Яндекс Облако

Анализ логов с помощью баз данных

https://github.com/rekby/dump-2019

Тимофей Кулин, разработчик

Работа с логами в базе данных

- Хранение логов
- Базы данных
- Тестовый стенд
- Тестирование
- Итоги

Хранение логов

Текстовый лог

Время, Событие, Сообщение
Время, Событие, Сообщение

Тривиально в реализации

Лёгкая фильтрация по датам

Легко удалять старые данные

Писать программу для обработки

Полная обработка каждый раз

Построчное хранение в базе данных

Время	Событие	Сообщение
Время	Событие	Сообщение

Обработка сразу нужных полей

Анализ данных через запросы

Чтение лишних данных с диска

Засорение кэша ОС

Колоночное хранение данных

С диска читаются только нужные столбцы
Эффективное сжатие

Медленное чтение отдельных записей

Тестовый набор, объём данных

Логи активности на github.com https://www.gharchive.org

Период	Строк	Объём	Объём, gzip
Январь 2015	7 028 566	1.2Гб	0.6Гб
2015 год	95 492 871	16.1Гб	7.4Гб
2015-2018	690 073 952	118.2Гб	52.4Гб

Тестовый набор, структура

Структура данных

Поле	Тип
id	int64
actor_id	int32
actor_login	string
repo_id	int32
repo_name	string
created_at	date time
head	**bytes 20**
before	**bytes 20**
size	int16
distinct_size	int32

Задача

- Хранить логи
- Исследование логов человеком
- Возможность удалять устаревшие данные
- Найти самый популярный коммит на github.com

Базы данных

Postgres
Postgres-partitions
Postgres-cstore_fdw
Clickhouse
Memsql

Postgres-11

- Популярная строковая база данных
- work_mem увеличен до 3Гб

Варианты

- BRIN-index по дате
- Partitions

Postgres-cstore_fdw

- Расширение postgres
- Колоночное хранение
- Нельзя менять или удалять данные внутри таблицы
- Только автоматические индексы пропуска

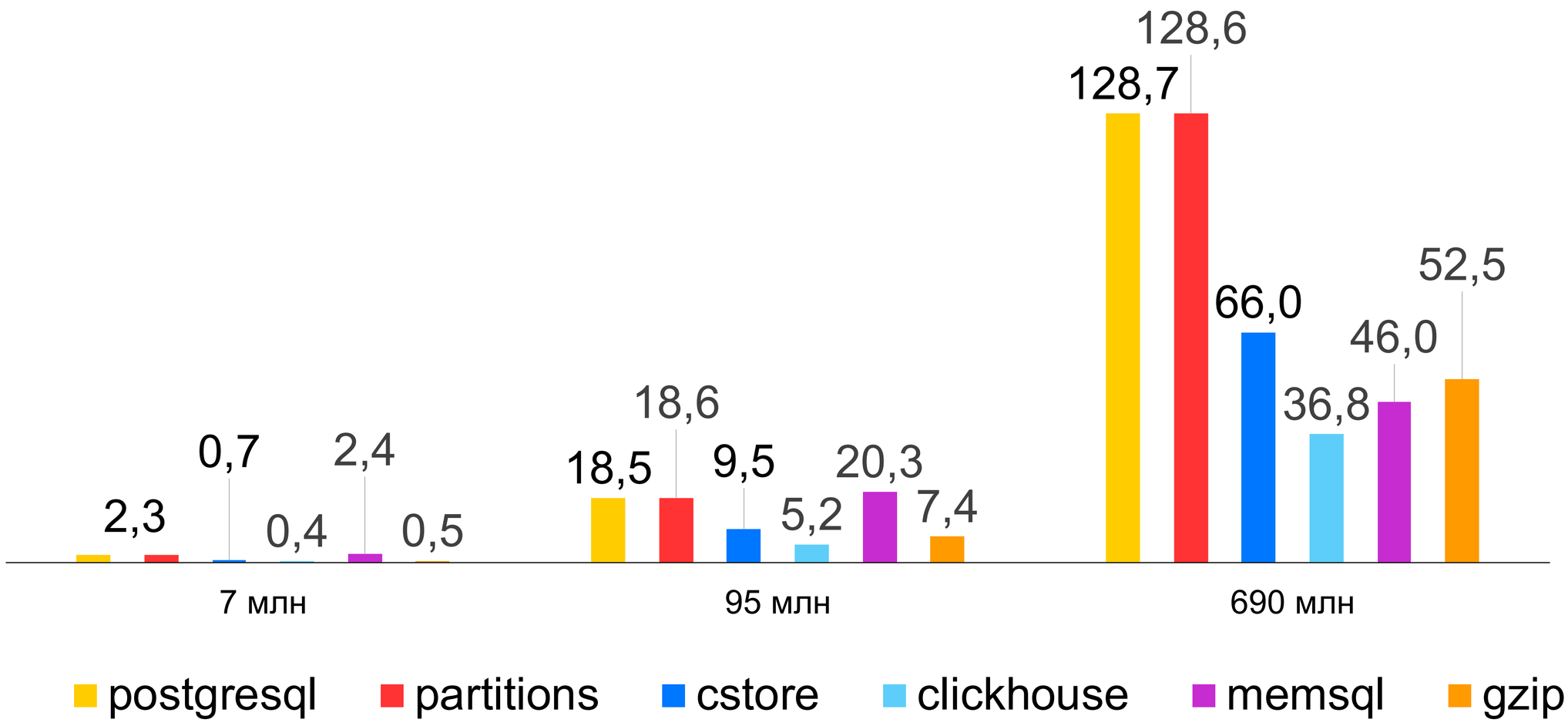
Clickhouse

- Колоночная база данных
- Оптимизирована на скорость обработки
- Сложно менять данные
- Удаление только партициями

Memsql

- Гибридная база данных
- Можно менять данные
- Можно использовать драйвер MySQL
- Конвейеры загрузки данных – например из S3
- Закрытый код, платная (при кластерах > 4 нод)

Место на диске (Гб)



Тестовый стенд

- CPU: Intel(R) Xeon(R) CPU E5-2620 v2 @ 2.10GHz (6*2 ядер)
- RAM: 8G (Ограничение через параметр ядра mem)
- ОС: Debian GNU/Linux 10
- Базы данных на выделенном диске SATA HDD 7200 оборотов
- Система, swap – на SSD
- Docker
- Свободен от других задач

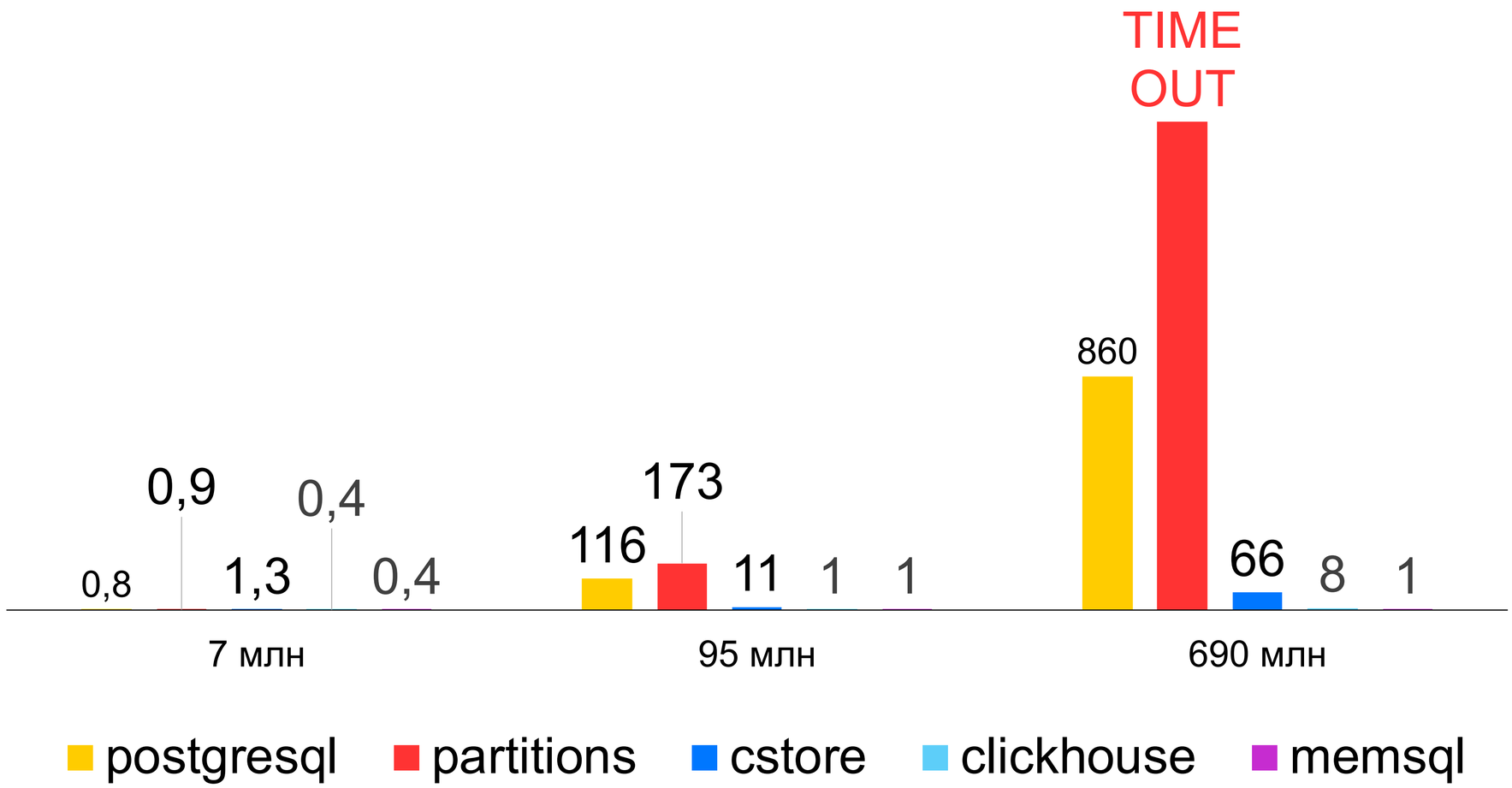
Способ измерения

- Единый порядок сортировки
- Максимальное время выполнения запроса – 30 минут
- Запрос выполняется 3 раза подряд
- Записывается средний из результатов

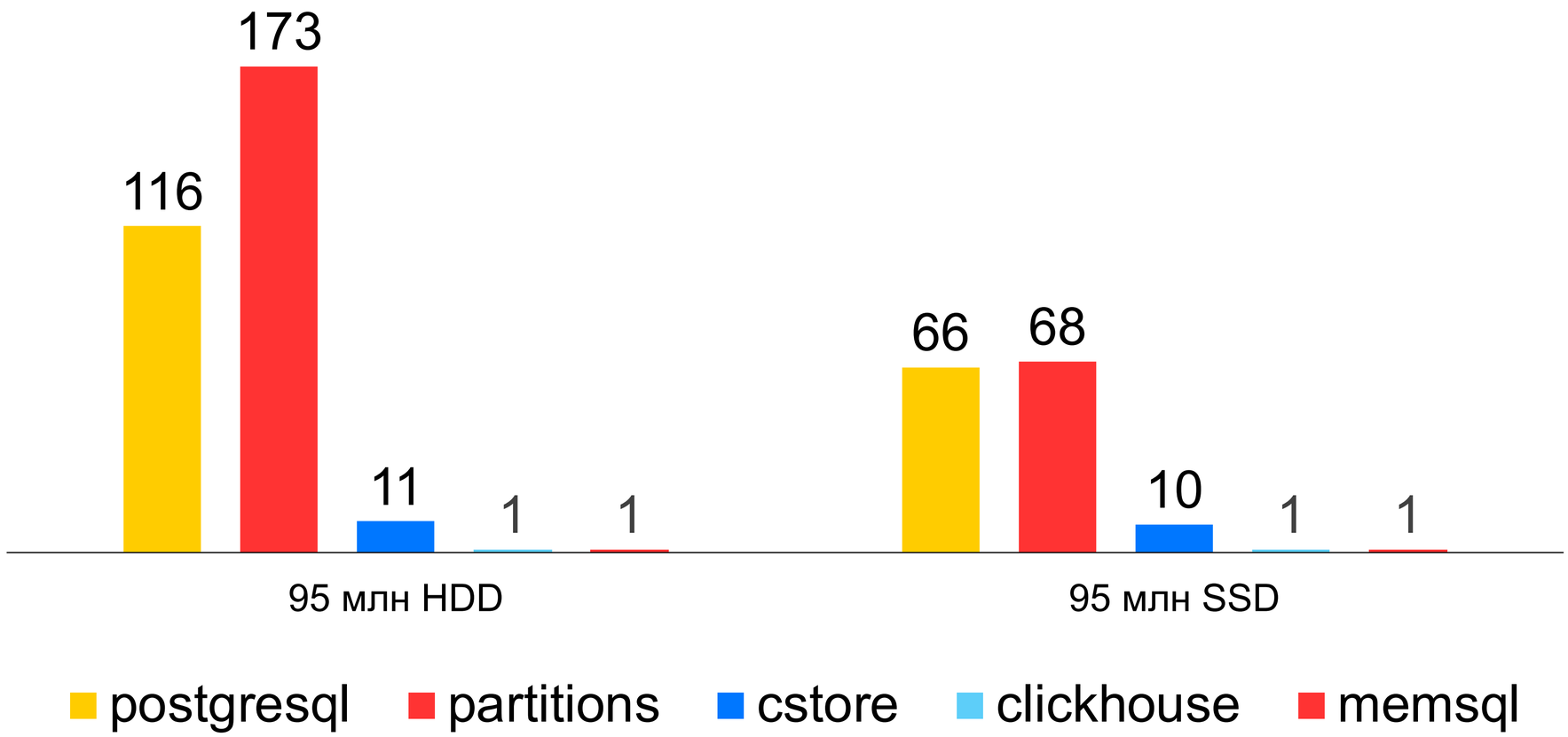
Простой запрос - количество

```
SELECT COUNT(*) FROM push
```

Количество, секунды



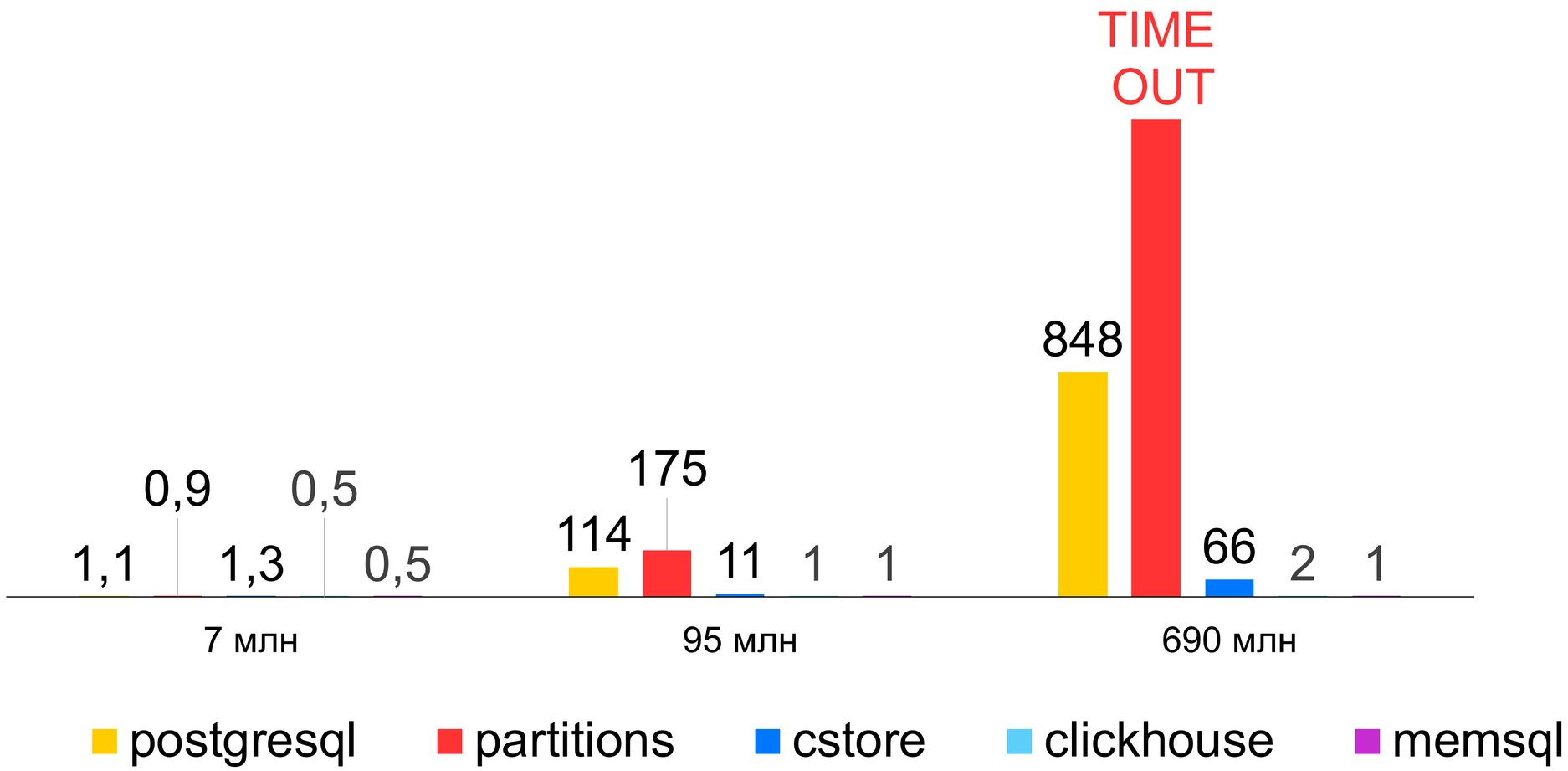
Количество, секунды



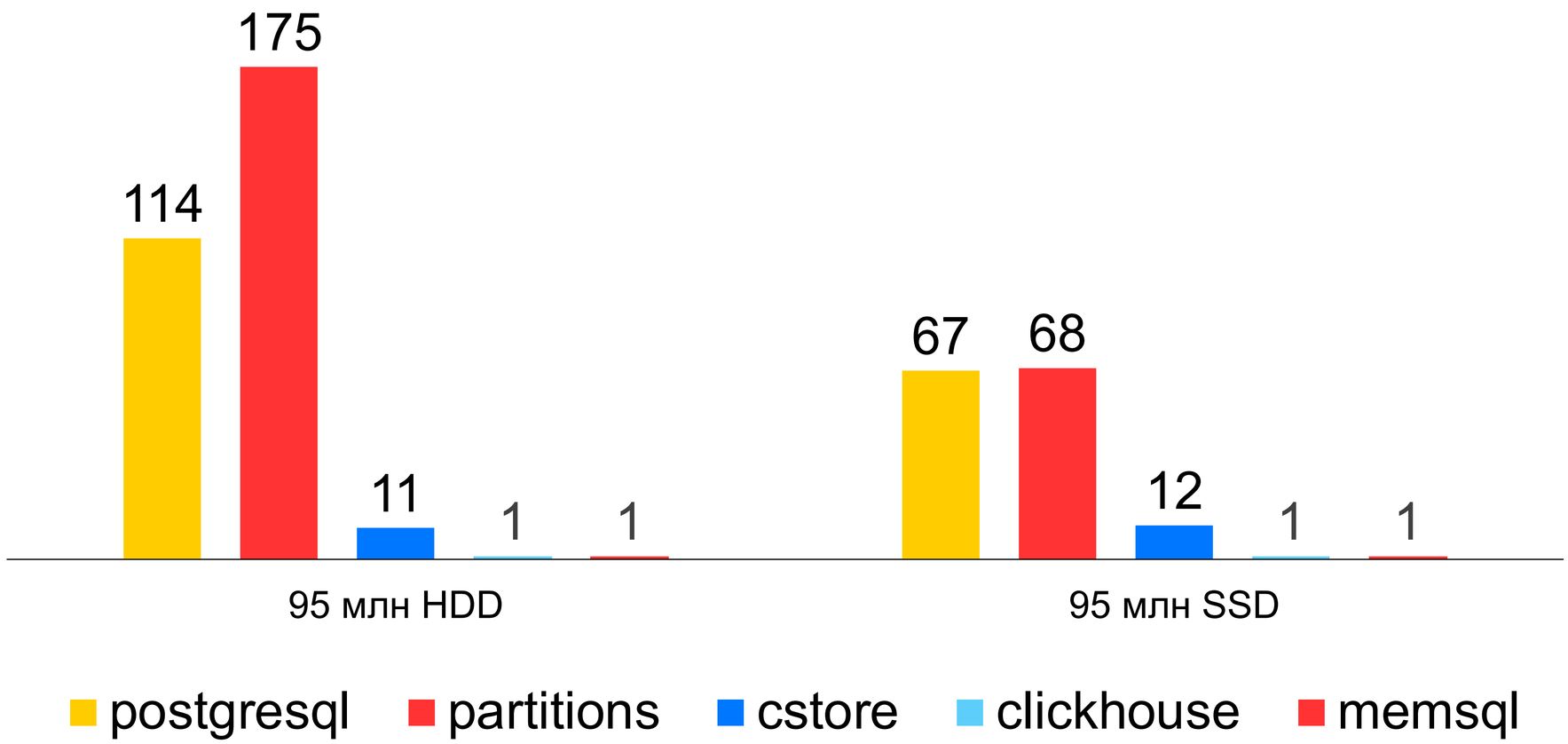
Фильтр

```
SELECT COUNT(*) FROM push
WHERE actor_login='rekby'
```

Фильтр, секунды



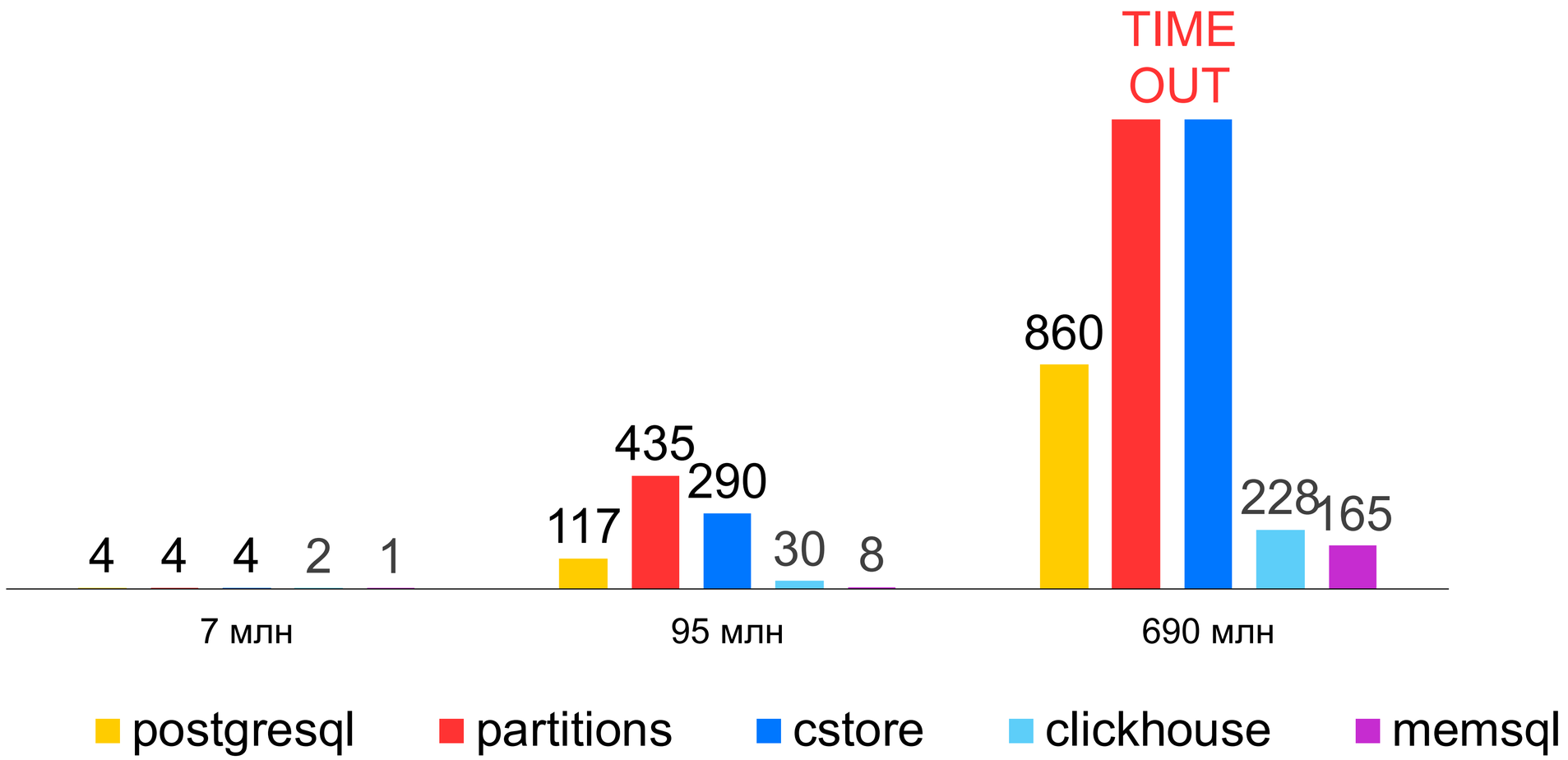
Фильтр, секунды



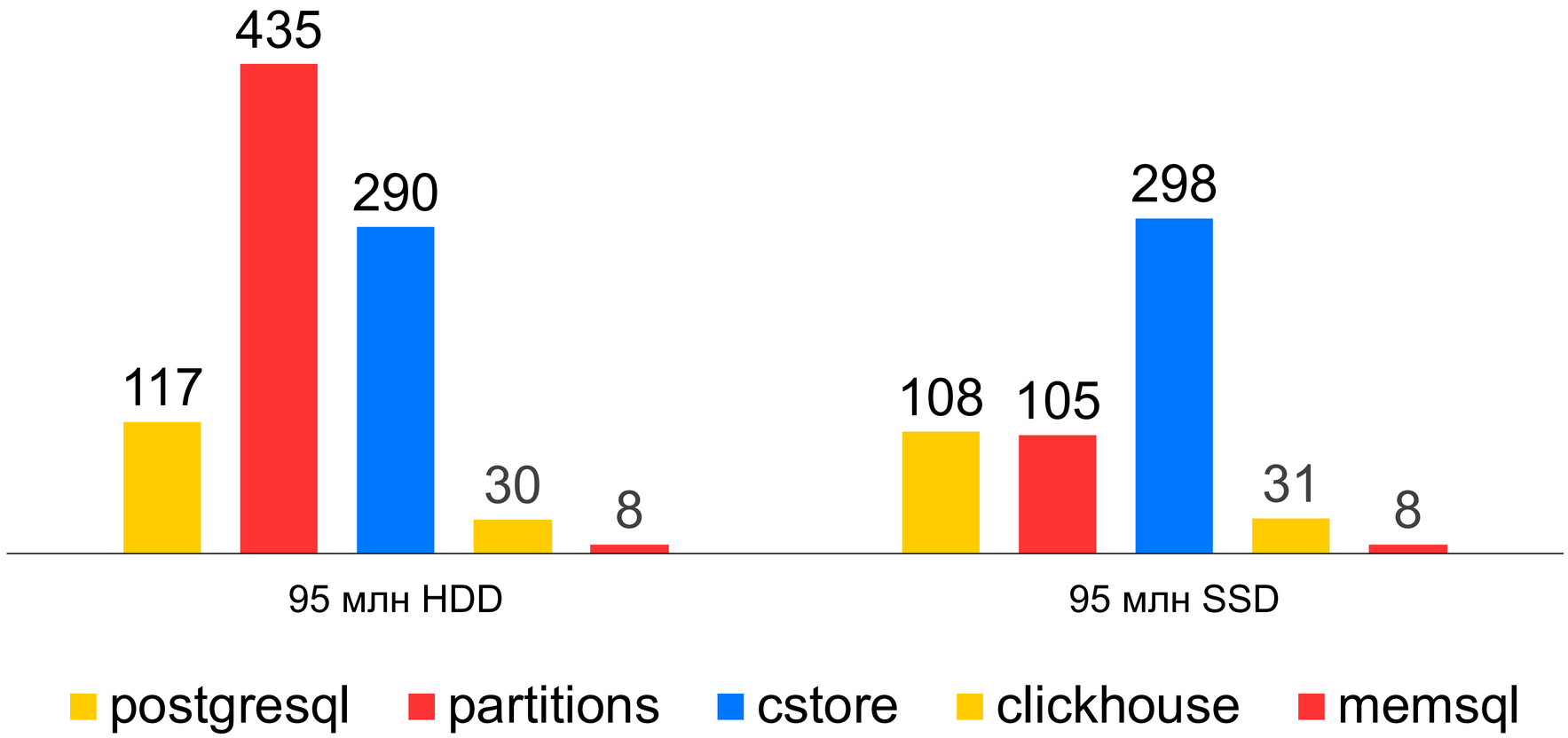
Фильтр по регулярному выражению

```
SELECT COUNT(*) FROM push
WHERE substring(repo_name from '(^.*)/') = 'rekby'
```

Регулярное выражение, секунды



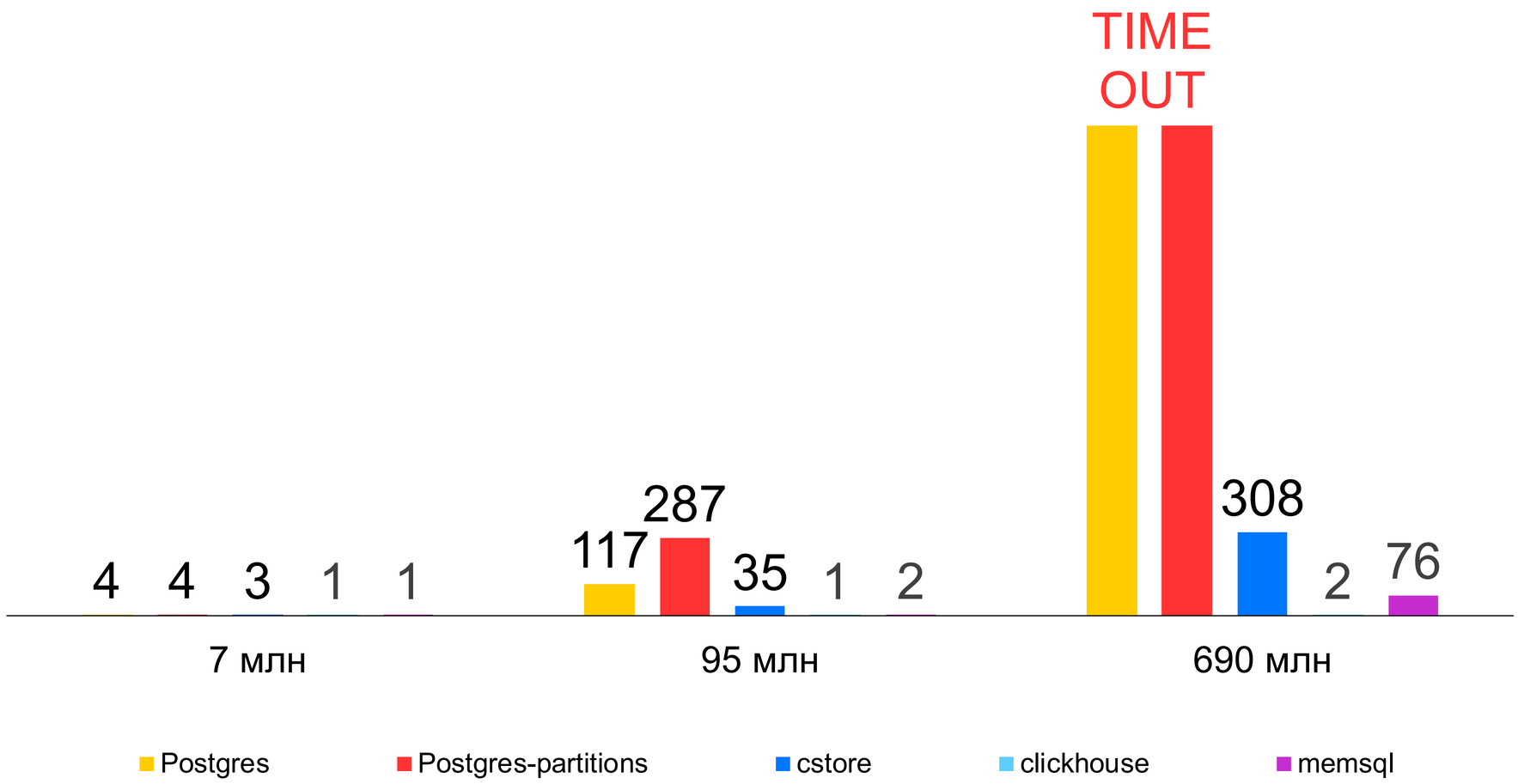
Регулярное выражение, секунды



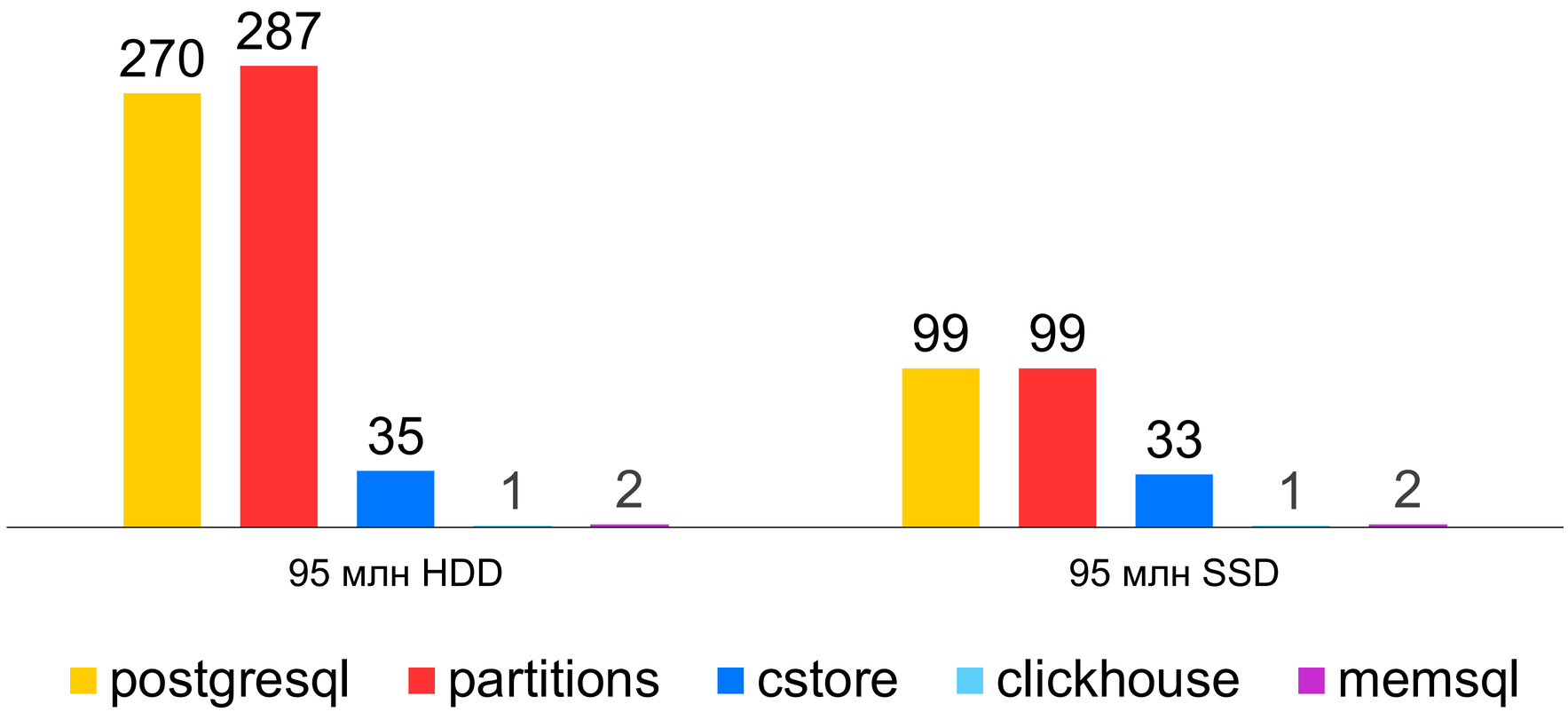
Простая статистика

```
SELECT
extract('hour' FROM created_at) AS hour, count(*) AS cnt
FROM push
GROUP BY hour
ORDER BY hour
```

Простая статистика



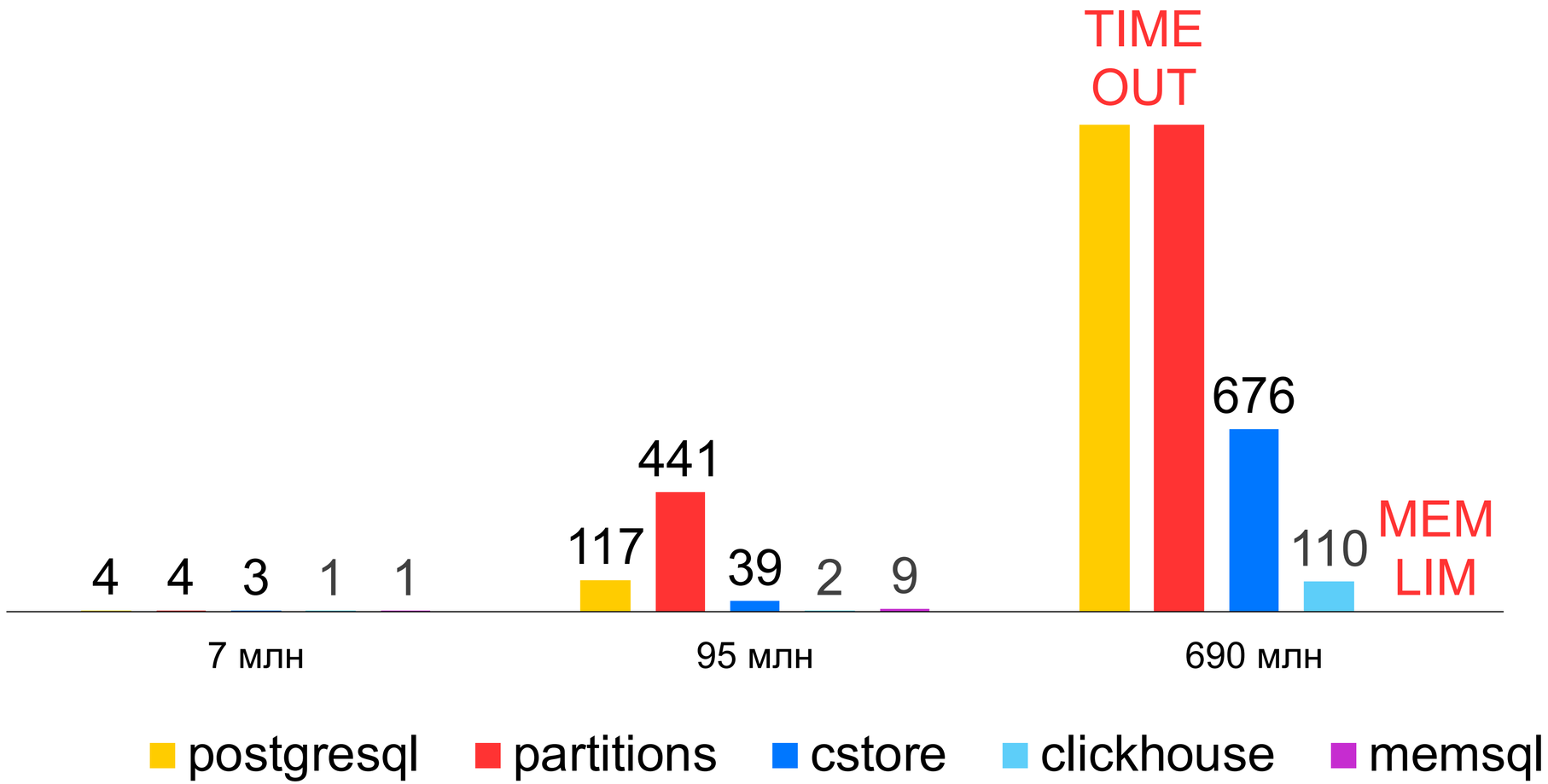
Простая статистика



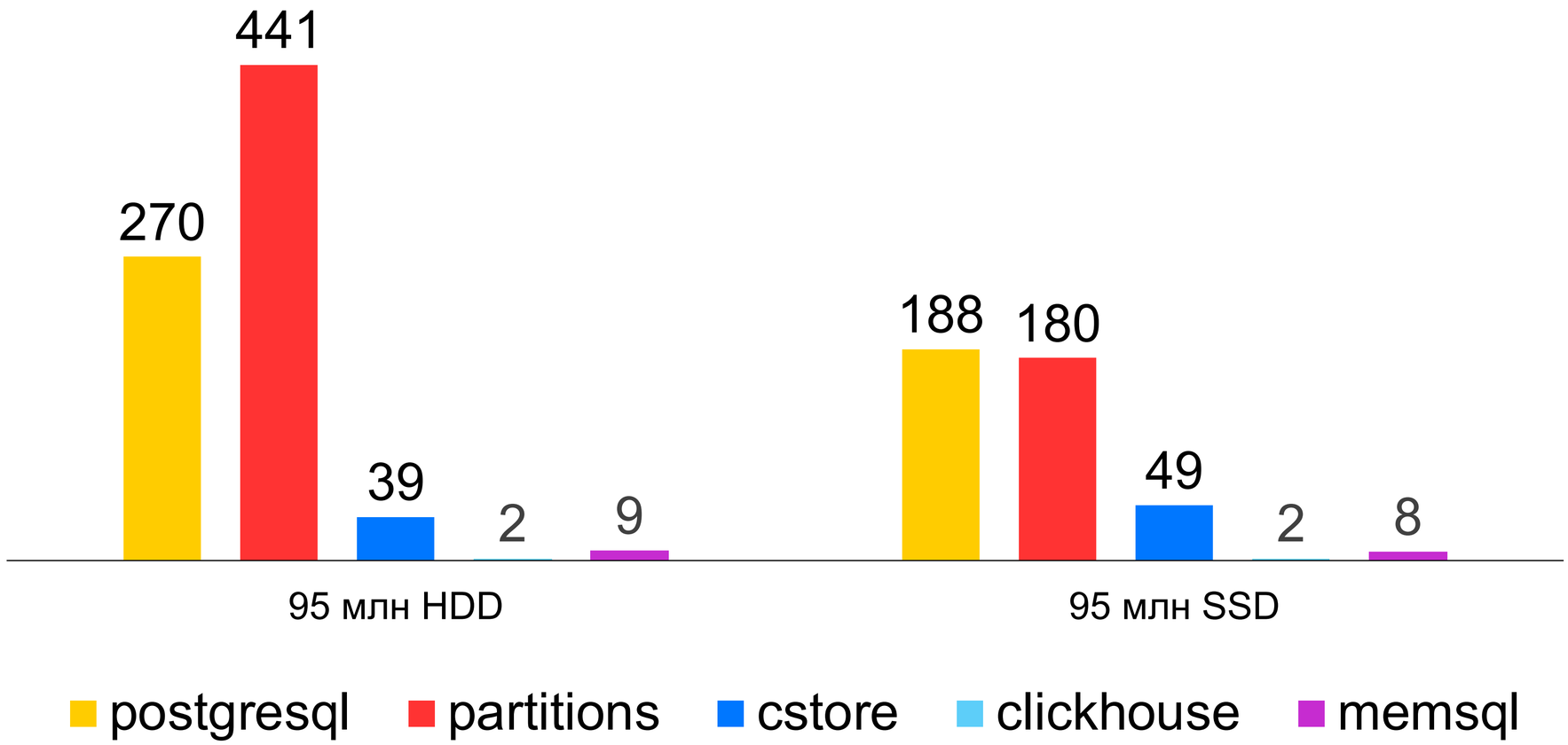
Средняя статистика

```
SELECT repo_name, count(*) as cnt
FROM push
GROUP BY repo_name
ORDER BY cnt DESC
LIMIT 5
```

Статистика, средняя



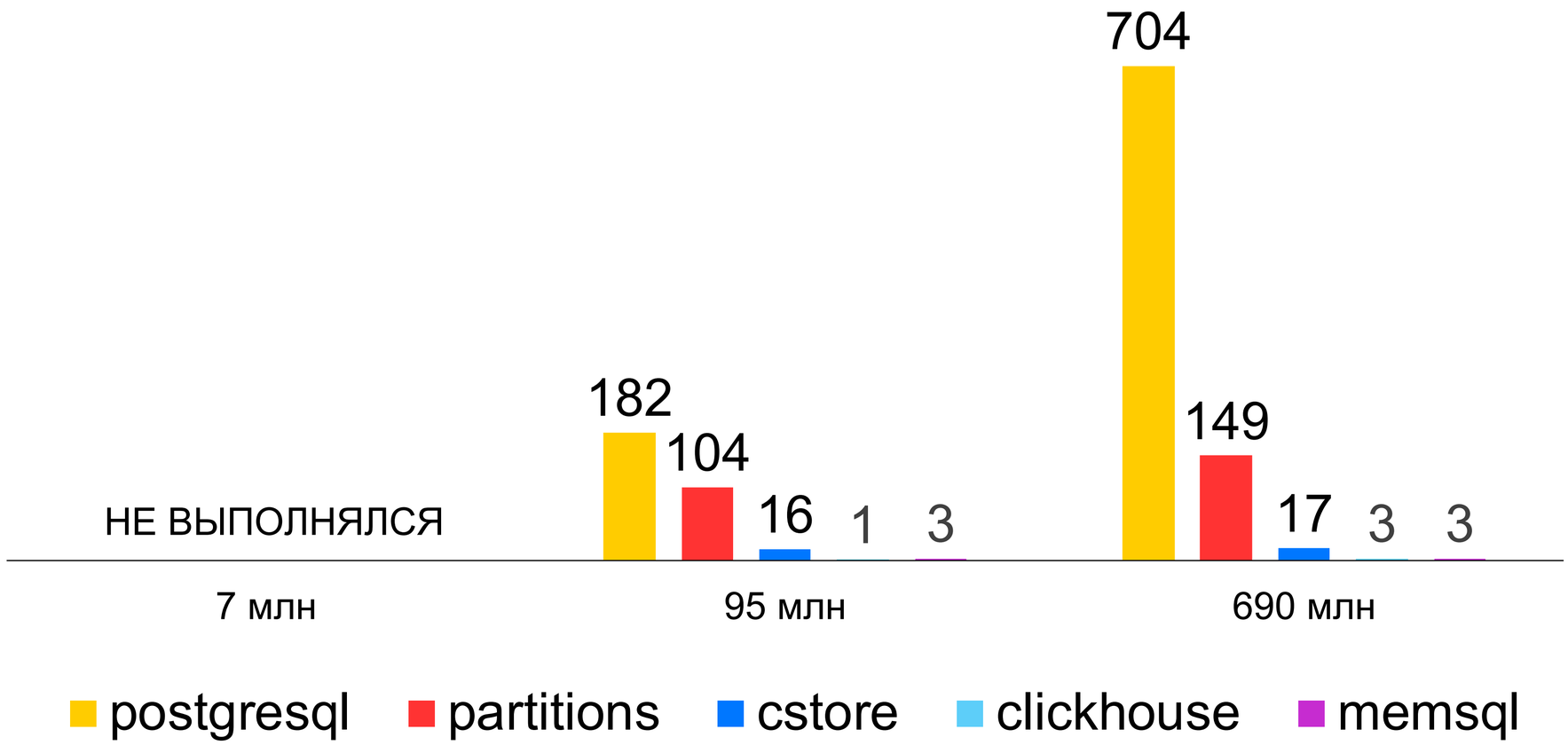
Статистика, средняя



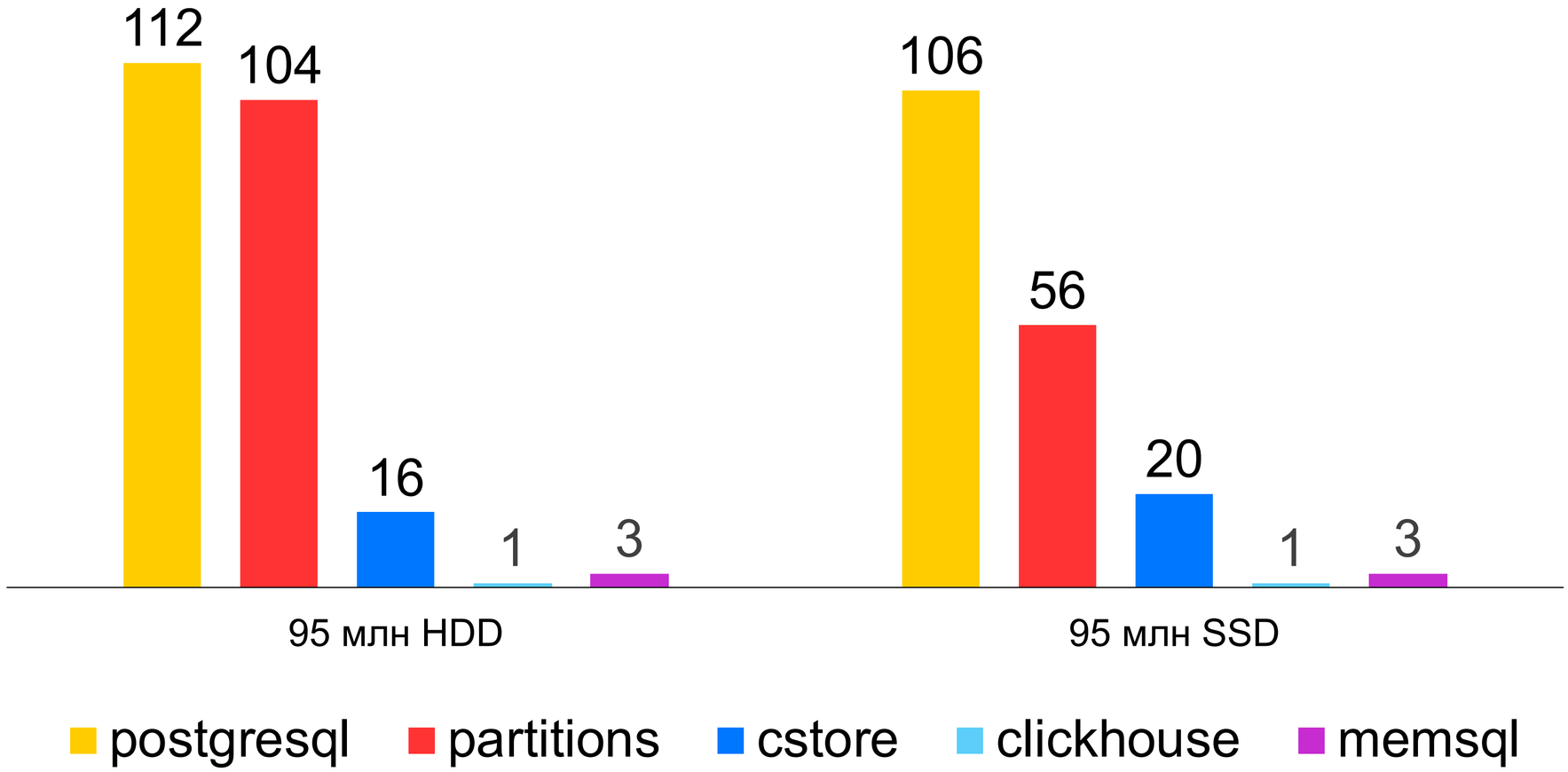
Средняя статистика, с фильтром по времени

```
SELECT repo_name, count(*) AS cnt
FROM push
WHERE
'2015-06-01' <= created_at AND
created_at < '2015-10-01’
GROUP BY repo_name
ORDER BY cnt DESC LIMIT 5
```

Средняя статистика, с фильтром по времени



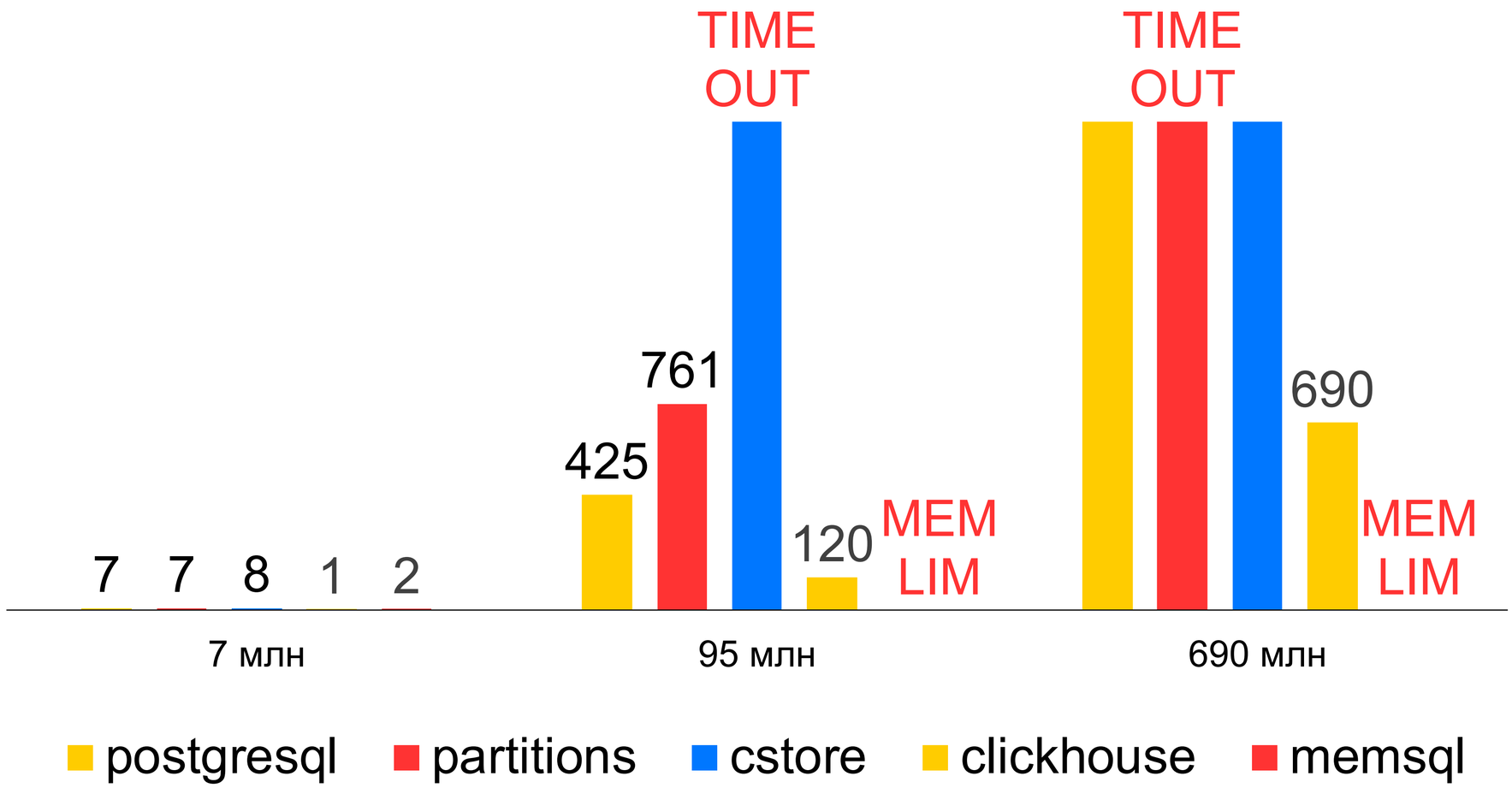
Средняя статистика, с фильтром по времени



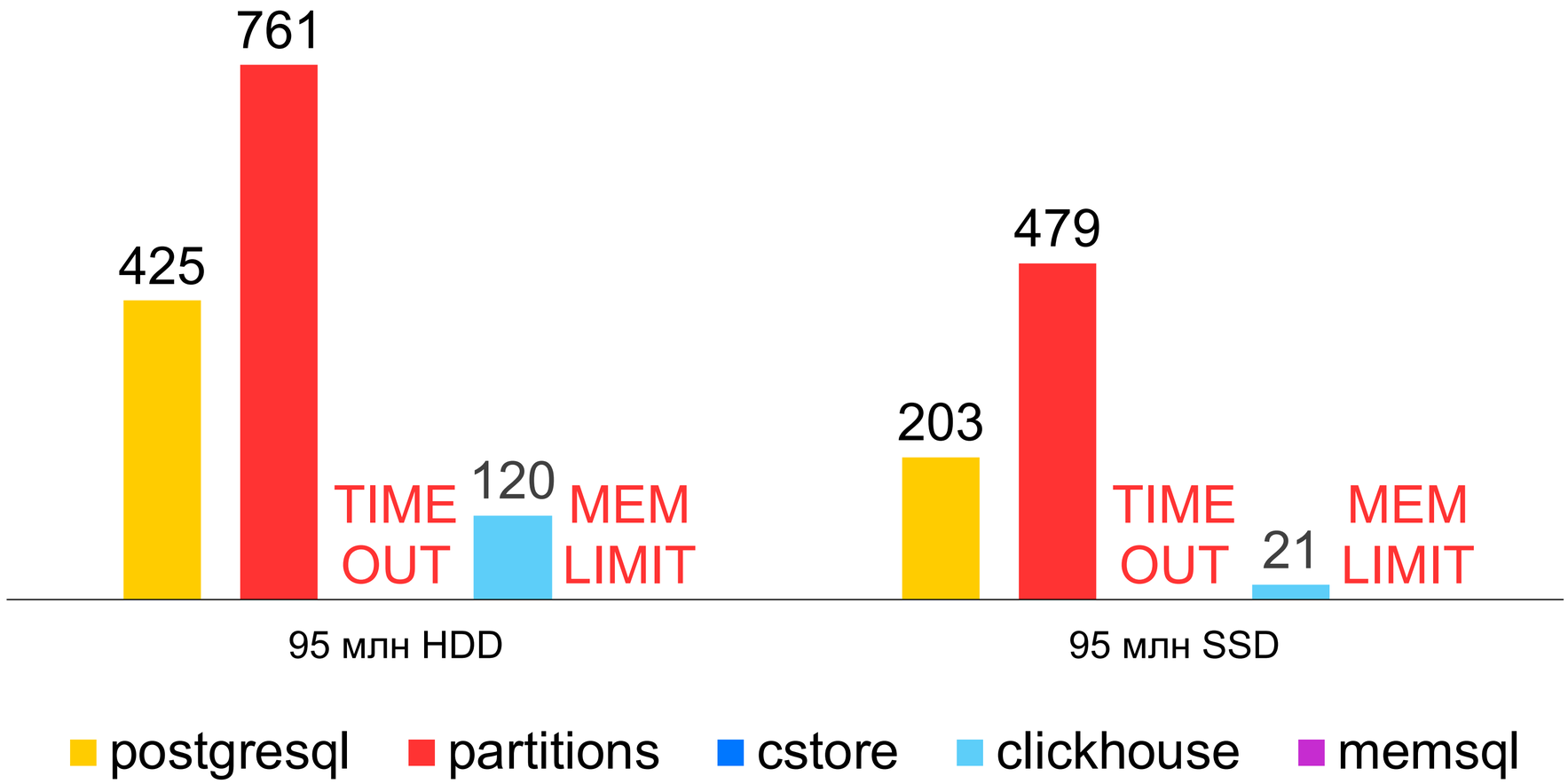
Большая статистика

```
SELECT before, count(*) AS cnt
FROM push
GROUP BY before
ORDER BY cnt DESC
LIMIT 5
```

Большая статистика, популярный коммит



Большая статистика, популярный коммит



Итог HDD, 95 млн записей

База	Место	Количество	Фильтр	Регулярка	Мал стат	Сред стат	Сред стат Огр	Бол стат
PostgreSQL	18,5 Гб	116 с	114 с	117 с	117 с	117 с	112 с	425 с
PostgreSQL partitions	18,5 Гб	173 с	175 с	435 с	287 с	441 с	104 с	761 с
PostgreSQL-cstore_fdw	9,5 Гб	11 с	11 с	290 с	35 с	39 с	16 с	TIME OUT
ClickHouse	**5,2 Гб**	**1 с**	**1 с**	30 с	**1 с**	**2 с**	**1 с**	**120 с**
MemSQL	20,3 Гб	**1 с**	**1 с**	**8 с**	2 с	9 с	3 с	MEM LIMIT

Итог SSD, 95 млн записей

База	Место	Количество	Фильтр	Регулярка	Мал стат	Сред стат	Сред стат Огр	Бол стат
PostgreSQL	18,5 Гб	66 с	67 с	108 с	99 с	188 с	106 с	203 с
PostgreSQL (partitions)	18,5 Гб	68 с	68 с	105 с	99 с	180 с	56 с	479 с
PostgreSQL-cstore_fdw	9,5 Гб	10 с	12 с	298 с	33 с	49 с	20 с	TIME OUT
ClickHouse	**5,2 Гб**	**1 с**	**1 с**	31 с	**1 с**	**2 с**	**1 с**	**21 с**
MemSQL	20,3 Гб	**1 с**	**1 с**	**8 с**	2 с	8 с	3 с	MEM LIMIT

Итог, 690 млн записей

База	Место	Количество	Фильтр	Регулярка	Мал стат	Сред стат	Сред стат Огр	Бол стат
PostgreSQL	128 Гб	860 с	848 с	860 с	TIME OUT	TIME OUT	116 с	TIME OUT
PostgreSQL (partitions)	128 Гб	TIME OUT	TIME OUT	TIME OUT	TIME OUT	TIME OUT	149 с	TIME OUT
PostgreSQL-cstore_fdw	66 Гб	66 с	113 с	TIME OUT	308 с	676 с	17 с	???
ClickHouse	**37 Гб**	8 с	**2 с**	228 с	**2 с**	**110 с**	**3 с**	**690 с**
MemSQL	46 Гб	**1 с**	**2 с**	**165 с**	76 с	MEM LIMIT	**3 с**	MEM LIMIT

Итоги, личное мнение

- SSD – полезны для postgres, не очень полезны колоночным решениям

На основе эксперимента для хранения логов я бы взял

- Postgres (BRIN-index) – если уже работаю с базой и нужно хранить небольшой объем логов, разбивать данные на партиции я бы не стал
- Postgres (partitions) – только на SSD
- Postgres-cstore – тоже, при этом я хочу сэкономить место и готов к особенностям расширения
- Clickhouse - для хранения и сбора статистики по логам в общем случае
- MemSQL – если уже используется MySQL и хочется работать с логами через те же инструменты, если требуется хранить логи и другие данные в одной базе или анализировать меняющиеся данные

Самый популярный commit

38984c582bee39f1cc327d8941a4e5dfca494b13 – ?

Сколько раз поверх этого коммита был сделан push за 2018 год?

Самый популярный commit

38984c582bee39f1cc327d8941a4e5dfca494b13 - **815 027** пушей

Первый – 23.01.2018

В основном это репозиторий
https://github.com/unitydemo2/Docworks

› Создание пустого файла

https://github.com/unitydemo2/Docworks/commit/38984c582bee39f1cc327d8941a4e5dfca494b13

Яндекс Облако

Спасибо

Тимофей Кулин

Разработчик

rekby@yandex-team.ru

https://github.com/rekby/dump-2019