Pgpool-II: Past, Present and Future

SRA OSS, Inc. Japan
Tatsuo Ishii
Yugo Nagata

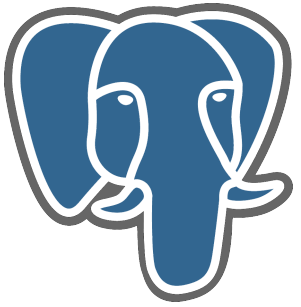
SRA OSS, INC.

Who are we?

- Tatsuo Ishii
  - Pgpool-II Community Lead
  - PostgreSQL Committer
  - SRA OSS, Inc. Japan Branch Manager
- Yugo Nagata
  - Pgpool-II Developer
  - Working on PostgreSQL consulting and support at SRA OSS, Inc. Japan

About SRA OSS, Inc. Japan

- Started PostgreSQL support and other OSS businesses in 1999. Established as SRA OSS in 2005
- Our businesses include:
  - Supoort, consulting and deployment of PostgreSQL, Zabbix and other OSS
  - Postgres Plus
  - PowerGres
  - PostgreSQL Training



PowerGres

ZABBIX

CERTIFIED PARTNER

EDB

ENTERPRISEDB

SRA OSS, INC.

27th June 2003: the birth of pgpool

- Only connection pooling and fail over were implemented
- Up to 2 PostgreSQL servers
- Only version 2 protocol is supported (version 3 protocol = PostgreSQL 7.4 was not released yet)
  - 4,719 lines in C

Palaeoloxodon naumanni (acient elephant)

Friday 27th June 22:54:46 JST
[pgsql-jp: 30256] A connection pool server for PostgreSQL: pgpool

Hi,

I have created a connection pool server for PostgreSQL for any programing language including PHP. Although it is still under development, please try it if you like.

ftp://ftp.sra.co.jp/pub/cmd/postgres/pgpool/pgpool-0.1.tar.gz
(note: this URL is still valid!)

Of course pgpool is under open source license, similar to the BSD license of PostgreSQL.

The reason why I developed pgpool is, we cannot use connection pooling with PHP.

I know PHP has "persistent connection" which caches the connection to DB, but it creates as many as the number of Apache process and it stresses DB. Pgpool will limit the number of connections to DB and improve the performance in this regard.
(Original text was in Japanese)

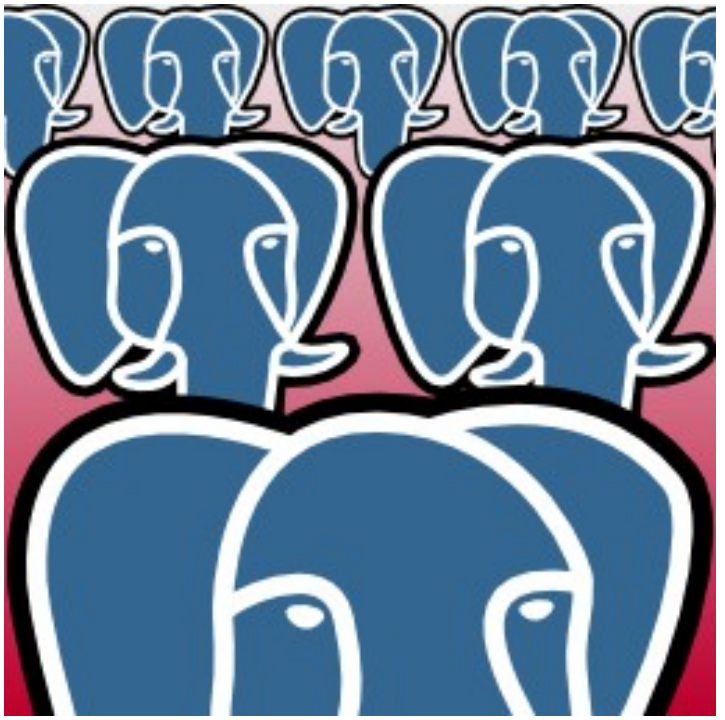
April 2004:pgpool 1.0

- Implemented “native replication mode”(at that time PostgreSQL does not have replication)
- Deal with canceling query and large objects
- 5,890 lines in C
- Random minor release policy (e.g. add new functions to minor releases)

Became modern elephant but toddling

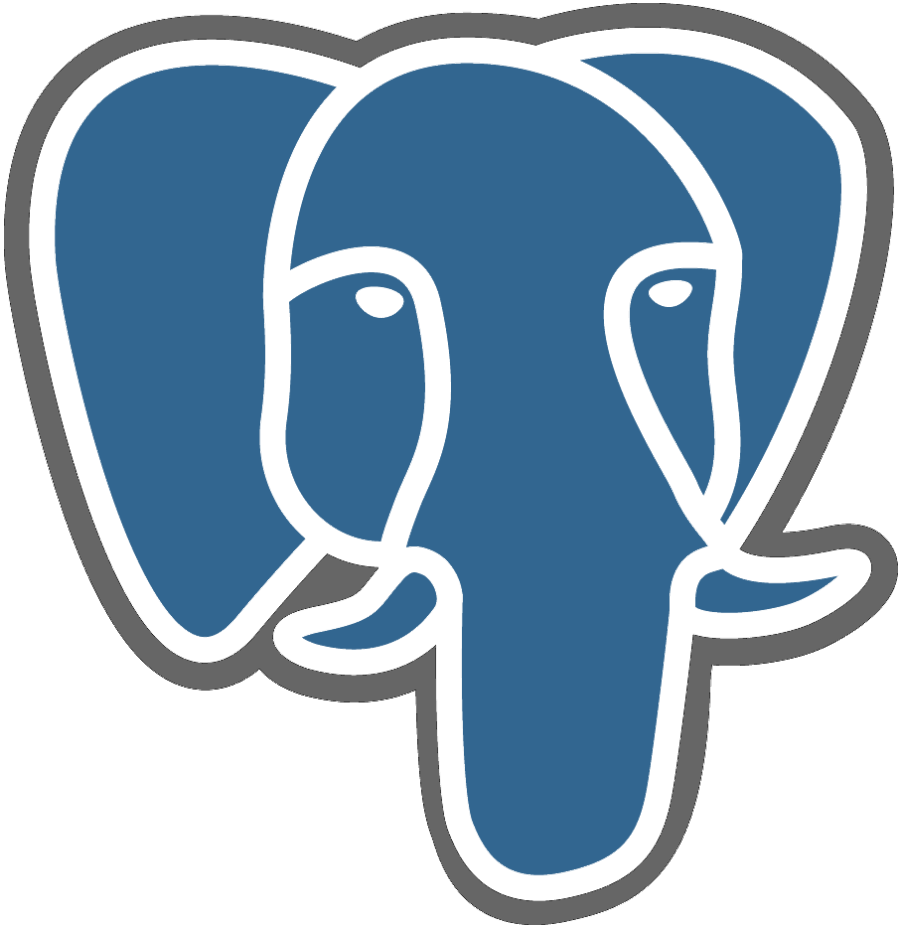
Evolved to pgpool 2.0

- June 2004 merely two months after 1.0
- I seemed to work hard :-)
- Deal with V3 protocol
- 7,750 lines in C
- Feburary 2005 2.5 was released. Health checking and master slave mode added2.5
  - the last release as “pgpool”



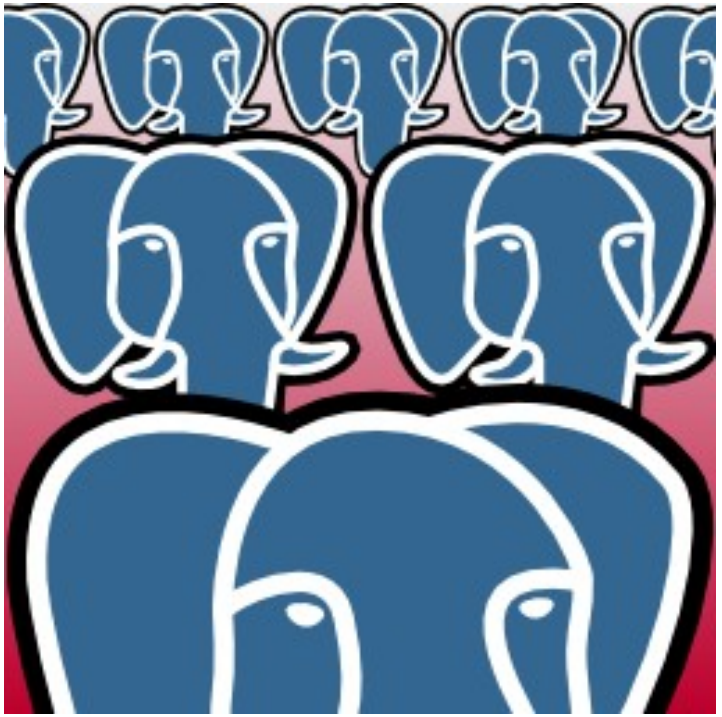
PostgreSQL at that time

- PostgreSQL 7.4 (November 2003)
  - V3 protocol was introduced, and protocol level prepared statement was available
  - autovacuum
  - Full text search contrib
  - Not ported to Windows yet
  - No built-in replication



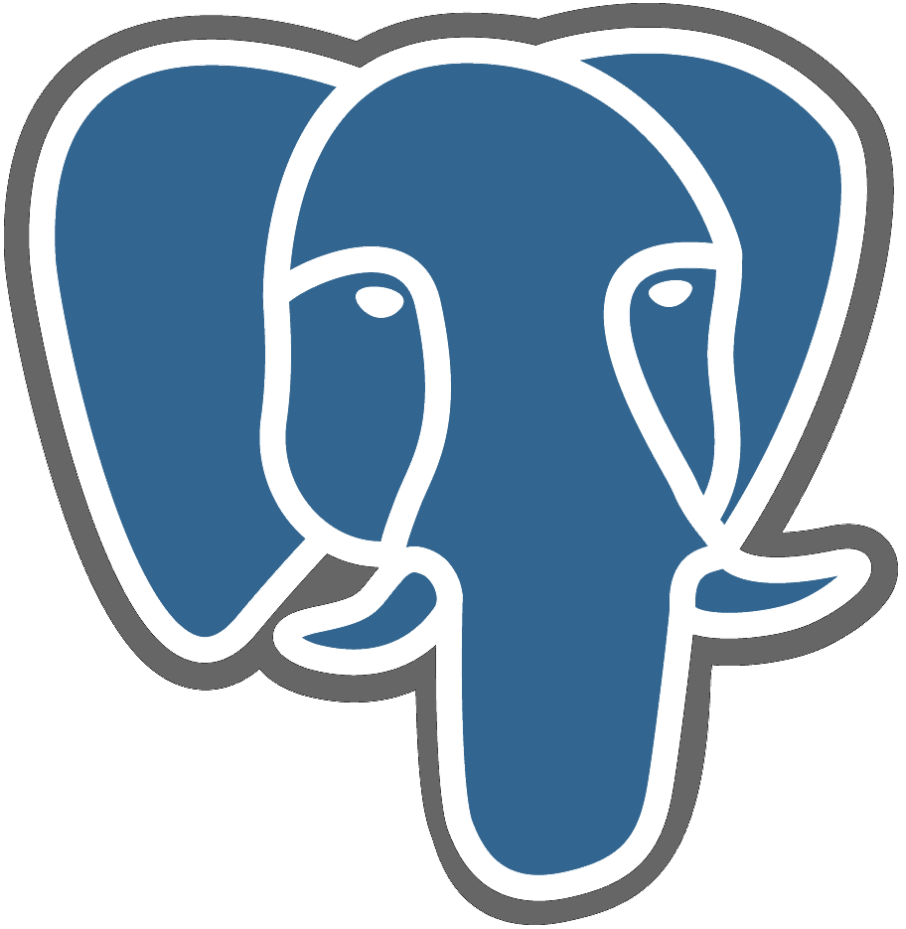
September 2006:pgpool-II

- Team project, rather than a personal project of pgpool
  - Grant from IPA
- Lots of new functions
  - Up to 128 PostgreSQL servers possible
  - SQL parser imported from PostgreSQL
  - pgpool-II control command (PCP)
  - A GUI tool (pgpoolAdmin) added
  - Parallel query
  - 73,511 lines in C, 10 times bigger than pgpool



PostgreSQL at that time

- PostgreSQL 8.2 (2006)
  - Native Windows port (8.0)
  - PITR (8.0)
  - Save point (8.0)
  - Table spaces (8.0)
  - Two phase commit (8.1)
  - Bitmap index scan (8.1)
  - Row locks (8.1)
  - No built-in replication



November 2011:Moving to pgpool.net

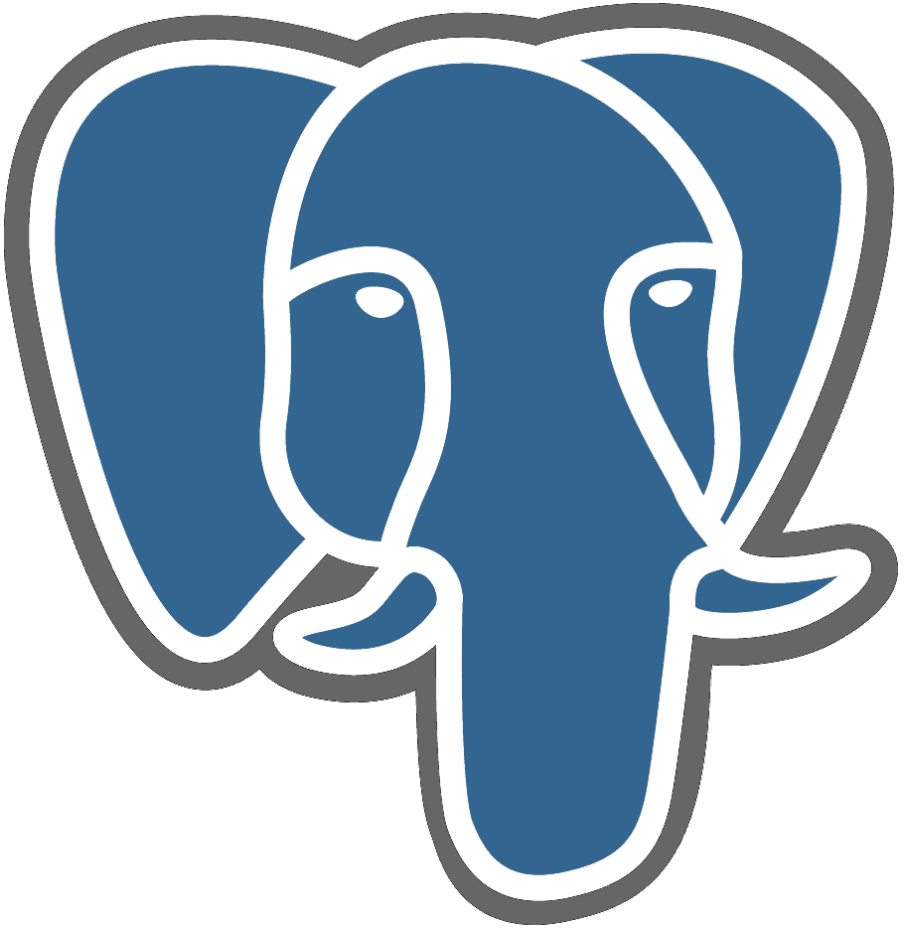
- Until that time hosted in pgfoundry. But unstable...
- Decided to have new site pgpool.net
- Moved from CVS to git
  - Great help from French PostgreSQL community. Thanks!

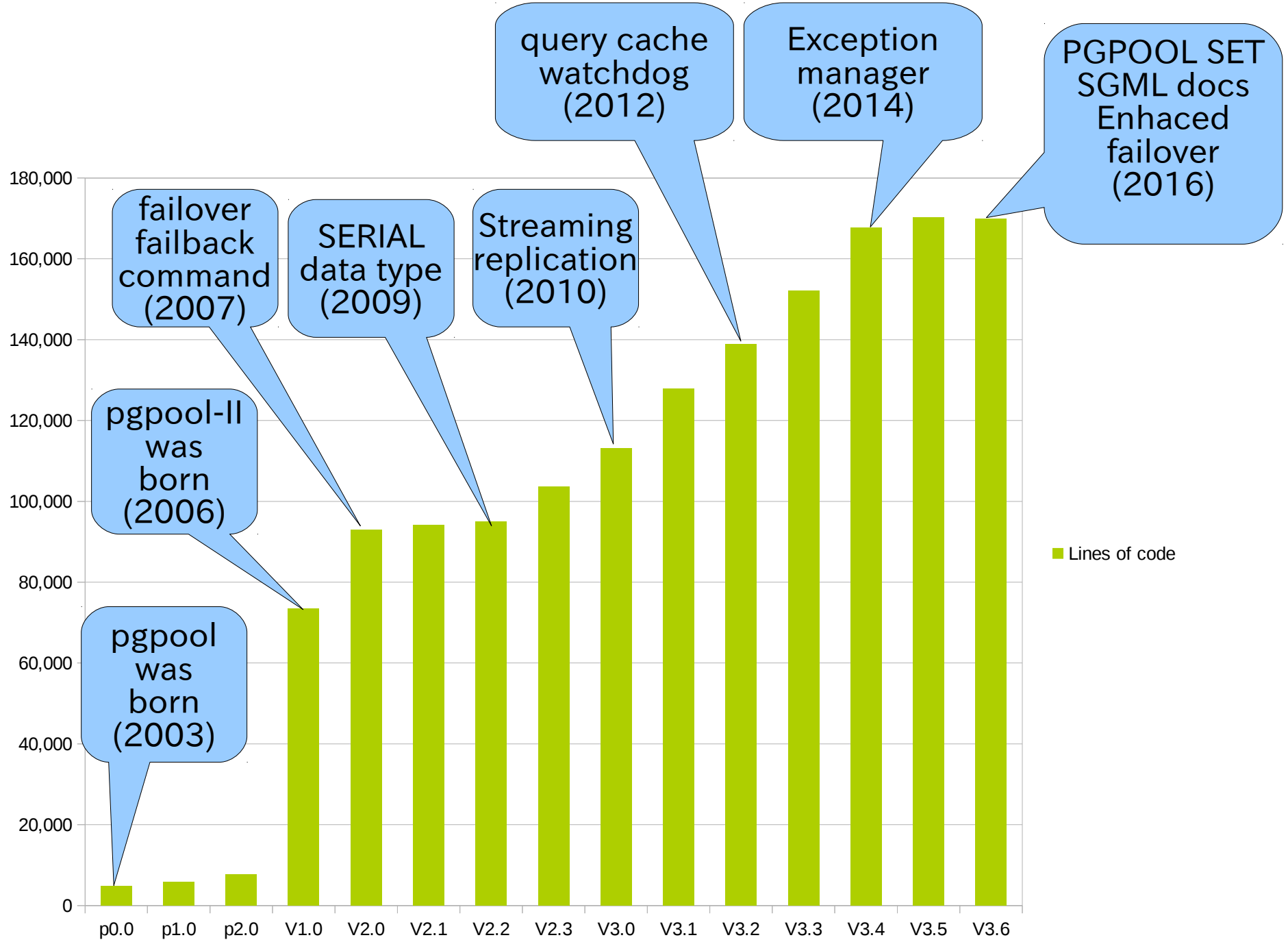
Moving is not easy...

- In pgpool.net we provide info in English and Japanese (if no time to write in Japanese, English info is provided at least.

PostgreSQL at that time

- PostgreSQL 9.1(2011)
  - Streaming replication(9.0)
  - Windows 64bit (9.0)
  - pg_upgrade (9.0)
  - Synchronous replication (9.1)
  - Foreign tables (9.1)
  - Recursive query (8.4)





Current development team

- Tatsuo Ishii (Japan)
  - Organizing the team. Writing codes
- Ahsan Hadi (Pakistan)
  - User relationship, benchmarking
- Muhammad Usama (Pakistan)
  - Watchdog and others. Committer
- Yugo Nagata (Japan)
  - In charge of watchdog. Committer
- Nozomi Anzai (Japan)
  - In charge of pgpoolAdmin and the installer. Committer
- Peng Bo (China)
  - Release engineering and RPM packaging and others. Committer

Current status of pgpool-II

- Total clustering management tool for PostgreSQL
  - Managing streaming replication
  - Query dispatching to primary and standby servers
  - Load balancing
  - Managing fail over
  - In memory query cache
- High availability of pgpool-II itself
  - watchdog

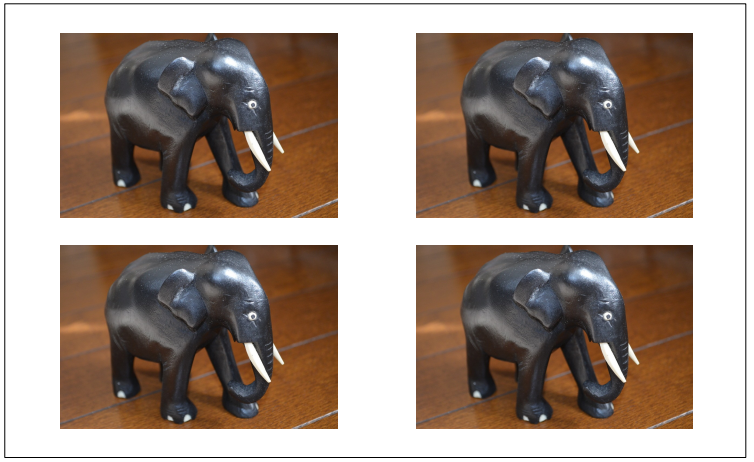
Major functions of Pgpool-II

Performance	Connection pooling Load balancing In memory query cache
High availability	automatic fail over fail over script follow master script watchdog
Cluster management	On line recovery
Application and the cluster relationship	Query dispatching

Introduction to Pgpool-II

A herd of PostgreSQL

- An elephant is powerful
- However, a herd of elephants is even more powerful!
- Problem is, how to manage the herd of elephants?



Problems of the herd of elephants include:

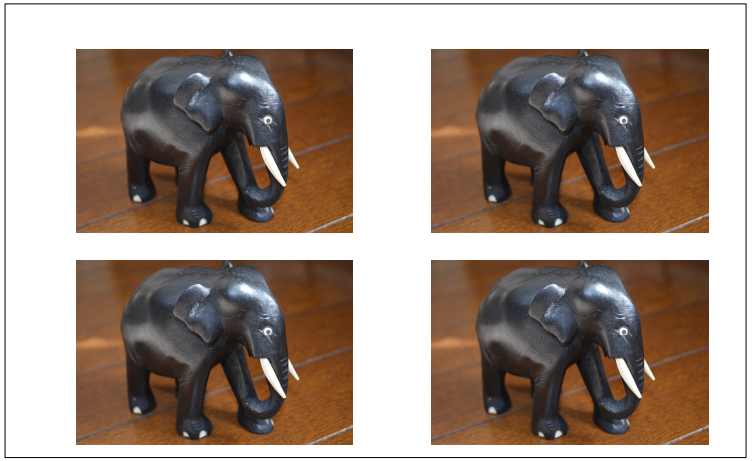
- It needs a leader (a primary server)
- If the leader retires, a new leader must take over its role (fail over, promote)
- Other elephants must follow the new leader
- If a non leader elephant cannot continue to work, it must retire and leave the herd (standby fail over)
- If a new elephant wants to join the herd, it should be accepted without disturbing the herd
- Elephants should help each other to perform a task in an efficient way (load balancing)
- There are tasks which can only be performed by the leader (write queries – needs query dispatching)

Solving the problems by using Pgpool-II

- By using Pgpool-II, a streaming replication PostgreSQL cluster almost looks like a single PostgreSQL server
- User supplied “fail over script”could define which standby server should take over when the primary server goes down
- User supplied “follow master command” allow other standbys follow the new primary server
- If a standby server goes down, it is removed from the cluster definition and clients can continue to use the cluster
- Pgpool-II examines each query. If the query is a read only one or not. If not, it is forwarded to one of servers (load balancing).
- If a query is a write query, it will be forwarded to the primary server

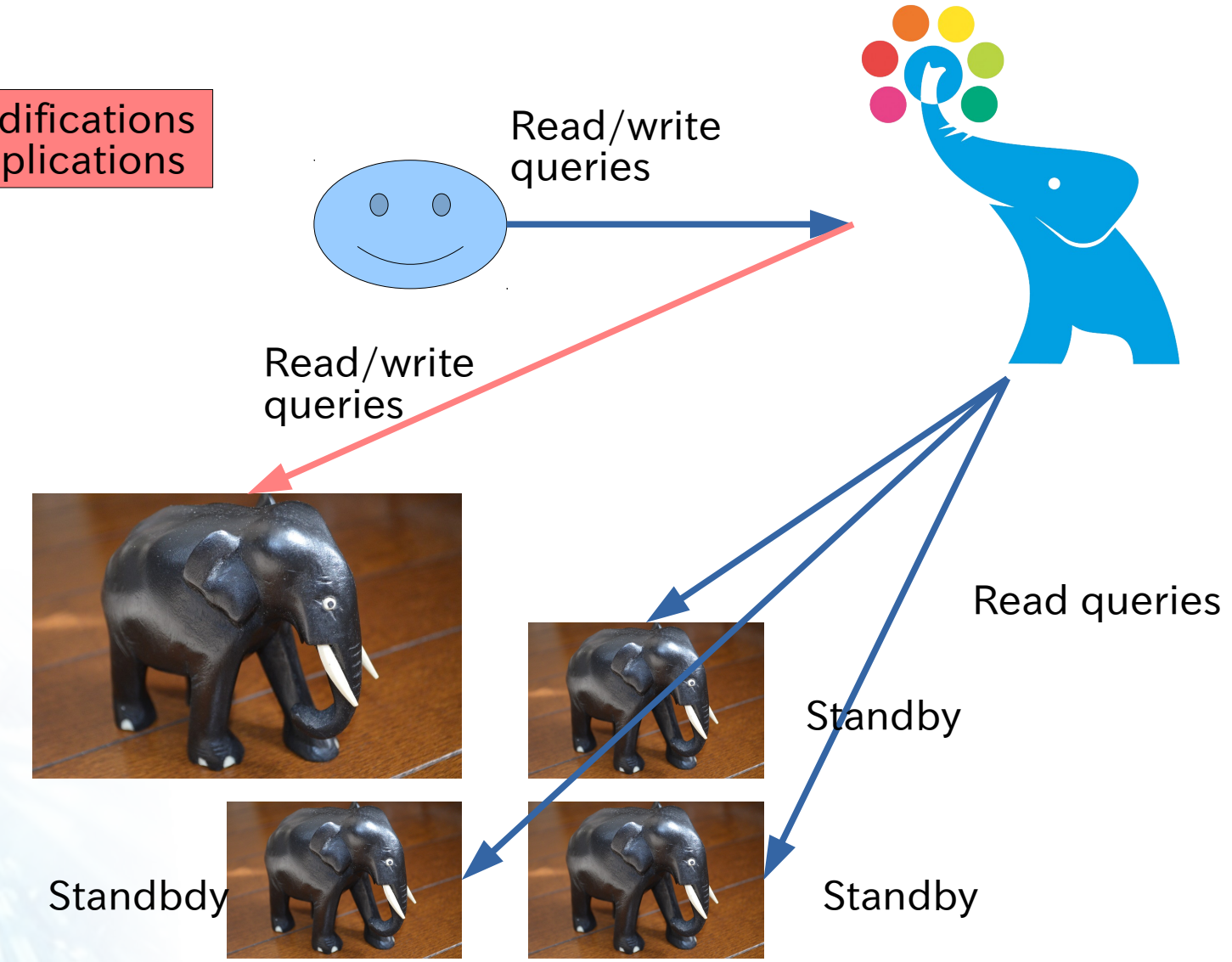
Pgpool-II

A herd of elephants looks like single big elephant

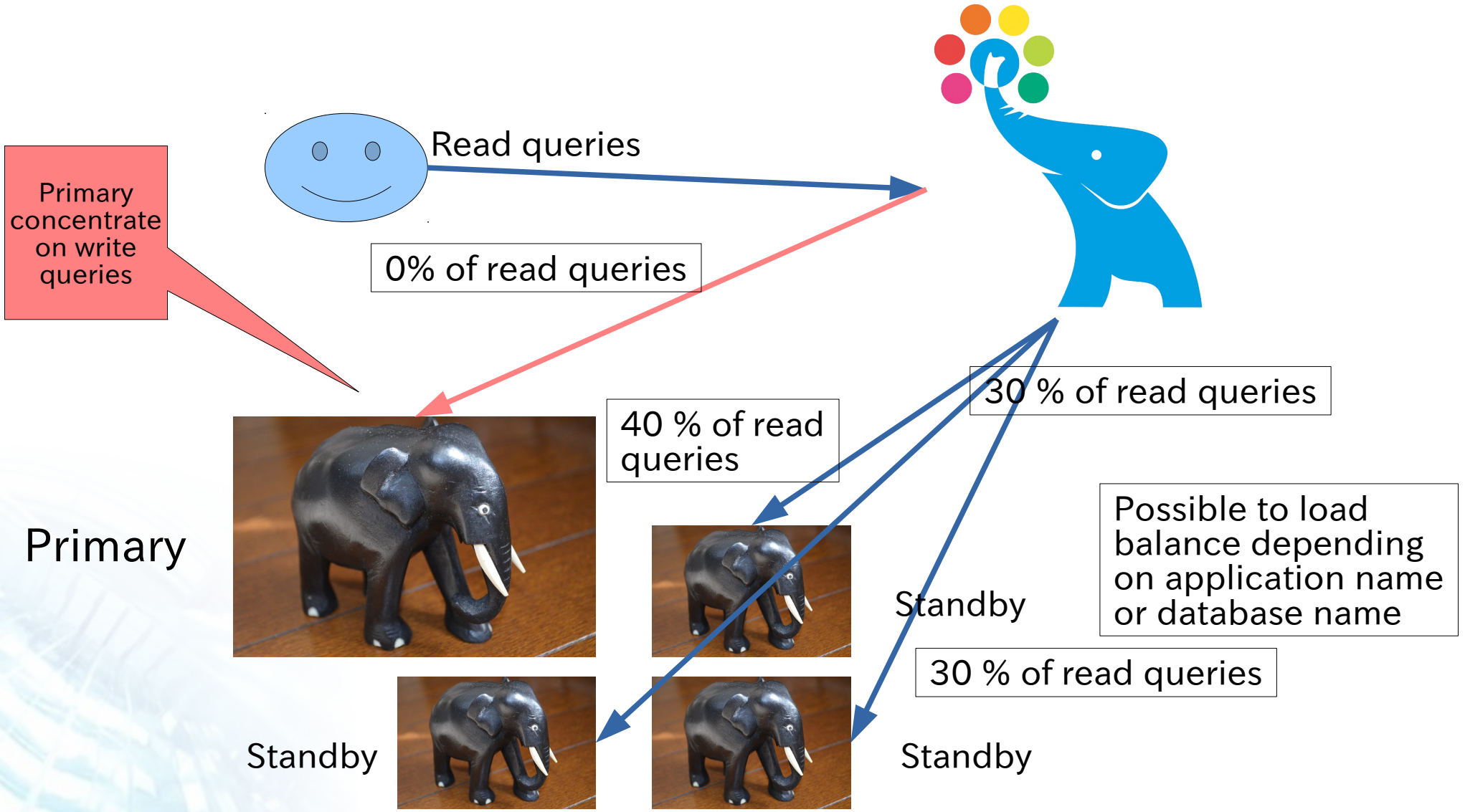


Query dispatching/routing

Minimum modifications to existing applications



Read query load balancing



When a standby goes down

Pgpool-II removes broken standby

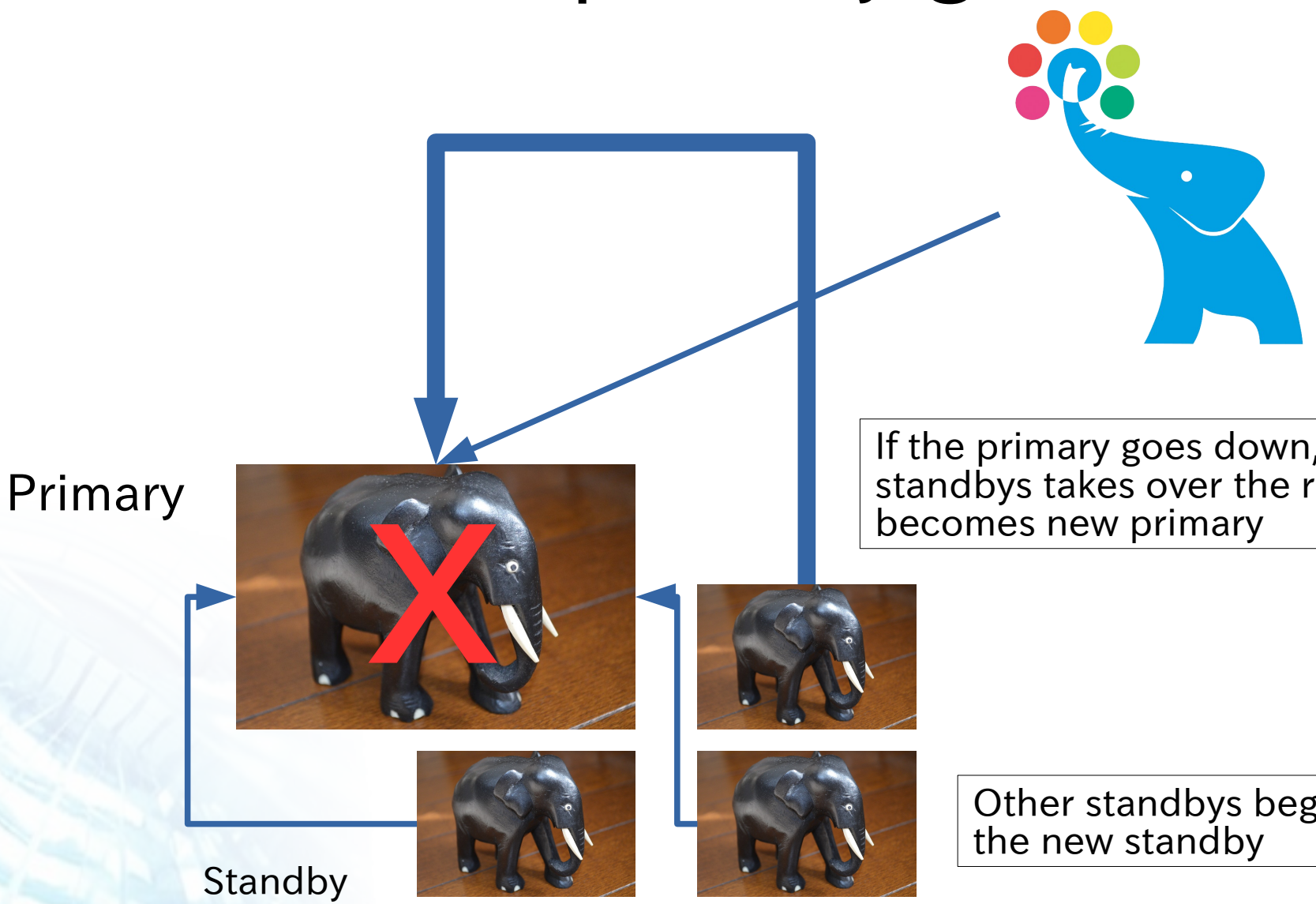
Users may need to reconnect to Pgpool-II

Primary

Standby

SRA OSS, INC.

When the primary goes down

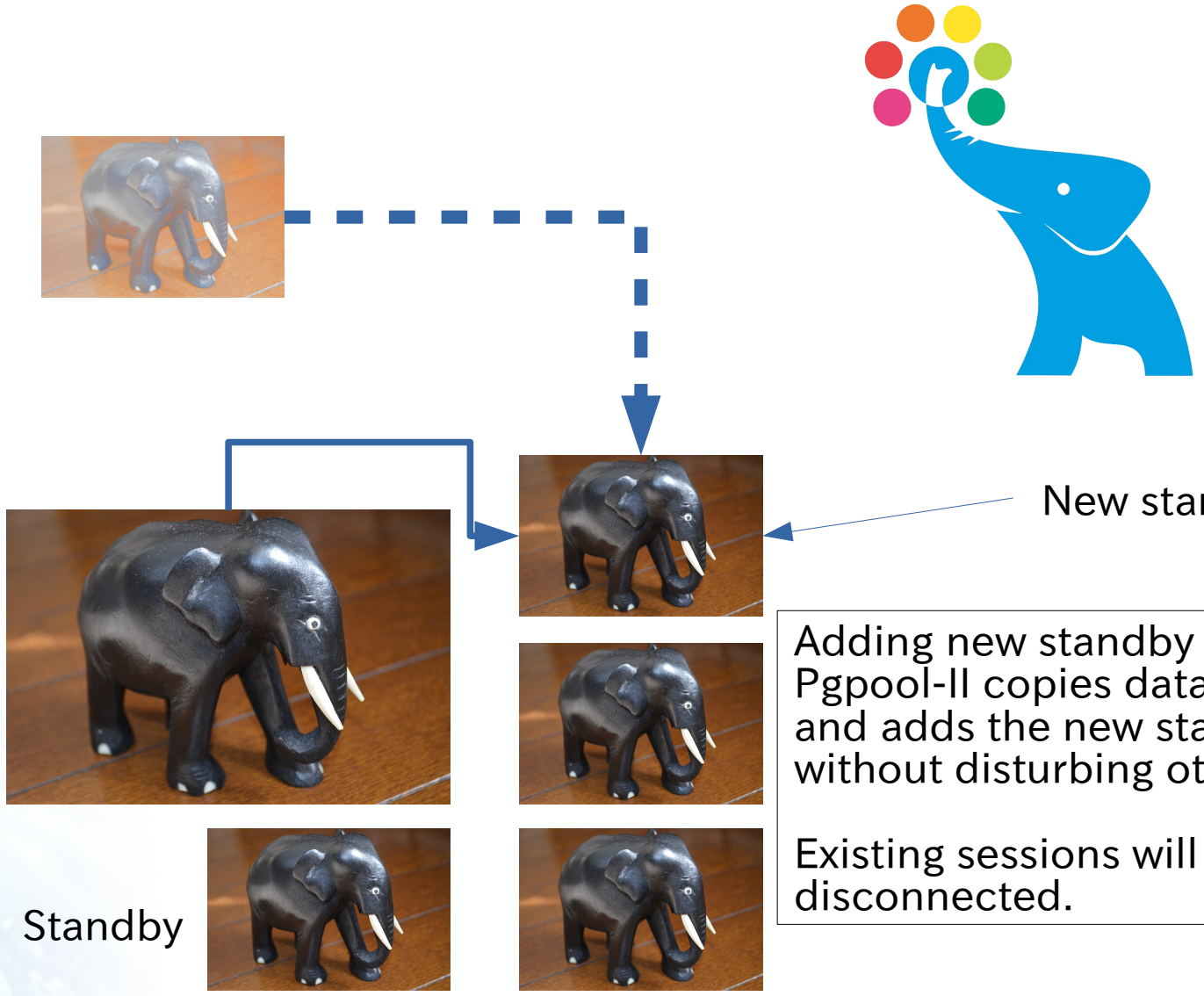


If the primary goes down, one of standbys takes over the role, and becomes new primary

Other standbys begin to follow the new standby

SRA OSS, INC.

Adding new standby



Adding new standby is easy.
Pgpool-II copies data from primary, and adds the new standby without disturbing other servers.

Existing sessions will not disconnected.

Watchdog: Built-in HA for Pgpool-II

Active Pgpool

Promotion

Standby Pgpool

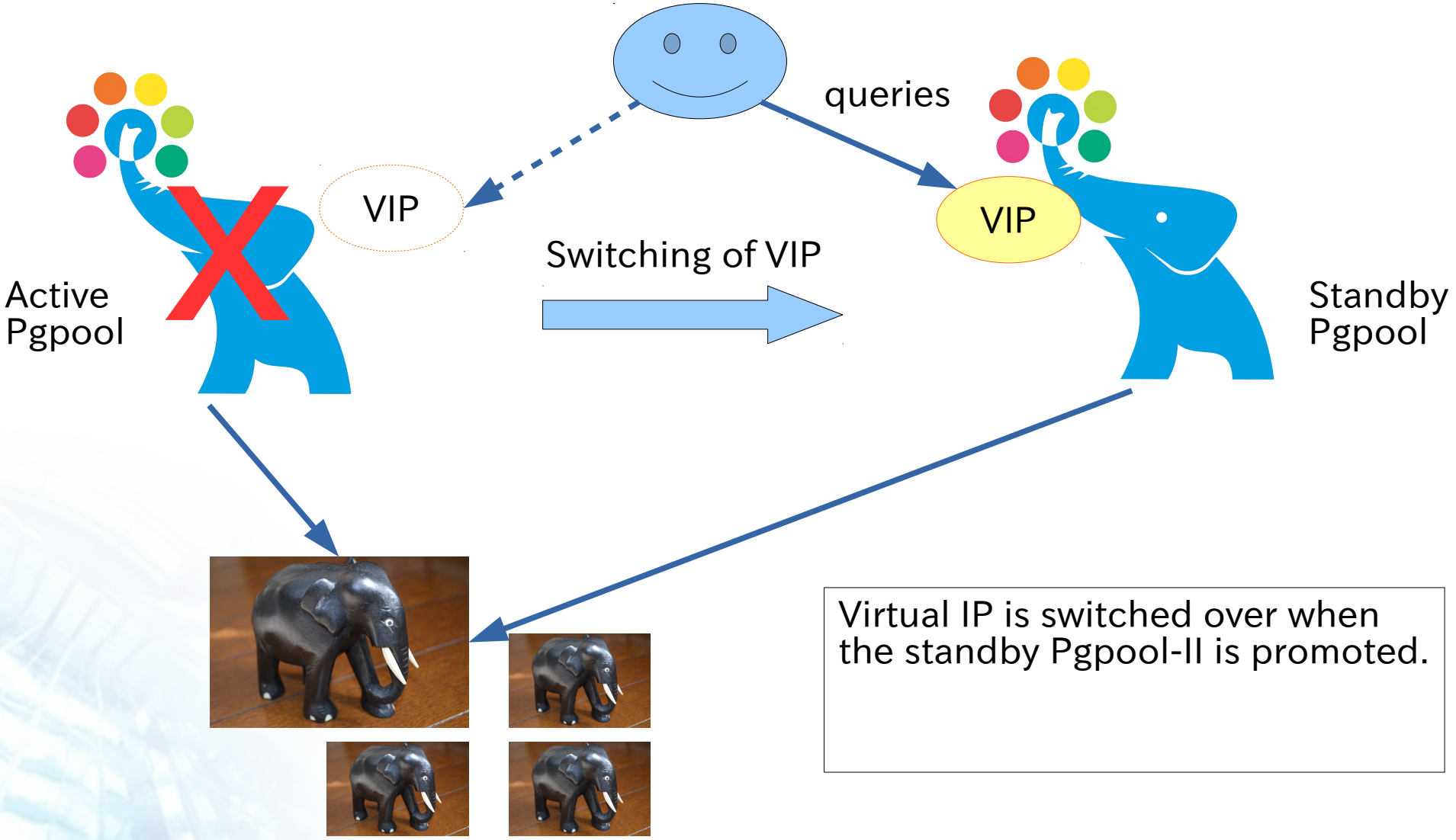
Primary

Standby

When active Pgpool-II goes down, standby Pgpool-II is promoted

SRA OSS, INC.

Watchdog: Virtual IP Switching



Virtual IP is switched over when the standby Pgpool-II is promoted.

Coordination with Watchdog

Active Pgpool

Mutual Monitoring

Information Sharing

Standby Pgpool

Primary

Standby

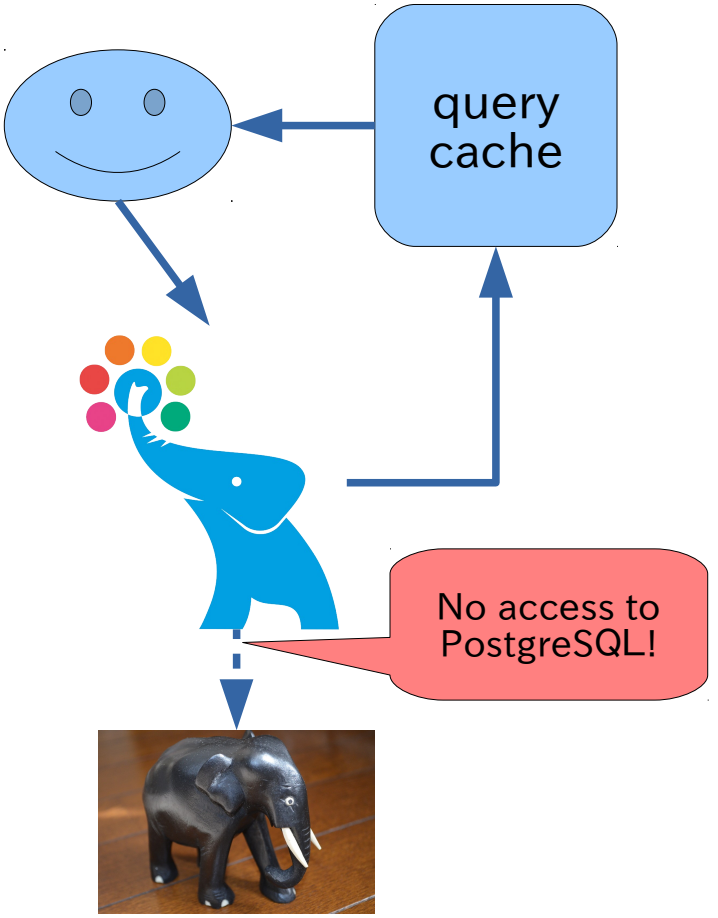
Monitoring each other's health.

Sharing information about node down with other Pgpool-II.

SRA OSS, INC.

In memory query cache

- Pgpool-II reuses a query result by using the query cache
- Query cache is fast because the cache is in memory
- No access to PostgreSQL while returning the query result using the query cache
- Cache storage has two choices: shared memory or memcached
- When a table is updated, all the cache entries referring to the table will be invalidated
- It is possible to invalidate the query cache by timeout



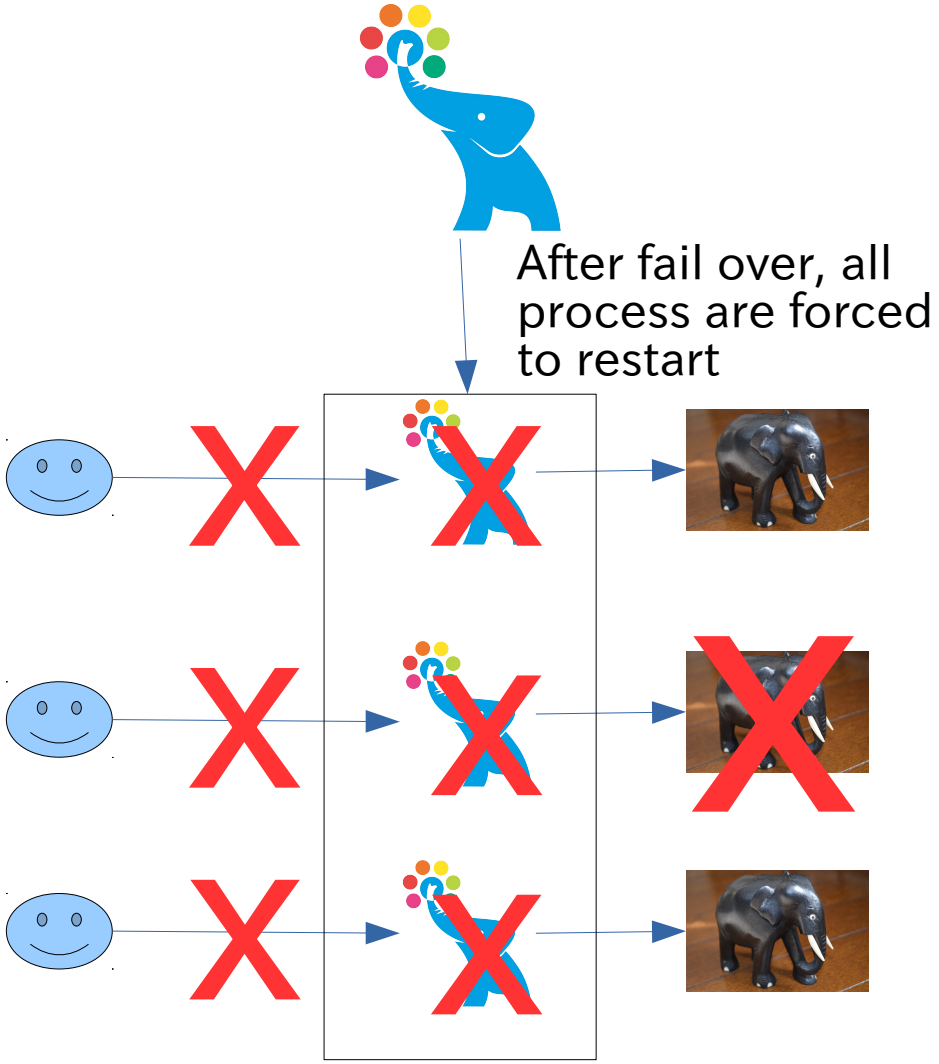
Introducing the latest version of Pgpool-II 3.6

- Enhanced fail over
- Adopting PostgreSQL 9.6
- More

Enhancing fail over

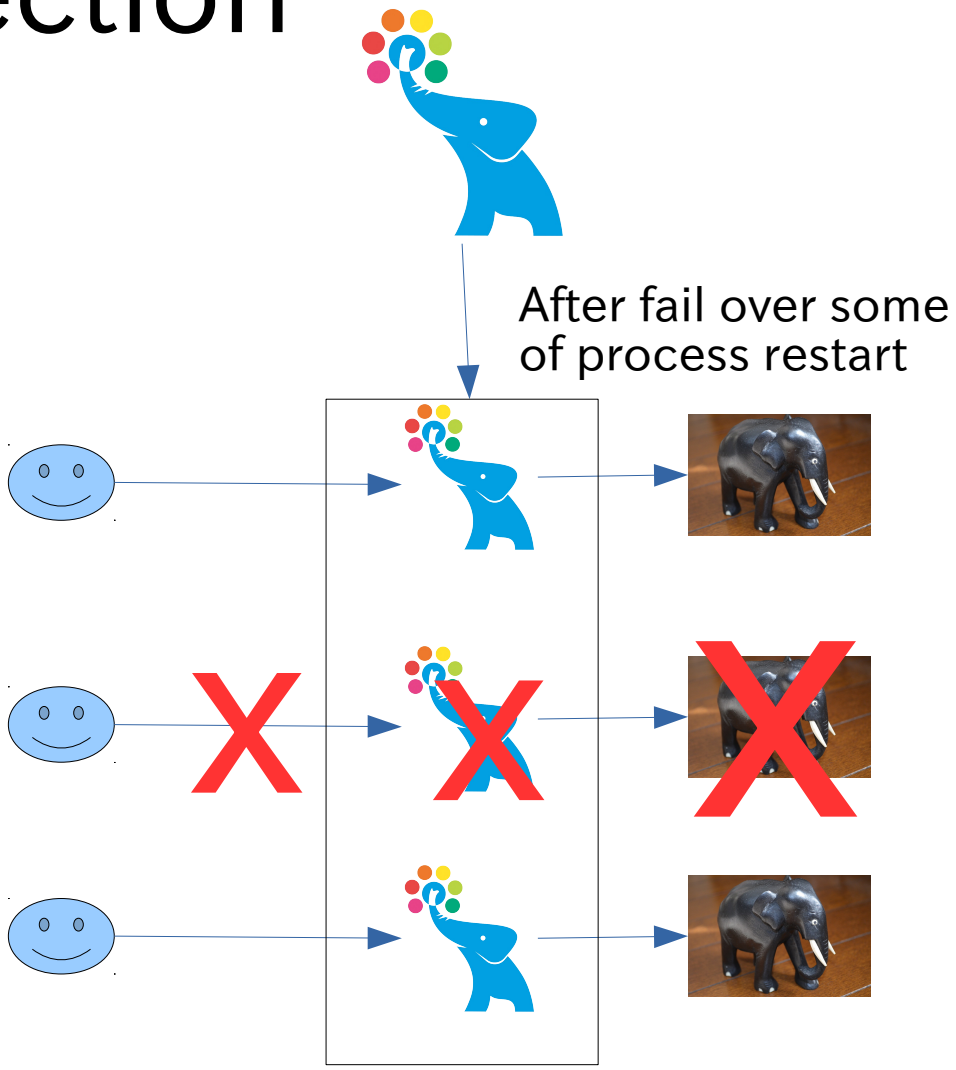
Issue of fail over

- When fail over happens, all existing session are disconnected even if a session does not use the broken database



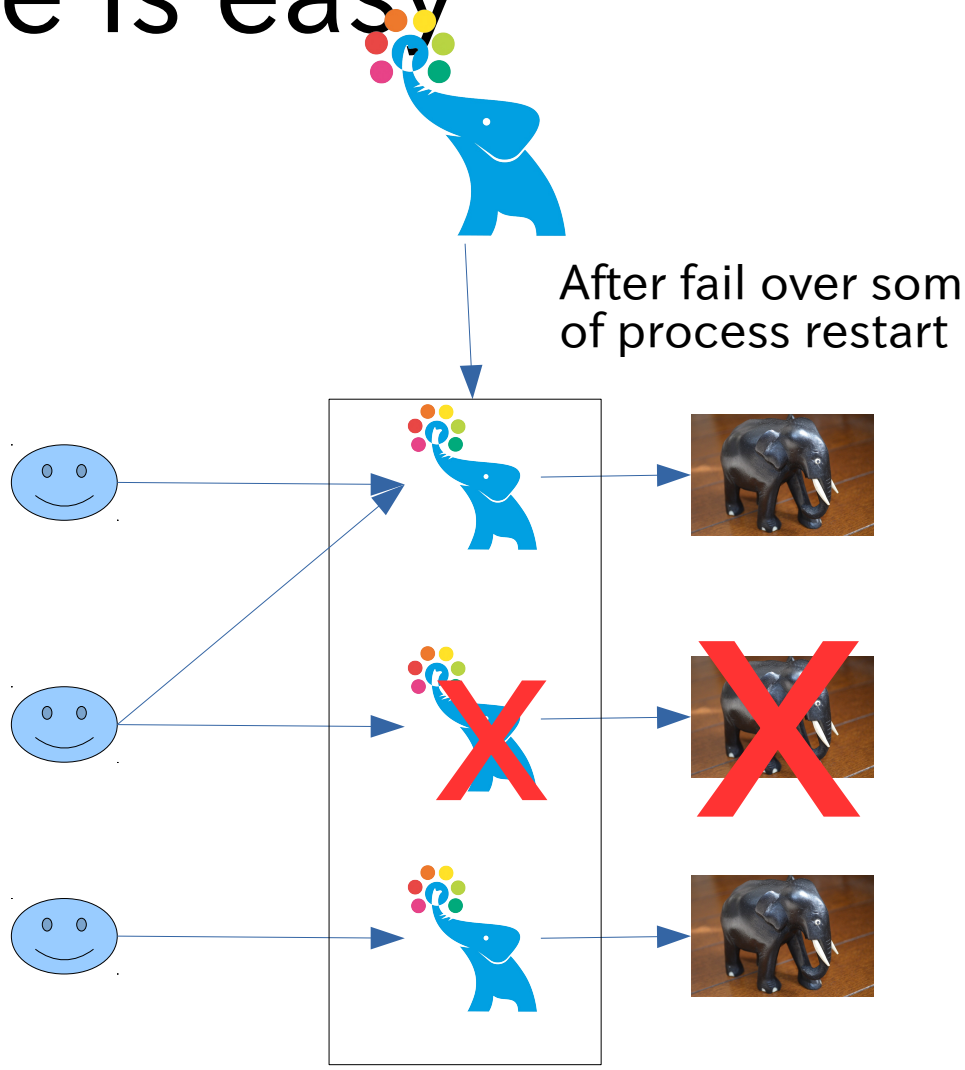
Pgpool-II 3.6 minimize the session disconnection

- Precondition：operated in streaming replication mode. The broken server is standby
- Only sessions using the downed server is forced to restart corresponding Pgpool-II child process
- In other sessions corresponding Pgpool-II child process will not restart. After the session exits, it automatically restarts to reflect the down status



Planning PostgreSQL maintenance is easy

- Set load balance weight to 0 for the DB server which is scheduled to be down beforehand
- New sessions will not use the DB
- No session is affected by the DB down



Allow to check which DB server is used for a session

- Because the load balance DB node is a session local property, it is not possible for PCP command to check it
- In 3.6 you can check the property by using “show pool_nodes”
- You could also check the replication delay

```
psql -p 11000 -c ”show pool_nodes” test
 node_id | hostname | port  | status | lb_weight |  role   | select_cnt | load_balance_node | replication_delay
---------+----------+-------+--------+-----------+---------+------------+-------------------+-------------------
 0       | /tmp     | 11002 | up     | 0.333333  | primary | 0          | false             | 0
 1       | /tmp     | 11003 | up     | 0.333333  | standby | 0          | true              | 0
 2       | /tmp     | 11004 | up     | 0.333333  | standby | 0          | false             | 0
(3 rows)
```

Adopting PostgreSQL 9.6

Importing PostgreSQL 9.6 SQL Parser

- Pgpool-II has its own SQL parser to precisely analyze SQL statements
  - the SQL parser is imported from the latest version of PostgreSQL
- New syntaxes are supported
  - COPY FROM INSERT … RETURNING TO …
  - ALTER FUNCTION … DEPENDS ON EXTENSION …
  - ALTER TABLE ADD COLUMN IF NOT EXISTS …

Pgpool-II 3.6 improvement: Others

PGPOOL SET

- Similar to PostgreSQL's “SET” command
- Allow to change some of configuration parameters of Pgpool-II in a session
  - Reset after session finises
  - Usage: “PGPOOL SET variable-name TO value”
- Following variables can be set by PGPOOL SET
  - client_idle_limit
  - client_idle_limit_in_recovery
  - log_statement
  - log_per_node_statement
  - log_min_messages
  - client_min_messages
  - log_error_verbosity
  - allow_sql_comments
  - check_temp_table
  - check_unlogged_table

PGPOOL SHOW

- Show individual Pgpool-II configuration variable (unlike “show ppool_status”)
  - Syntax: PGPOOL SHOW variable-name
- “logical group”: grouping frequently used variables
  - “backend”, “other_pgpool”(other watchdog nodes)、“heartbeat”

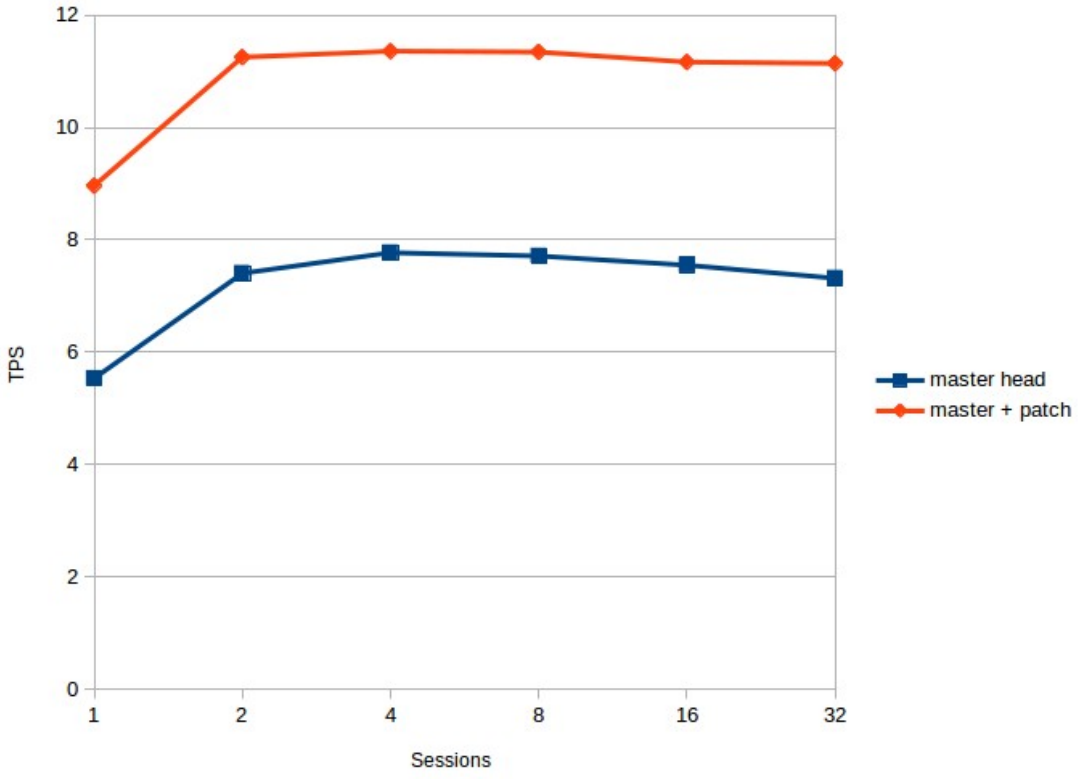
```
postgres=# pgpool show backend;
          item           |             value             |                  description
-------------------------+-------------------------------+-----------------------------------------------
 backend_hostname0       | 127.0.0.1                     | hostname or IP address of PostgreSQL backend.
 backend_port0           | 5434                          | port number of PostgreSQL backend.
 backend_weight0         | 0                             | load balance weight of backend.
 backend_data_directory0 | /var/lib/pgsql/data           | data directory of the backend.
 backend_flag0           | ALLOW_TO_FAILOVER             | Controls various backend behavior.
 backend_hostname1       | 192.168.0.1                   | hostname or IP address of PostgreSQL backend.
 backend_port1           | 5432                          | port number of PostgreSQL backend.
 backend_weight1         | 1                             | load balance weight of backend.
 backend_data_directory1 | /home/usama/work/installed/pg | data directory of the backend.
 backend_flag1           | ALLOW_TO_FAILOVER             | Controls various backend behavior.
(10 rows)
```

Dealing with pg_terminate_backend

- What is pg_terminate_backend?
  - One of PostgreSQL built-in functions
  - Allow to kill backend process by specifying process ID
  - Can be used to terminate backends fall into an infinite loop or, to grab a lock
- What's the problem with Pgpool-II?
  - Unwanted fail over because pg_terminate_backend returns exactly the same error code as PostgreSQL shutdown
- The solution
  - Examine the argument of pg_terminate_backend. If the argument is the same as the backend process ID used by Pgpool-II session, avoid fail over. Just terminate the session
- Restrictions
  - The arugument of pg_terminate_backend must be a simple integer constant
    - No good: SELECT pg_terminate_backend(pid) from strange_table;
  - Not supported if the extended protocol is used

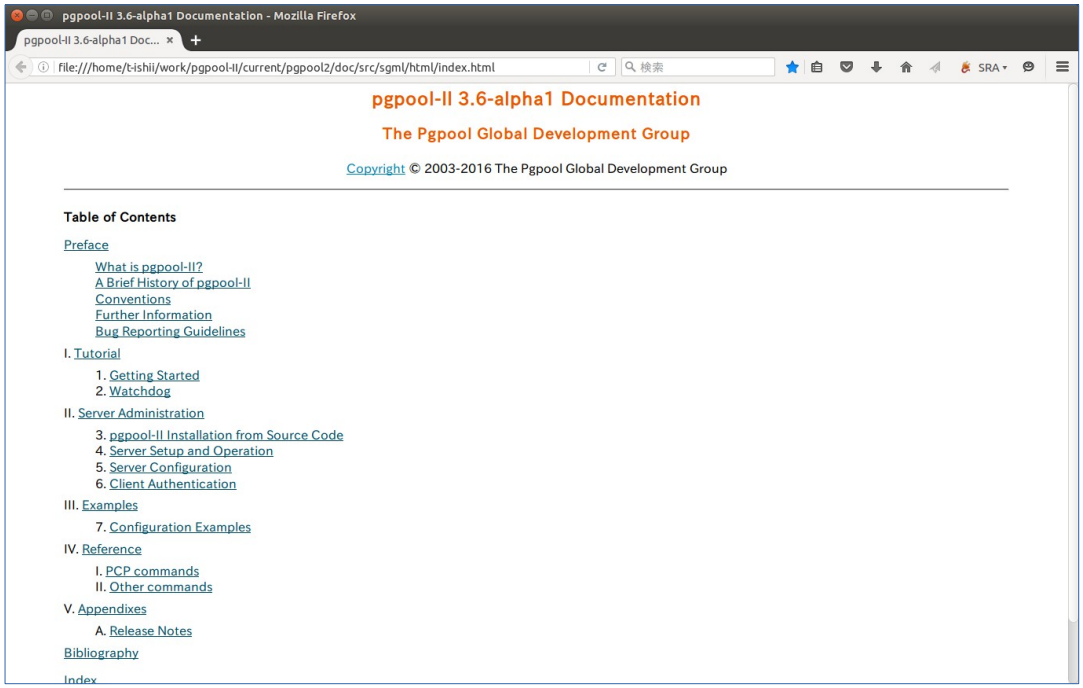
Enhance performance when many rows are returned

- Reduce the overhead while returning rows to client, enhance the performance from 47% to 62% in particular case
- Before Pgpool-II called a system call for each row. Now it does a buffering for each row instead, and call the system call at the end of process to enhance the performance



Changing document format

- Previously we were using hand waved HTML
  - Hard to maintain
  - Unable to create indexes or table of contents automatically
- Pgpool-II 3.6 employs SGML format then generates HTML, which is already done in PostgreSQL
- Expecting this brings easier maintenance, though initial work is painful



Future of Pgpool-II

- Pgpool-II will evolve and follow the progress of PostgreSQL replication technique
  - Synchronized replication
  - Cascading replication
  - Multi master replication
- Easier to manage whole cluster
  - Managing whole cluster is not an easy task
  - Pgpool-II will help the task
- Automatic fail back
  - Allow to automatically fail back standbys or newly added standbys by recognizing them
- Plug-in architecture
  - Example: Allow users to define their own load balancing logic

Conclusions

- Pgpool-II has evolved from a connection pooling software to a feature-rich clustering management software
- It started as a personal project, then migrated to an Open Source project
- Pgpool-II has been keeping pace with evolution of PostgreSQL: e.g. deal with streaming replication in addition to its native replication
- Pgpool-II will maintain the concept “Sitting close by PostgreSQL” and will keep on walking with PostgreSQL

URLs

- Pgpool-II official site
  - http://www.pgpool.net
    - Download sources
    - Git repository
    - RPMs
- Twitter
  - @pgpool2
- GitHub
  - https://github.com/pgpool/pgpool2
    - Currently it's just a source code mirror. May not respond too quickly to pull requests and issues

Thank you!