



Postgres 10



Oleg Bartunov
Moscow University,
Postgres Professional

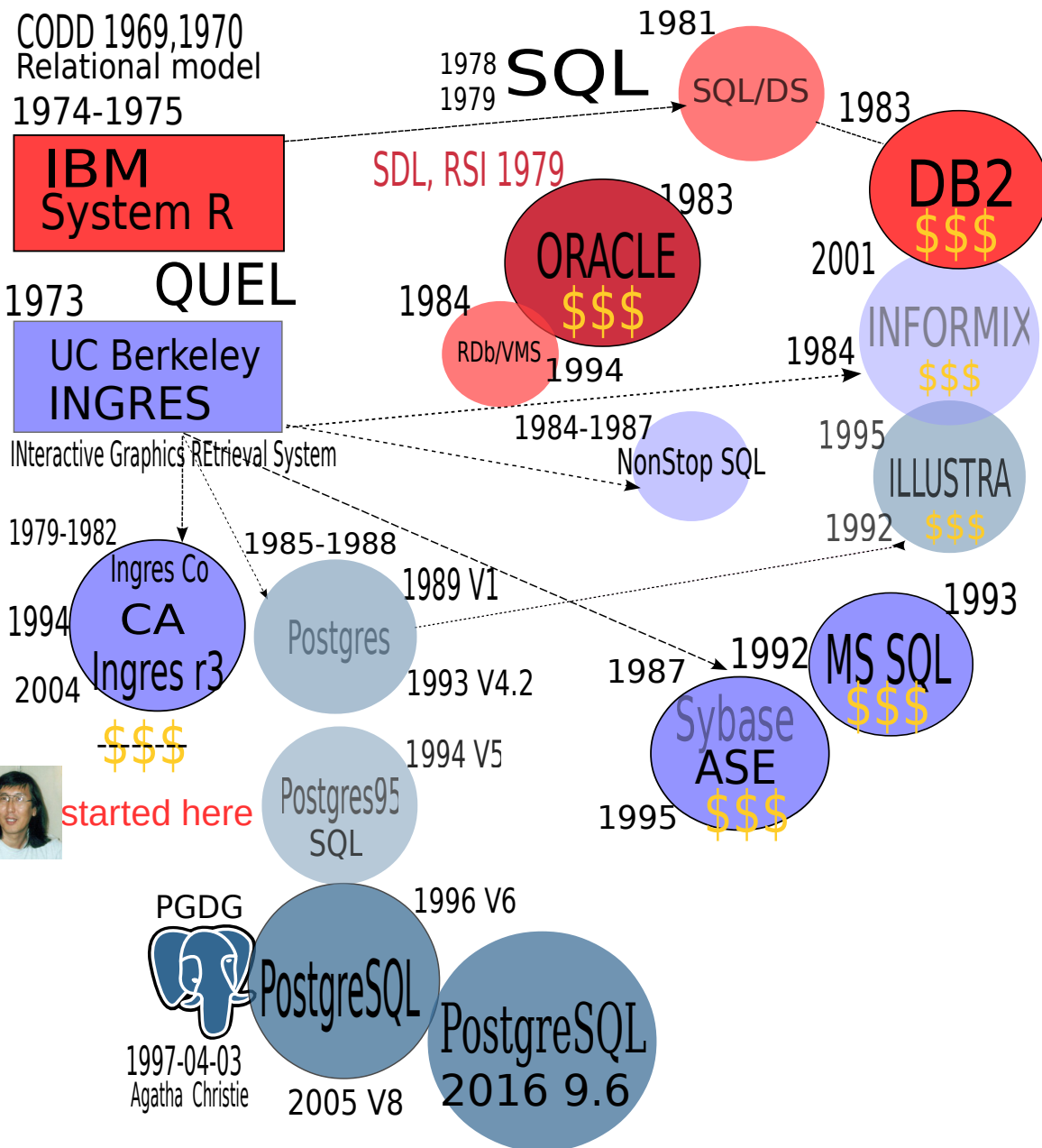
Oct 5, 2017, Moscow



PostgreSQL History >20 years



Michael Stonebreaker
Turing Award, 2015

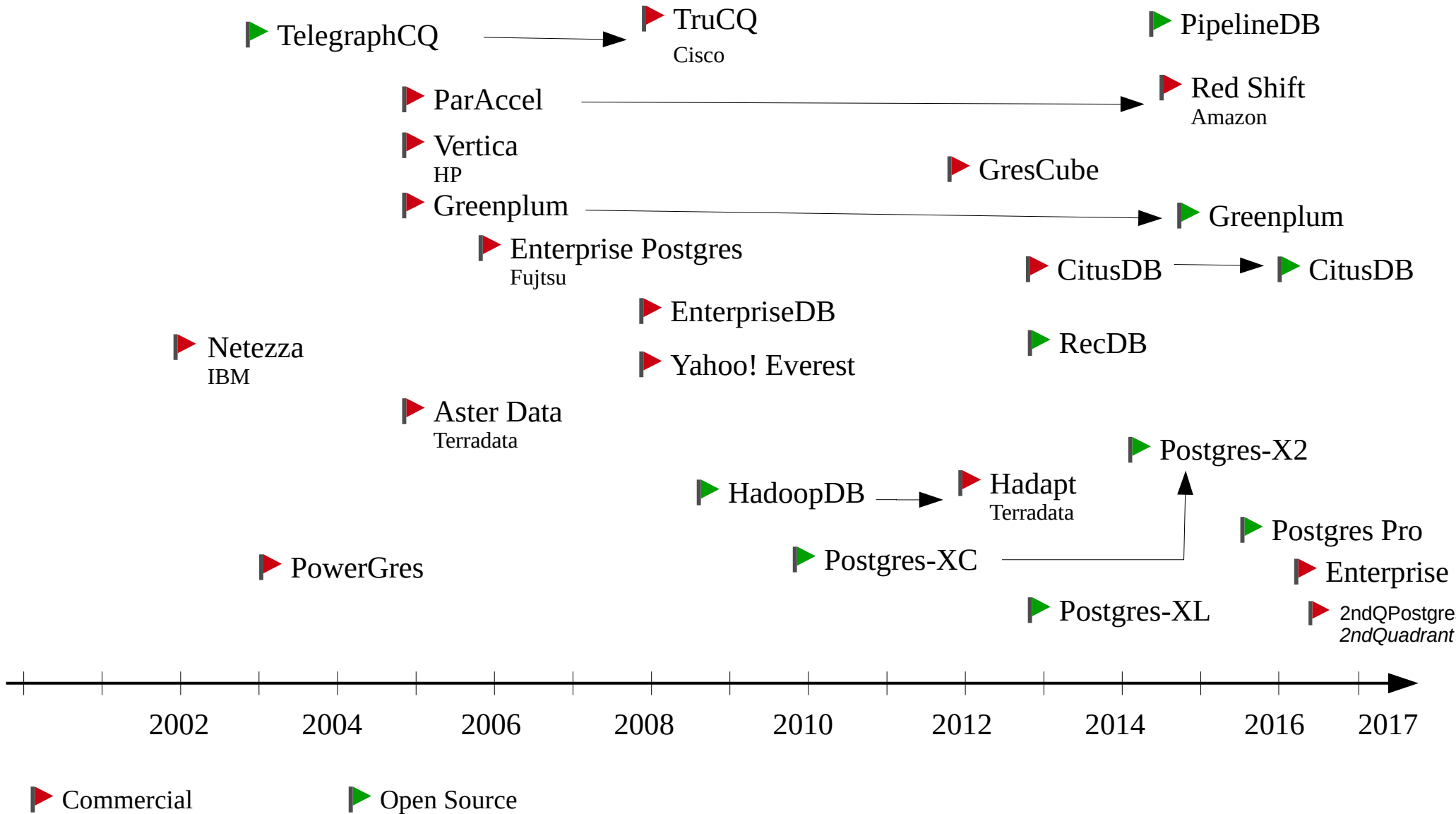


started here

2017 10





















PostgreSQL Forks (we love forks!)



PostgreSQL is #4 !

334 systems in ranking, September 2017

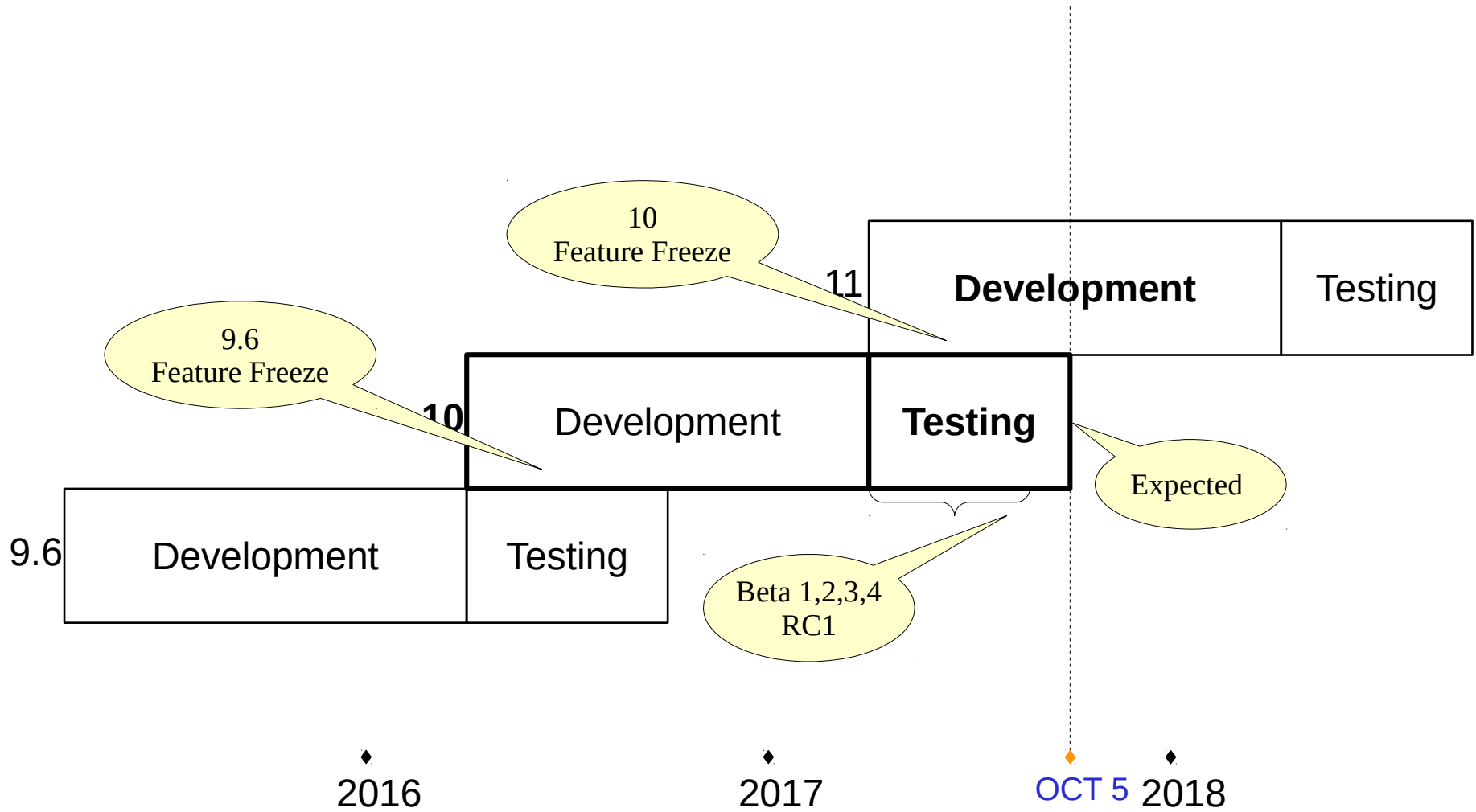
Rank			DBMS	Database Model	Score		
Sep 2017	Aug 2017	Sep 2016			Sep 2017	Aug 2017	Sep 2016
1.	1.	1.	Oracle  	Relational DBMS	1359.09	-8.78	-66.47
2.	2.	2.	MySQL  	Relational DBMS	1312.61	-27.69	-41.41
3.	3.	3.	Microsoft SQL Server  	Relational DBMS	1212.54	-12.93	+0.99
4.	4.	4.	PostgreSQL  	Relational DBMS	372.36	+2.60	+56.01
5.	5.	5.	MongoDB  	Document store	332.73	+2.24	+16.74
6.	6.	6.	DB2 	Relational DBMS	198.34	+0.87	+17.15
7.	7.	 8.	Microsoft Access	Relational DBMS	128.81	+1.78	+5.50
8.	8.	 7.	Cassandra 	Wide column store	126.20	-0.52	-4.29
9.	9.	 10.	Redis 	Key-value store	120.41	-1.49	+12.61
10.	10.	 11.	Elasticsearch 	Search engine	120.00	+2.35	+23.52

PostgreSQL users

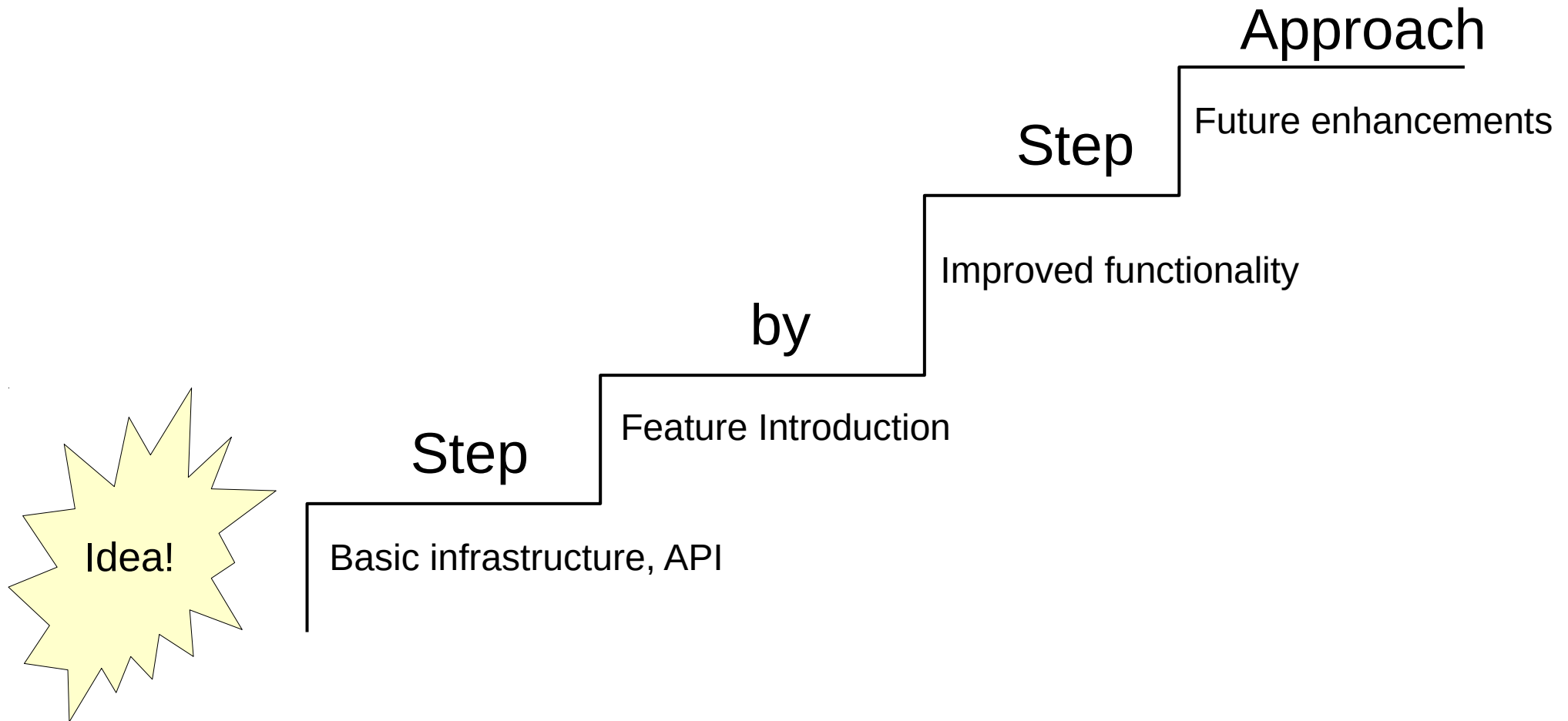


+BIG RUSSIAN Enterprise !

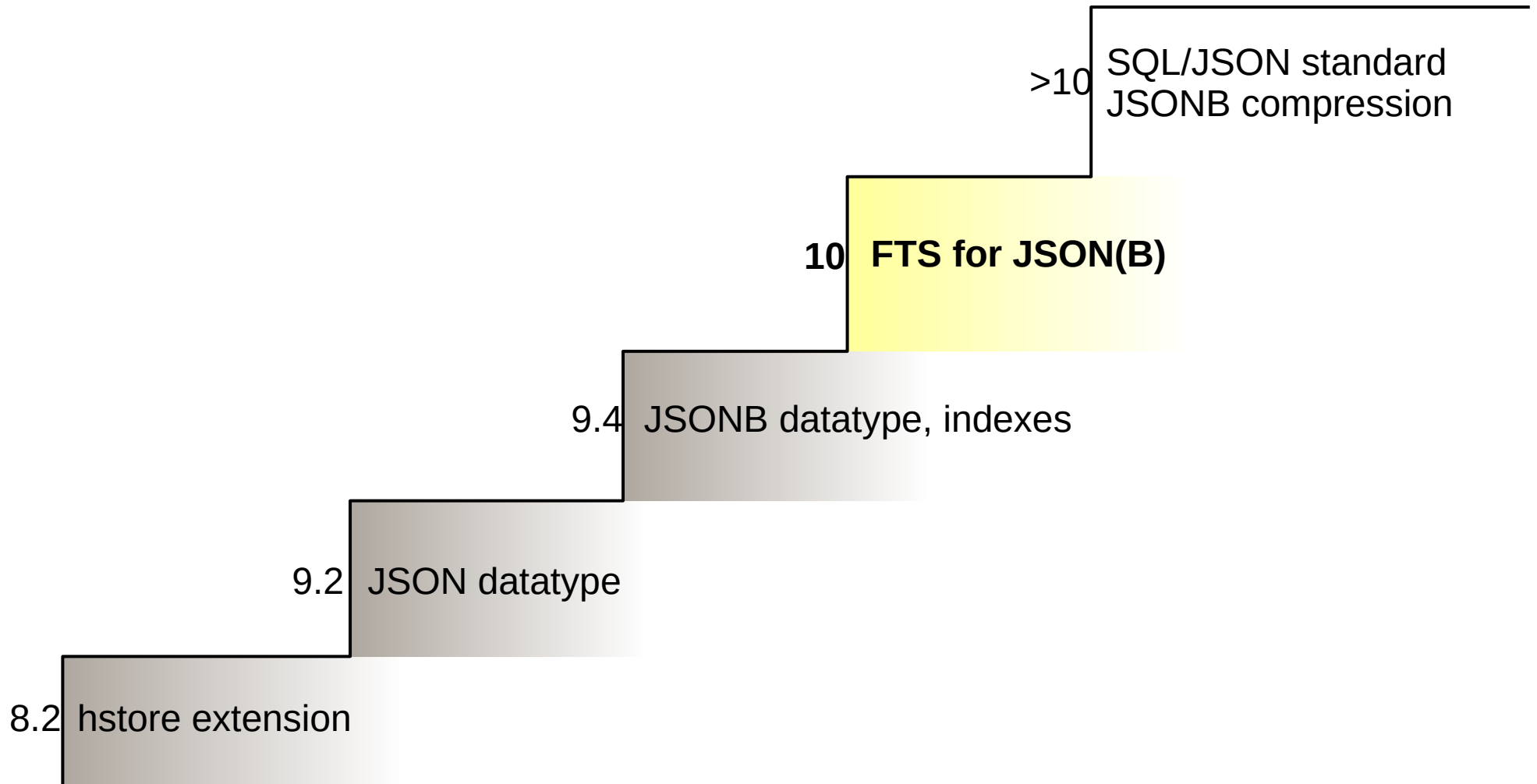
PostgreSQL Release Cycle



Major feature evolution



JSON Roadmap



- New version numbering
- DBA visible changes
- Logical Replication
- Native Table Partitioning
- Improved Query Parallelism
- Performance improvement
- Quorum Commit for Sync Replication
- Assorted improvements

New version numbering

- Postgres version numbering was always weird
 - Check <https://www.postgresql.org/docs/10/static/release.html>
0.01, 0.02, 0.03, 1.0, 1.01,...,1.02.1, 1.09 - Postgres95
6.0,...,6.5.3
7.0,...,7.4.30
8.0,...,8.4.22
9.0,...9.6.5
 - Generally version looks like major1.major2.minor1, difficult to decide which major number to advance
 - 6.0 — PostgreSQL, Postgres95 was known as Postgres Release 5
commit 9b41da6ce48e3bed6730faa6347a5461175cff83
Author: Bruce Momjian <bruce@momjian.us>
Date: Wed Dec 11 00:28:15 1996 +0000
Rename postgres95 to PostgreSQL. Add comment for SELECT NULL
 - 7.0 — really usable server (FK, SQL 92 JOIN, better optimizer)
 - 8.0 - Microsoft Windows Native Server
 - 9.0 — Built-in replication
- Now version numbering is simple: major.minor
- Expect 10.0 release Sep 25, 2017
- Next major release will be 11.0

- Fool-tolerance
 - Directories pg_xlog to pg_wal, pg_clog to pg_xact,
 - References „xlog“ → «wal»
 - log_directory (for log files) from pg_log to log
 - Use «lsn» instead of «location» for example, pg_xlog_location_diff → pg_wal_lsn_diff
- HASH indexes must be rebuilt after pg_upgrade
- ICU library (--with-icu, ICU4C needed), stable collation support

DBA visible changes 2/3

- `wal_level = replica` - supports `pg_basebackup` new default to include required WALs
 - `max_wal_senders = 10`, `max_replication_slots = 10`
 - `wal_level` can be one of {minimal, replica (replaced archive and hot_standby), or logical}
- Replication in `pg_hba.conf`
 - Allow replication connections from localhost by a user with the replication privilege (^^Gitlab). Lines below were commented before.

```
local    replication    all                                     trust
host     replication    all             127.0.0.1/32    trust
host     replication    all             ::1/128         trust
```

DBA visible changes 3/3

- password_encryption is md5 (on, default)
 - #password_encryption = **md5**, scram-sha-256
 - no plain, no UNENCRYPTED option in CREATE/ALTER USER, --unencrypted option removed from createuser command
- + ssl_dh_params_file (Diffie-Hellman parameters)
- - {create,drop}lang (create/drop extensions) contrib/tsearch2
- + idle_in_transaction_session_timeout = 0 # in milliseconds, 0 is disabled
- - sql_inheritance = on
- Better commenting importance of fsync = on (eat my data off)
 - # flush data to disk for crash safe (turning this off can cause unrecoverable data corruption)

Logical Replication

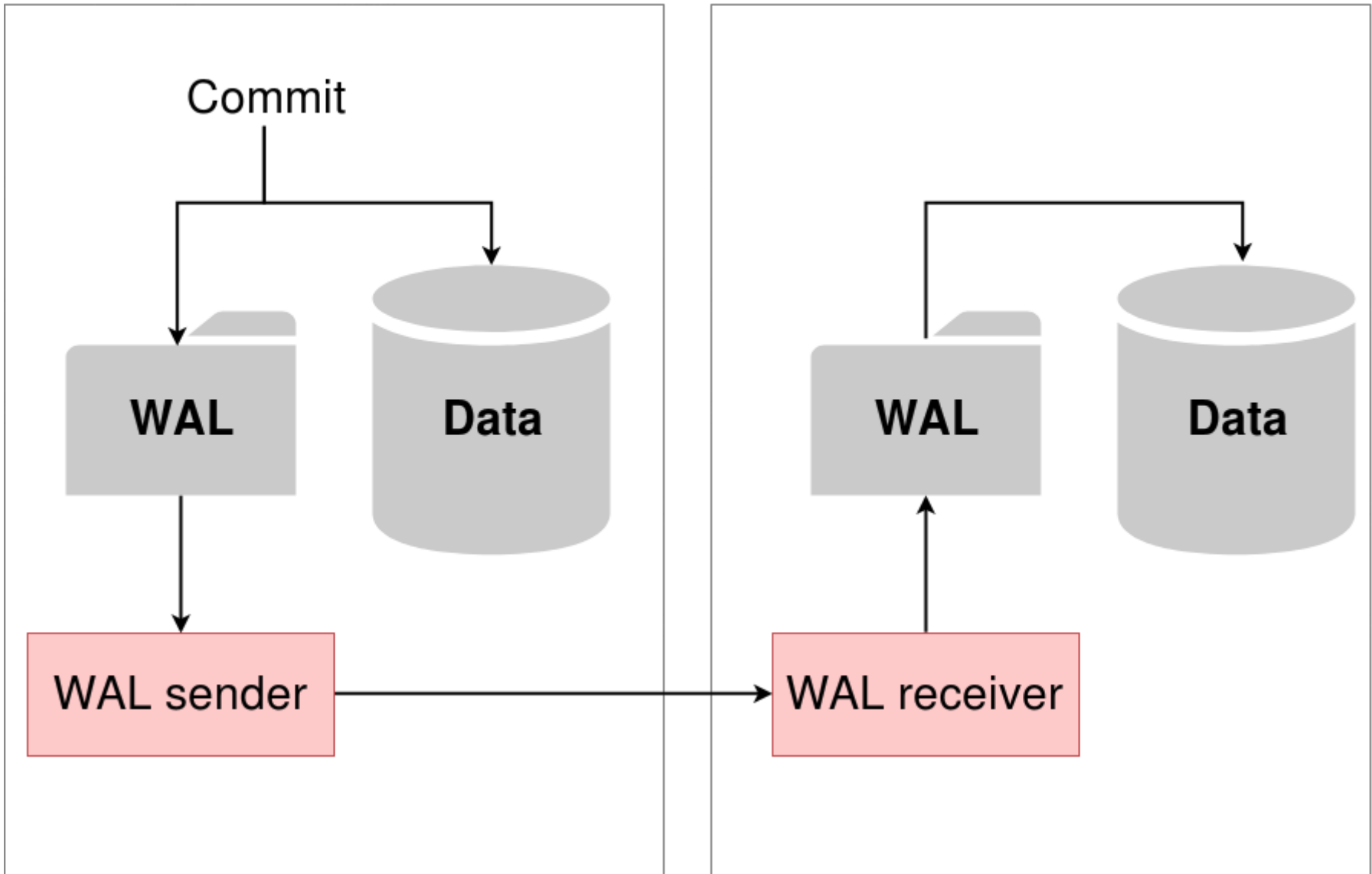
- 9.4: logical decoding of WAL records
- 10.0: functionality (some) pglogical extension was ported to the core, added SQL interface
- Big step to multimaster

- Partial replication (individual objects)
 - One-to-many, many-to-one
- Replication between different versions (starting from 10.0), different platforms (Linux to Windows)
- Upgrades (with minimal or no downtime)
- Write operations on secondary servers are possible

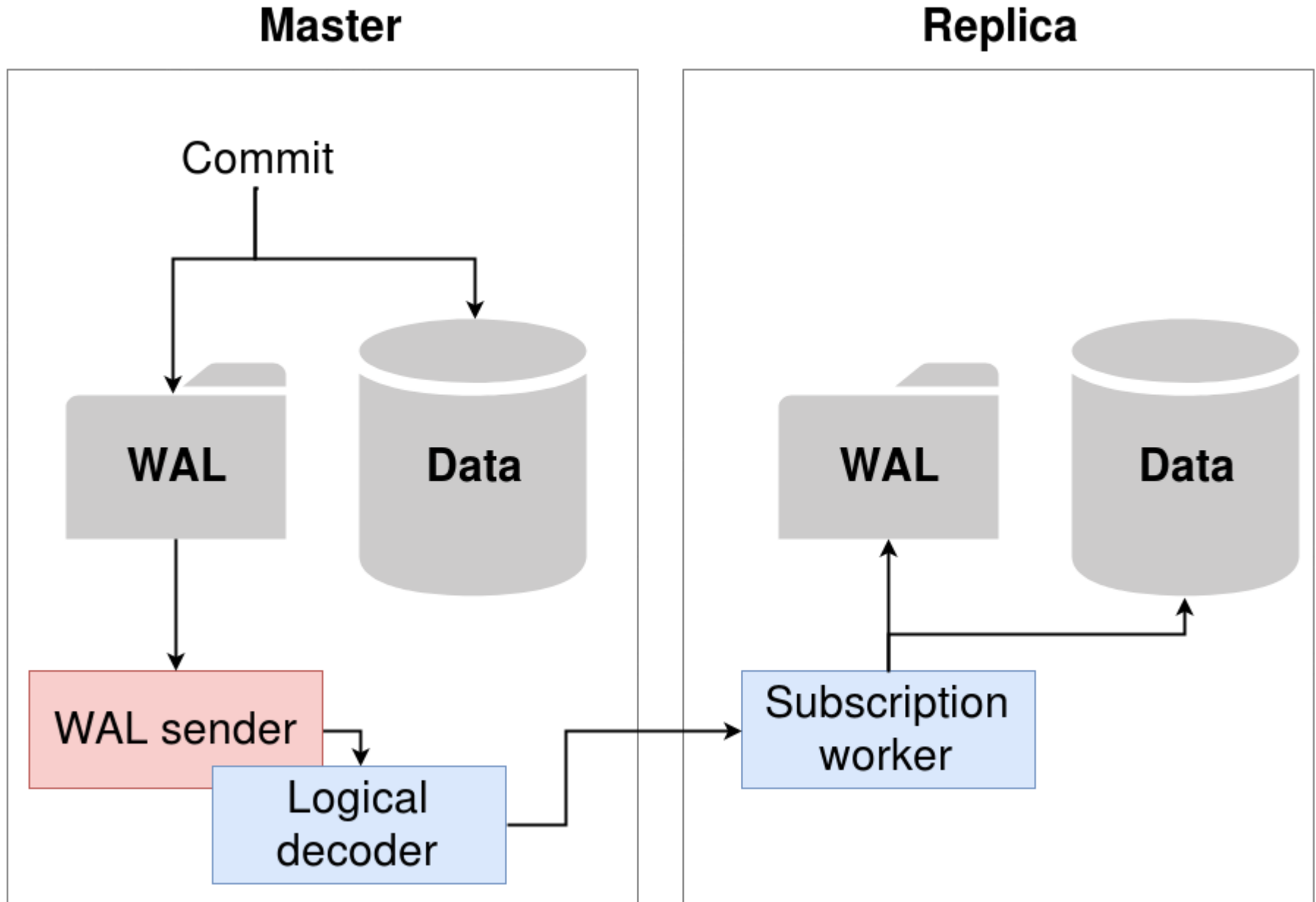
Physical Replication

Master

Replica



Logical Replication



- SQL interface:

```
ON MASTER: wal_level = logical
```

```
CREATE/ALTER/DROP PUBLICATION name  
  [ FOR TABLE [ ONLY ] table_name [ * ] [, ...]  
  | FOR ALL TABLES ]  
  [ WITH ( publication_parameter [= value] [, ... ] ) ]
```

```
WITH (publish = 'insert, delete')
```

```
ON SECONDARY:
```

```
CREATE/ALTER/DROP SUBSCRIPTION subscription_name  
  CONNECTION 'conninfo'  
  PUBLICATION publication_name [, ...]  
  [ WITH ( subscription_parameter [= value] [, ... ] ) ]
```

```
WITH ( copy_data = false )
```

Логическая репликация

«Master»

port 5432, database 'test'

```
CREATE TABLE  
test(x int PRIMARY KEY);
```

```
INSERT INTO test VALUES(1);
```

```
CREATE PUBLICATION mypub  
FOR TABLE test;
```

«Replica»

port 5433, database 'test'

```
CREATE TABLE test(x int  
PRIMARY KEY);
```

```
CREATE SUBSCRIPTION mysub  
CONNECTION 'dbname=test  
port=5432' PUBLICATION  
mypub;
```

```
SELECT * FROM test;
```

1

Logical Replication

- Limitations in 10.0
 - does not replicate schema/DDL
 - does not replicate sequences
 - does not replicate TRUNCATE
 - only supports replicating base (normal) table to base table
- Wait for the next releases !

Table partitioning

- Before 10.0: table inheritance + constraint exclusion
- Manual setup, slow for partitions pruning
- 10.0: still table inheritance+metadata
- Declarative syntax, still slow for partition pruning
- But, metadata makes possible to improve planner in future releases !
- `pg_pathman` for really fast partitioning
 - It doesn't uses table inheritance
 - It demonstrate how fast could be native partitioning - orders of magnitude faster (for 500 partitions)
 - https://github.com/postgrespro/pg_pathman

Table Partitioning

- Declarative Partitioning provides SQL syntax for:
 - range and list partitioning, Multi-level partitioning
 - Attach/detach partitions, creating partitions as foreign tables
 - Fast tuple routing

By range:

```
CREATE TABLE t1(created timestamp)
  PARTITION BY RANGE(EXTRACT(YEAR FROM created));
CREATE TABLE t1_2017
  PARTITION OF t1 FOR VALUES FROM (2017) TO (2018);
```

By list:

```
CREATE TABLE t2(category text)
  PARTITION BY LIST(category);
CREATE TABLE t2_books
  PARTITION OF t2 FOR VALUES IN ('books');
```

Table Partitioning

- Limitations:
- Need to manually create indexes on partitioned tables
- No automatic creation of partitions
- No routing tuples to foreign partitions
- No splitting or merging partitions

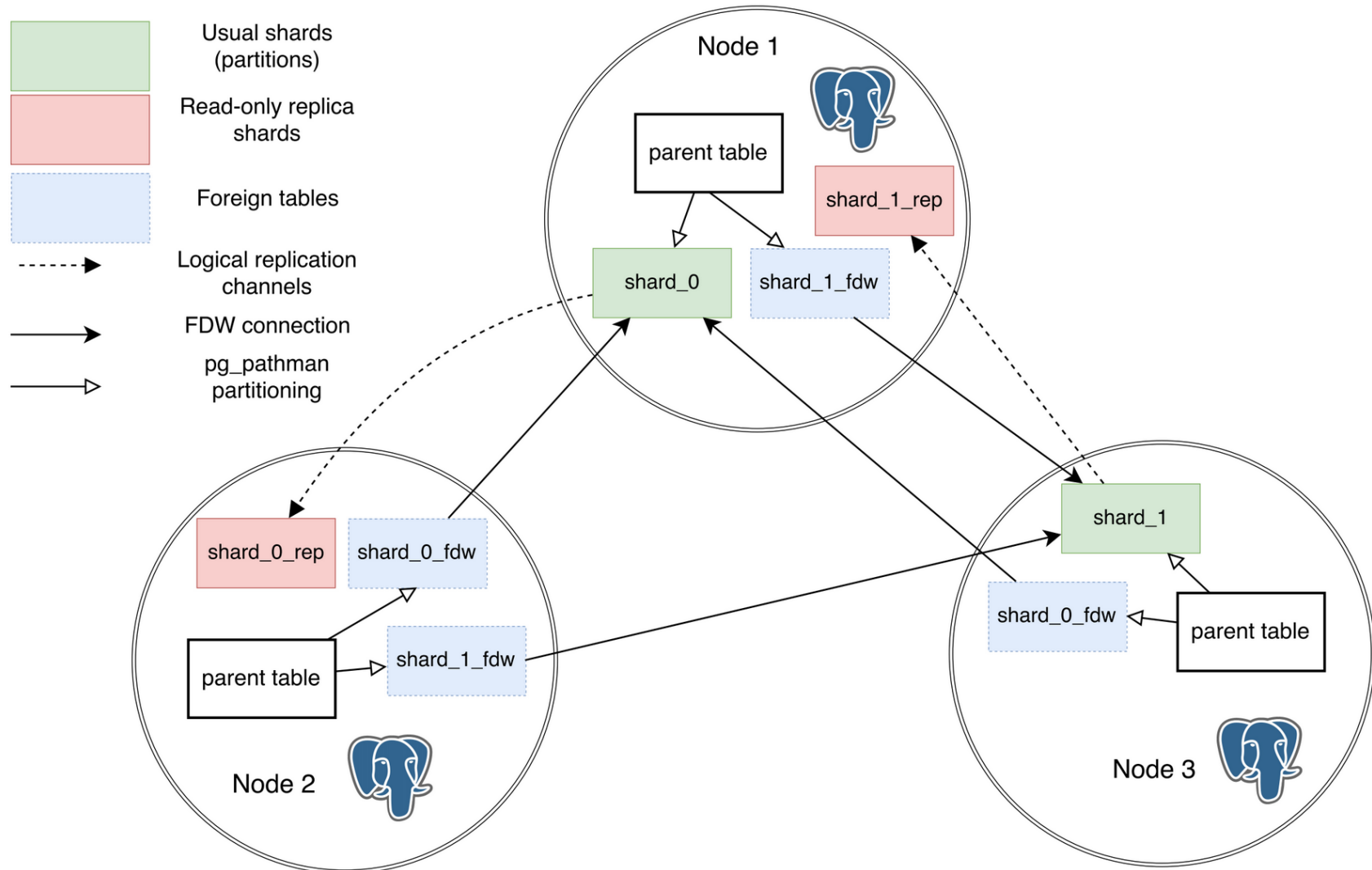
Future improvements:

- Add hash partitioning
- Global index
- Sharding

Sharding

pg_shardman: sharding via pg_pathman, postgres_fdw and logical replication

3 nodes, 2 shards, one replica per shard



- 9.3: Infrastructure
 - background workers
- 9.6: Feature introduction
 - Parallel sequential scans
 - Hash joins
 - Nested Loops
- 10.0: Improvement
 - Bitmap heap scans
 - Index scans
 - Merge joins
 - Subqueries
- >10.0: Even better
 - Create index
 - Parallel Append

Performance improvements

- Faster expression evaluations in executor
 - Currently benefit is about 6-20%
 - But it made possible future JIT-ing (expected several times improvements)

Transition tables

- Complete SQL standard for AFTER triggers

```
CREATE [ CONSTRAINT ] TRIGGER name { BEFORE | AFTER | INSTEAD OF } { event [ OR ... ] }
ON table_name
[ FROM referenced_table_name ]
[ NOT DEFERRABLE | [ DEFERRABLE ] [ INITIALLY IMMEDIATE | INITIALLY DEFERRED ] ]
[ REFERENCING { { OLD | NEW } TABLE [ AS ] transition_relation_name } [ ... ] ]
[ FOR [ EACH ] { ROW | STATEMENT } ]
[ WHEN ( condition ) ]
EXECUTE PROCEDURE function_name ( arguments )
```

- Referencing old and new table result sets (in function_name) using transition tables

- **OLD TABLE** (UPDATE,DELETE) – all rows before, **NEW TABLE** (UPDATE, INSERT) – all rows after

```
REFERENCING NEW TABLE AS new_table OLD TABLE AS old_table
```

```
FOR new_r in select * from new_table LOOP
  Raise notice «NEW: %» new_r;
```

- Infrastructure for automatic update of Materialized View !

- Examples:

- Depesz - Waiting for Postgresql 10

- <http://www.dataarchitect.cloud/david-fetter-cool-stuff-in-postgresql-10-transition-table-triggers/>

- 9.6: primary wait for commit confirmation from N of M
 - Priority set of N nodes with M standbys (order of standbys is important)
 - GUC variable `synchronous_standby_names`
 - `synchronous_standby_names = N(standby_1,...,standby_M)`
- 10.0: Quorum Commit
 - Quorum set of N nodes (order of standbys is not important)
 - `synchronous_standby_names = ANY N(standby_1,...,standby_M)`
 - Use `FIRST` instead of `ANY` to emulate 9.6 — this is default

- XMLTABLE (better standard, infrastructure for json_table)
- Durable HASH indexes
- FDW aggregate pushdown

```
db01=# explain (analyze, verbose) SELECT name, count(*) FROM t_test GROUP BY 1;
```

```
-----  
Foreign Scan (cost=107.31..137.93 rows=200 width=40)  
(actual time=192.244..192.245 rows=1 loops=1)  
Output: name, (count(*))  
Relations: Aggregate on (public.t_test)  
Remote SQL: SELECT name, count(*) FROM public.t_test GROUP BY name  
Planning time: 0.063 ms  
Execution time: 192.581 ms  
(6 rows)
```

- Transaction traceability
 - txid_status(BIGINT) — useful to recover from indeterminate COMMIT.
 - <https://blog.2ndquadrant.com/traceable-commit-postgresql-10/>
- pg_stat_activity
 - More wait events: client reads/writes, server reads/writes and fsync ops, synchronous replication
 - Worker processes, WAL senders and more

Assorted Improvements 2/2

- Extended Statistics - Functional Dependencies, Multivariate N-Distinct Counts

- CREATE STATISTICS stname (dependencies, ndistinct) ON col1, col2,... FROM tablename;

```
CREATE TABLE test (id int, data int);  
CREATE STATISTICS test_stats (dependencies) ON id, data FROM test;
```

- Security Technical Implementation Guide (STIG DoD), 1st OSS database
- RLS (permissive +restrictive) — policies can be AND-ed
- Better (than md5) authentication - SCRAM-SHA-256
- FTS for JSONB
(<https://obartunov.livejournal.com/194683.html>)
- +many (>100) features

References

- Documentation: [Release Notes](#) for version 10
- Postgres Wiki: [New in Postgres 10](#)
- Bruce Momjian: [Major Features: Postgres 10](#)
- Robert Haas: [New Features Coming in PostgreSQL 10](#)
- Michael Paquier: [Postgres 10 highlight ... series](#)
- Hubert (depesz) Lubaczewski: [Waiting for PostgreSQL 10 ... series](#)
- Robert Haas: [Parallel Query v2](#)
- Robert Haas: [Partitioning plans for v11](#)
- Simon Riggs: [News and Roadmap for BDR](#)
- Petr Jelinek: [Logical Replication in PostgreSQL 10](#)

Several Postgres groups are working on



Postgres Distributed

Postgres Vectorized

Postgres Parallel

Postgres Asynchronous

Postgres Extendable+

Postgres NoSQL — check SQL/JSON

<http://sqlfiddle.postgrespro.ru:6081/#!21/>

Postgres Scalable (Vertical & Horizontal)

Conclusions

- PostgreSQL is the universal database with clear roadmap
- Proven technology of developing major features
- Postgres 10 is a big step in product evolution





Thanks !