

# **New features of PostgreSQL 9.4**

In the field of NoSQL and advanced replication

Michael Paquier

Tokyo, Japan

2014/12/5

# About the lecturer

- Michael Paquier
  - Working on Postgres for VMware
  - Community hacker and blogger
  - Based in Tokyo
- Contacts
  - Twitter: @michaelpq
  - Blog: <http://michael.otacoo.com>
  - Contact: michael@otacoo.com

# Resources

- [planet.postgresql.org](http://planet.postgresql.org)
- Blogs
  - Depesz => [www.depesz.com](http://www.depesz.com)
  - Michael Paquier => [michael.otacoo.com](http://michael.otacoo.com)
  - Documentation:  
<http://www.postgresql.org/docs/devel/static/release-9-4.html>

# Current status

- Postgres 9.4 RC1 released on 20<sup>th</sup> Nov
- Beta period
  - 6-months!
  - Beta1 => 11<sup>th</sup> May
  - Issues related to some key-features

# Development quantity

- For 9.4
  - 2183 files changed
  - 374421 insertions(+)
  - 209439 deletions(-)
- For 9.3
  - 2610 files changed
  - 228600 insertions(+)
  - 184221 deletions(-)

```
git diff \  
$(git merge-base REL9_3_STABLE master) \  
$(git merge-base REL9_4_STABLE master) \  
--stat | tail -n 1
```

# Categories

- **SQL-related features**
- Administration and DBA things
- Infrastructure and replication

# FILTER and aggregates

- Filter tuples evaluated by aggregates in GROUP BY clause
- Forget CASE ... THEN NULL!

```
=# CREATE TABLE aggtab AS SELECT generate_series(1,10) as id;  
-- No more!  
=# SELECT id,  
        count(id),  
        count(case when id < 5 then id else NULL END)  
FROM aggtab GROUP BY id;  
=# SELECT id,  
        count(id),  
        count(id) FILTER (WHERE id < 5)  
FROM aggtab GROUP BY id;
```

# Ordered-set aggregates (1)

- WITHIN GROUP
- Hypothetical aggregates
- mode(), most present value in a subset

```
=# CREATE TABLE modetab (id int);  
=# INSERT INTO modetab VALUES (1), (2), (3), (2), (4);  
=# SELECT mode() WITHIN GROUP (ORDER BY id)  
   FROM modetab;  
mode  
-----  
   2  
(1 row)
```



# Ordered-set aggregates (2)

- Percentiles

```
=# CREATE TABLE percent_tab AS
  SELECT generate_series(1,200) AS id;
=# SELECT percentile_cont(0.3) WITHIN GROUP (ORDER BY id),
        percentile_disc(0.3) WITHIN GROUP (ORDER BY id)
   FROM percent_tab;
percentile_cont | percentile_disc
-----+-----
              60.7 |              60
(1 row)
```

- Rank

```
=# SELECT rank(8) WITHIN GROUP (ORDER BY id DESC) FROM percent_tab;
rank
-----
   193
(1 row)
```

# unnest() and WITH ORDINALITY

- unnest() through multiple array series
- WITH ORDINALITY => column to order results

```
=# SELECT *
```

```
FROM unnest(array[1,2], array[3,4,5])
```

```
WITH ORDINALITY;
```

```
unnest | unnest | ordinality
```

```
-----+-----+-----
```

```
1 | 3 | 1
```

```
2 | 4 | 2
```

```
null | 5 | 3
```

```
(3 rows)
```

# GIN indexes

- Improved key lookup (common AND rare)
- Improved compression using delta method
  - Less space, up to 6x
  - Cost some extra CPU
- Example, pg\_trgm on “Les Misérables”
  - 68116 lines
  - gin\_trgm\_ops
  - 9.3 => 24MB
  - **9.4 => 8MB!**

# JSONB - datatype

- Binary representation of JSON
- Key order not preserved
- Faster than JSON datatype
  - Which only stored text
  - With a format validator
- Key uniqueness: last value wins

```
=# SELECT '{"key1":"val1", "key1":"val1bis"}'::jsonb;  
      jsonb  
-----  
{"key1": "val1bis"}  
(1 row)
```

# JSONB - GIN indexing

- Lookup to all keys possible with 1 index
  - No more index **per** key
- Compact
- Faster than 9.3 and... MongoDB

```
=# SELECT data->population FROM geodata  
  WHERE data @> '{"country_code": "JP", "asciiame": "Tokyo"}';  
      [...]  
-> Bitmap Index Scan on geodata_gin (cost=0.00..128.86 rows=8648 width=0)  
   Index Cond: (data @> '{"asciiame": "Tokyo", "country_code": "JP"}'::jsonb)
```

# Auto-updatable views (1)

- Column-base update
- Automatically detected
- Here: a => OK, b => OK, c => No

```
=# CREATE TABLE aa (a int, b int);  
=# CREATE TABLE bb (a int);  
=# CREATE VIEW aav AS  
    SELECT aa.*,  
    (SELECT avg(a) FROM bb) AS c FROM aa;
```

# Auto-updatable views (2)

- WITH CHECK
  - LOCAL, on current view only
  - CASCADE, with parents (default)
- INSERT and UPDATE control for new tuples

```
=# CREATE VIEW aav2 AS
    SELECT aa.* FROM aa WHERE a > 5 WITH CHECK OPTION;
CREATE VIEW
=# INSERT INTO aav2 VALUES (1,2);
ERROR: 44000: new row violates
        WITH CHECK OPTION for view "aav2"
DETAIL: Failing row contains (1, 2).
LOCATION: ExecWithCheckOptions, execMain.c:1676
```

# Categories

- SQL-related features
- **Administration and DBA things**
- Infrastructure and replication



# Materialized views

- Refresh CONCURRENTLY
  - Unique index on relation
  - Share Update Exclusive lock => READ/WRITE OK!
- In 9.3, REFRESH = exclusive lock

```
=# CREATE TABLE basetab AS
    SELECT generate_series(1,100) as id;
=# CREATE MATERIALIZED VIEW matbase
    AS SELECT * FROM basetab;
=# CREATE UNIQUE INDEX mati ON matbase(id);
=# REFRESH MATERIALIZED VIEW
    CONCURRENTLY matbase;
```

# ALL IN TABLESPACE

- Move all objects of a database to new place
  - ALTER TABLE (Only table data, **not indexes**)
  - ALTER INDEX
  - ALTER MATERIALIZED VIEW

```
=# CREATE TABLESPACE fast_ssd LOCATION '/to/fast/ssd/path';  
CREATE TABLESPACE  
=# ALTER TABLE ALL IN TABLESPACE  
    pg_default SET TABLESPACE fast_ssd;  
ALTER TABLE  
=# ALTER INDEX ALL IN TABLESPACE  
    pg_default SET TABLESPACE fast_ssd;  
ALTER INDEX
```

# pg\_stat\_statements

- Tracking of query ID
  - Hash value
  - Calculated using parse tree
- Save stat file at shutdown

# ALTER SYSTEM

- Manipulate configuration settings
- postgresql.auto.conf in PGDATA
  - Takes priority on postgresql.conf
- SIGHUP params OK, but restart tricky

```
=# ALTER SYSTEM  
    SET synchronous_standby_names TO 'node_5433';  
=# select pg_reload_conf();
```

# session\_preload\_libraries

- Automatically load library for given session
- Can be changed with ALTER ROLE
- Useful for debugging
  - auto\_explain
- local\_preload\_libraries for all sessions
- No need to LOAD

# pg\_prewarm

- Module in contrib/pg\_prewarm
- Useful for tests to avoid/reduce ramp-up
- Preload buffers in OS or Postgres cache

```
=# \dx+ pg_prewarm
      Objects in extension "pg_prewarm"
      Object Description
-----
function pg_prewarm(regclass,text,text,bigint,bigint)
(1 row)
```

# Stacktrace using pl/pgsql

- Useful for debugging of multiple levels!

```
=# CREATE FUNCTION foo() RETURNS int AS $$  
DECLARE  
    stack text;  
BEGIN  
    GET DIAGNOSTICS stack = PG_CONTEXT;  
    RAISE NOTICE E'--- Call Stack ---\n%', stack;  
    RETURN 1;  
END;  
$$ LANGUAGE plpgsql;
```

# Categories

- SQL-related features
- Administration and DBA things
- **Infrastructure and replication**



# Replication slots

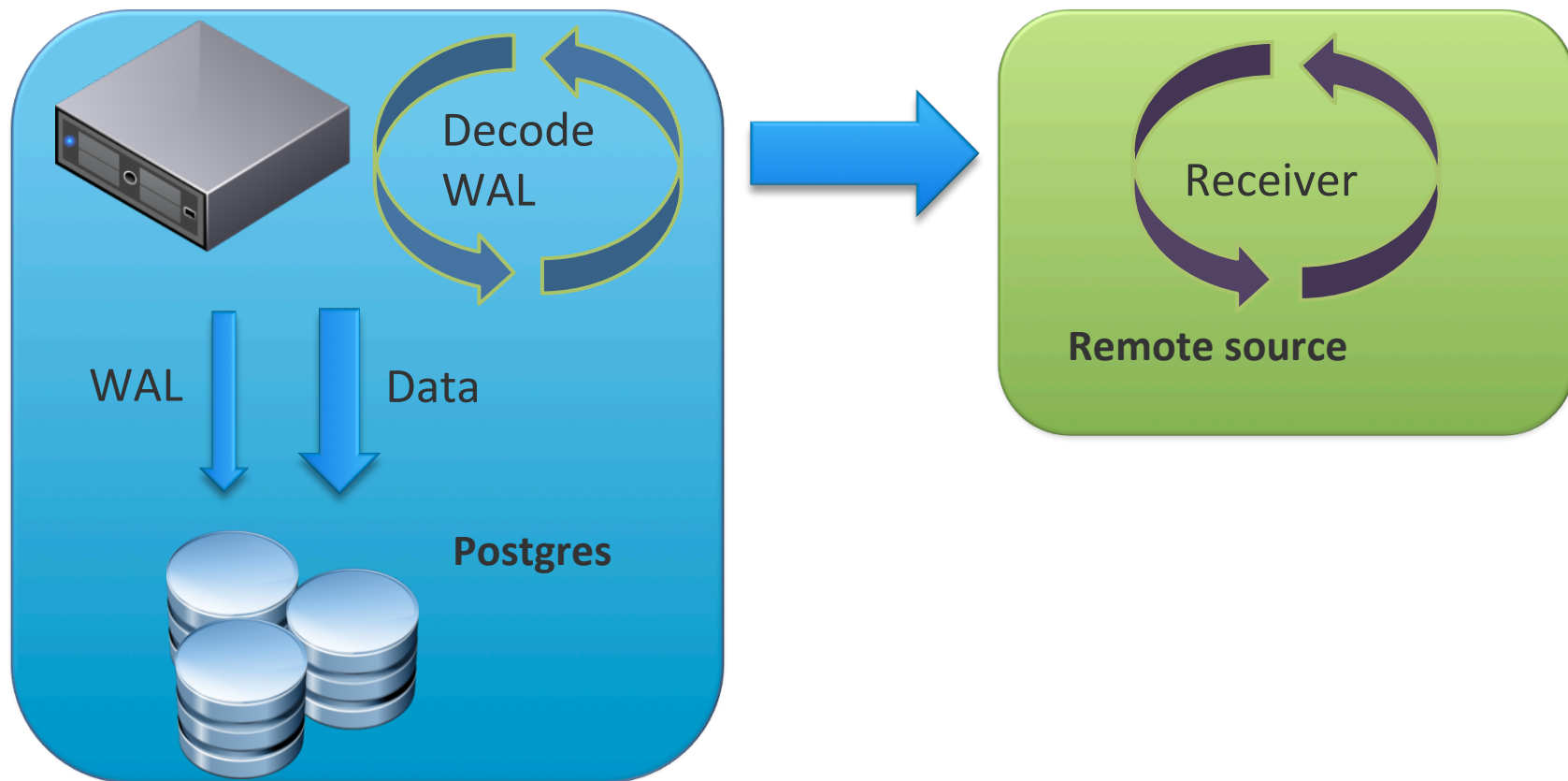
- Block WAL file deletion until consumption
- Prevent replication conflicts with xmin horizon report
- Danger: monitoring of pg\_xlog partition.
- **wal\_keep\_segments** not needed!

# Time delayed standbys

- Delayed live backups
- Can fix unfortunate transactions
- Not untransactional, like DROP DATABASE
- WAL replayed, commit delayed
- `recovery_min_apply_delay='$TIME'`  
in `recovery.conf`
- With `hot_standby_feedback`, risk of bloat on master

# Logical decoding (1)

- Get logical changes from WAL



# Logical decoding (2)

- Examples:
  - Client: pg\_recvlogical
  - Output plugin: contrib/test\_decoding
- Use cases
  - Online upgrade
  - Replication solutions (BDR, Slony)
  - Audit

# Dynamic bgworkers

- 9.3 limitation
  - Spawn at server startup
  - Connection to only one database
- Can be started at will
  - Max with max\_worker\_processes
- Build your own structure launcher/worker!
- Example: contrib/worker\_spi
- Base for parallel query processing

# wal\_log\_hints

- wal\_log\_hints
- Log hint-bits (TX state on page) to WAL
- Required for rewind tools when not using checksums (pg\_rewind!)
- Page checksum does it

# Backup features

- `pg_stat_archiver`, view archiving activity
  - Number of WAL files archived
  - Failure/success tracking
- `pg_basebackup`
  - `--xlogdir` for custom location of `pg_xlog` dir
  - Backup throttling with `--max-rate`
  - Relation of tablespaces in backup copy

# And more, far more...

- Huge community work
- Performance improvements
- etc.
- Complete list is here:  
<http://www.postgresql.org/docs/devel/static/release-9-4.html>



Thanks!  
Questions?

# JSONB, No2 on Hacker news!

**Y** **Hacker News** new | comments | show | ask | jobs | submit

1. ▲ **Keeping Secrets** (medium.com)  
118 points by ghosh 5 hours ago | 7 comments
2. ▲ **Postgres 9.4 feature highlight: Indexing JSON data with jsonb data type** (otacoo.com)  
105 points by ghosh 5 hours ago | 17 comments
3. ▲ **Unraveling NSA's TURBULENCE Programs** (sesek.com)  
35 points by acqq 3 hours ago | 3 comments
4. ▲ **The Programmer's Price: Want to hire a coding superstar? Call the agent** (newyorker.com)  
107 points by eroo 6 hours ago | 70 comments
5. ▲ **Young Brits in Silicon Valley** (theguardian.com)  
40 points by kul 3 hours ago | 8 comments
6. ▲ **Good Game: The Rise of the Professional Cyber Athlete** (newyorker.com)  
39 points by austinz 4 hours ago | 2 comments
7. ▲ **Show HN: A Python Spider System with Web UI** (github.com)  
102 points by binux 7 hours ago | 18 comments
8. ▲ **The Dutch Village Where Everyone Has Dementia** (theatlantic.com)  
39 points by prawn 9 hours ago | 10 comments
9. ▲ **Reproducible Development Environments with GNU Guix** (dthompson.us)  
49 points by rev 6 hours ago | 6 comments
10. ▲ **What does the NSA think of academic cryptographers?** (scottaaronson.com)  
186 points by robinhouston 18 hours ago | 27 comments