Сдвигаем тестирование БД влево

Николай Самохвалов
nik@postgres.ai

Postgres.ai

Свежая версия данных слайдов

https://bit.ly/pgday2021

(комментирование открыто!)


I'M CHECKING NEW INDEXES
FOR OUR POSTGRES DATABASE
NOT ON PRODUCTION, RIGHT?
NOT ON PRODUCTION, RIGHT?
imgflip.com

Some examples of failures due to lack of testing

- Incompatible changes – production has different DB schema than dev & test
- Cannot deploy – hitting `statement_timeout` – too heavy operations

- During deployment, we've got a failover
- Deployment lasted 10 minutes, the app was very slow (or even down)

- Two weeks after deployment, we realize that the high bloat growth we have now has been introduced by that deployment
- Deployment succeeded, but then we have started to see errors

Ландшафт тестирования БД (только dev-часть)

	Схема БД	Данные
Стат. анализ	Тесты схемы	Тесты данных
Управление изменениями	Тесты DDL	Тесты DML
Дин. анализ (производ-ть)	*“микро”* `EXPLAIN (ANALYZE, BUFFERS)`	
	“макро” Нагрузочное тестирование	

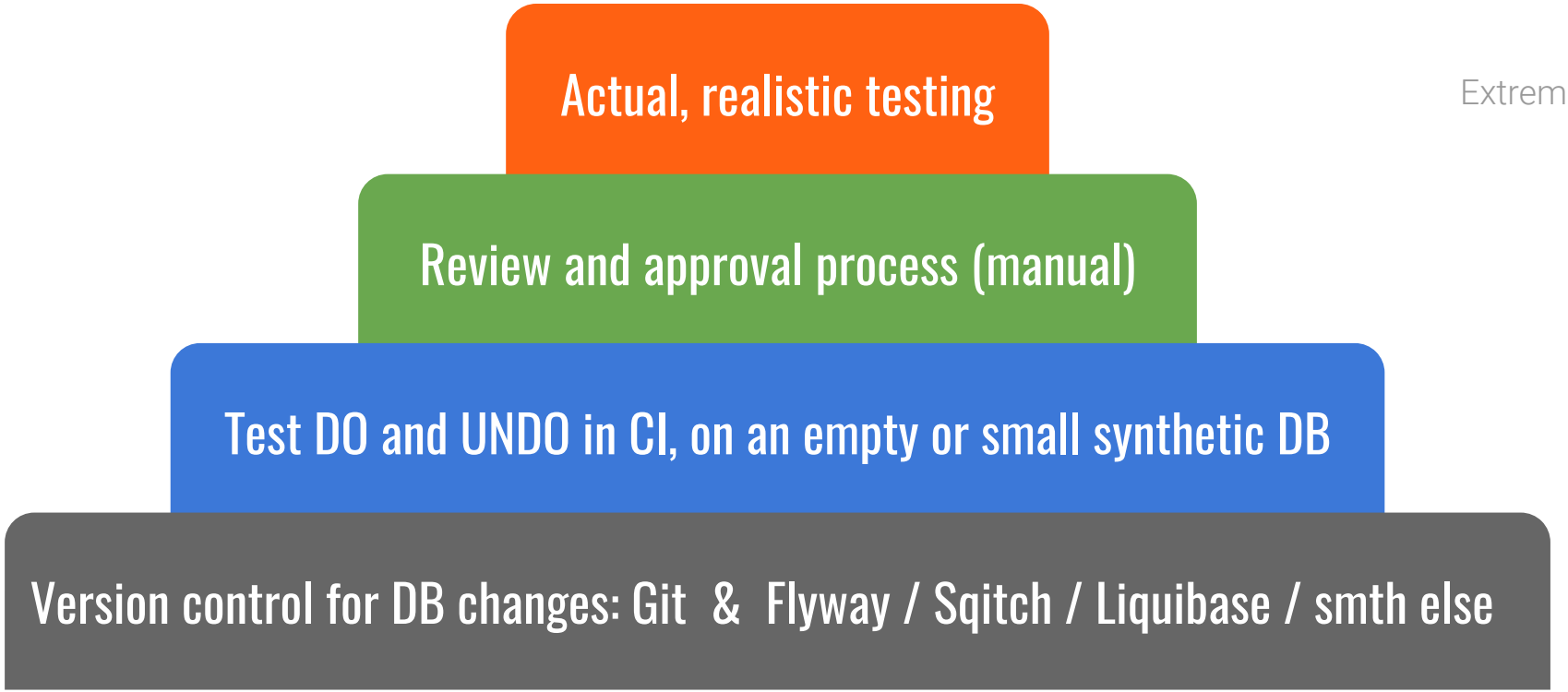
“Меняйся или умри”

НО: Изменения → больше рисков получить сбой

Миграции БД (DDL, DML) — риски:

- Ухудшение поведения систем (locking issues, resource saturation)
- Ошибки и/или деградация производительности *после* изменения
- Сбой выкатки (некорректное изменение, упираемся в квоты и т.д.)

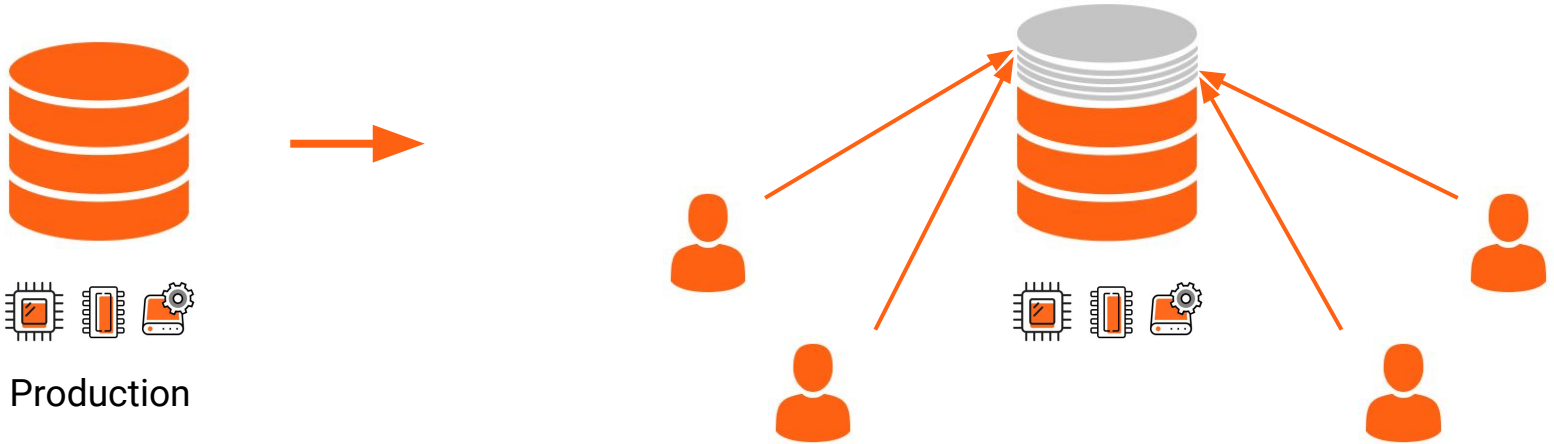
Reliable database changes – the hierarchy of needs



Traditional DB experiments – **thick** clones

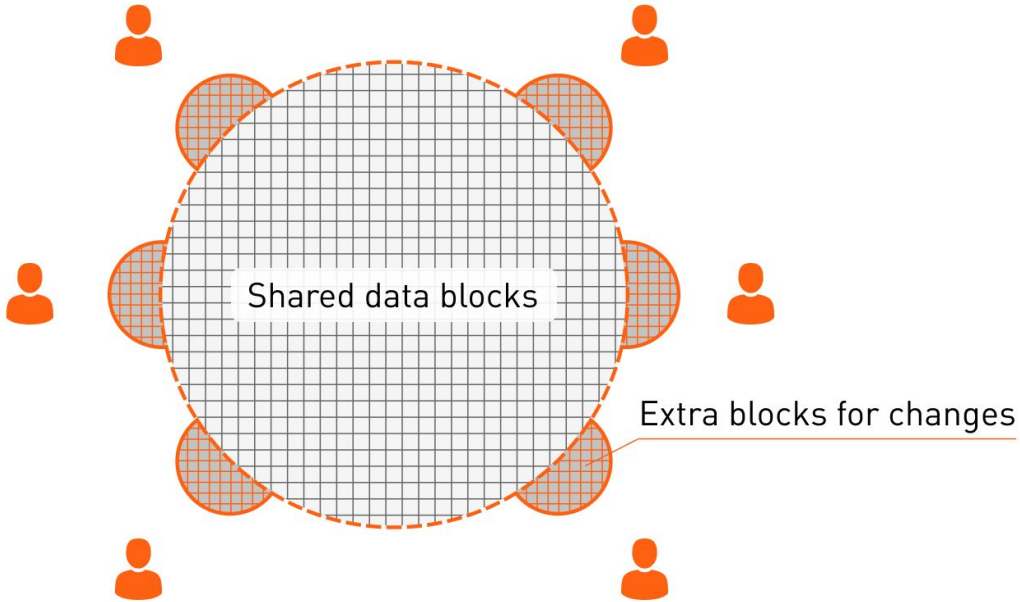
“1 database copy – 10 persons”

Database Lab: use *thin* clones



“1 database copy – 1 person”

“Thin clones” – Copy-on-Write (CoW)

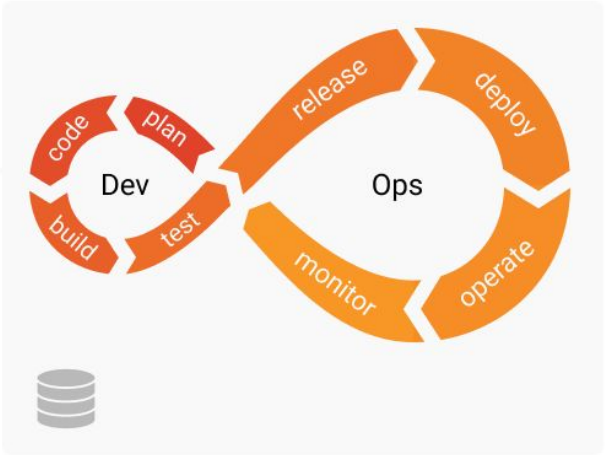


Thick copy of production (any size)

Thin clone (size starts from 1 MB, depends on changes)

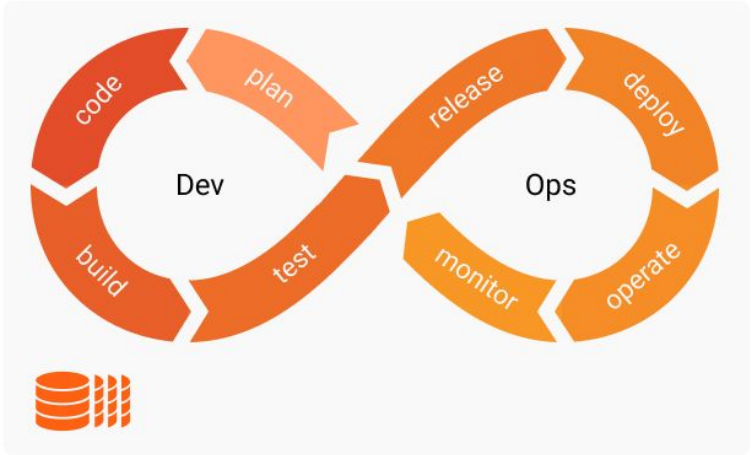
Database Lab unlocks "Shift-left testing"

Development bottlenecks
(with standard staging DB)

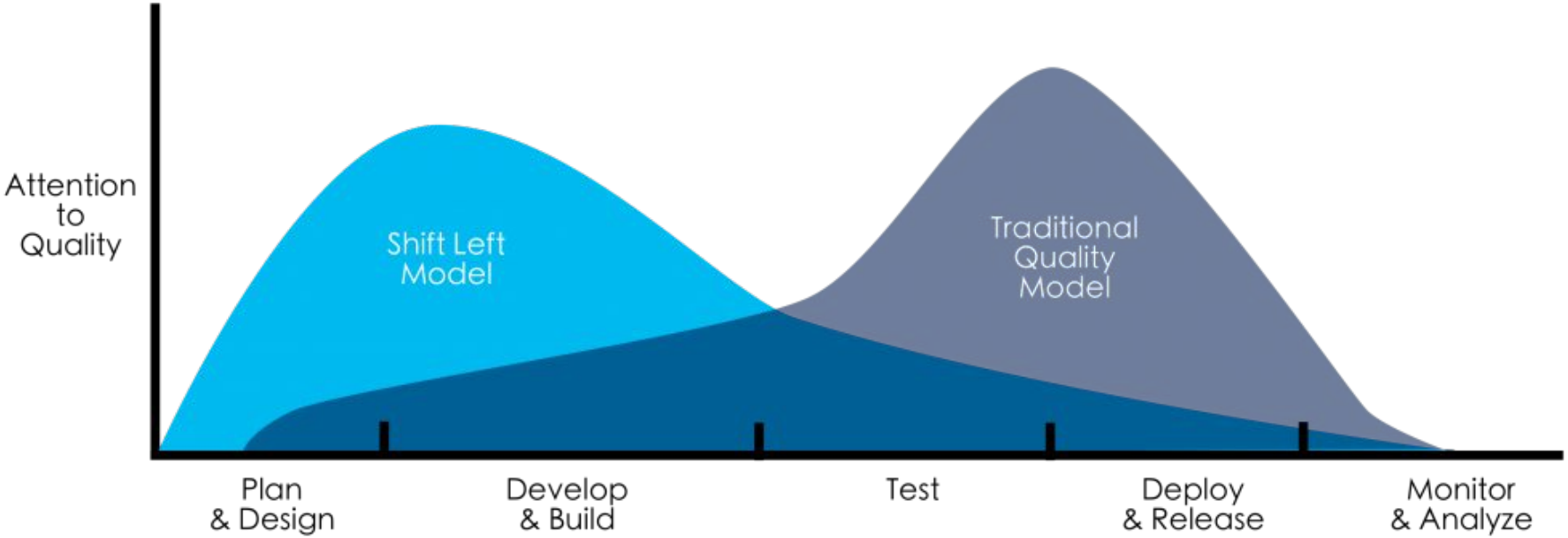


- ✕ Bugs: difficult to reproduce, easy to miss
- ✕ Not 100% of changes are well-verified
- ✕ SQL optimization is hard
- ✕ Each non-prod big DB costs a lot
- ✕ Non-prod DB refresh takes hours, days, weeks

Frictionless development
(with Database Lab)



- ✓ Bugs: easy to reproduce, and fix early
- ✓ 100% of changes are well-verified
- ✓ SQL optimization can be done by anyone
- ✓ Non-prod DB refresh takes seconds
- ✓ Extra non-prod DBs doesn't cost a penny



https://devopedia.org/shift-left

Для каких тестов БД подходят тонкие клоны?

ДА

- Check execution plan – Joe bot 

  - EXPLAIN w/o execution
  - EXPLAIN (ANALYZE, BUFFERS)
    - (timing is different; structure and buffer numbers – the same)
- Check DDL
  - index ideas (Joe bot)
  - auto-check DB migrations (CI Observer)
- Heavy, long queries: analytics, dump/restore
  - No penalties!
    (think hot_standby_feedback, locks, CPU)

НЕТ

- Load testing
- Execution time check (exact)

Database Lab – Open-core model

The Database Lab Engine (DLE)

Open-source (AGPLv3)

- Thin cloning – API & CLI
- Automated provisioning and data refresh
- Data transformation, anonymization
- Supports managed Postgres (AWS RDS, etc.)

https://gitlab.com/postgres-ai/database-lab

The Platform (SaaS)

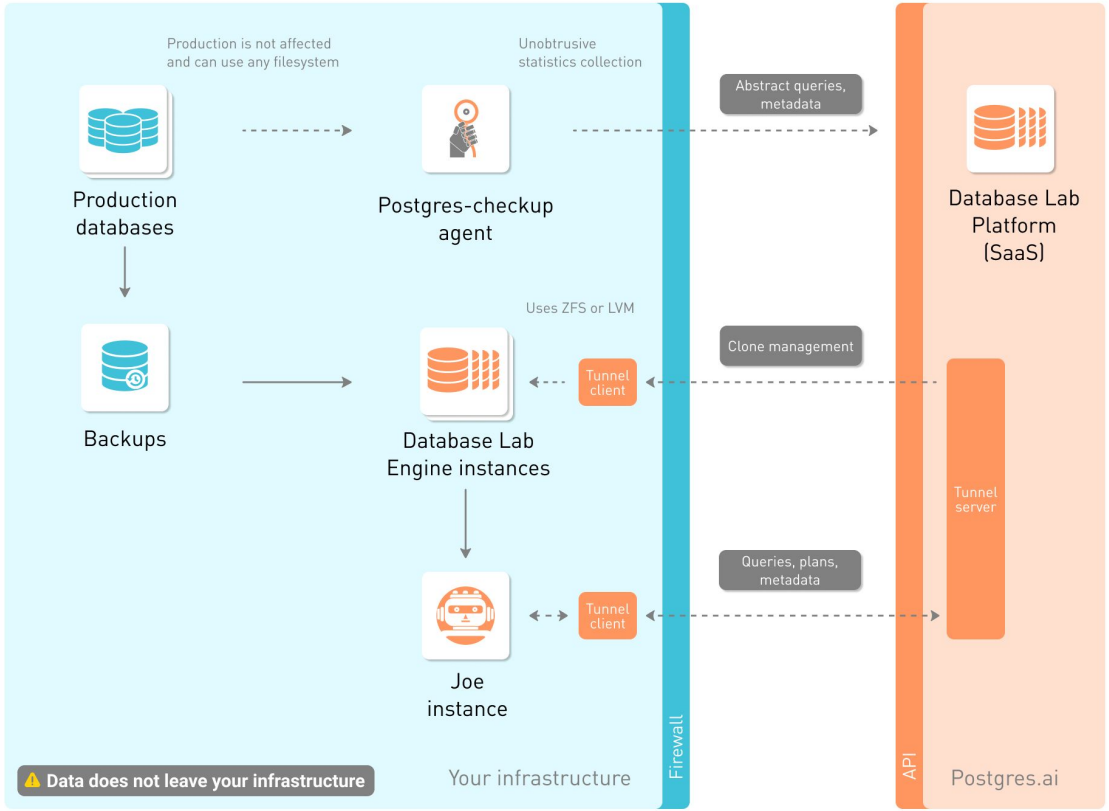
Proprietary (freemium)

- Web console – GUI
- Access control, audit
- History, visualization
- Support

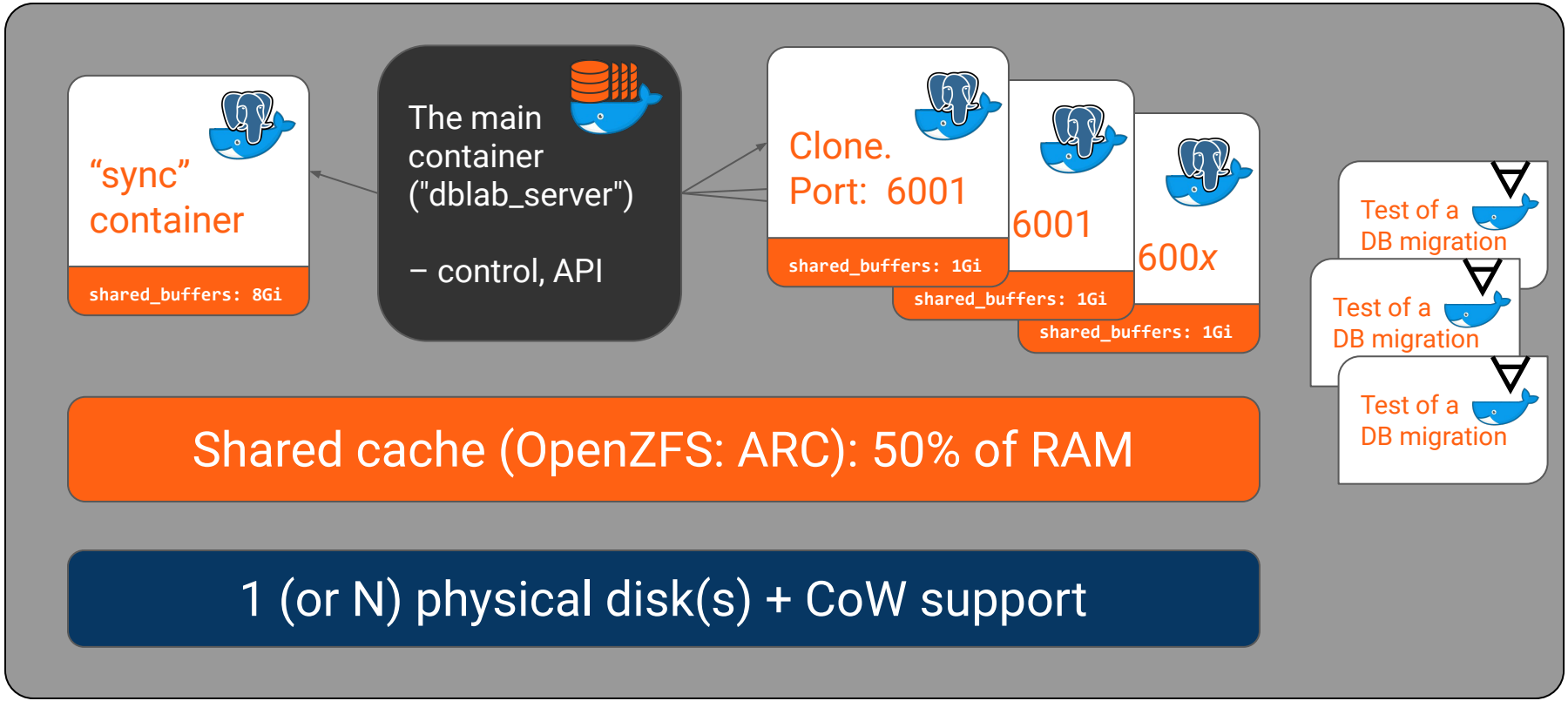
https://postgres.ai/

^^ use these links to start using it for your databases ^^

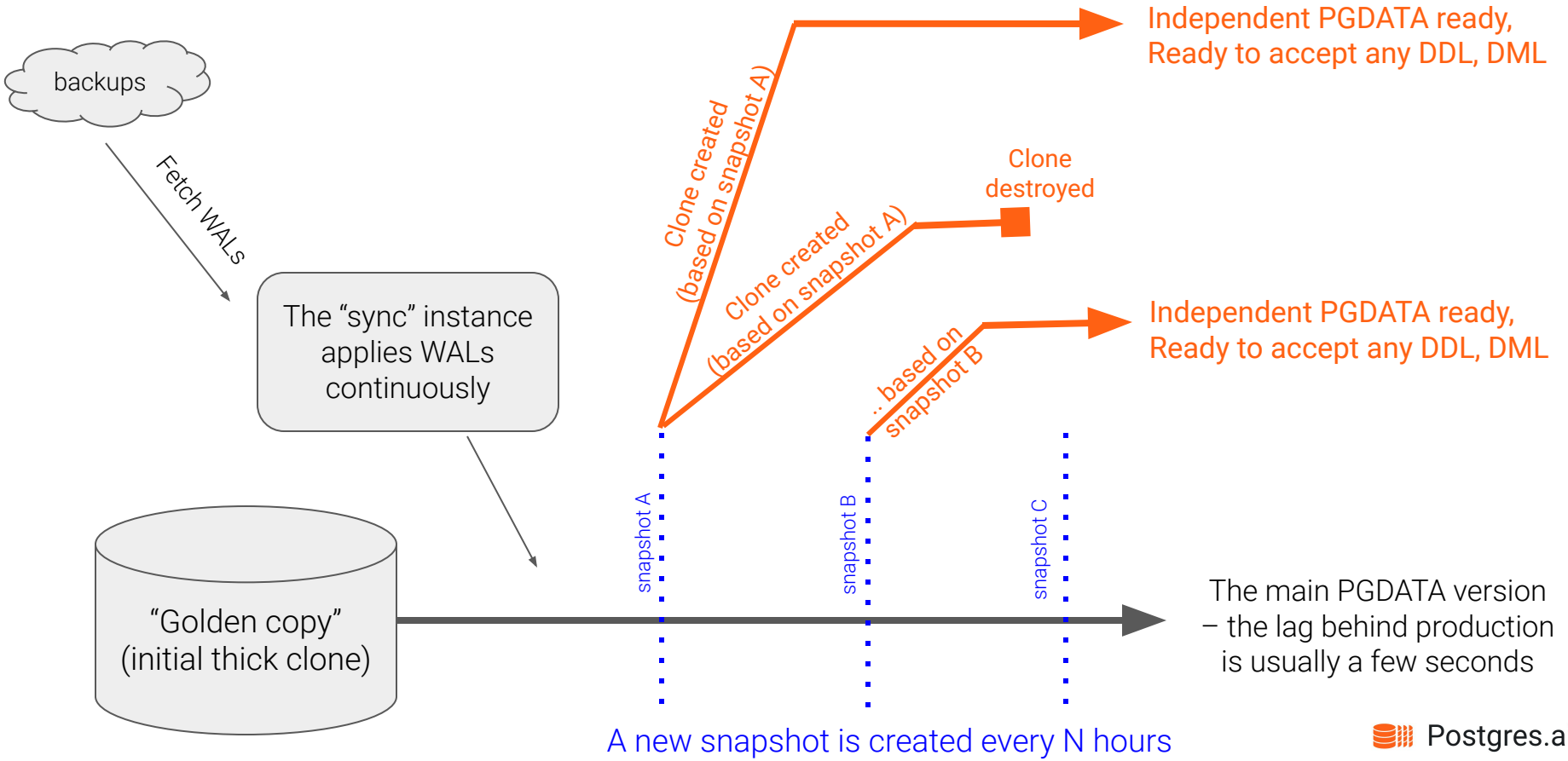
Database Lab – a high-level overview (with SaaS)



Inside the Database Lab Engine 2.x



DLE – the data flow (physical mode)



Postgres.ai

How snapshots are created (ZFS version)

- Create a “pre” ZFS snapshot (R/O)
- Create a “pre” ZFS clone (R/W)
- DLE launches a temporary “promote” container
  - If needed, performs “preprocessing” shell scripts (optional)
  - Uses “pre” clone to run Postgres and promote it to primary state
  - If needed, performs “preprocessing” SQL queries (optional)
  - Performs a clean shutdown of Postgres
- Create a final ZFS snapshot that will be used for cloning

Major topics of automated (CI) testing on thin clones

- Security

  https://postgres.ai/docs/platform/security

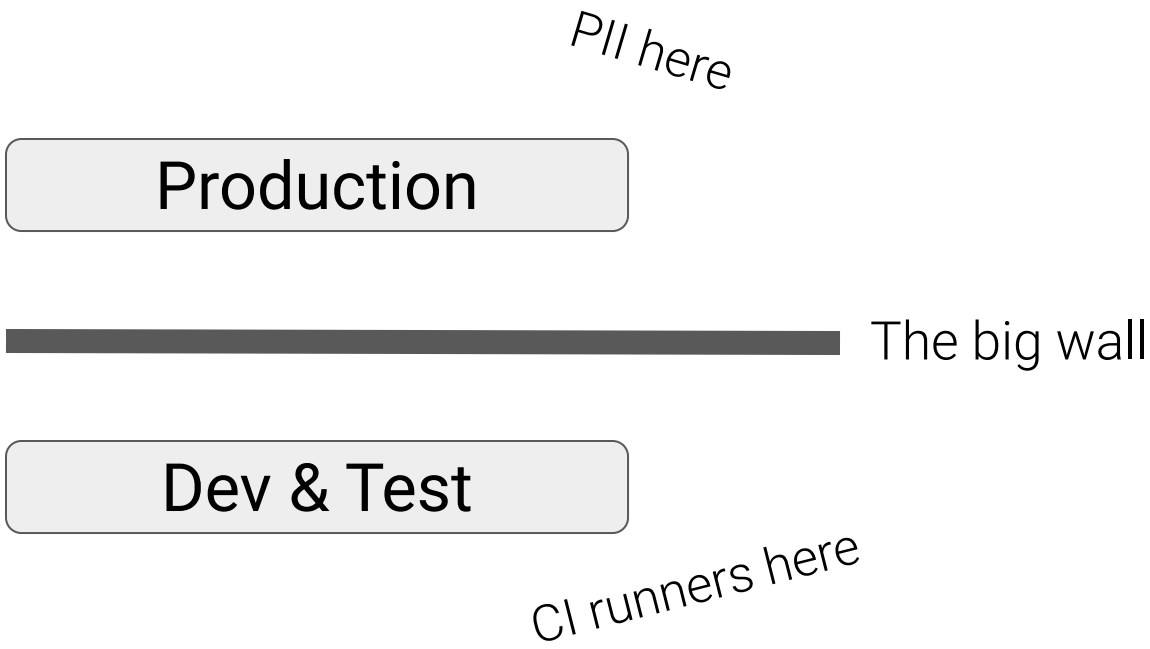
- Capturing dangerous locks

  CI Observer: https://postgres.ai/docs/database-lab/cli-reference#subcommand-start-observation

- Forecast production timing

  Timing estimator: https://postgres.ai/docs/database-lab/timing-estimator

Making the process secure: where to place the DLE?



DLE as part of production

PII here

Production

High-level API calls

GitHub

The big wall

GitLab

Dev & Test

CI runners here

Jenkins

DLE as part of production

CI check container:
- “db migrate” (any)
- cannot talk to external world
- git clone is done separately (mounted as a volume)

PII here

Production

High-level API calls

GitHub

The big wall

GitLab

Dev & Test

Jenkins

CI runners here

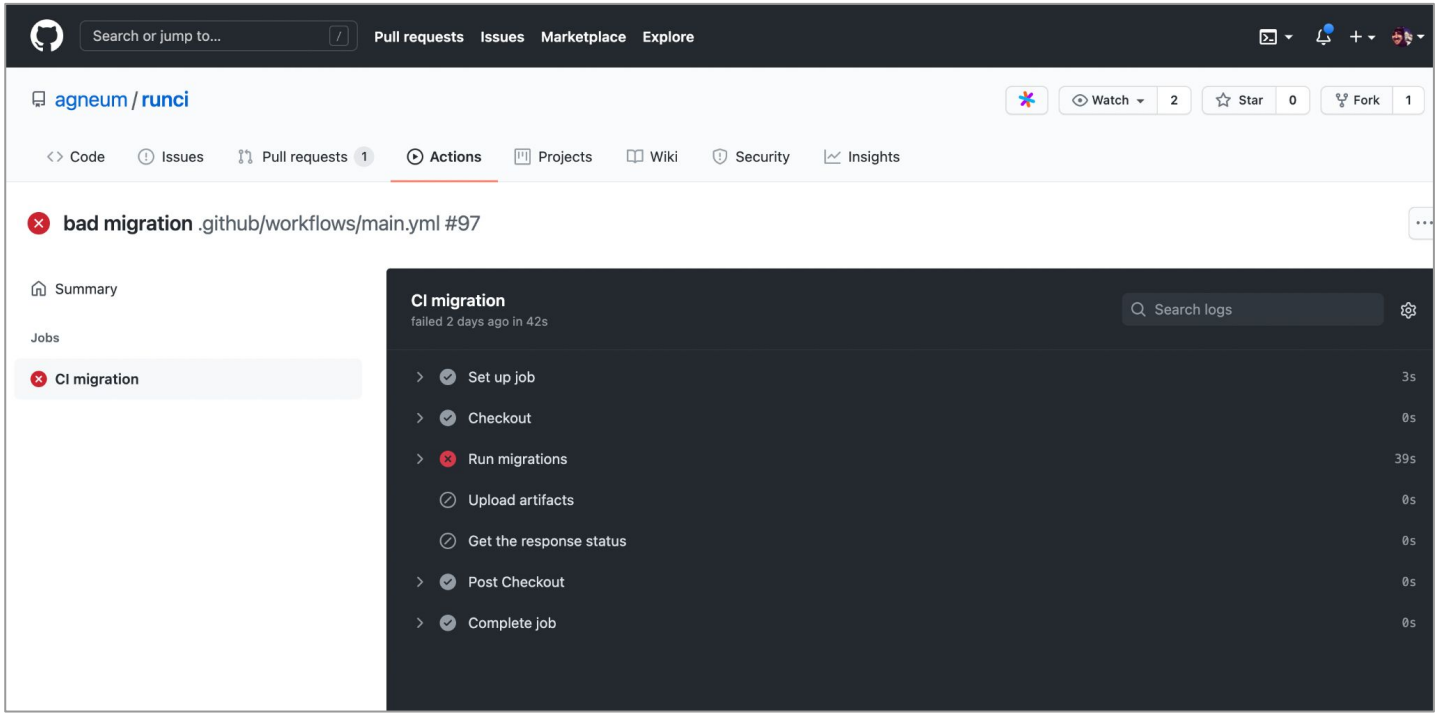
DB testing: Engineering vs. Legal

1. EXPLAIN (ANALYZE, BUFFERS)
2. DB migration check
    a. manual
    b. in CI
3. Product testing
4. DBA's benchmarks

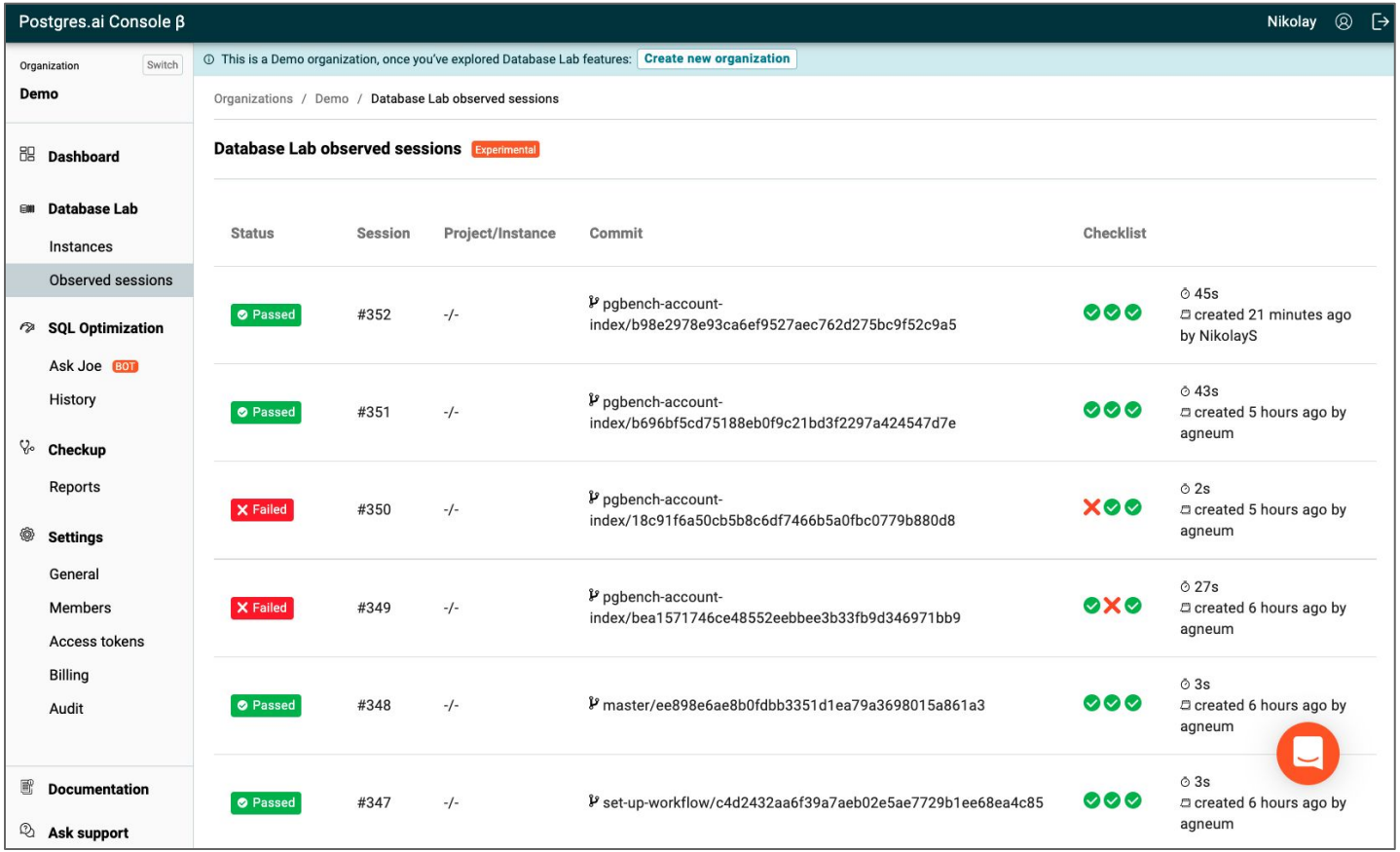
- what can (has to) be done with raw production data, containing PII?

How it looks like: CI part

Example: GitHub Actions:
https://github.com/agneum/runci/runs/2519607920?check_suite_focus=true



Postgres.ai SaaS



Postgres.ai

Postgres.ai SaaS

Database Lab observed sessions Experimental

Status	Session	Project/Instance	Commit	Checklist	
Passed	#352	-/-	pgbench-account-index/b98e2978e93ca6ef9527aec762d275bc9f52c9a5	✓✓✓	45s created 21 minutes ago by NikolayS
Passed	#351	-/-	pgbench-account-index/b696bf5cd75188eb0f9c21bd3f2297a424547d7e	✓✓✓	43s created 5 hours ago by agneum
Failed	#350	-/-	pgbench-account-	✗✓✓	2s created 5 hours ago by

Database Lab observed session #349 Experimental

Summary

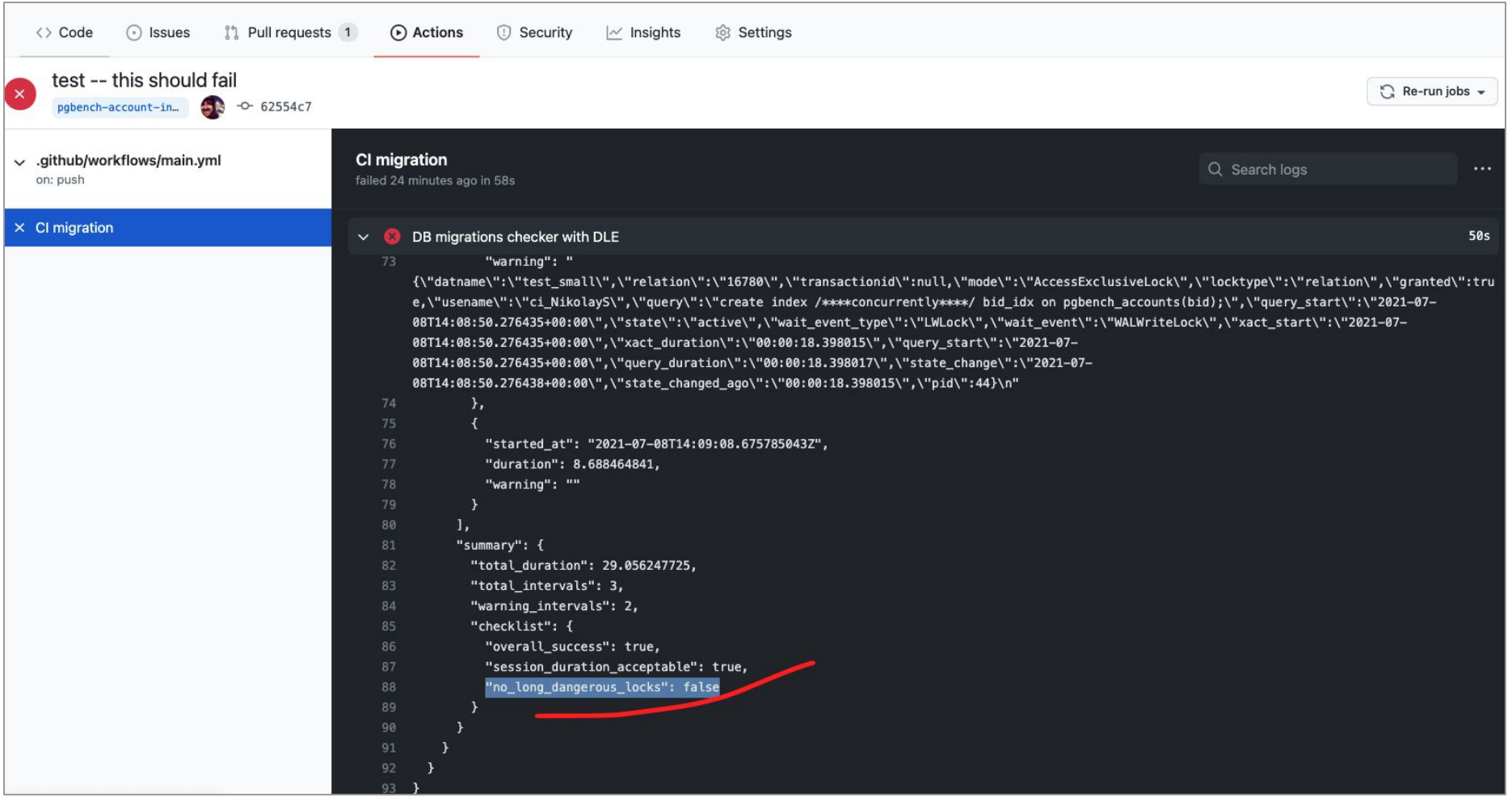
Status:	✕ Failed
Session:	#349
Project:	-
DLE instance:	
DLE version:	2.4.0-beta.3-3-g8b57d6d-20210707-0301
Data state at:	2021-07-07 11:39:05 UTC
Duration:	27s
Created:	6 hours ago
Branch:	pgbench-account-index
Commit:	bea1571746ce48552eebbee3b33fb9d346971bb9
Triggered by:	agneum (akartasov)
PR/MR:	-
Changes:	-

Checklist

✓ Passed	overall_success
✕ Failed	Dangerous locks are not observed during the session (3 intervals with locks of 10 allowed)
✓ Passed	Session duration is within allowed interval (spent 27s of the allowed 10m)

Observed intervals and details

Show intervals ⌄

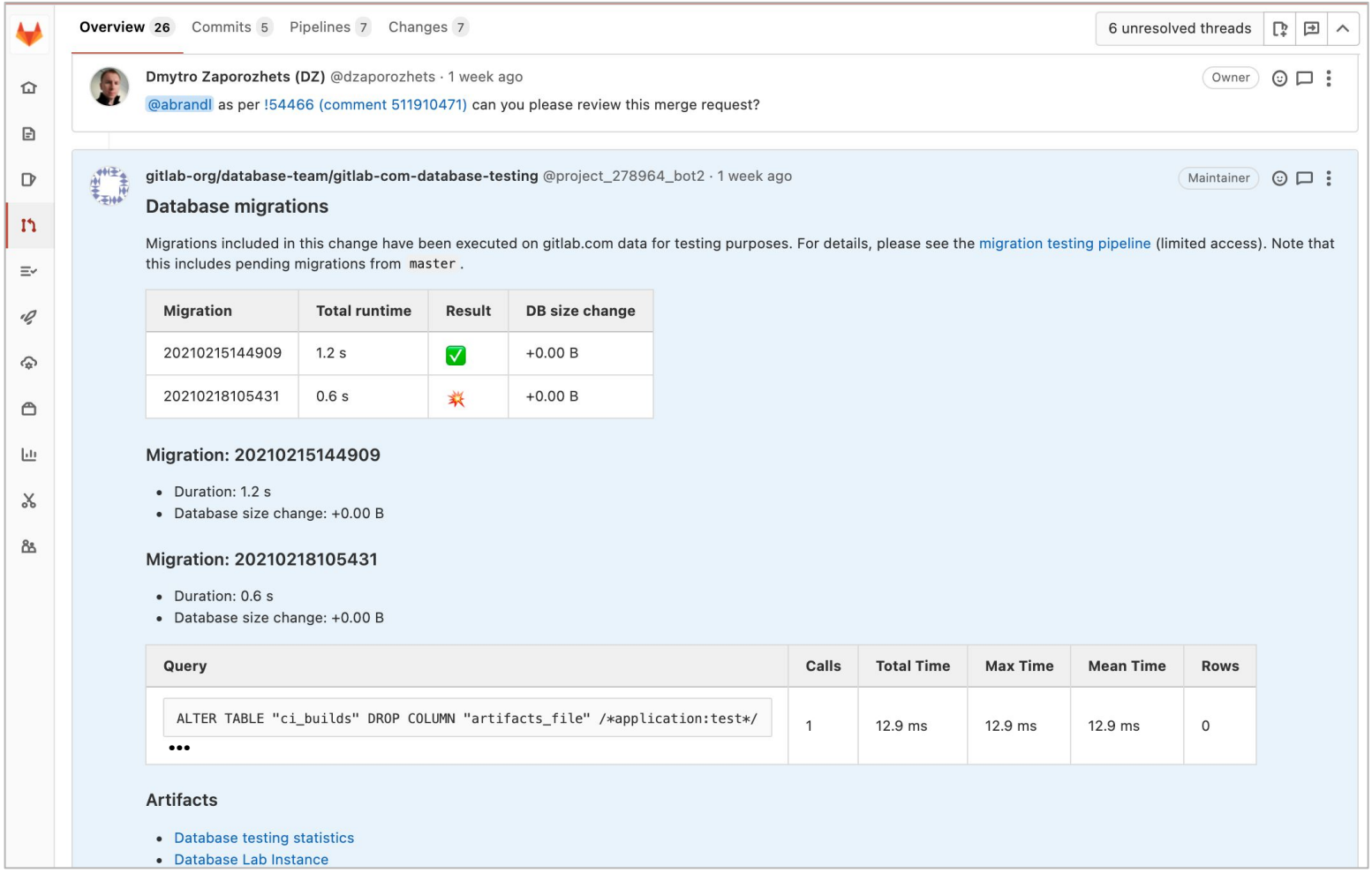

Code
Issues
Pull requests 1
Actions
Security
Insights
Settings
test -- this should fail
pgbench-account-in…
62554c7
Re-run jobs
.github/workflows/main.yml
on: push
CI migration
CI migration
failed 24 minutes ago in 58s
Search logs
DB migrations checker with DLE
50s
"warning": "
{\"datname\":\"test_small\",\"relation\":\"16780\",\"transactionid\":null,\"mode\":\"AccessExclusiveLock\",\"locktype\":\"relation\",\"granted\":true,\"usename\":\"ci_NikolayS\",\"query\":\"create index /****concurrently****/ bid_idx on pgbench_accounts(bid);\",\"query_start\":\"2021-07-08T14:08:50.276435+00:00\",\"state\":\"active\",\"wait_event_type\":\"LWLock\",\"wait_event\":\"WALWriteLock\",\"xact_start\":\"2021-07-08T14:08:50.276435+00:00\",\"xact_duration\":\"00:00:18.398015\",\"query_start\":\"2021-07-08T14:08:50.276435+00:00\",\"query_duration\":\"00:00:18.398017\",\"state_change\":\"2021-07-08T14:08:50.276438+00:00\",\"state_changed_ago\":\"00:00:18.398015\",\"pid\":44}\n"
},
{
"started_at": "2021-07-08T14:09:08.675785043Z",
"duration": 8.688464841,
"warning": ""
}
],
"summary": {
"total_duration": 29.056247725,
"total_intervals": 3,
"warning_intervals": 2,
"checklist": {
"overall_success": true,
"session_duration_acceptable": true,
"no_long_dangerous_locks": false
}
}
}
}
}

Case study: GitLab.com, testing database changes using Database Lab

- Full automation
- GitLab CI/CD pipelines securely work with Database Lab
- Database Lab clones ~14 TiB database in ~15 seconds

More:

- https://docs.gitlab.com/ee/architecture/blueprints/database_testing/
- https://postgres.ai/resources/case-studies/gitlab



Overview 26 Commits 5 Pipelines 7 Changes 7

6 unresolved threads

Dmytro Zaporozhets (DZ) @dzaporozhets · 1 week ago

Owner

@abrandl as per !54466 (comment 511910471) can you please review this merge request?

gitlab-org/database-team/gitlab-com-database-testing @project_278964_bot2 · 1 week ago

Maintainer

Database migrations

Migrations included in this change have been executed on gitlab.com data for testing purposes. For details, please see the migration testing pipeline (limited access). Note that this includes pending migrations from `master`.

Migration	Total runtime	Result	DB size change
20210215144909	1.2 s	✅	+0.00 B
20210218105431	0.6 s	💥	+0.00 B

Migration: 20210215144909

- Duration: 1.2 s
- Database size change: +0.00 B

Migration: 20210218105431

- Duration: 0.6 s
- Database size change: +0.00 B

Query	Calls	Total Time	Max Time	Mean Time	Rows
`ALTER TABLE "ci_builds" DROP COLUMN "artifacts_file" /*application:test*/` •••	1	12.9 ms	12.9 ms	12.9 ms	0

Artifacts

- Database testing statistics
- Database Lab Instance

February 19, 2021 – https://gitlab.com/gitlab-org/gitlab/-/merge_requests/54564#note_512678910

More about production timing estimation

Experimental, WIP: https://postgres.ai/docs/database-lab/timing-estimator

```
estimator:
    readRatio: 1
    writeRatio: 1
    profilingInterval: 20ms
    sampleThreshold: 100
```

```
LOG: Profiling process 63 with 10ms sampling
% time      seconds wait_event
------ ------------ ------------------------------
57.30     17.715111 IO.DataFileRead
25.53      7.893916 Running
3.55       1.097738 IO.DataFileExtend
2.55       0.787341 LWLock.WALWriteLock
2.25       0.696663 IO.BufFileRead
2.14       0.662457 IO.BufFileWrite
2.12       0.654081 IO.WALInitWrite
1.62       0.499461 IO.WALInitSync
1.09       0.335660 IO.WALWrite
0.98       0.301637 IO.DataFileImmediateSync
0.81       0.250249 IO.WALSync
0.07       0.020805 LWLock.WALBufMappingLock
------ ------------ ------------------------------
100.00    30.915119
```

Summary:

```
Time: 3.148 s
  - planning: 0.168 ms
  - execution: 3.147 s (estimated* for prod: 2.465...2.693 s)
    - I/O read: 627.267 ms
    - I/O write: 3.644 ms

Shared buffers:
  - hits: 1016393 (~7.80 GiB) from the buffer pool
  - reads: 16395 (~128.10 MiB) from the OS file cache, including disk I/O
  - dirtied: 16395 (~128.10 MiB)
  - writes: 280 (~2.20 MiB)
```

Summary – available in PR/MR and visible to whole team

- When, who, status
- Duration (in the Lab + estimated for production)
- Size changes, new objects
- Dangerous locks
- Error stats
- Transaction stats
- Query analysis summary
- Tuple stats
- WAL generated, checkpoitner/bgwriter stats
- Temp files stats

Example (WIP): https://gitlab.com/postgres-ai/database-lab/-/snippets/2083427

More artifacts, details – restricted access

- System monitoring (resources utilization)
- pg_stat_*
- pg_stat_statements, pg_stat_kcache
- logerrors
- Postgres log
- pgBadger (html, json)
- wait event sampling
- perf tracing, flamegraphs; or eBPF
- Estimated production timing

https://gitlab.com/postgres-ai/database-lab/-/issues/226

Database Lab Roadmap

https://postgres.ai/docs/roadmap

- Lower the entry bar
  - Simplify installation // Terraform: https://gitlab.com/postgres-ai/database-lab-infrastructure/
  - Simplify the use
  - Easy to integrate
  - *** **** * *******

С чего начать

Postgres.ai/docs/

Вопросы — телеграм: t.me/databaselabru

Спасибо!

Телеграм (RU): t.me/databaselabru

Twitter: @samokhvalov & @Database_Lab

Email: nik@postgres.ai

Extra slides

О докладчике: Николай Самохвалов

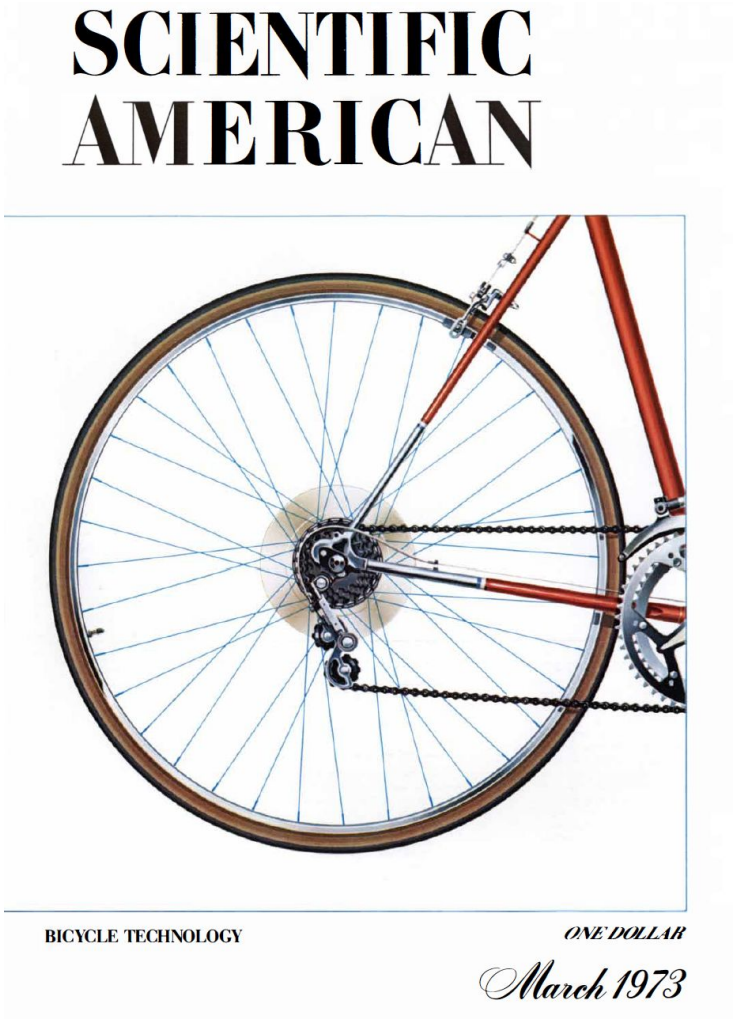
- СУБД
  - 2002-2005: Oracle 8i, Microsoft SQL Server 2000
  - since 2005: PostgreSQL
- Типа данных и функции XML в Постгресе (2005-2007)
- «Общественная» деятельность – #RuPostgres, Postgres.tv

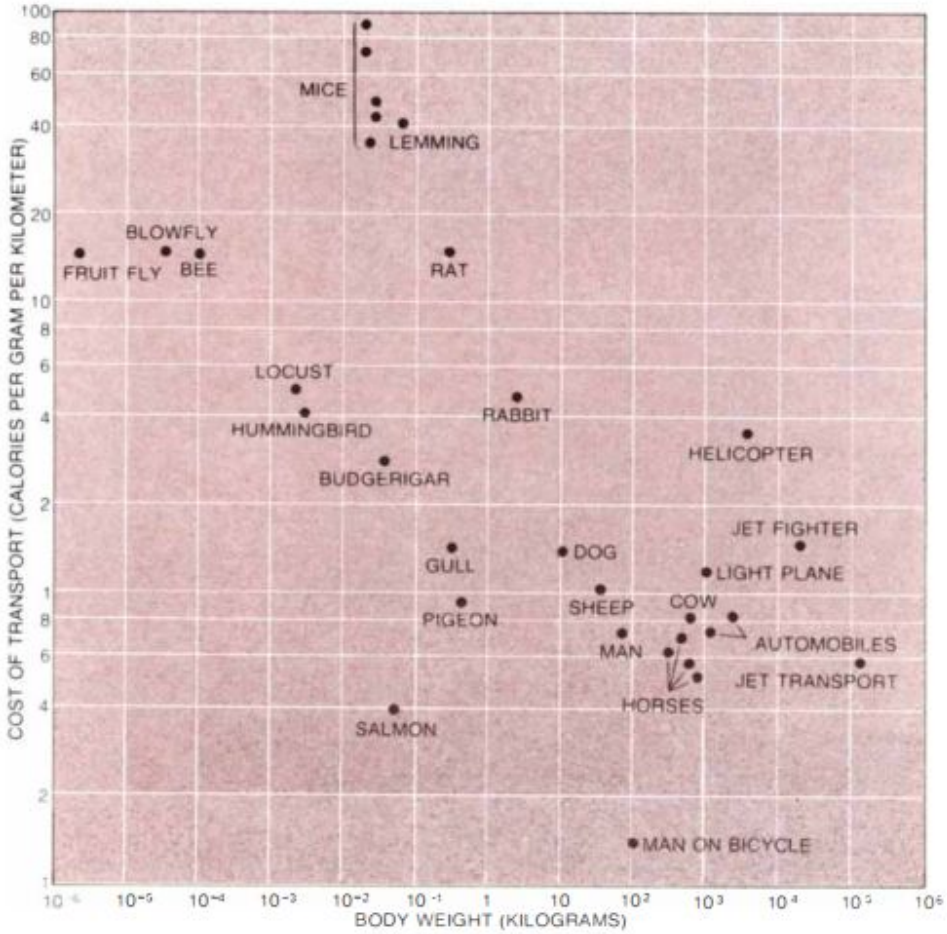
- ПК конференций highload++, Backend Conf 2019, PGIBZ и т.д.
- Главное: Postgres.ai

Postgres.ai

— amplify the database engineering power using new tools for database development, testing, and management

We need better tools


SCIENTIFIC
AMERICAN
BICYCLE TECHNOLOGY
ONE DOLLAR
March 1973


COST OF TRANSPORT (CALORIES PER GRAM PER KILOMETER)
BODY WEIGHT (KILOGRAMS)
MICE
LEMMING
BLOWFLY
FRUIT FLY
BEE
RAT
LOCUST
HUMMINGBIRD
RABBIT
HELICOPTER
BUDGERIGAR
JET FIGHTER
DOG
GULL
LIGHT PLANE
SHEEP
COW
PIGEON
AUTOMOBILES
MAN
JET TRANSPORT
HORSES
SALMON
MAN ON BICYCLE

Postgres.ai

Steve Jobs (1980)

1) We, humans, are great tool-makers.
   We amplify human abilities.

2) Something special happens
   when you have 1 computer and 1 person.

   It’s very different that having 1 computer and 10 persons.

DB migration testing – “stateful tests in CI”

What we want from testing of DB changes:

- Ensure the change is valid
- It will be executed in appropriate time
- It won’t put the system down

…and:

- What to expect? (New objects, size change, duration, etc.)

Perfect Lab for database experiments

- Realistic conditions – as similar to production as possible
  - The same schema, data, environment as on production
  - Very similar background workload
- Full automation
- “Memory” (store, share details)
- Low iteration overhead (time & money)
- Everyone can test independently

allowed to fail → allowed to learn

Database experiments with Database Lab today (2021)

- Realistic conditions – as similar to production as possible
  - The same schema, data, environment as on production
  - ~~Very similar background workload~~
- Fine automation
- “Memory” (store, share details)
- Low iteration overhead (time & money)
- Everyone can test independently

  able to fail → able to learn

Why Database Lab was created

- Containers, OverlayFS (file-level CoW)

  CI: `docker pull ... && docker run ...`

    – OK only for tiny (< a few GiB) databases

- Existing solutions: Oracle Snap Clones, Delphix, Actifio, etc.
  $$$$, not open

    – OK only for very large enterprises

Companies that do need it today

- 10+ engineers
- Multiple backend teams (or plans to split soon)
- Microservices (or plans to move to them)
- 100+ GiB databases
- Frequent releases