hagander.net
Magnus Hagander

• PostgreSQL
  • Core Team member
  • Committer
  • PostgreSQL Europe
• Redpill Linpro
  • Infrastructure services
  • Principal database consultant
Do you read...

- planet.postgresql.org
Thanks to!

- depesz
- Michael Paquier
- (others)
- And the developers of course!
Development schedule

- June 14, 2013 - branch 9.3
- June 2013 - CF1
- September 2013 - CF2
- November 2013 - CF3
- January 2014 - CF4
Current status

• In final Commit Fest
• Started Jan 14
• Continues until the end
• Please help with review and testing!
Current status

• Statistics!
  • 1892 files changed
  • 81018 insertions (+)
  • 55533 deletions (-)
• Significantly lower than 9.3
  • So far
• CF1 still in progress!
So what's really new

- Need some sort of categorization
- Developer and SQL features
- Infrastructure
- DBA and administration
- Replication and recovery
New features

• Developer and SQL features
• Infrastructure
• DBA and administration
• Replication and recovery
Smaller aggregate changes

• Allow variadic aggregates
• None shipped by default
• For user defined
• Improved EXPLAIN information

HashAggregate  (cost=2.94..2.98  rows=4  width=8)
   Group Key: a
   ->  Seq Scan on agg  (cost=0.00..2.11  rows=111  width=8)
FILTER aggregates

• Add support for FILTER clause to aggregates
• No more case-then-null!

```sql
SELECT a,
    count(*),
    count(*) FILTER (WHERE b > 5)
FROM agg GROUP BY a
```
Ordered-set aggregates

- New class of aggregates
  - "Offset in group"
- WITHIN GROUP
- Also Hypothetical aggregates
Ordered-set aggregates

• Most common value in group

```sql
SELECT a, 
       mode() WITHIN GROUP (ORDER BY b) 
FROM agg GROUP BY a
```
Ordered-set aggregates

- Percentiles

```sql
SELECT a,
    percentile_cont(0.3) WITHIN GROUP (ORDER BY b),
    percentile_disc(0.3) WITHIN GROUP (ORDER BY b)
FROM agg GROUP BY a
```
Ordered-set aggregates

• Hypothetical rows

```
SELECT  a,
        rank(4) WITHIN GROUP (ORDER BY b),
        percent_rank(4) WITHIN GROUP (ORDER BY b)
FROM    agg GROUP BY a
```
Improved updatable views

- Partially updatable views
- Some columns can be updated, others not
- Automatically detected
Improved updatable views

• WITH CHECK OPTION
• Only allow rows visible through view
• LOCAL
  • Only conditions on current view checked
• CASCADE
  • Recursively check on parents
  • Default once CHECK OPTION is specified
UNNEST

• Multi-argument UNNEST
• Unnest multiple arrays at once

```sql
SELECT * FROM unnest(
    array['a', 'b', 'c'],
    array['d', 'e', 'f']
)
```
WITH ORDINALITY

• Automatic row number for unnested rows

```sql
SELECT * FROM unnest(
    array['a', 'b', 'c'],
    array['d', 'e', 'f']
) WITH ORDINALITY
```
You can now get a stacktrace!

```sql
CREATE OR REPLACE FUNCTION public.inner_func() RETURNS integer AS $$
DECLARE
    stack text;
BEGIN
    GET DIAGNOSTICS stack = PG_CONTEXT;
    RAISE NOTICE E'--- Call Stack ---\n%', stack;
    RETURN 1;
END;
$$ LANGUAGE plpgsql;
```
New features

- Developer and SQL features
- Infrastructure
- DBA and administration
- Replication and recovery
Dynamic background workers

- 9.3 got background workers
- Only at postmaster startup
- Can now be started dynamically
Dynamic shared memory

• Shared memory can be allocated on request
• Main segment still fixed at startup
• Requested by e.g. bgworkers
• Also supports lightweight message queue
MVCC catalog access

• **SnapshotNow** has been removed
• All catalog access is now MVCC
• Extensions relying on it will break
  • This is intentional
• Simpler and more robust code
• Future decreased locking
New features

• Developer and SQL features
• Infrastructure
• DBA and administration
• Replication and recovery
MATERIALIZED VIEWS

• 9.3 added materialized views
  • Limited usability due to locking
• 9.4 adds concurrent refresh

REFRESH MATERIALIZED VIEW CONCURRENTLY myview

• Requires UNIQUE index on view
Tablespace options on creation

- Tablespace options can now be set on CREATE
- Previously required CREATE + ALTER

```
CREATE TABLESPACE ssd
    LOCATION '/ssd'
    WITH (random_page_cost = 1.1);
```
Move objects in tablespaces

- Move all objects in tablespaces
- Or all tables, all indexes, etc.

```
ALTER TABLESPACE pg_default
    MOVE INDEXES TO ssd;

ALTER TABLESPACE ssd
    MOVE ALL TO pg_default;
```
pg_prewarm

• Prewarm your cache
• Extension with `pg_prewarm()` function
• Prewarm OS or Postgres caches
Configuration

```
ALTER SYSTEM
  SET work_mem='10MB';

SELECT pg_reload_conf();
```
ALTER SYSTEM SET

- Variables in separate config file
- Overrides what's in postgresql.conf
- Reload still required
- Contexts still applies
  - Restart can be tricky!
New configuration parameters

- `autovacuum_work_mem`
- Default `-1 = use maintenance_work_mem`
- Can now be controlled independently
New configuration parameters

- `session_preload_libraries`
- Loaded at session startup
- But not just from `plugins` directory
New configuration parameters

- `wal_log_hints`
- Log hintbit changes to WAL
- Required for rewind tools when not using checksums
- Hint about checksum log increments
pg_stat_statements

• pg_stat_statements now exposes query id
• Internal hash value
  • Based on parse tree
• **NOT** stable across versions
  • Or platforms
  • Or schema modifications (some)
New features

• Developer and SQL features
• Infrastructure
• DBA and administration
• Replication and recovery
recovery_target=immediate

- New recovery target: immediate
- End recovery as soon as consistent state reached
Regular logging of running xacts

• Running transactions are logged every 15 secs
• Faster "consistency reached" on hot standbys
• Nothing sent on completely idle
Time delayed standbys

• Delay WAL application on slave
• Replays all normal WAL, delays at commit
• "fast recovery starting point"
• min_recovery_apply_delay=30min
• in recovery.conf
There's always more

- Lots of smaller fixes
- Performance improvements
- etc, etc
- Can't mention them all!
Still pending

- Commitfest is **in progress**
- Features in the queue may or may not be included
Foreign tables

- Triggers on foreign tables
- Foreign table inheritance
Performance

• Reduced locks for ALTER TABLE
• Reduced WAL volume for updates
• Partial sort
• GIN index fastscan
Backups

- Backup throttling
- Relocating tablespaces in pg_basebackup
- pg_stat_archiver
UPSERT

• INSERT INTO ON DUPLICATE KEY LOCK FOR UPDATE
• Makes it possible to implement UPSERT
  • Maybe in combination with wCTE
HStore 2.0

- Nested hstore
- Typed (partially)
- Casts <-> json
- jsonb
Tiny favorite?

- Dynamic library loading logs to DEBUG1
- Particularly useful for local_preload_libraries
- Less logspam!
Just because we're Postgres

• Date parsing now supports years >5 digits
• ISO parsing already supported this!
• Non-standard formats now supported as well
Thank you!

Magnus Hagander
magnus@hagander.net
@magnushagander