leboncoin

Large databases
lots of servers
on premises
in the cloud
GET THEM ALL!

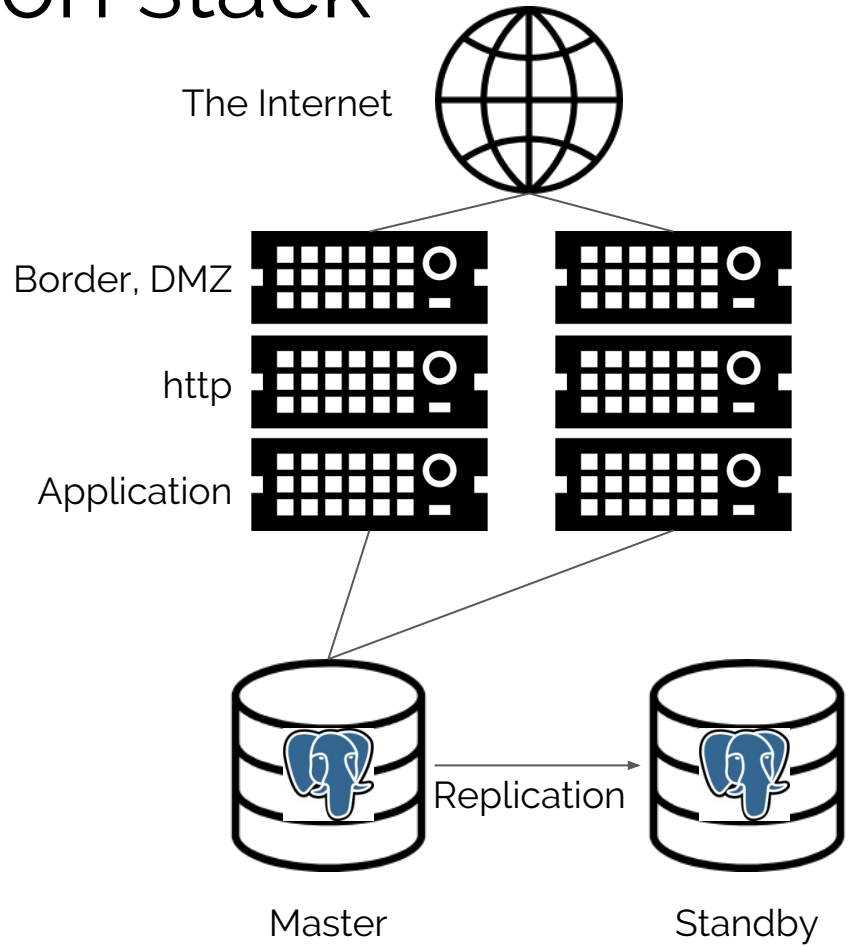
Flavio Gurgel

DBA leboncoin

pgDay Paris 2019

Mars 12, 2019

A common stack



Trouvez la bonne affaire parmi 28 351 126 petites annonces sur leboncoin.

Déposer une annonce

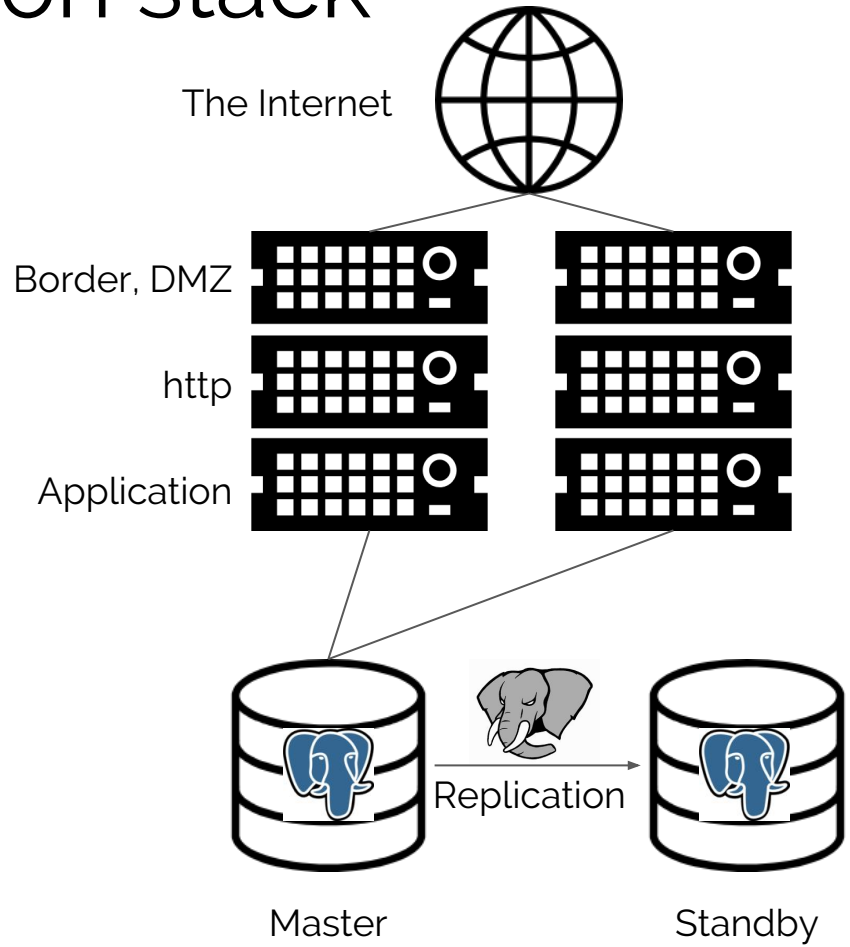
Rechercher autour de moi

Alsace
Aquitaine
Auvergne
Basse-Normandie
Bourgogne
Bretagne
Centre
Champagne-Ardenne
Corse
Franche-Comté
Haute-Normandie
Ile-de-France
Languedoc-Roussillon
Limousin
Lorraine
Midi-Pyrénées
Nord-Pas-de-Calais
Pays de la Loire
Picardie
Poitou-Charentes
Provence-Alpes-Côte d'Azur
Rhône-Alpes

Guadeloupe
Martinique
Guyane
Réunion

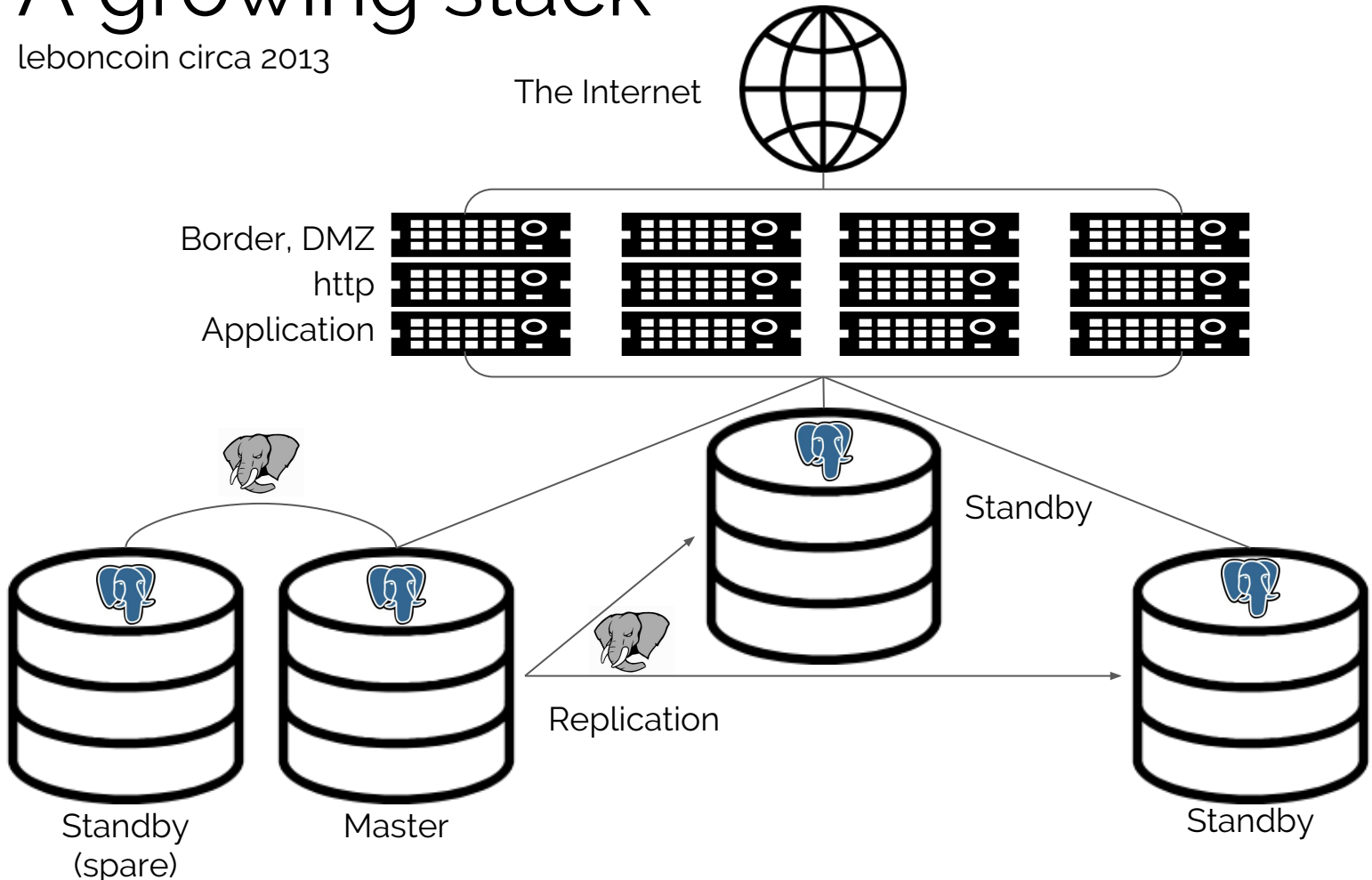
A common stack

leboncoin circa 2009



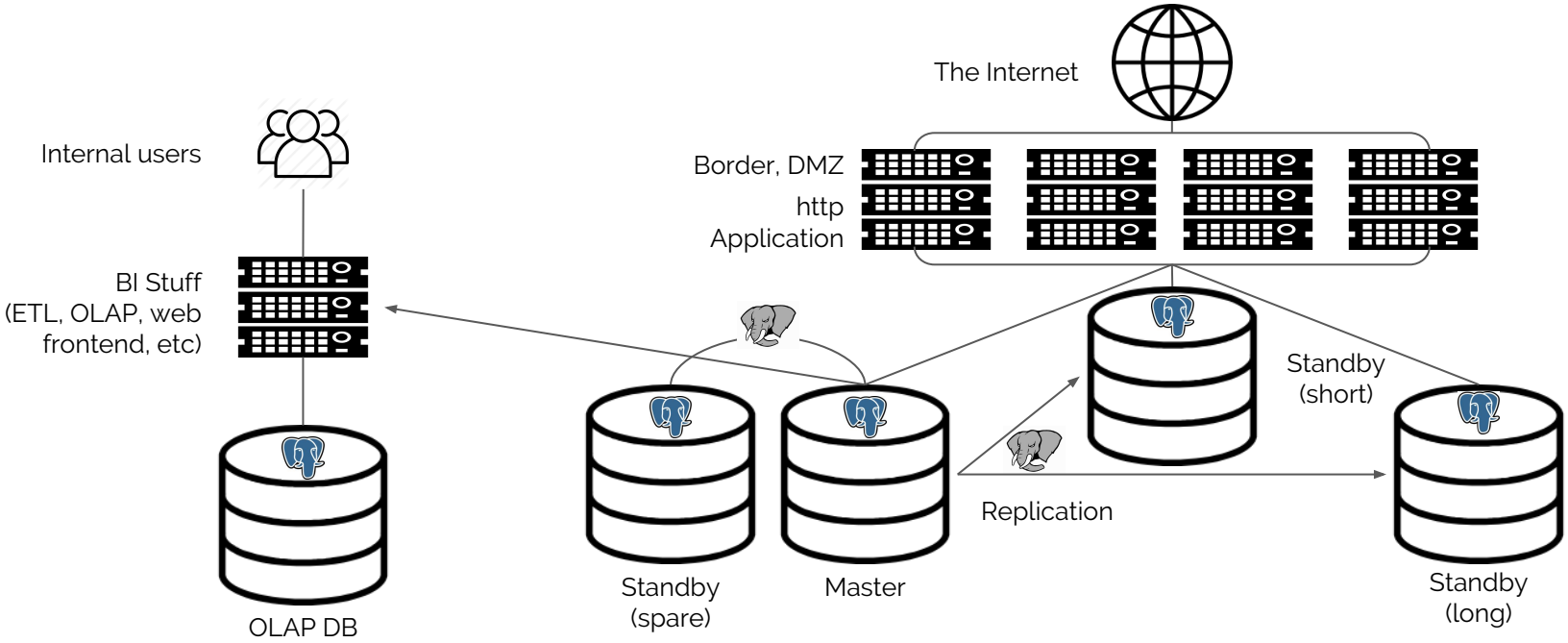
A growing stack

leboncoin circa 2013



A growing company

leboncoin still at 2013



28,1 million
unique visitors*

more than
27 million
classifield ads online

800 000
new ads every day

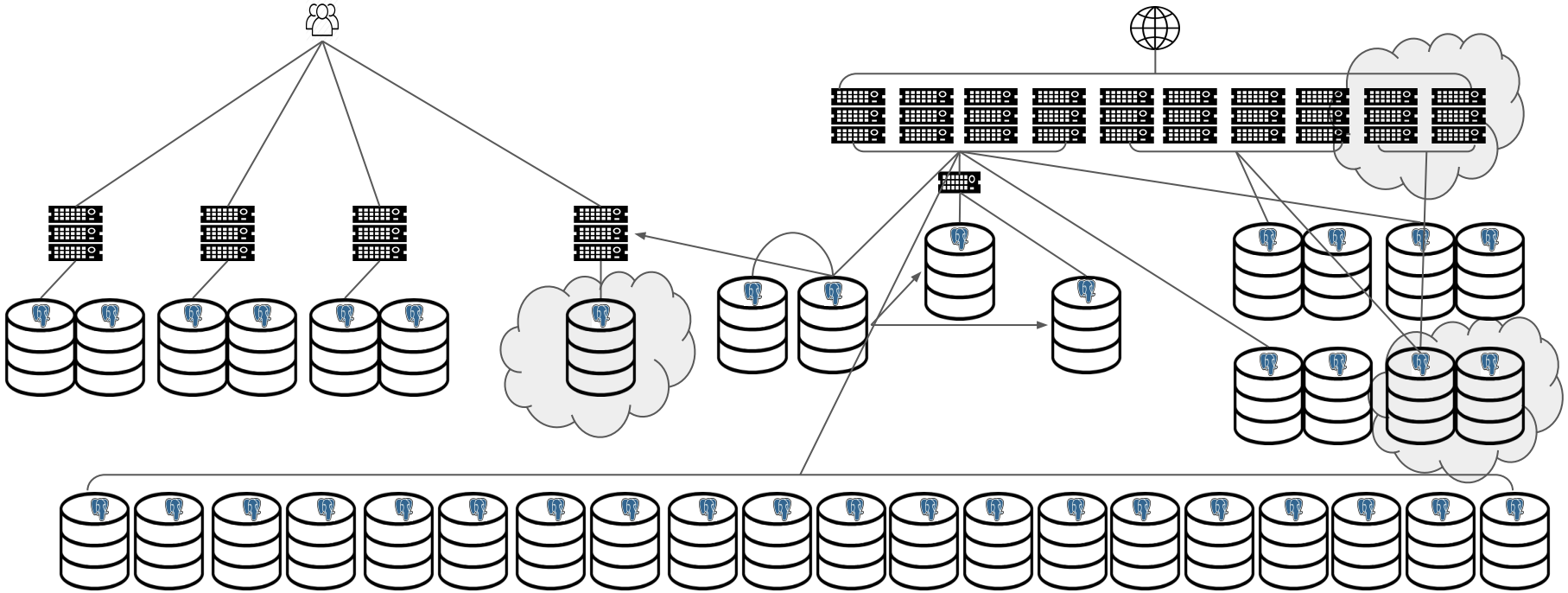
73
categories

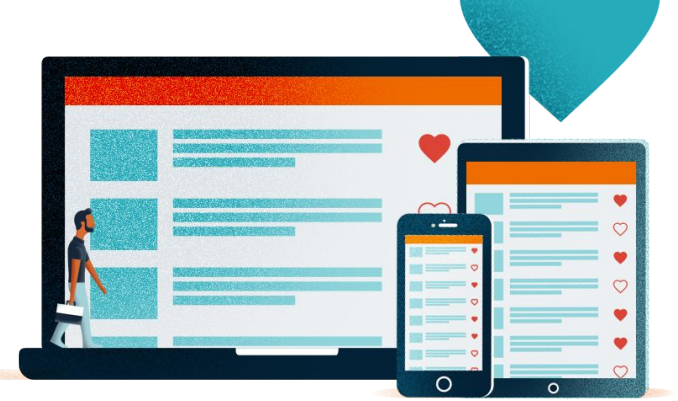
*Source Médiamétrie Net Ratings, avril 2018

leboncoin

Stack? Incomplete chart - 2 DCs + AWS

leboncoin at 2019





Technical stack

2
Datacenters
&
1
Cloud provider

2000
Virtual machines

>20 Gbits/s
outflows & a database of
3 To

300M
images
50k
req/s on leboncoin

Tech team of more than
200
people

A strong « open-source » culture:
PostgreSQL, Go, React, Python, Hadoop, Kubernetes
...

leboncoin

To handle all that: automation

AWS CloudFormation

Collins

Availability - is replication enough?

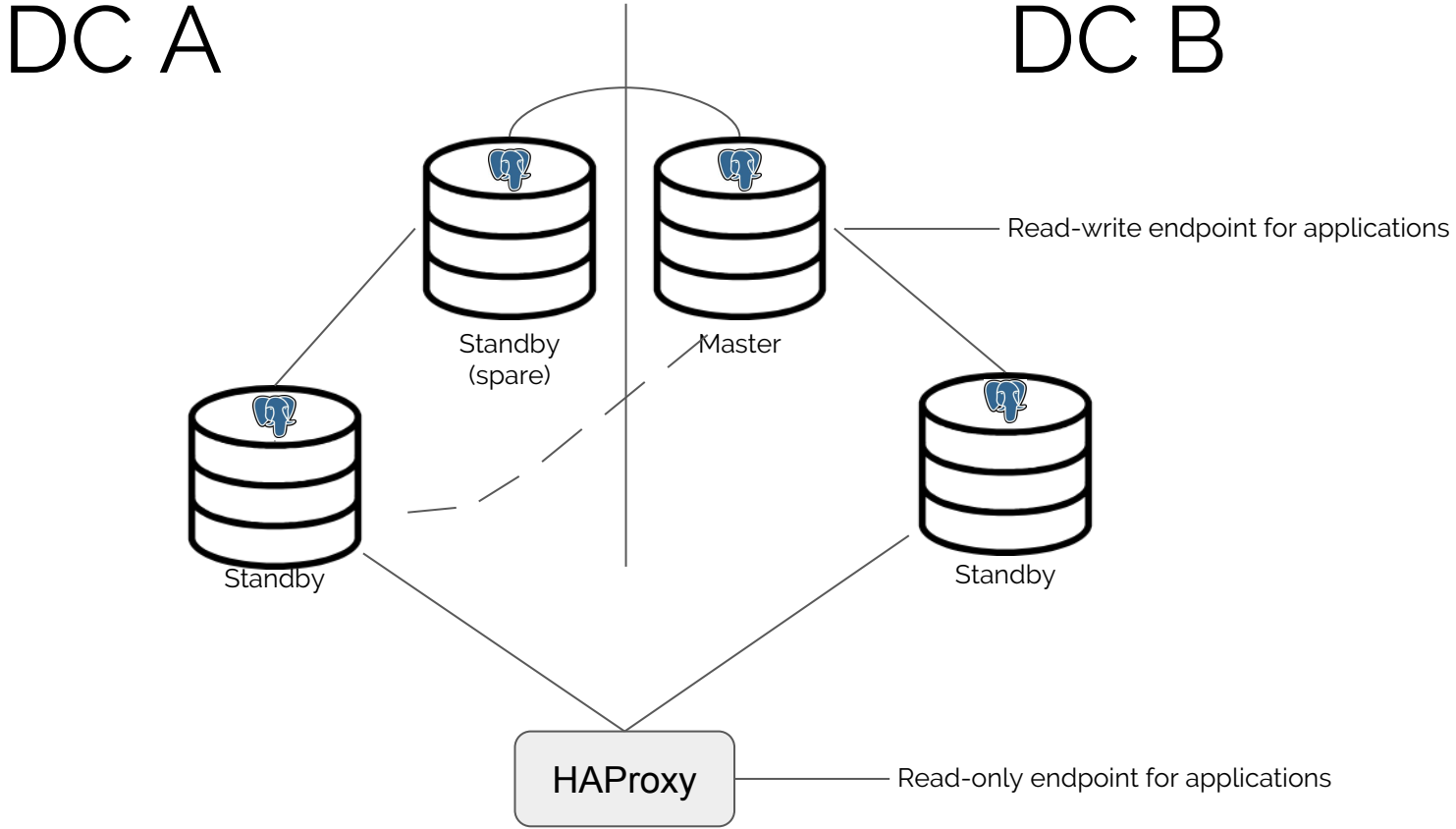
- Hardware
- Warranty
- Power
- RAID 10
- Battery
- ECC RAM
- Network
- Fans
- Alerting

Replication minimum requirements

And let's think about load-balancing too

- Standby
- Streaming
- Replication slots
- Geographic distribution
- Path
- Load balancing
- Spare

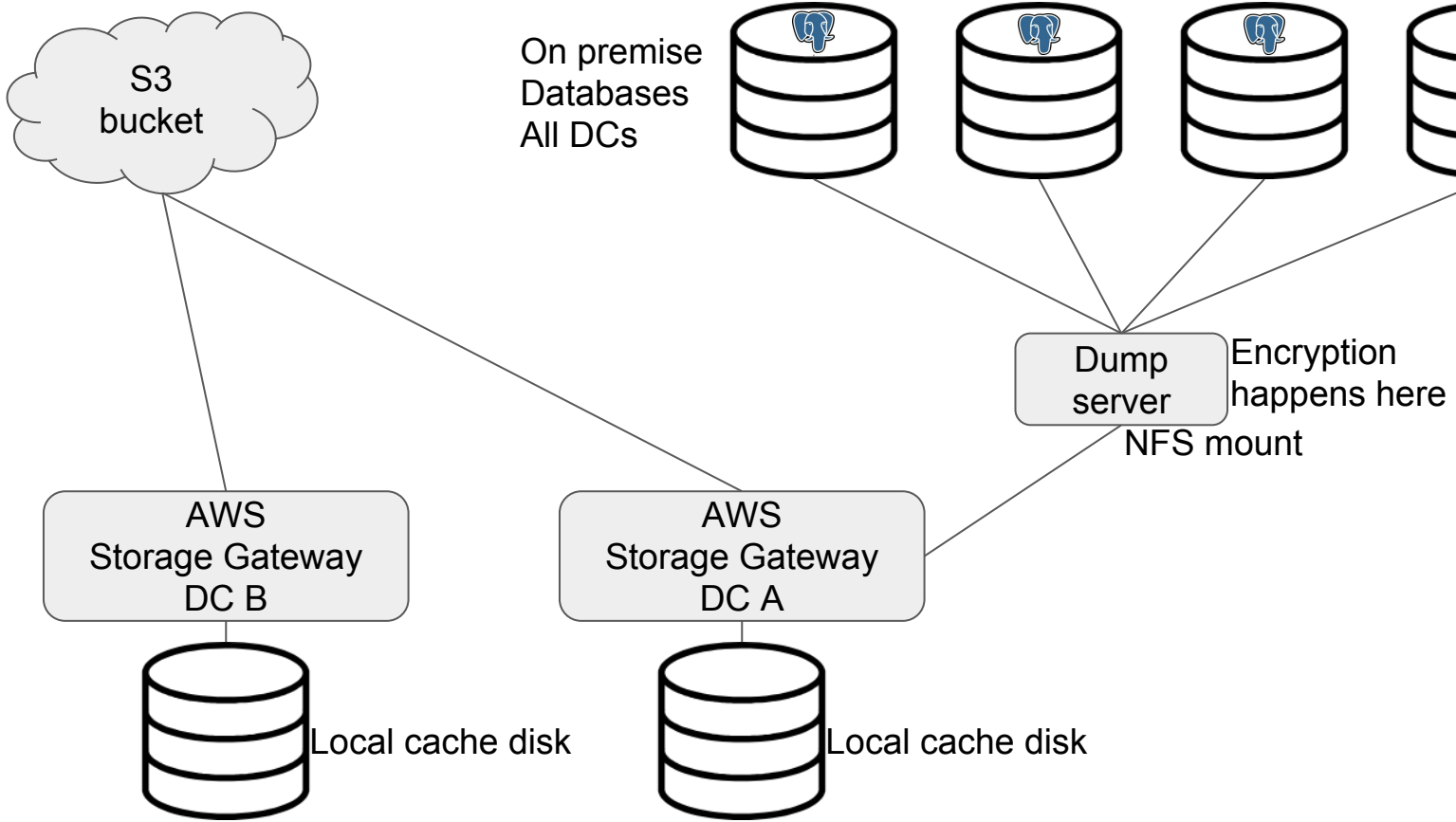
Getting critical



pg_dump

- Nightly
- Archiving DBs not dumped
- Custom mode
- Directory mode (from 300 GB)
- Encrypted
- Sent to the cloud
- Retention -> GDPR

pg_dump (and restore) strategy



Testing pg_dumps

- Mandatory
- Corruption
- Procedures
- Bugs
- **Time to restore**

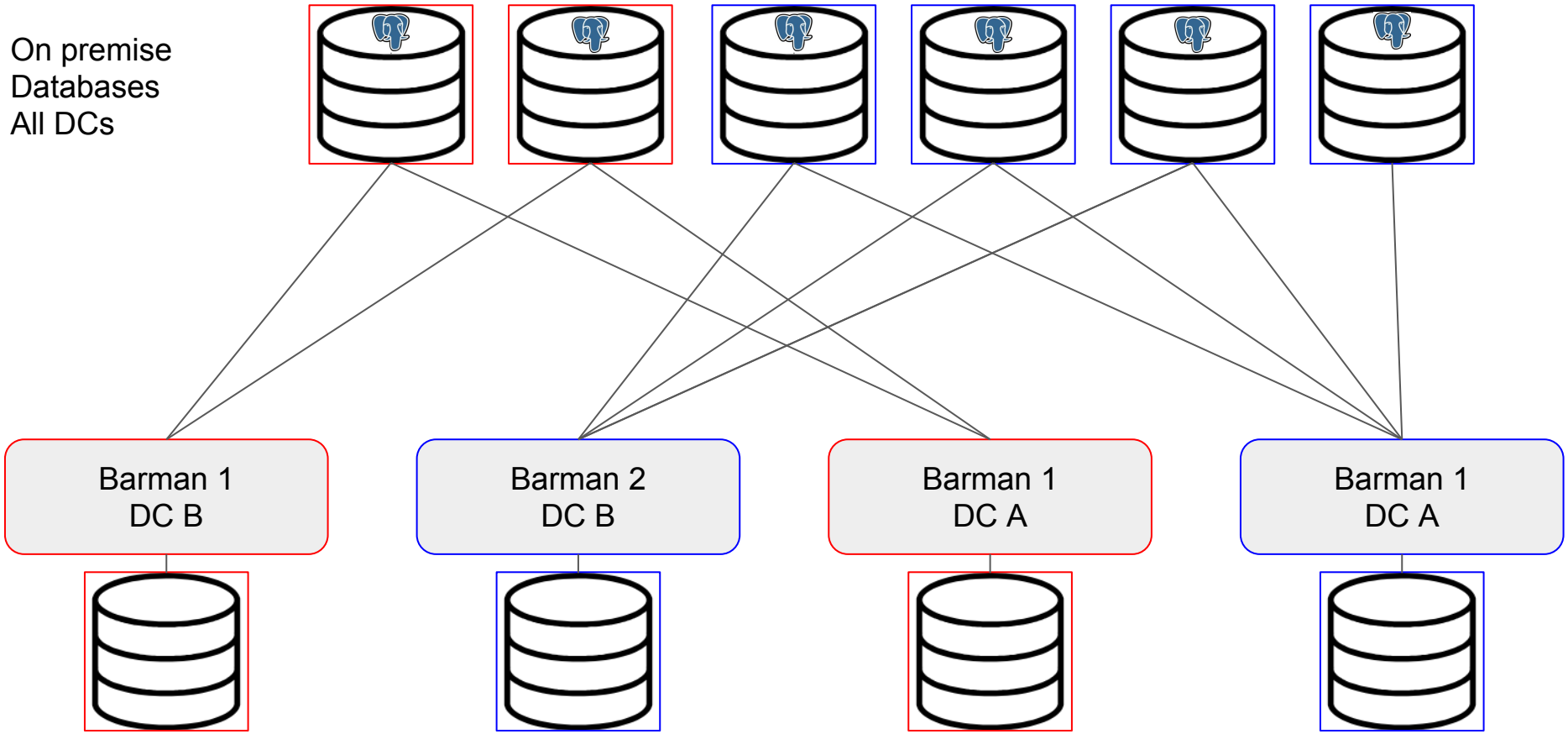
Physical backups

- Barman
- Basebackups (based on WAL/day)
  - Daily -> from 1 TB
  - Twice a week -> between 100 GB and 1 TB
  - Twice a month -> up to 100 GB
- PITR
- Tests

Barman tips

- Postgres method
  - pg_receivewal
  - Pg_basebackup
  - Replication slot
- Geographic distribution
- Archive
- Disk space
- Retention

Barman strategy



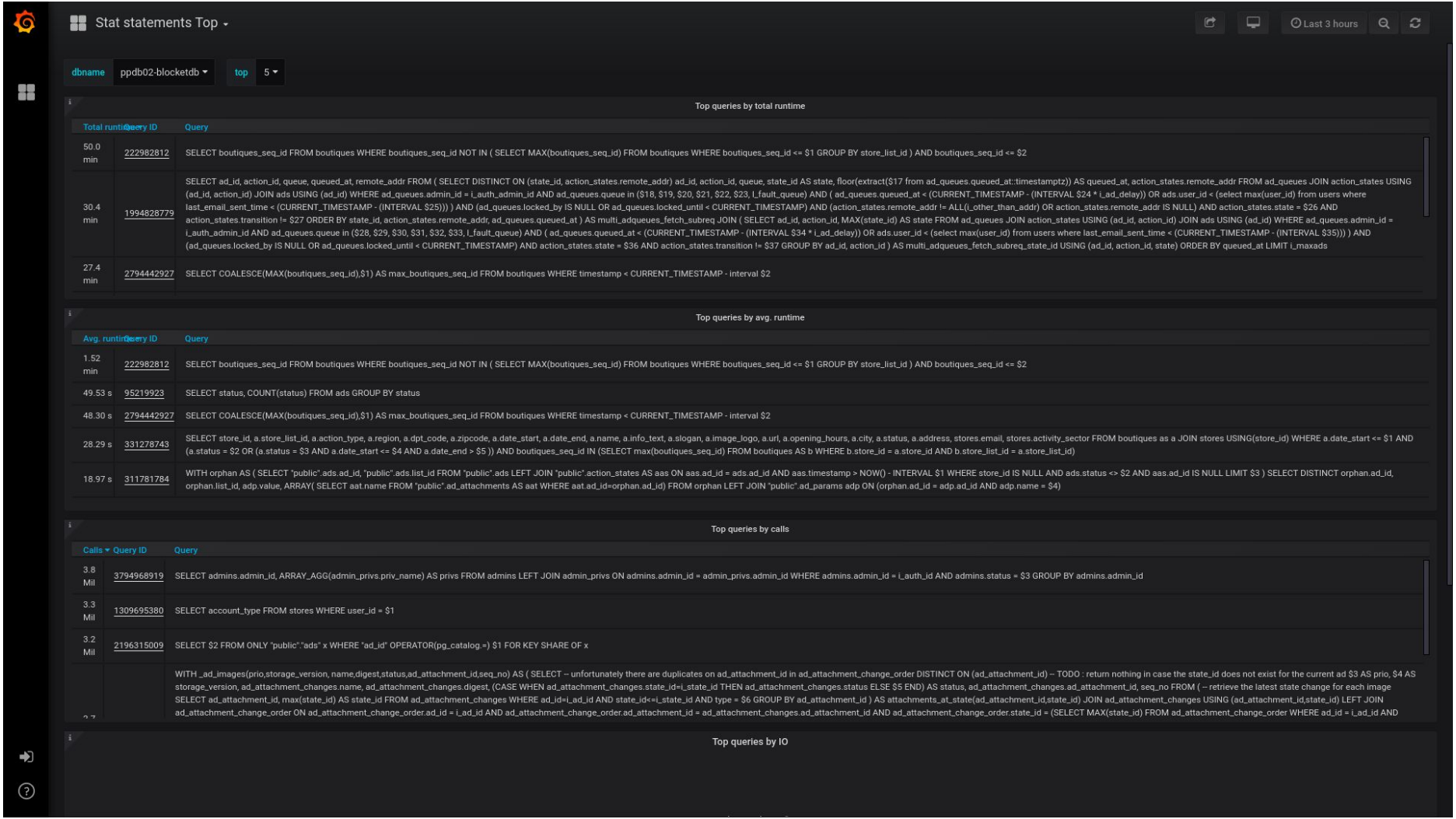
Monitoring and Alerting

pagerduty

CYBERTEC
pgwatch2


Master
conn cpu dfnfs disk files fio hw hw_bat hw_disk hw_fan hw_power hw_ram hw_temp idleintrans index info inode ipcs lag_pg libs lostslot memory monit msgs net ntpq ports postgresql pp ppenv procs queue serenity sirq slowquery ssh stale trends yum
ppdb01
ppdb02
ppdb03
ppdb04
ppdb06
Mutu
apt conn cpu disk files hw hw_bat hw_disk hw_fan hw_power hw_ram hw_temp idleintrans index info inode ipcs lag_pg libs lostslot memory monit msgs net ntpq ports postgresql pp ppenv procs sirq slowquery ssh stale trends
ppmutudb01
ppmutudb02
ppmutussddb01
ppmutussddb02
ppmutuinfradb01
ppmutuinfradb02
Serenity
conn cpu disk files hw hw_bat hw_disk hw_fan hw_power hw_ram hw_temp idleintrans index info inode ipcs lag_pg libs lostslot memory monit msgs net ntpq ports postgresql pp ppenv procs sirq slowquery ssh stale trends yum
ppserenitydb01
ppserenitydb02
Token
apt conn cpu disk files history hw hw_bat hw_disk hw_fan hw_power hw_ram hw_temp idleintrans index info inode ipcs lag_pg libs lostslot memory monit msgs net ntpq ports postgresql pp ppenv procs sirq slowquery ssh stale trends
pptokendb05
pptokendb06
Trans_queue
conn cpu disk files fio hw hw_bat hw_disk hw_fan hw_power hw_ram hw_temp idleintrans index info inode ipcs lag_pg libs lostslot memory monit msgs net ntpq ports postgresql pp ppenv procs queue sirq slowquery ssh stale trends yum
pptransq01
pptransq02
pg_dump
apt conn cpu disk dump files info inode ipcs libs memory monit msgs ntpq ports pp ppenv procs sirq ssh trends
vpdump01

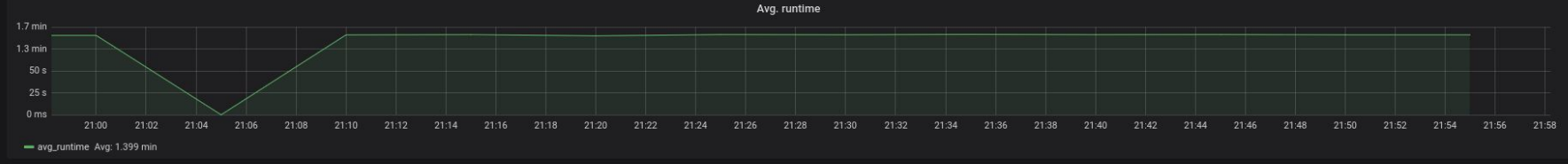

DB overview Unprivileged / Developer mode
Last 3 hours
dbname
ppdb02-blocketdb
TPS
Last 6 hours
837
QPS
Last 6 hours
2735
Query runtime
Last 6 hours
0.4 ms
Sessions
Last 6 hours
275.8
Temp bytes 1h
Last 6 hours
0 B
DB size ch. 1h
Last 6 hours
0 B
Tuple Ins. / Upd. / Del. statistics (1h rate)
10 Mil
1 Mil
100 K
10 K
19:00
19:30
20:00
20:30
21:00
21:30
INSERT Avg: 2.436 Mil
UPDATE Avg: 946 K
DELETE Avg: 136 K
Buffer hit ratio + Rollback ratio
100%
50%
0%
Shared buffers hit ratio Avg: 94.9%
TX rollback ratio Avg: 0.0%
Avg. query runtime
0.5 ms
0.4 ms
0.3 ms
avg_query_runtime Avg: 0.4 ms
WAL rate + DB size
20.0 MB/s
15.0 MB/s
10.0 MB/s
5.0 MB/s
0 B/s
3.20597 TiB
3.20551 TiB
3.20506 TiB
WAL rate Avg: 6.4 MB/s
DB Size Avg: 3.2 TiB
Sessions + Deadlocks + Temp bytes
400
300
200
100
0
1.0 B/s
0.5 B/s
0 B/s
Sessions
Deadlocks (1h rate)
Temp bytes written
TPS / QPS
15 K
10 K
5 K
0
TPS Avg: 2.5 K
QPS Avg: 8.7 K
Seq. scans per minute (>10MB tables)
40.0
30.0
20.0
10.0
0
seq_scan Avg: 27.9
Exclusive locks
150
100
50
0
AccessExclusiveLock Avg: 0
ExclusiveLock Avg: 0
RowExclusiveLock Avg: 28.3
ShareRowExclusiveLock Avg: 0
ShareUpdateExclusiveLock Avg: 1.6
Brought to you by:
CYBERTEC
The PostgreSQL Database Company

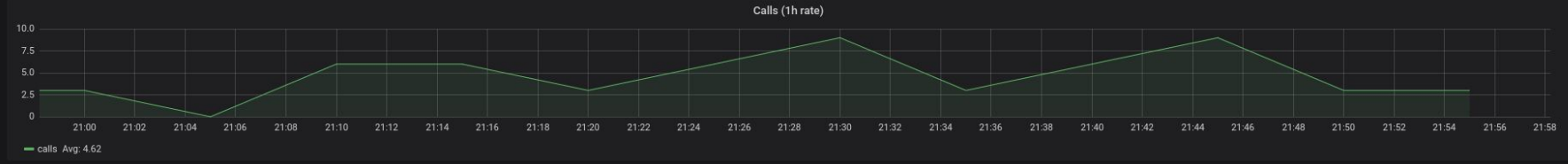

Stat statements Top
Last 3 hours
dbname
ppdb02-blocketdb
top
5
Top queries by total runtime
Total runtime
Query ID
Query
50.0 min
222982812
SELECT boutiques_seq_id FROM boutiques WHERE boutiques_seq_id NOT IN (SELECT MAX(boutiques_seq_id) FROM boutiques WHERE boutiques_seq_id <= $1 GROUP BY store_list_id) AND boutiques_seq_id <= $2
30.4 min
1994828779
SELECT ad_id, action_id, queue, queued_at, remote_addr FROM (SELECT DISTINCT ON (state_id, action_states.remote_addr) ad_id, action_id, queue, state_id AS state, floor(extract($17 from ad_queues.queued_at::timestamptz)) AS queued_at, action_states.remote_addr FROM ad_queues JOIN action_states USING (ad_id, action_id) JOIN ads USING (ad_id) WHERE ad_queues.admin_id = i_auth_admin_id AND ad_queues.queue in ($18, $19, $20, $21, $22, $23, i_fault_queue) AND (ad_queues.queued_at < (CURRENT_TIMESTAMP - (INTERVAL $24 * i_ad_delay)) OR ads.user_id < (select max(user_id) from users where last_email_sent_time < (CURRENT_TIMESTAMP - (INTERVAL $25)))) AND (ad_queues.locked_by IS NULL OR ad_queues.locked_until < CURRENT_TIMESTAMP) AND (action_states.remote_addr != ALL(i_other_than_addr) OR action_states.remote_addr IS NULL) AND action_states.state = $26 AND action_states.transition != $27 ORDER BY state_id, action_states.remote_addr, ad_queues.queued_at) AS multi_adqueues_fetch_subreq JOIN (SELECT ad_id, action_id, MAX(state_id) AS state FROM ad_queues JOIN action_states USING (ad_id, action_id) JOIN ads USING (ad_id) WHERE ad_queues.admin_id = i_auth_admin_id AND ad_queues.queue in ($28, $29, $30, $31, $32, $33, i_fault_queue) AND (ad_queues.queued_at < (CURRENT_TIMESTAMP - (INTERVAL $34 * i_ad_delay)) OR ads.user_id < (select max(user_id) from users where last_email_sent_time < (CURRENT_TIMESTAMP - (INTERVAL $35)))) AND (ad_queues.locked_by IS NULL OR ad_queues.locked_until < CURRENT_TIMESTAMP) AND action_states.state = $36 AND action_states.transition != $37 GROUP BY ad_id, action_id) AS multi_adqueues_fetch_subreq_state_id USING (ad_id, action_id, state) ORDER BY queued_at LIMIT i_maxads
27.4 min
2794442927
SELECT COALESCE(MAX(boutiques_seq_id),$1) AS max_boutiques_seq_id FROM boutiques WHERE timestamp < CURRENT_TIMESTAMP - interval $2
Top queries by avg. runtime
Avg. runtime
Query ID
Query
1.52 min
222982812
SELECT boutiques_seq_id FROM boutiques WHERE boutiques_seq_id NOT IN (SELECT MAX(boutiques_seq_id) FROM boutiques WHERE boutiques_seq_id <= $1 GROUP BY store_list_id) AND boutiques_seq_id <= $2
49.53 s
95219923
SELECT status, COUNT(status) FROM ads GROUP BY status
48.30 s
2794442927
SELECT COALESCE(MAX(boutiques_seq_id),$1) AS max_boutiques_seq_id FROM boutiques WHERE timestamp < CURRENT_TIMESTAMP - interval $2
28.29 s
331278743
SELECT store_id, a.store_list_id, a.action_type, a.region, a.dpt_code, a.zipcode, a.date_start, a.date_end, a.name, a.info_text, a.slogan, a.image_logo, a.url, a.opening_hours, a.city, a.status, a.address, stores.email, stores.activity_sector FROM boutiques as a JOIN stores USING(store_id) WHERE a.date_start <= $1 AND (a.status = $2 OR (a.status = $3 AND a.date_start <= $4 AND a.date_end > $5)) AND boutiques_seq_id IN (SELECT max(boutiques_seq_id) FROM boutiques AS b WHERE b.store_id = a.store_id AND b.store_list_id = a.store_list_id)
18.97 s
311781784
WITH orphan AS (SELECT "public".ads.ad_id, "public".ads.list_id FROM "public".ads LEFT JOIN "public".action_states AS aas ON aas.ad_id = ads.ad_id AND aas.timestamp > NOW() - INTERVAL $1 WHERE store_id IS NULL AND ads.status <> $2 AND aas.ad_id IS NULL LIMIT $3) SELECT DISTINCT orphan.ad_id, orphan.list_id, adp.value, ARRAY(SELECT aat.name FROM "public".ad_attachments AS aat WHERE aat.ad_id=orphan.ad_id) FROM orphan LEFT JOIN "public".ad_params adp ON (orphan.ad_id = adp.ad_id AND adp.name = $4)
Top queries by calls
Calls
Query ID
Query
3.8 Mil
3794968919
SELECT admins.admin_id, ARRAY_AGG(admin_privs.priv_name) AS privs FROM admins LEFT JOIN admin_privs ON admins.admin_id = admin_privs.admin_id WHERE admins.admin_id = i_auth_id AND admins.status = $3 GROUP BY admins.admin_id
3.3 Mil
1309695380
SELECT account_type FROM stores WHERE user_id = $1
3.2 Mil
2196315009
SELECT $2 FROM ONLY "public"."ads" x WHERE "ad_id" OPERATOR(pg_catalog.=) $1 FOR KEY SHARE OF x
WITH _ad_images(prio,storage_version, name,digest,status,ad_attachment_id,seq_no) AS (SELECT -- unfortunately there are duplicates on ad_attachment_id in ad_attachment_change_order DISTINCT ON (ad_attachment_id) -- TODO : return nothing in case the state_id does not exist for the current ad $3 AS prio, $4 AS storage_version, ad_attachment_changes.name, ad_attachment_changes.digest, (CASE WHEN ad_attachment_changes.state_id=i_state_id THEN ad_attachment_changes.status ELSE $5 END) AS status, ad_attachment_changes.ad_attachment_id, seq_no FROM (-- retrieve the latest state change for each image SELECT ad_attachment_id, max(state_id) AS state_id FROM ad_attachment_changes WHERE ad_id=i_ad_id AND state_id<=i_state_id AND type = $6 GROUP BY ad_attachment_id) AS attachments_at_state(ad_attachment_id,state_id) JOIN ad_attachment_changes USING (ad_attachment_id,state_id) LEFT JOIN ad_attachment_change_order ON ad_attachment_change_order.ad_id = i_ad_id AND ad_attachment_change_order.ad_attachment_id = ad_attachment_changes.ad_attachment_id AND ad_attachment_change_order.state_id = (SELECT MAX(state_id) FROM ad_attachment_change_order WHERE ad_id = i_ad_id AND
Top queries by IO

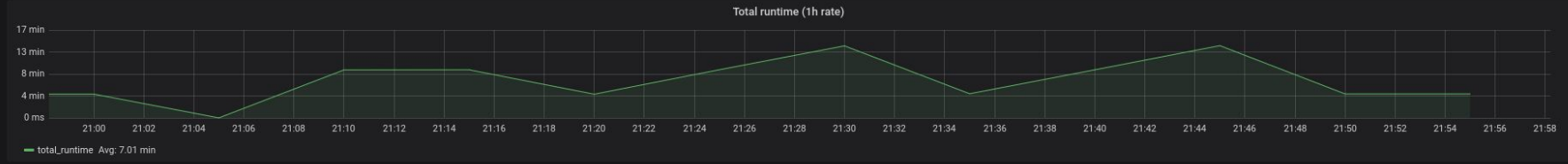
Single query details

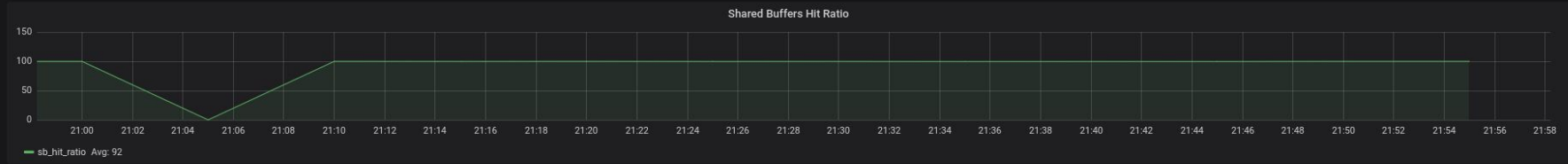
Last 1 hour

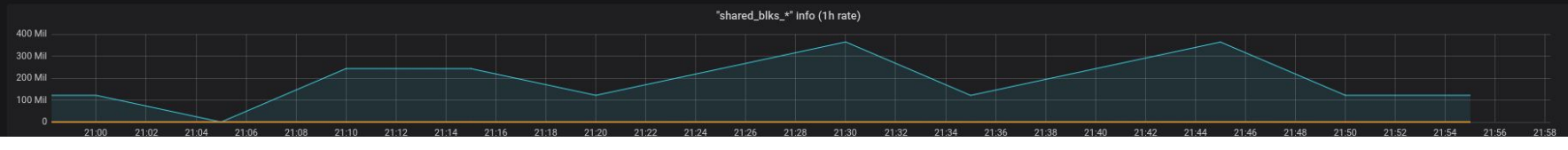
dbname ppdb02-blocketdb | queryid 222982812 | rate_unit 1h



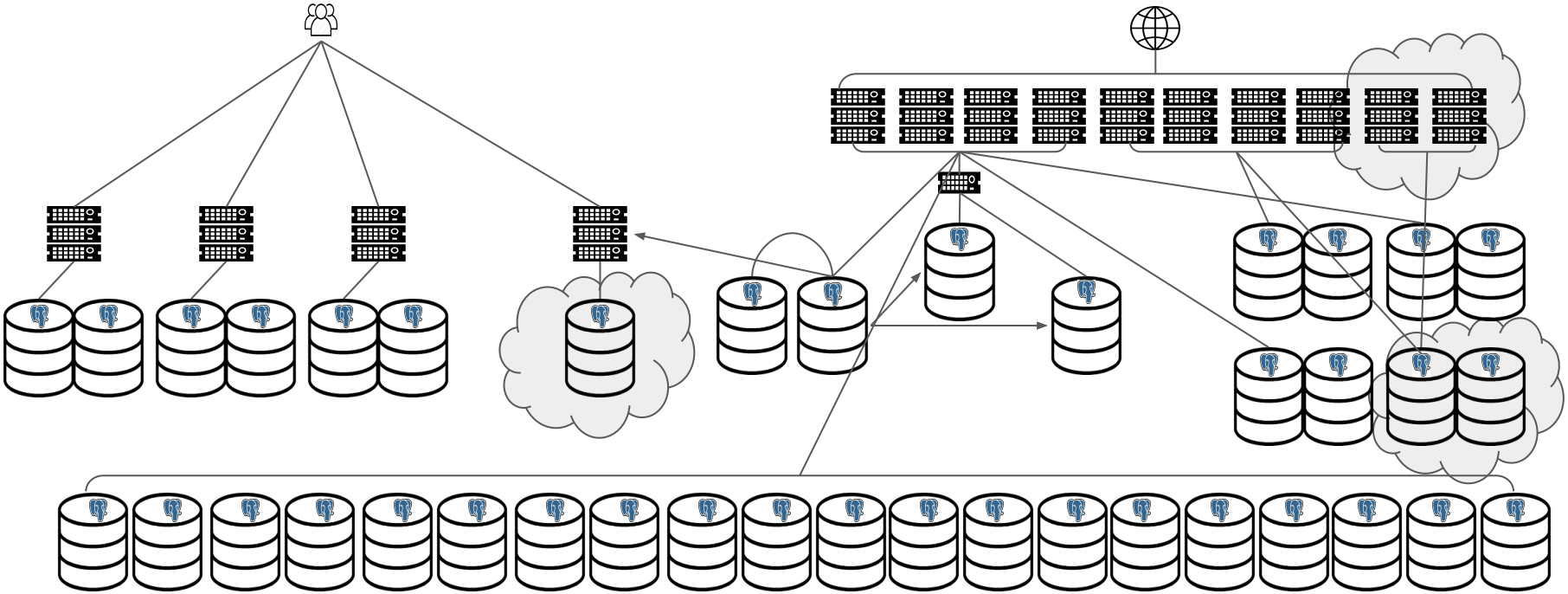








Data flows



Minor version upgrades

- Release notes
- DBA + SRE + Developper
- Standby -> Master
- Automation
- 1h total
- Site always up
- Services cut for seconds

Major version upgrades

- Same version everywhere
- Current is 10
- Decide
- QA + Staging
- Production
- pg_upgrade
- 3h total
- Site always up

Major version (new generation) upgrades

- Near zero downtime
- Logical replication to 11
- Stop origin on 10
- Update sequences
- Point to new origin
- Start production
- New physical replica

How it was to migrate from 9.3 to 10?

- row_to_json
- Parallel query
- Plan
- Auto-analyze
- Function
- Replication lag
- DDL locks
- Replication slots

Applying DDL

Incidents we faced

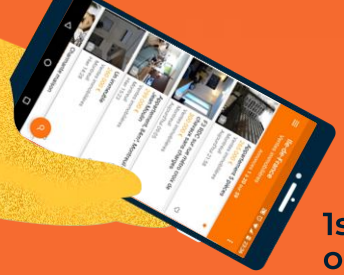
- Replication lag
- Changing execution plans
- Sqitch
- Unattended upgrade

Cloud? (speaking only of databases)

- Instance types
- On premises cost
- Variables
- Physical backups/replicas
- Lock-in
- New scenarios
- Small, elastic, internal services
- Decommissioned DBs

Other DB engine? NoSQL?

- NoSQL
  - InfluxDB
  - Elasticsearch
  - Redis
- Other engines
  - PostgreSQL - more than 70 servers
  - MySQL - some servers
  - MSSQL - one server
- Ditch PostgreSQL for (generic NoSQL here)
  - Never



1st french website on the top 10 on audience

50,6 Google

45,9 facebook.

44,8 YouTube

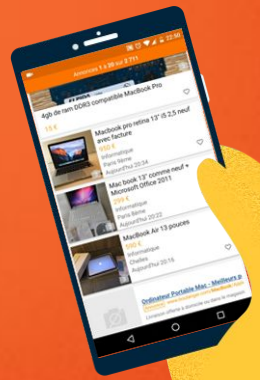
29,5 WIKIPÉDIA

28,2 leboncoin

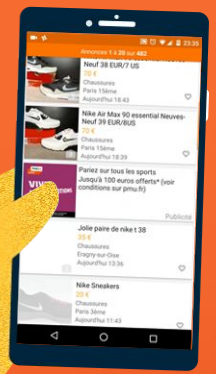
27,7 amazon

An app (iOS + Android)
downloaded 27 million times

70% of the audience
is on device mobile



The satisfaction rate of active users 85%**



*Source Médiamétrie Net Ratings, avril 2018
**Source Baromètre de satisfaction BVA novembre 2017

leboncoin

Let’s keep in touch…

github.com/leboncoin

leboncoin Engineering blog

@leboncoinEng

leboncoin