PostgreSQL как сервер приложений

Дмитрий Дорофеев

PGBootCamp 23, Москва

Развитие датацентричных паралельных вычислений

- Закон Мура: он всё ещё работает?
- СУБД нового поколения: NoSQL, NewSQL, MPP
  - REST API-enabled, масштабируемые
- Кажется это хороший момент затащить сервер приложений в СУБД ?
- Есть уникальная возможность сравнить 2-х звенку и классическую 3-х звенку на примере Luxms BI.
  - Luxms BI – аналитическая BI платформа

Расслабьтесь! Хайп NoSQL прошёл, SQL победил.

- **В NoSQL нет поддержки .... SQL!**
- **Гибридный NewSQL: какой-то SQL, distributed, in-memory**
  - Но нет полной поддержки стандарта ANSI SQL 99, нет хранимых процедур
- **PostgreSQL/Greenplum готовы к будущему:**
  - Основаны на реляционной алгебре (строгая математика)
  - Гарантированные, консистентные, достоверные ответы на ваши запросы
  - Горизонтально масштабируемые, экосистема расширений
  - Языки программирования внутри СУБД

SQL победил!

Kafka = KSQL: Streaming SQL for Apache Kafka
Spark = Spark SQL & DataFrames
Flink = Flink Table API & SQL
Hadoop: Hive, Phoenix, ...
Redis: RediSQL => zeeSQL
MongoDB = Atlas SQL Interface

Как работать с BIG DATA?

- **Перенести вычисления ближе к данным, ~~перенести данные ближе к вычислениям~~:**
  - **Датацентричный MapReduce = Hadoop ecosystem**
  - **Сюрприз: PL/* внутри MPP СУБД**
  - **PL/pgSQL в GPDB можно использовать как MapReduce**
- gpmapreduce
  - **Runs Greenplum MapReduce jobs as defined in a YAML specification document.**

Можно ли эту идею применить для обычных приложений?

Рецепт давно известен: 2-х звенка
https://dl.acm.org/doi/10.1145/3274856.3274869 (ACM, ноябрь 2018)

Недостатки:

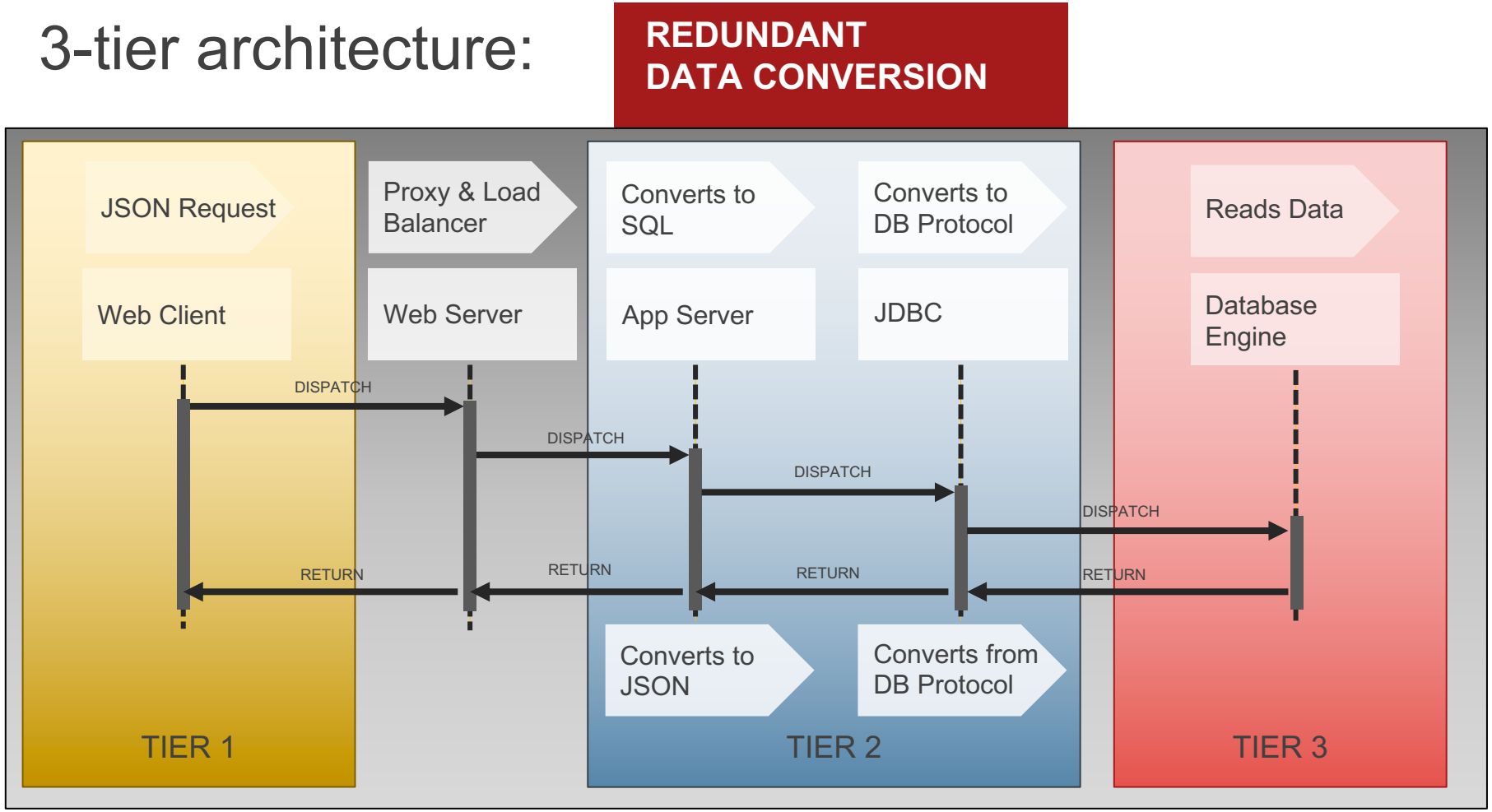
- Нарушен принцип разделения ответственности
- Сложное обновление версий
- Сложная эксплуатация и поддержка (нет :)

Достоинства:

- Экосистема и надёжность PostgreSQL
- Изменение кода в работающей системе без перезапуска ???
- Скорость

Датацентричный подход для разработки приложений

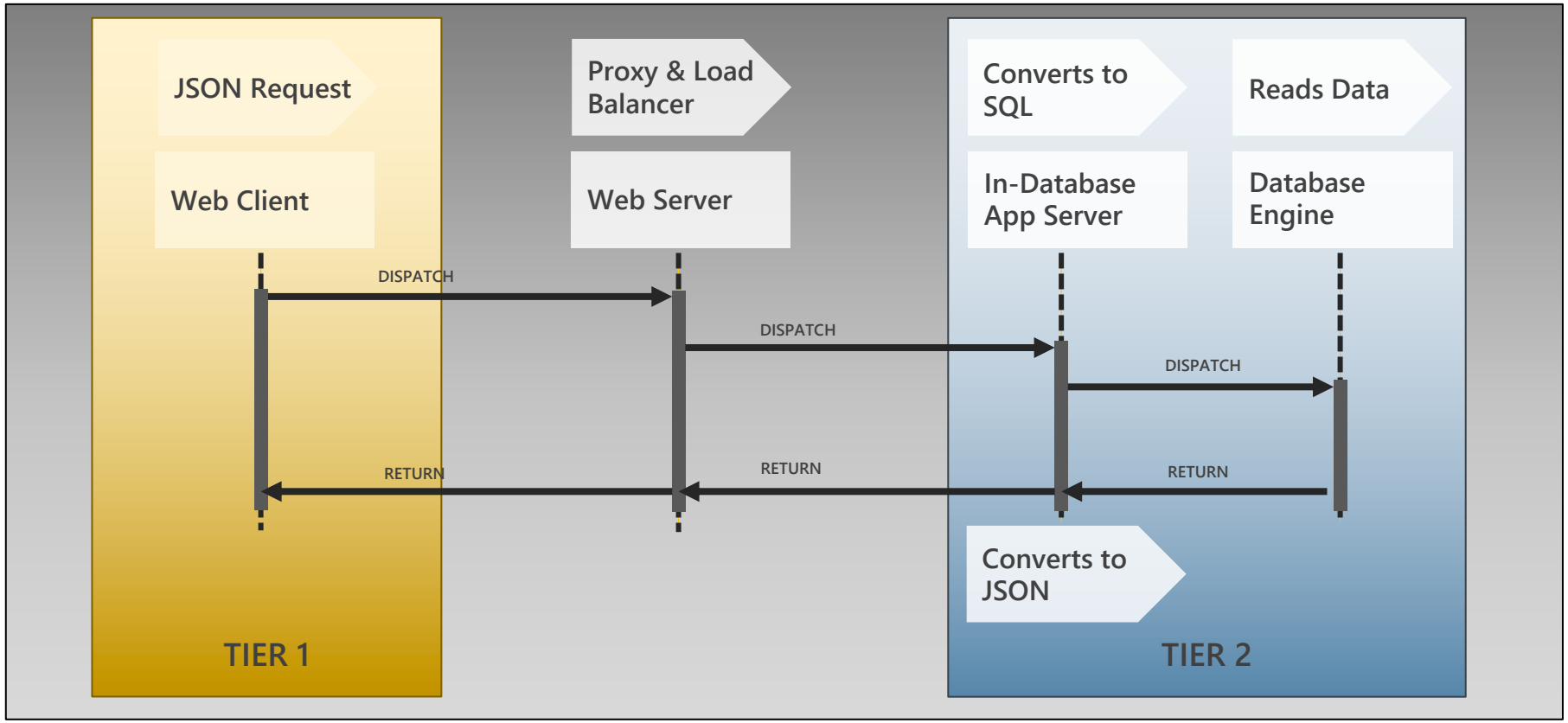
3-tier architecture:



Датацентричный подход для разработки приложений

Data-centric 2-tier architecture:

IN-DATABADE ANALYTICAL SERVER (PostgreSQL)



Разработка на PL/pgSQL

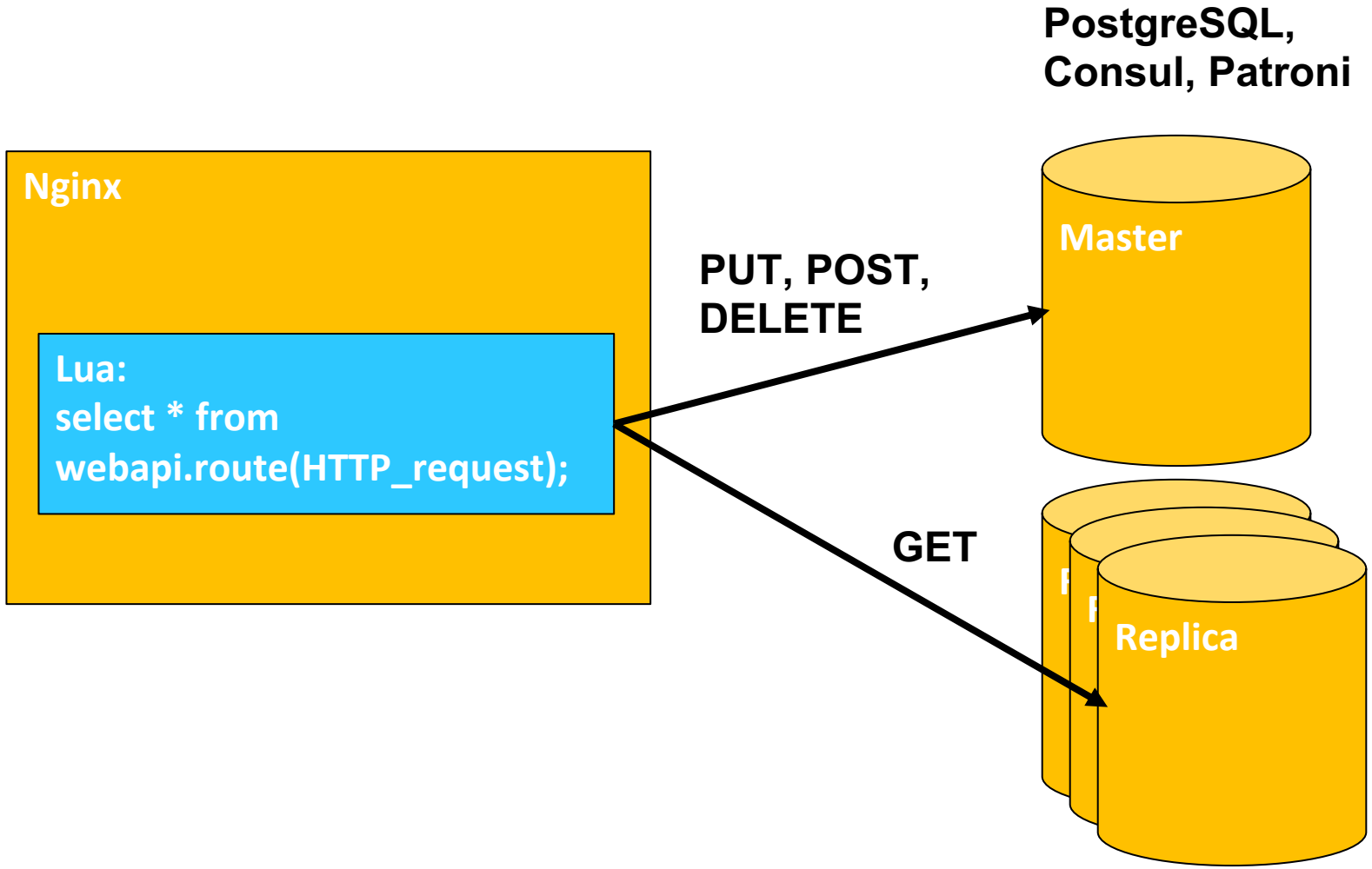
Это очень похоже на Smalltalk/LISP/Erlang:

Правка кода в живой, постоянно работающей системе

Мы должны учиться у проекта Pharo (современный и бесплатный Smalltalk):

	Pharo	PostgreSQL
Source code management	Iceberg	??????
Debugging	Live debugger	omnidb.org
Code quality	Live Quality Assistant (lint on steroids)	plpgsql_check
Unit testing	Sunit	pgTAP

Nginx + Lua


PostgreSQL,
Consul, Patroni
Nginx
Lua:
select * from
webapi.route(HTTP_request);
PUT, POST,
DELETE
Master
GET
Replica

Как сделать HTTP запрос с помощью SQL

```
SELECT * FROM webapi.route('POST',
'/api/data/ds_270/colors',
'{"luxmsbi-user-session":"secret-session-key"}'::JSON,
'{"version":"2.0","cube":{
    "parameters":[2800],
    "metrics":[35],
    "locations":[10001,10002],
    "periods":[16052]}}'::JSON,
'{"debug":true}'::JSON);
```

HTTP METHOD

URL

HEADER

BODY

QUERY STRING

```
-[ RECORD 1 ]-------------------------------------------------------------
status | 200
headers| {"content-type": "application/json"}
body   | [{"metric_id":35, "loc_id":10008, "period_id":16052,"color":"red"}]
```

HTTP Middleware

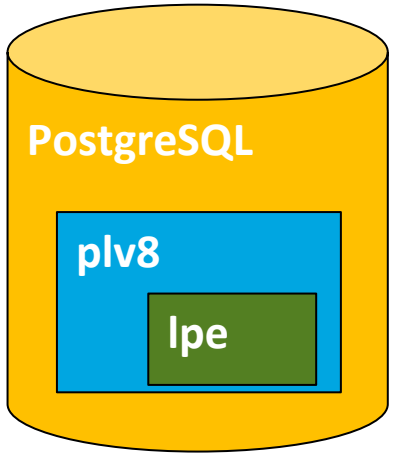
http_method	src_uri_template	dst_handler_name	auth_functions
PATCH	/api/db/:dataset:ds.:table.:column/:id:any	db_bipath	{cookie,user_crud_operations}
PATCH	/api/db/:dataset:ds.:table	db_multirows	{cookie,user_crud_operations}
DELETE	/api/dataset/:dataset:ds/data	clear_dataset	{cookie,user_to_single_dataset}
GET	/srv/presentations/:presentation.pdf	presentation_pdf	{cookie}
GET	/api/bookmarks	bookmarks	{cookie}
GET	/api/bookmarks/:id:bipath	bookmarks_bipath	{cookie}
POST	/api/bookmarks	bookmarks	{cookie,user_to_single_dataset}

аутентификация/авторизация
Шаблоны для URI
Разбор URI
Передача параметров URI в plSQL функции
Локализованные сообщения об ошибках

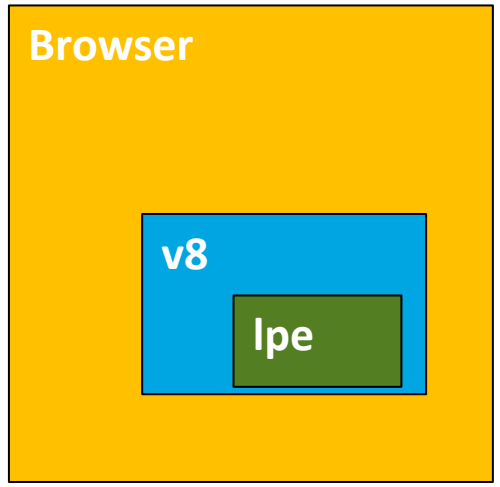
LPE: встроенный язык программирования

Десятое правило Гринспена:

Любая достаточно сложная программа на Си или Фортране содержит заново написанную, неспецифицированную, глючную и медленную реализацию половины языка Common Lisp.



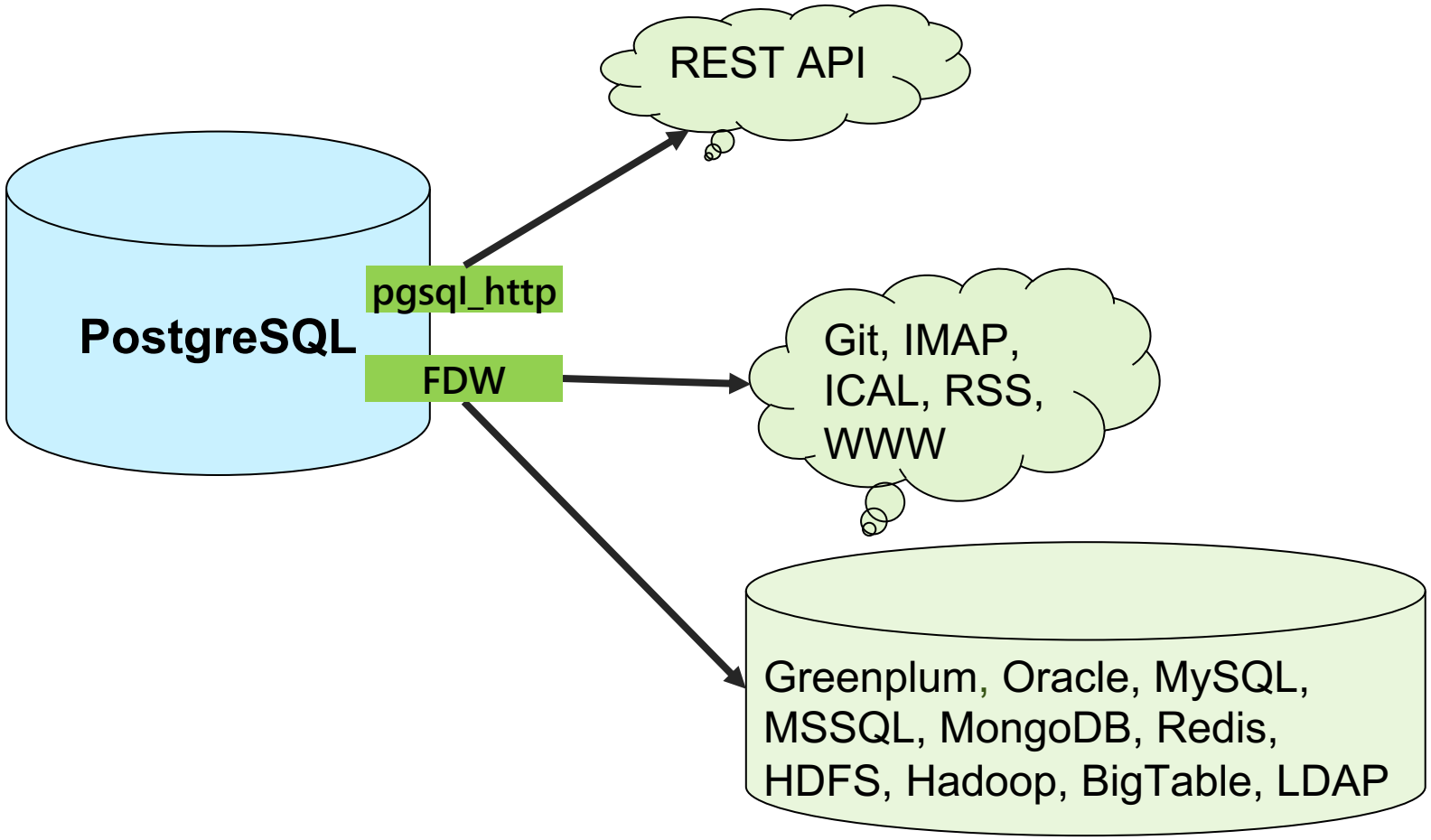
Одна реализация LISP работает и в PostgreSQL и в браузерах



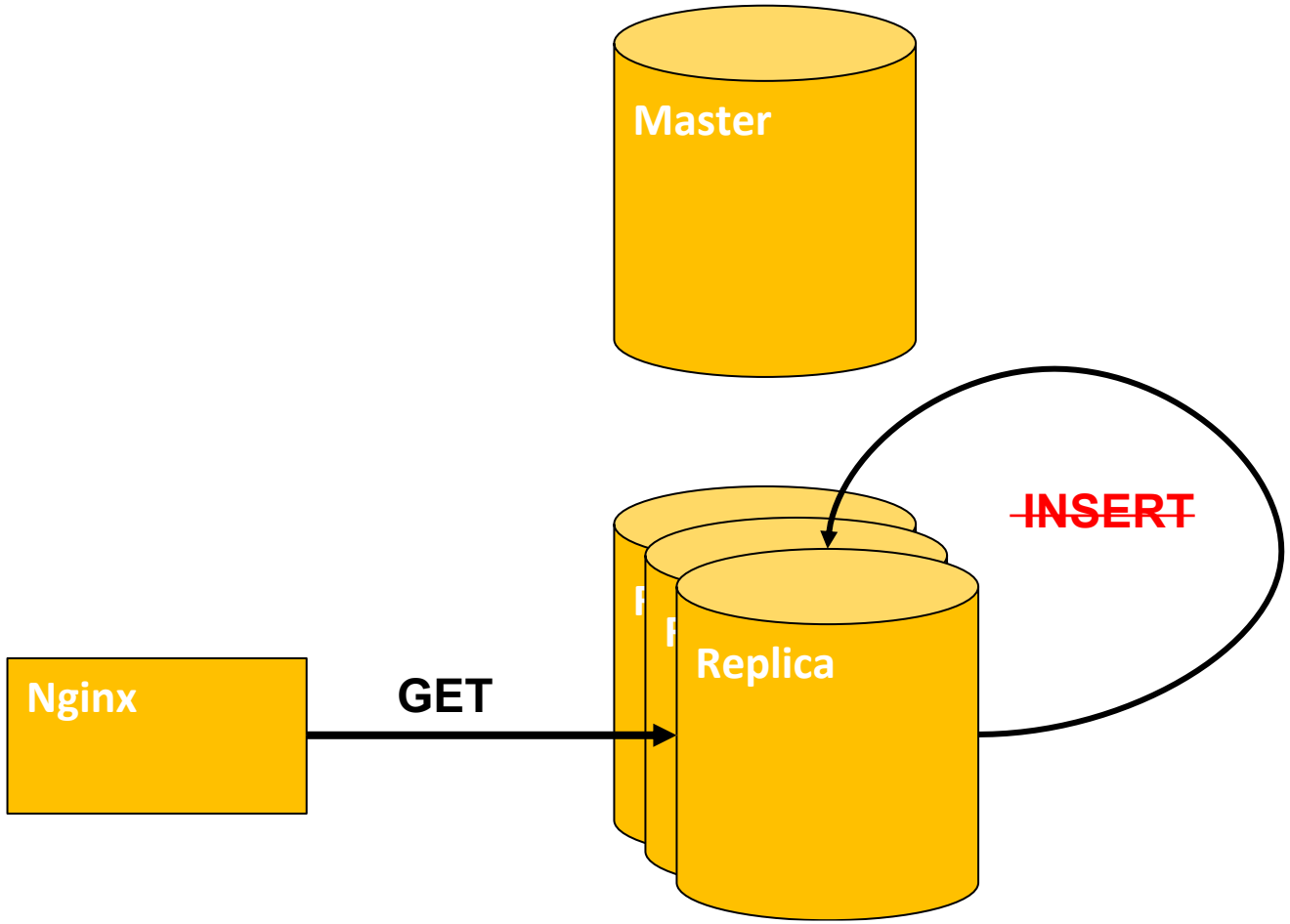
LPE

```
begin(
cp(["http", "resp", "body", "preferred_username"],
   ["db", "adm", "users", "username"]),
assoc_in(db,
         ["adm","users","sys_config","ext_groups"],
         []),
assoc_in(db,
          ["adm","users","sys_config","ext_domain"],
          "DMN"),
assoc_in(db,
         ["adm","users","email"],
         str(
              http.resp.body.preferred_username,
              "@nowhere.com")))
```

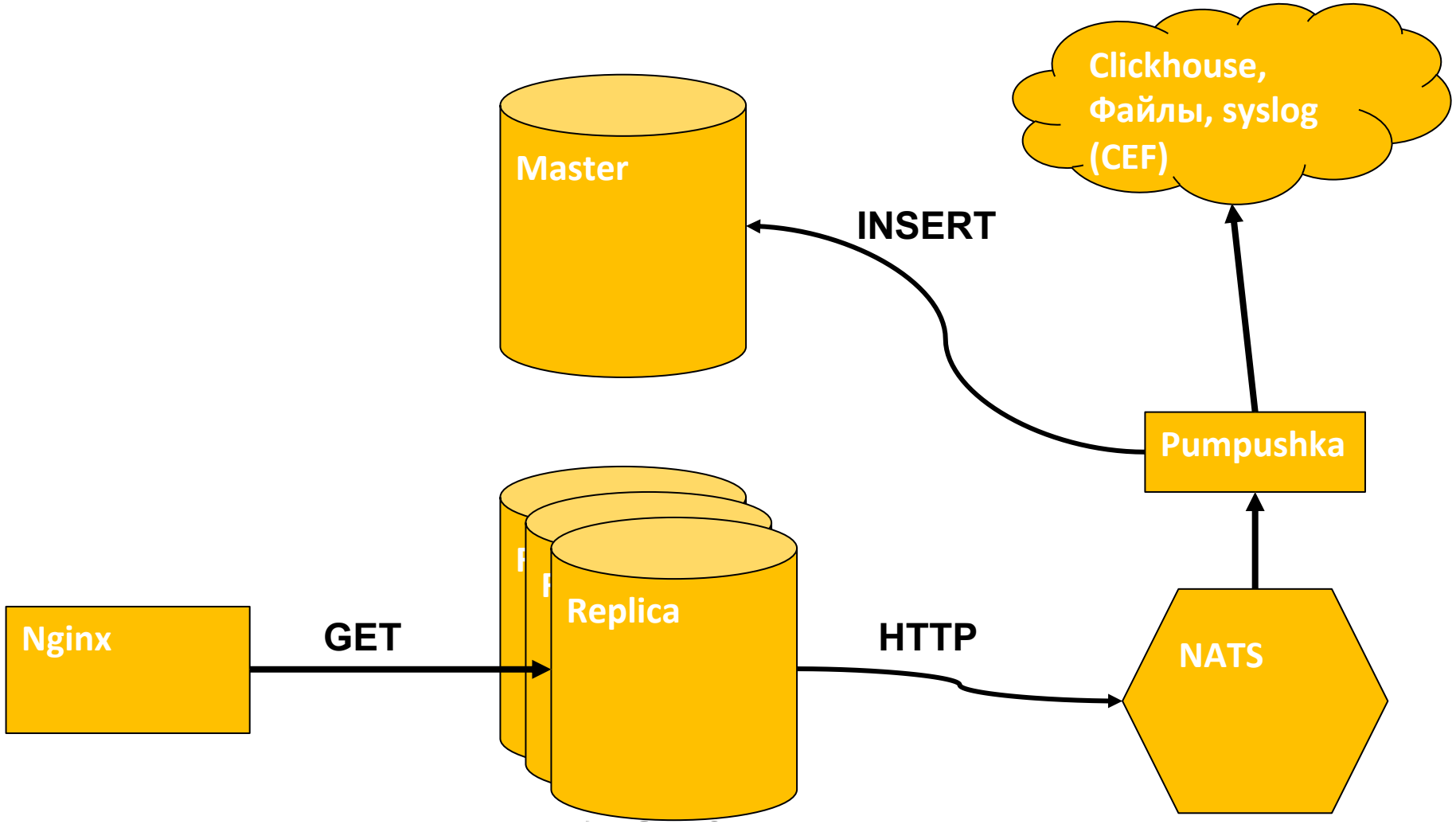
POSTGRESQL EXTENSIONS



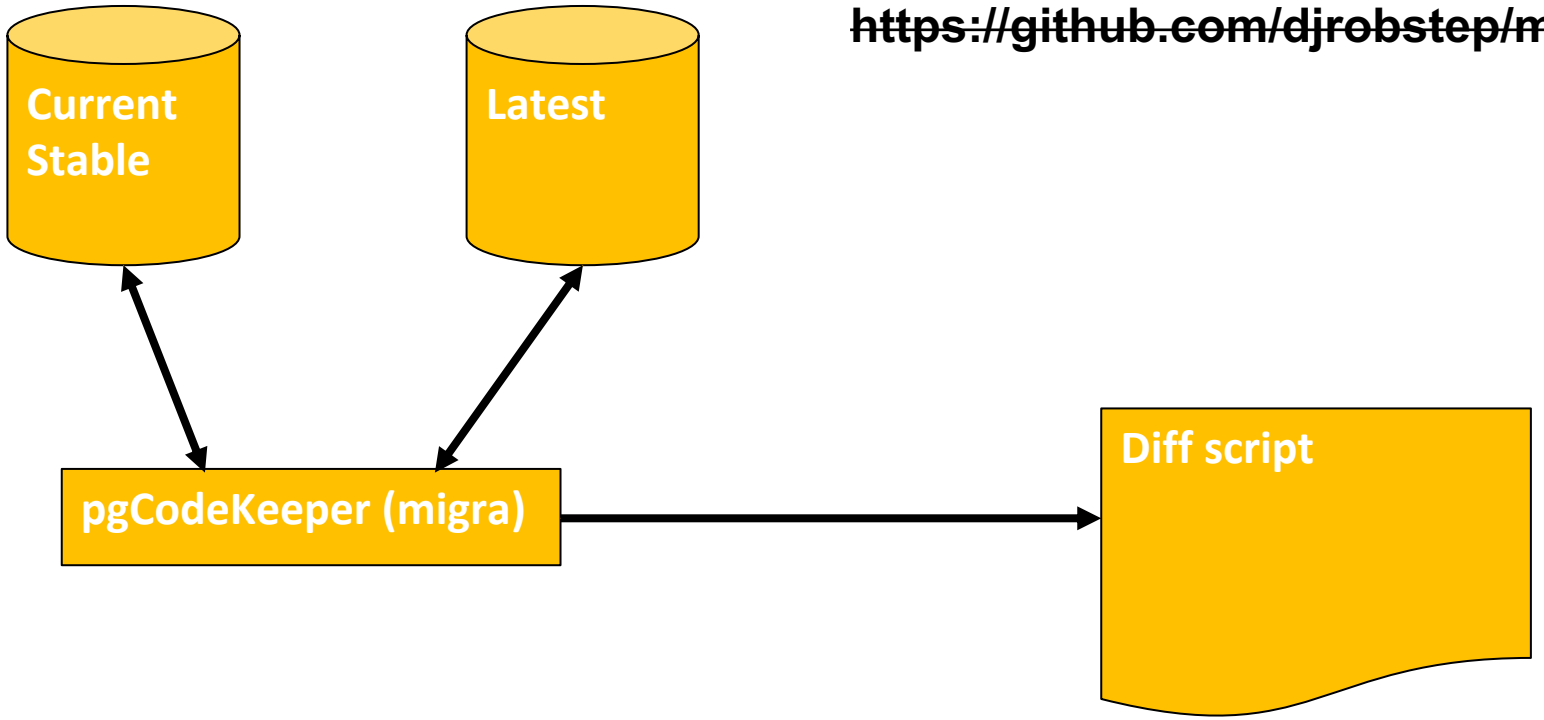
Аудит действий пользователей 1



Аудит действий пользователей 2



Выпуск новых версий



pgcodekeeper.org

~~https://github.com/djrobstep/migra~~

Tracing (немного помечтаем)

https://opentracing.io/ = нет поддержки PL/pgSQL

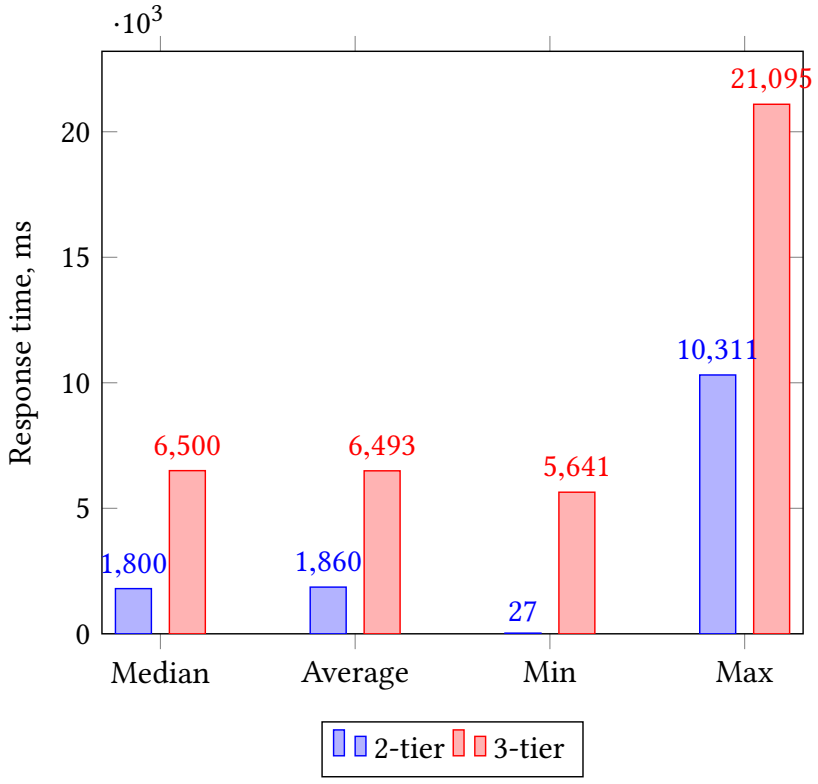
Но есть поддержка DTrace в PostgreSQL :-)

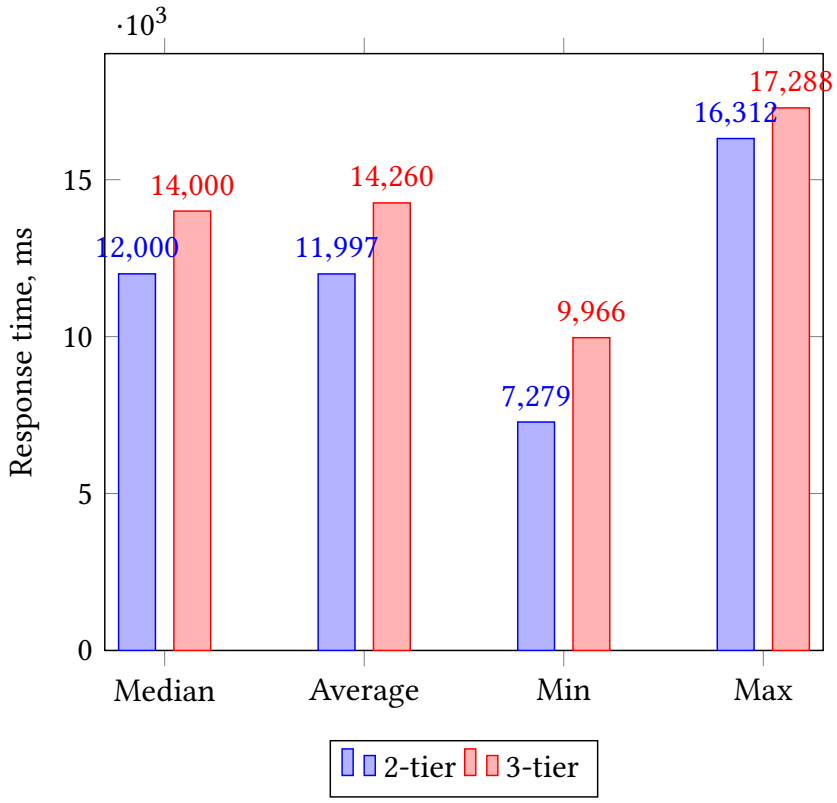
Но там нет поддержки PL/pgSQL :-(

https://github.com/bigsql/plprofiler
https://github.com/EnterpriseDB/pldebugger

Производительность 1

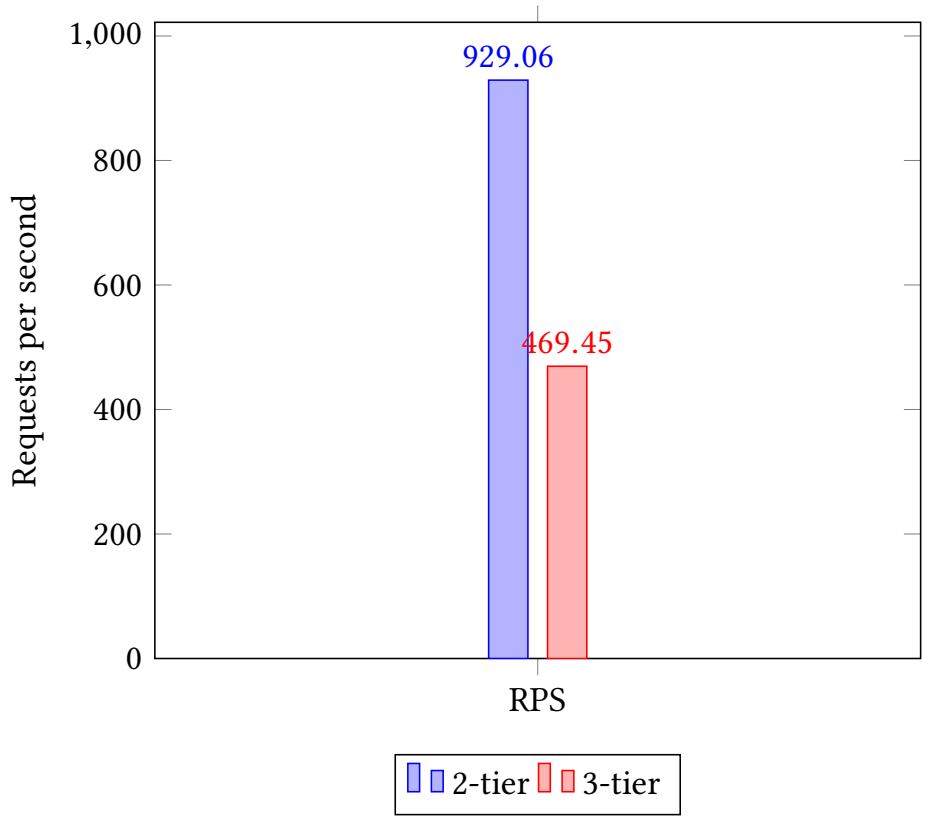
Lightweight and Medium Workloads: Latency

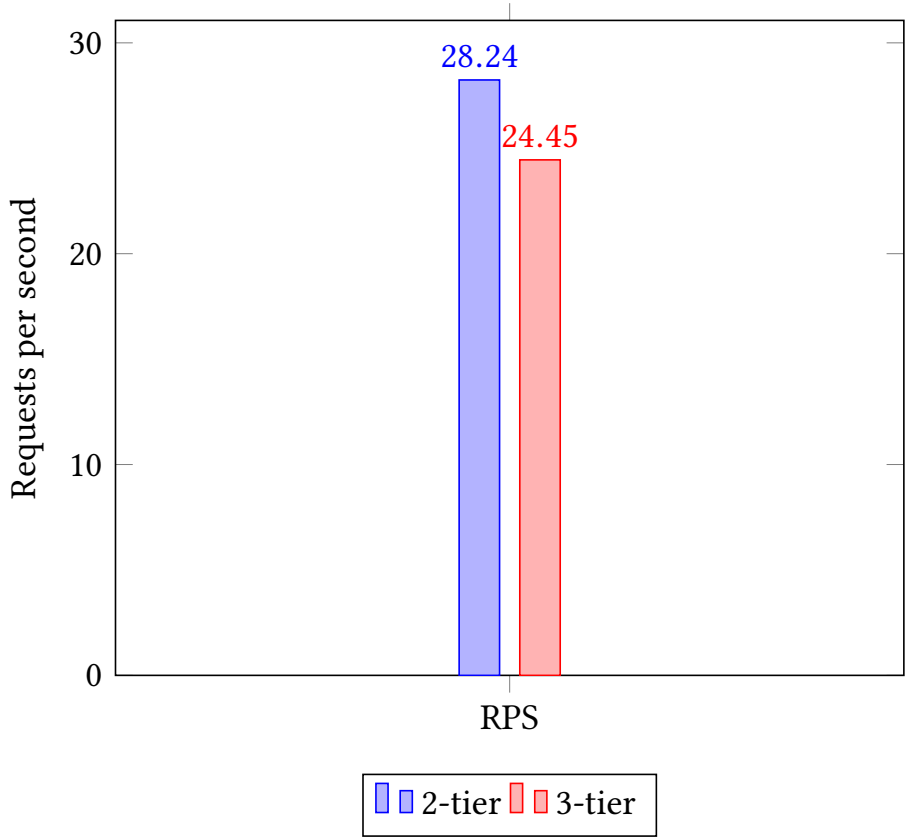




Производительность 2

Lightweight and Medium Workloads: RPS





Результаты тестов в облаке VK

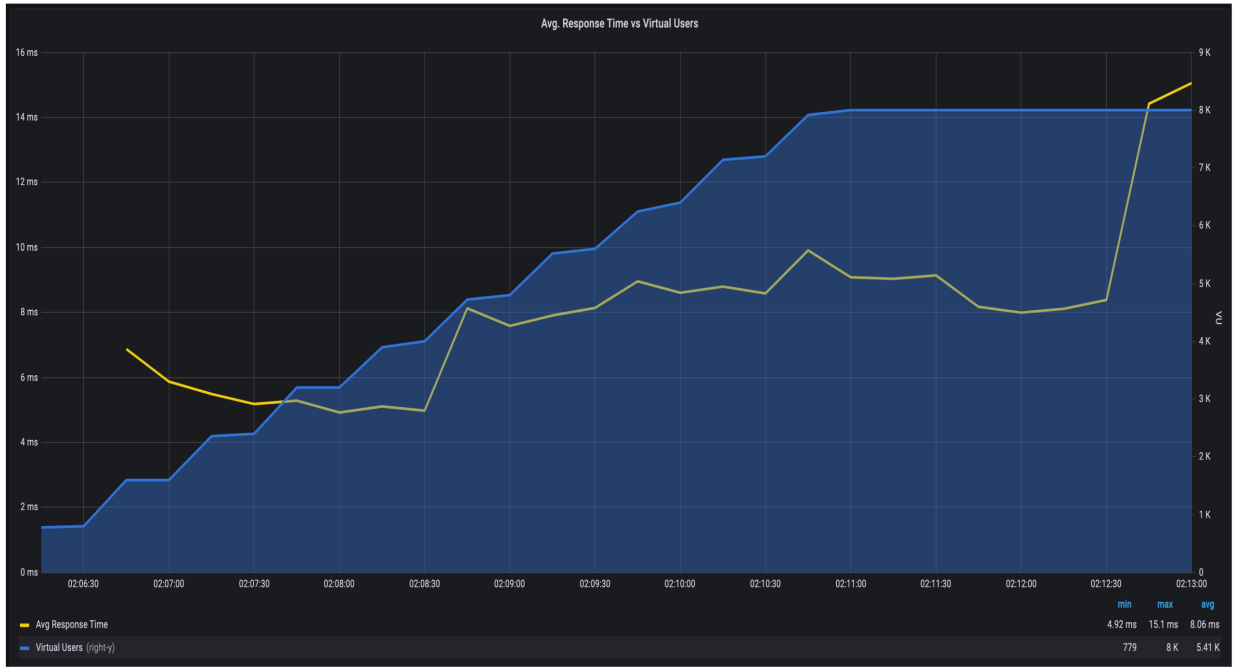
№ исп.	RPS(max)	RPS(avg)	CPU-avg(BI)	CPU-max(BI)	CPU-avg(Ch)	CPU-max(Ch)	Virtual-users
1	540	426	23%	32%	37%	100%	1200
2	494	356	48%	75%	29%	94%	1000
3	272	199	53%	79%	12%	41%	550

№ исп.	CPU	RAM
1	16	16
2	8	16
3	4	16

Конфигурация серверных ресурсов(Clickhouse) - 16 CPU / 64 GB RAM

Результаты на железе 36CPU / 16GB RAM

№ исп.	RPS(max)	RPS(avg)	CPU-avg(BI)	CPU-max(BI)	Virtual-users
1	5481	4400	15%	23%	8000



Спасибо за внимание!
Задавайте вопросы

Дмитрий Дорофеев dima@luxmsbi.com

PGBootCamp 23, Москва