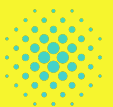


YTsaurus в дикой природе – плюсы, минусы, подводные камни

Никита Благодарный
telegram @nblagodarnyy

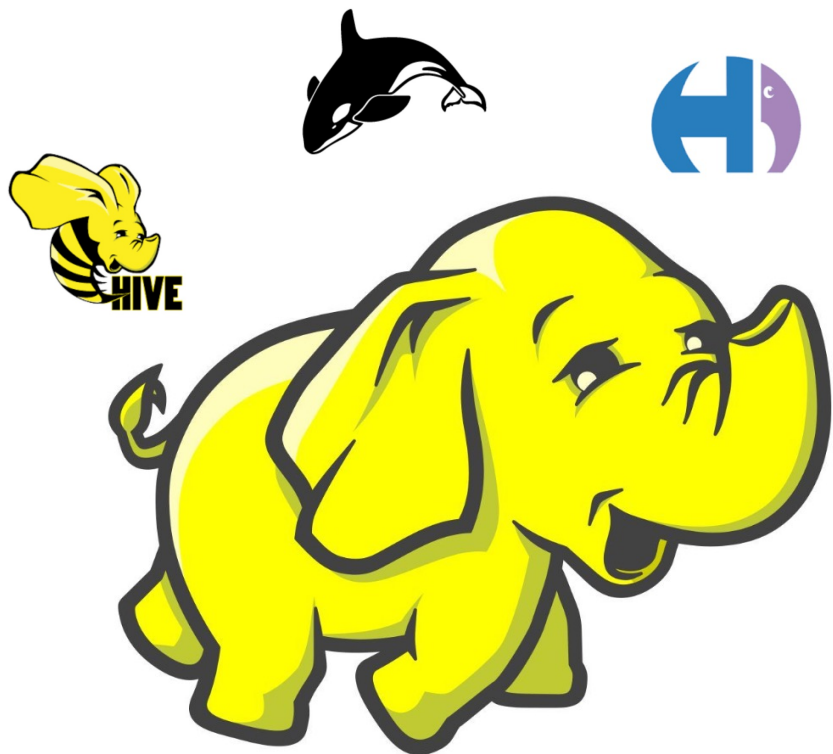


Немного вводных

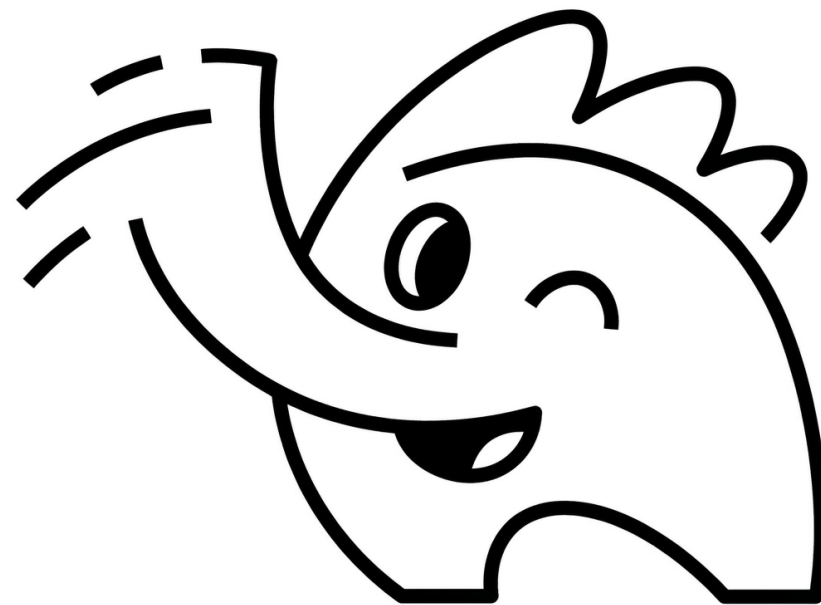


Про что этот доклад?

YTsaurus не в Яндексе



vs



Содержание

Brief - сравнение компонентов Hadoop+ и YTsaurus

Цели и подходы к апробации/тестированию, состав работ

Эффективность хранения и вычислений

Мнения участников внедрения

Схоластика

Выводы/пожелания





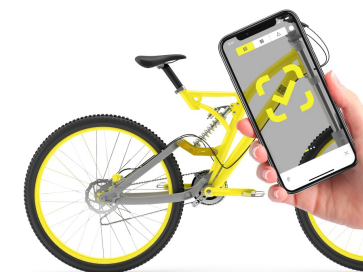
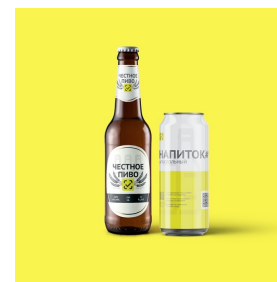
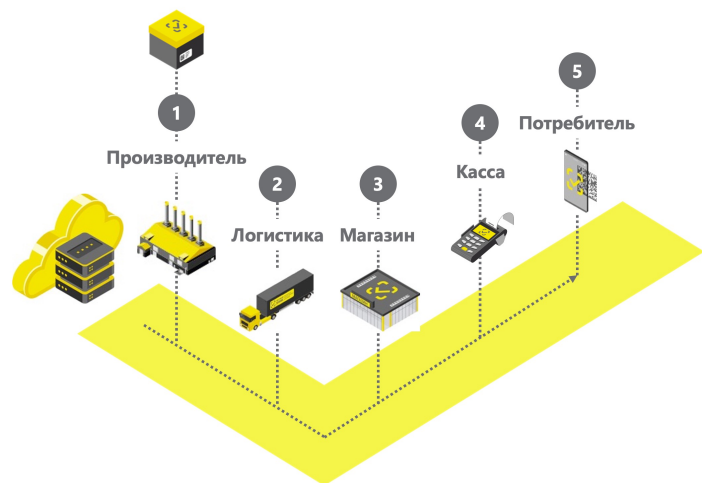
Data Engineer, 19 лет в IT, из них 16 в области BI/DE

- 2007-2011 Сбербанк, разработка OLTP-решений на Oracle, ХД/Reporting на стеке Microsoft
- 2012-2016 Сбербанк, разработка ЦХД на Oracle / Teradata / Informatica PC
- 2016-2018 КРОК, проекты на Oracle / Informatica / Hadoop (HDP, Arenadata) / Spark / Hive / NiFi
- 2018-2019 Газпромнефть, построение DataLake (Arenadata, MS SQL, Informatica, Spark/Hive)
- 2019-.... Честный знак, DataLake (Vanilla Hadoop/ Spark / Hive / HBase / ClickHouse / Postgres, YTsaurus etc.)
- 4-е выступление на SmartData

О компании Честный знак

Создаем систему цифровой маркировки и прослеживания товаров "Честный Знак" в России и странах ЕАЭС

- Основана в 2017 году
- > 1000 сотрудников
- Офисы разработки в Москве, Питере, Пензе, Ульяновске, Нижнем Новгороде
- Разнообразные товарные группы (Табак, Молоко, Вода, Пиво, Лекарства, Одежда, Обувь, ...)



О компании Честный знак

Создаем систему цифровой маркировки и прослеживания товаров "Честный Знак" в России и странах ЕАЭС

Год	Инкремент - строк
2024	623 608 881 343
2025	1 231 278 669 090
2026 факт июль	1 239 440 705 267
2026 прогноз	2 478 881 410 534
2027 прогноз	4 957 762 821 068

Configured Capacity:	41.41 PB
Configured Remote Capacity:	0 B
DFS Used:	23.1 PB (55.78%)
Non DFS Used:	11.76 TB
DFS Remaining:	16.21 PB (39.15%)

Сравнение КОМПОНЕНТОВ

Распределенная система хранения



VS



Cypress

- Единицы хранения только файлы, API – только файловый
- Нет TTL для файлов
- Storage Policies – Hot, ALL_SSD, ONE_SSD etc.
- ЕС – до сих пор багованный
- Spark – основные кодеки (zstd, snappy, lz4...)

- Единицы хранения – файлы, таблицы, очереди, документы, транзакции и др.
- Есть TTL для узлов
- Медиумы (в том числе In-memory - RAM)
- ЕС – нормальный (говорят)
- Чуть больше кодеков хранения (+brotli)

Сравнение - хранение



yt saurus prod

Navigation

☆

▼

/ home / data / pharma / bronze / history_daily

✎

🔗

🔗

🔗

Content

Attributes

User attributes

ACL

Locks 0

Edit metadata

Filter...

⚙️

Default

Resources

Show resources

Medium: All

▼

Create object

All 4197 • Table 4196 • Map node 1

☐	Name ^	Locks ⚙️	Account ⚙️	Disk space ⚙️	Rows ⚙️	Modification time ⚙️	Creation time ⚙️	
☐	📁 20251106		sys	–	–	17 Nov 2025 14:37:19	17 Nov 2025 14:36:51	🗑️ 📄 ...
☐	📁 20250528		sys	72.16 GiB	24 298 006	29 May 2025 03:11:39	28 May 2025 15:12:54	🗑️ 📄 ...
☐	📁 20250529		sys	109.39 GiB	37 796 185	30 May 2025 03:11:41	29 May 2025 04:11:28	🗑️ 📄 ...
☐	📁 20250530		sys	92.33 GiB	30 265 668	31 May 2025 03:11:32	30 May 2025 04:11:37	🗑️ 📄 ...
☐	📁 20250531		sys	12.79 GiB	3 890 821	02 Jun 2025 14:10:05	31 May 2025 04:11:23	🗑️ 📄 ...
☐	📁 20250601		sys	1.99 GiB	554 059	02 Jun 2025 14:10:05	01 Jun 2025 04:12:04	🗑️ 📄 ...

Browse Directory

/hbase/

Go!

Show 25 entries

Search:

	Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name	
☐	drwxr-xr-x	hadoop	supergroup	0 B	Jan 16 2026	0	0 B	.hbase-snapshot	
☐	drwxr-xr-x	hadoop	supergroup	0 B	Mar 27 2021	0	0 B	.hbck	
☐	drwxr-xr-x	hadoop	supergroup	0 B	Oct 22 2025	0	0 B	.tmp	
☐	drwxr-xr-x	hadoop	supergroup	0 B	Aug 24 21:12	0	0 B	MasterProcWALs	
☐	drwxr-xr-x	hadoop	supergroup	0 B	Aug 24 19:21	0	0 B	WALs	
☐	drwxr-xr-x	hadoop	supergroup	0 B	May 28 2020	0	0 B	archive	
☐	drwxr-xr-x	hadoop	supergroup	0 B	Jun 22 2020	0	0 B	corrupt	
☐	drwxr-xr-x	hadoop	supergroup	0 B	Dec 17 2021	0	0 B	data	
☐	-rw-r--r--	hadoop	supergroup	42 B	Apr 20 2020	3	128 MB	hbase.id	
☐	-rw-r--r--	hadoop	supergroup	7 B	Apr 20 2020	3	128 MB	hbase.version	
☐	drwxr-xr-x	hadoop	supergroup	0 B	Aug 21 13:11	0	0 B	indexes	
☐	drwxr-xr-x	hadoop	supergroup	0 B	Aug 24 21:22	0	0 B	oldWALs	

Showing 1 to 12 of 12 entries

Previous

1

Next

Сравнение – scheduler



- Контейнерное выделение
- Управляет vCPU и RAM
- Монолитный RM (либо федерация монолитов)
- Очереди
- Preemption – смерть
- Смотрит на data locality

- Не контейнерное выделение (для своих)
- Управляет vCPU, RAM, сетью
- Scheduler + Controller Agents
- Пулы/деревья пулов
- Preemption – может приостановить
- Не смотрит на data locality

Сравнение – масштабирование storage/compute



- Очень сложно смешать с другими нагрузками
=> compute скейлится машинами

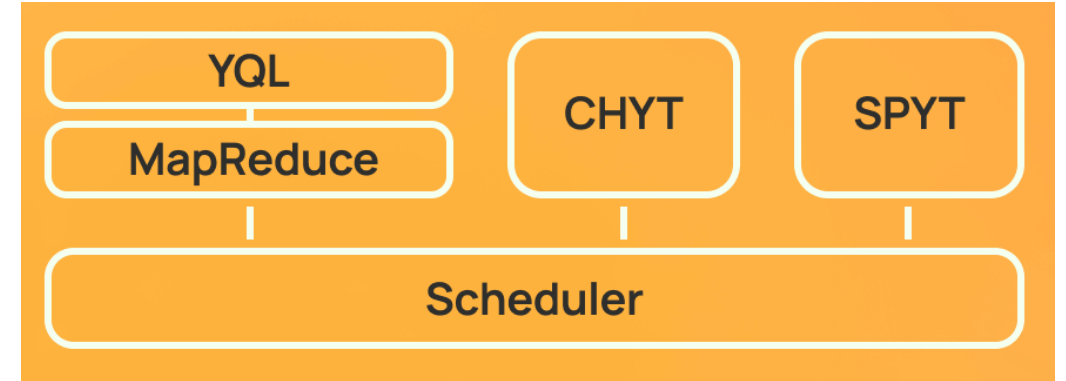
- Просто смешать, если в одном k8s
=> compute скейлится k8s

Сравнение – движки расчётов



MR2

VS



- Apache Spark + Hive Metastore
- Apache Flink

- Hive on Tez
- [Hive on] MR2

- Spark over YT (SPYT) – с ограничениями
- Flink over YT FLYT

- YQL – свой движок/язык
- CHYT – ClickHouse
- MapReduce с кодом или без

Сравнение – движки расчётов - YQL

YQL

CHYT

YT QL

2025.05

New

```
1 PRAGMA AnsiInForEmptyOrNullItemsCollections;
2 pragma yt.DefaultMemoryLimit = "2G";
3 PRAGMA yt.EnableFuseMapToMapReduce = 'true';
4 pragma yt.AutoMerge = "disabled";
5 pragma yt.QueryCacheMode = "disable";
6
7 --1. Get all history records by increment keys
8 $all_hist_records_by_inc_keys = (
9   select d.*
10   from `//home/data/table1` d
11   where d.col1 in
12         (select distinct col
13          from like(`//home/data/pharma/bronze/history_parsed`, "20251124%")
14         );
15 );
16 --2. Deduplicate and get new values for calculates fields
17 $increment_plus_its_history = (
18   select *
19   from (
20     select 0 as is_inc, 0 as flg_exclude, cs.*
21     from $all_hist_records_by_inc_keys cs
22     ) hist
23   union all
```

Run

Validate

Explain

Access: nobody

Completed

00:24:48

06 Aug 19:52

by n.blagodarny

Share

Edit ACO

Progress

Statistics

Meta

Graph

Timeline

SmartData

2026

Сравнение – движки расчётов - операции

- **Map, Reduce, MapReduce** — выполняют пользовательский код над входными данными;
- **Sort** — сортирует указанную на входе таблицу;
- **Merge** — выполняет слияние таблиц;
- **Erase** — удаляет из таблицы указанный диапазон данных;
- **RemoteCopy** — копирует данные между кластерами;
- **Vanilla** — запускает набор **пользовательских скриптов** в нужном количестве на узлах кластера и поддерживает их жизнедеятельность.

Сравнение – движки расчётов - операции

```
def main(app_name: str, step: int) -> None: 1 usage  n.blagodarny * 29.12.2025

    mb_per_job=2048

    src_full_tbl_path="//home/data/pharma/silver/table_big"
    src_inc_tbl_path="//home/data/pharma/silver/tmp/table_small"
    tgt_tbl_path=f"//home/data/pharma/silver/tmp/table_result_{mb_per_job}"

    tgt_tbl = yt.TablePath(
        tgt_tbl_path,
        append=False,
        optimize_for="lookup",
        compression_codec="zstd_5",
        #attributes={ "replication_factor": 1 }
    )

    spec = MergeSpecBuilder()\
        .begin_job_io()\
        .table_writer( {"upload_replication_factor": 1} )\
        .end_job_io()\
        .pool("research")\
        .alias(app_name)\
        .data_size_per_job(mb_per_job*1024*1024)\
        .combine_chunks(False)\
        .merge_by(["col1", "col2", "col3"])\
        .mode("sorted")\
        .output_table_path(tgt_tbl) \
        .input_table_paths([src_full_tbl_path, src_inc_tbl_path]) \
        .build()

    client = yt.YtClient(proxy="http://ytsaurus-proxy.crpt.ru", token="ytct-YoUrPeRsOnAlToKeN")
    yt.set( path: "//sys/controller_agents/config/enable_merge_schemas_during_schema_infer", value: True)
    client.run_merge(
        source_table=[src_full_tbl_path, src_inc_tbl_path],
        destination_table=tgt_tbl,
        spec=spec
    )
```

Сравнение – движки расчётов - CHYT

YQLCHYTYTQL

chyt_10

chyt_10

6ea5bc60-7b929037-103e8-10ba2522

test_devops inactive

466e7ff9-e90a34f3-103e8-50062f48

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

```
SELECT version() ver,
.....toStartOfMonth(col_date) mon_start,
.....uniqHLL12(sgtin) ch_func2
FROM (select sgtin, col_date
.....from `//home/data/pharma/table1`
.....limit 100)
GROUP BY mon_start
```

▶ Run

Access: nobody

✔ Completed

00:00:01

02 Sep 01:09

by n.blagodarny

Result

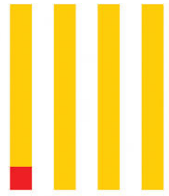
Chart

Meta

Rows 1-10 of 10

#	ver ⓘ	mon_start ⓘ	ch_func2 ⓘ
1	"24.8.15.1"	2020-07-01	19
2	"24.8.15.1"	2020-08-01	1
3	"24.8.15.1"	2020-09-01	2
4	"24.8.15.1"	2020-10-01	15
5	"24.8.15.1"	2020-11-01	9
6	"24.8.15.1"	2020-12-01	11
7	"24.8.15.1"	2021-01-01	13
8	"24.8.15.1"	2021-09-01	8
9	"24.8.15.1"	2022-06-01	2
10	"24.8.15.1"	2022-08-01	15

Сравнение – витринный слой



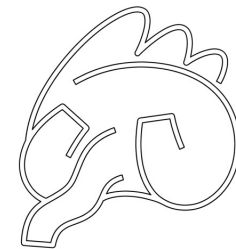
ClickHouse

VS

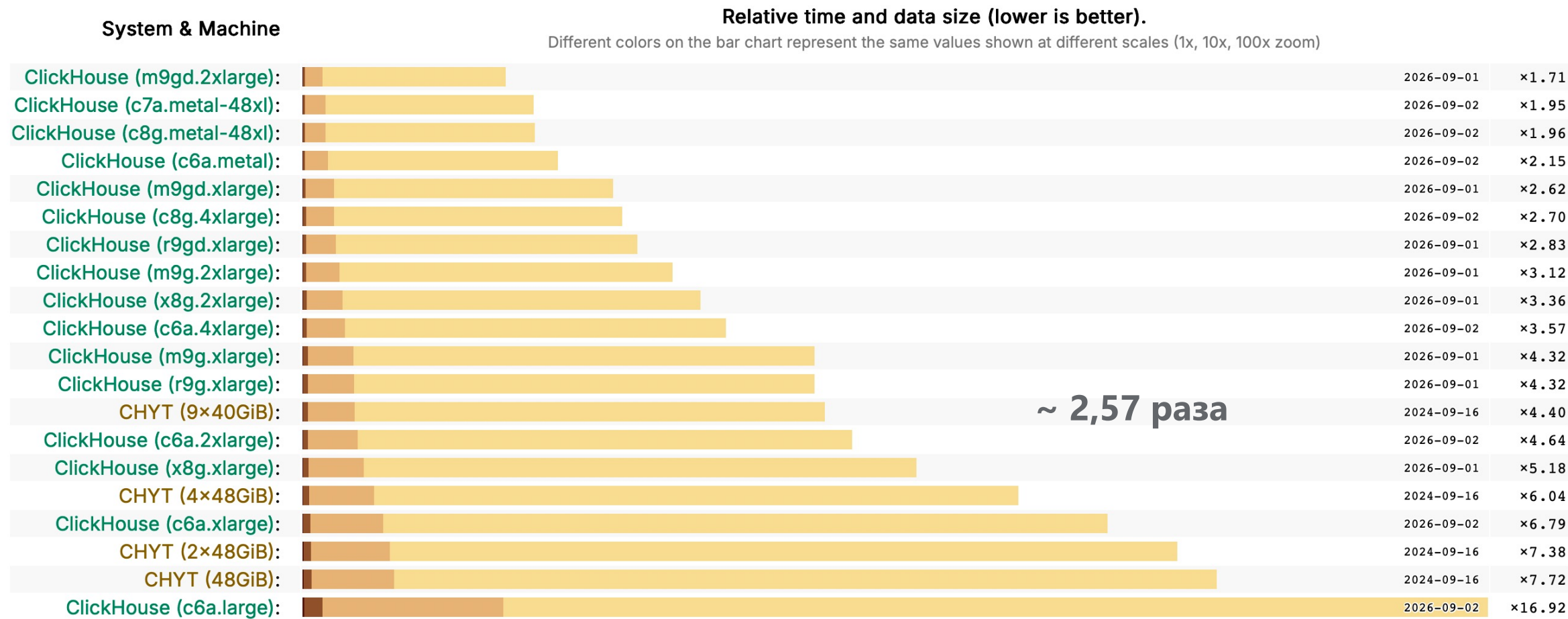


- Отдельный кластер на серверах
- Надо загружать данные (если не S3 и т.п.)
- Сложно быстро свернуть/развернуть
- Версия – по желанию
- Быстрее

- Живёт внутри YT (k8s)
- Не надо загружать данные
- Кнопки в UI
- Версия – как в дистрибе
- Медленнее



Сравнение – витринный слой



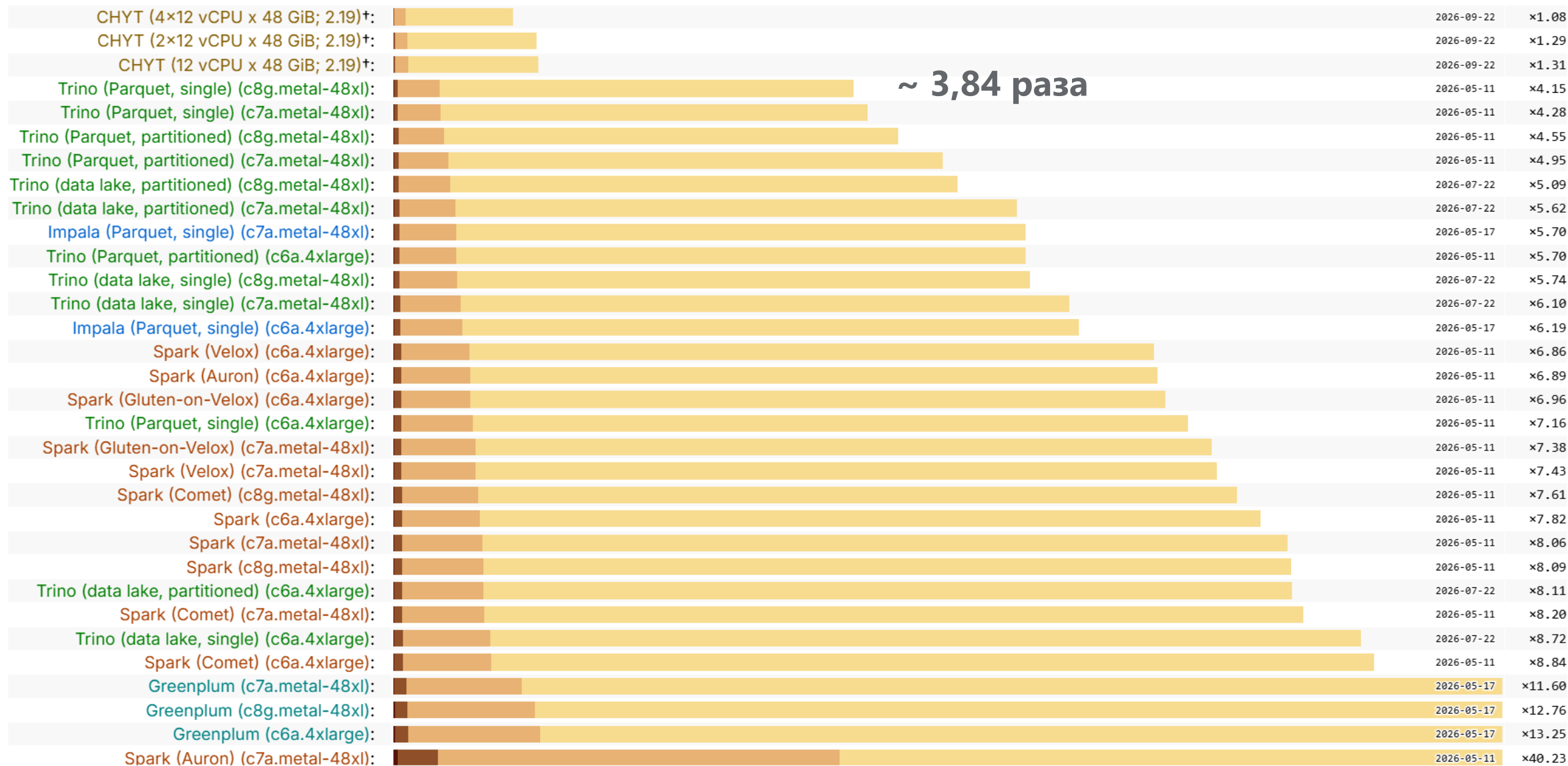
<https://benchmark.clickhouse.com>

Сравнение – витринный слой

System & Machine

Relative time and data size (lower is better).

Different colors on the bar chart represent the same values shown at different scales (1x, 10x, 100x zoom)

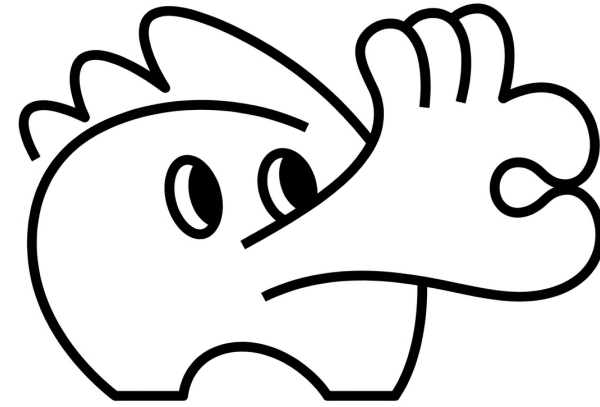


~ 3,84 раза

Сравнение – key-value



vs



Dynamic
Tables

- Отдельный кластер
- Нет батчевого SQL/вставки
- По ключу – мгновенно

- Тот же кластер
- Есть батчевый SQL/вставка
- По ключу – тоже мгновенно

Сравнение – metastore



VS

В Hive:

- ✓ Выделенный процесс
- ✓ Таблицы + схемы
- ✓ Партиции
- ✓ Бакеты
- ✓ Статистика
- ✓ Views

Метаданные на уровне таблицы Cypress

The screenshot shows the Cypress interface for a table named 'ordered_by_id'. The metadata is displayed in a structured format with various attributes and their values.

Metadata	
Id	1-10a3-10191-1ed112a9
Owner	operation_archivarius
Account	operations_archive
Creation time	18 Dec 2024 18:08:09
Modification time	25 Aug 2026 22:01:04
Access time	25 Aug 2026 22:01:05
Rows	= 1 248 206
Chunks	417
Uncompressed size	191.52 GiB
Compressed size	26.73 GiB
Primary medium	SSD
Total disk space	82.44 GiB
Data weight	191.32 GiB
Table type	dynamic
Optimize for	Scan
Compression ratio	0.13956
Compression codec	lz4
Replication factor	3
Sorted	true
Tablet cell bundle	sys
Tablet state	Mounted
In memory mode	Compressed
Tablet error count	0

Navigation: sys / operations_archive / ordered_by_id

Content: Attributes User attributes Mount config ACL Locks 0 Schema Tablets

Keys: 3/3 No offset

"id_hash" iL	"id_hi" iL	"id_lo" iL	"state"	"authenticated_user"	"operation_type"	"pr"
28082386160129u	285773360873261u	12479907737789556148u	"completed"	"robot-odin"	"map"	

В YT:

- Нет отдельного компонента
- ✓ Таблицы + схемы
- Партиции (=таблицы)
- Бакеты
- ✓ Статистика
- ✓ Views
- ✓ Тулинг вокруг отсутствия партиций



SmartData

2026

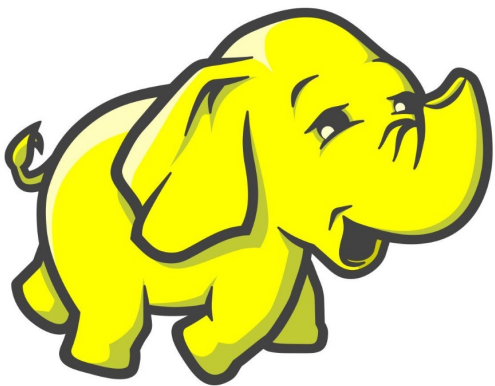
Сравнение – движки расчётов - YQL

```
from //home/data/pharma/bronze/history_parsed
where d.col1 in
    (select distinct col
     from like(//home/data/pharma/bronze/history_parsed, "20251124%"))
```

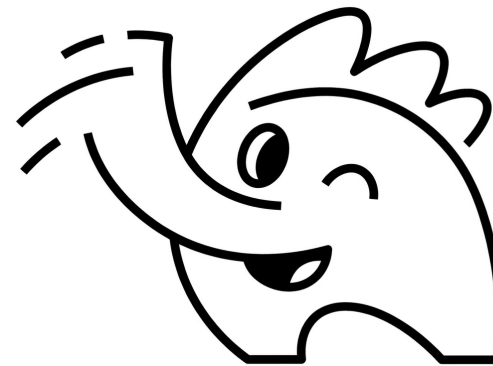


Сравнение – transactions

ICEBERG 



vs



- ACID в рамках 1 таблицы - с форматами
- Атомарный mv файлов

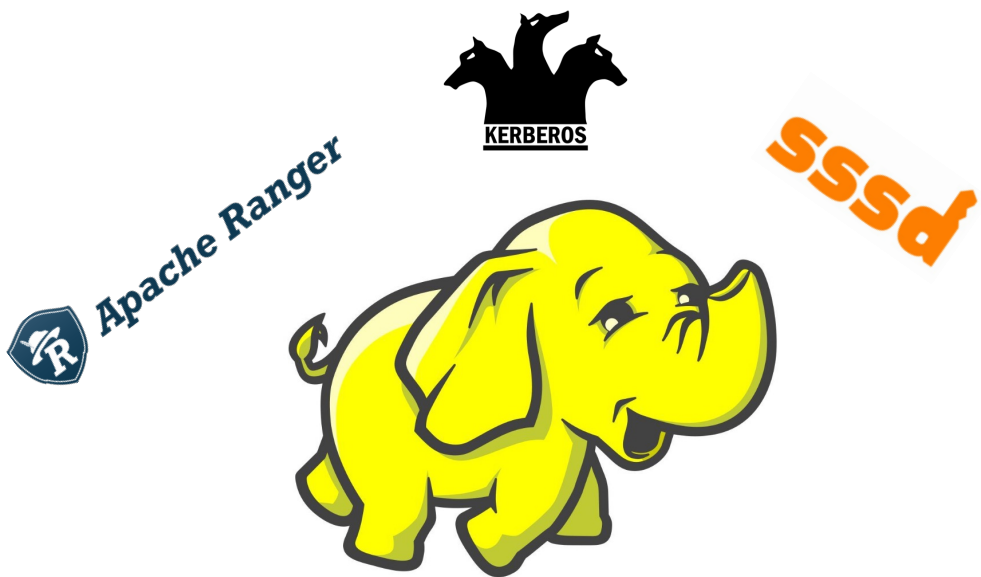
- Широкая модель транзакций и блокировки + настраивается изоляция



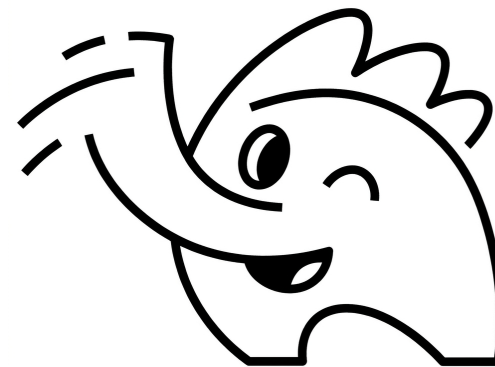
SmartData

2026

Сравнение – безопасность



VS



- LDAP – через Linux SSSD
- Authentication – krb5 + LDAP
- Authorization – Ranger plugins
- RLS, CLS – через Ranger

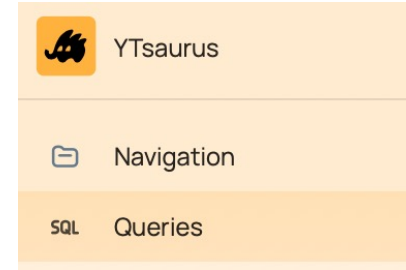
- LDAP – тула ytsaurus-identity-sync
- Authentication – токены, логонасс, OIDC
- Authorization – своя через master
- RLS, CLS – встроены



Сравнение – ноутбуки пользователей

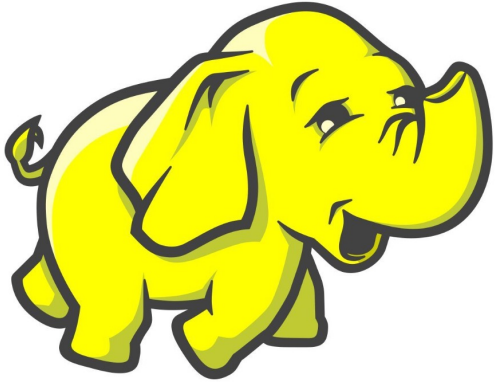


- Jupyter есть
- Ещё Zeppelin

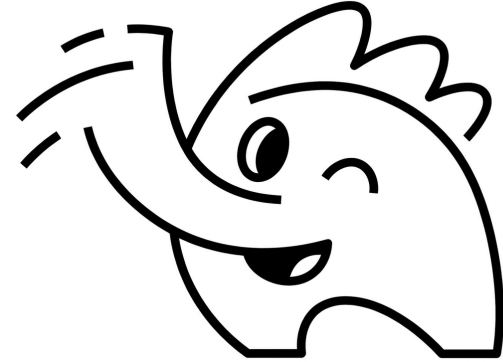


- Jupyter тоже есть
- Удобный UI запросов

Сравнение – в остальном



vs

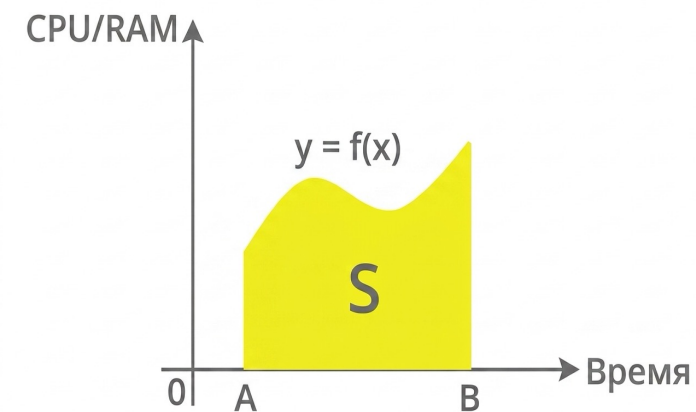


- Нет нормального cluster manager
- Груда UI под каждый компонент

- Есть k8s operator CRD + UI
- UI един – да здоровствует UI

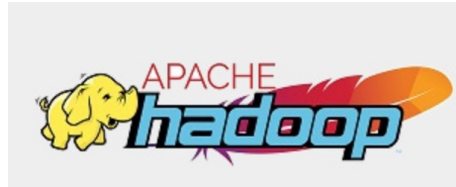
Цели и подходы к апробации

Зачем?



Методика

Одна и та же kafka с исходными данными



20 серверов

20 серверов

bronze, silver, gold – расчёты

bronze, silver, gold – расчёты

Оценка метрик вычисления/хранения

Эффективность хранения

**Показатель эффективности -
нормированная метрика -
сколько Gb занимает на диске 1
млн 1 и тех же строк**



Эффективность хранения



Данные	Формат	Примечания / codec / erasure	Gb на 1 млн строк	+ от лучшего	Строк	Gb
Bronze из kafka - json	Hadoop - parquet	snappy, до компакций	5,49182	1416%	164 756 161	904,81
	Hadoop - parquet	snappy, после компакций	5,67359	1466%	543 180 866	3081,78
	Hadoop - parquet	zstd_9, до компакций	1,37559	279%	180 951 810	248,91
	YT - static table	brothli_8	0,77260	113%	184 183 378	142,30
	YT - static table	brothli_8 + erasure	0,36220	0%	184 183 378	66,71
	YT - static table	lz4	4,61175	1173%	178 054 014	821,14
Silver - разобранные колоночные данные	Hadoop - parquet	snappy	0,59950	209%	164 756 161	98,77
	Hadoop - parquet	zstd_9	0,44257	128%	50 003 691	22,13
	Hadoop - parquet	zstd_9	0,41163	112%	162 972 022	67,08
	YT - static table	brothli_8	0,41894	116%	170 312 825	71,35
	YT - static table	brothli_8	0,37235	92%	53 460 566	19,91
	YT - static table	brothli_8 + erasure	0,19385	0%	184 183 378	35,70
Bronze из kafka -protobuf	Hadoop - parquet	snappy	0,51838	594%	134 841 038	69,90
	Hadoop - parquet	zstd	0,18117	143%	1 921 836 954	348,17
	YT - static table	brothli_8	0,16217	117%	2 039 982 227	330,82
	YT - static table	brothli_8 + erasure	0,07466	0%	2 039 982 227	152,31
	YT - static table	lz4	0,29634	297%	1 987 069 973	588,84

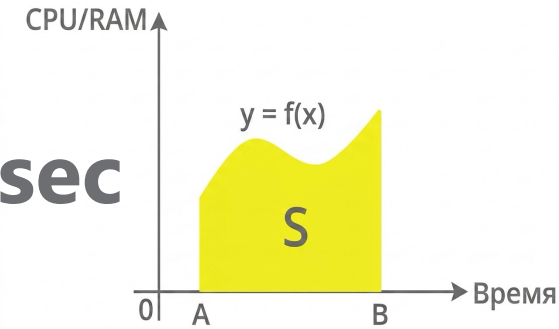
Выводы:

- EC - сильно эффективнее, x2 и более
- JSON - YT ~x2 лучше parquet zstd
- Protobuf - YT чуть лучше parquet zstd
- Типизированное - YT ~ равен zstd

Эффективность вычислений

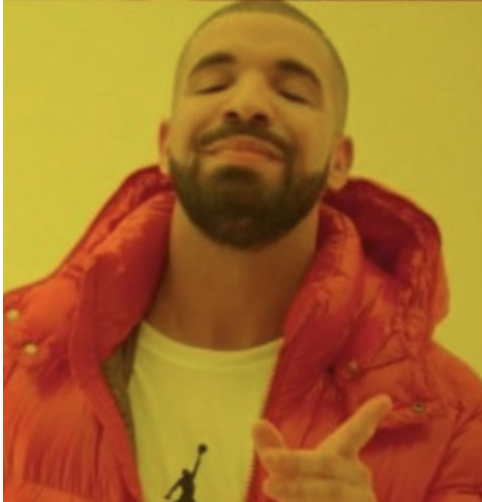
Экспликация метрик

- **duration_sec** - wall time
- **mem_MB_x_sec** - интеграл, RAM кластера * sec
- **CPU_sec** - суммарно секунд CPU
- **sec_per_10^N_row** - нормированная скорость обработки
- **mbsec_per_1000000_row** - нормированная производительность обработки по RAM
- **CPU_sec_per_1000000_row** - нормированная производительность обработки по CPU





**До оптимизации
кластера и/или расчётов**



**После оптимизации
кластера и/или расчётов**

Бронза - парсинг JSON

Бронза - парсинг JSON

YT			Spark	
Метрика	Значение	Выиграл	Значение	Метрика
queries_cnt	898		715	queries_cnt
duration_sec_min	25,00	YT	53,76	duration_sec_min
duration_sec_max	1 089,00	SPARK	199,21	duration_sec_max
duration_sec_quant_95	339,05	SPARK	132,35	duration_sec_quant_95
duration_sec_quant_999	1 089,00	SPARK	192,50	duration_sec_quant_999
mem_MB_x_sec_min	700,83	YT	1 418 029,87	mem_mb_x_sec_min
mem_MB_x_sec_max	3 851 558,69	YT	24 242 717,85	mem_mb_x_sec_max
mem_MB_x_sec_quant_95	2 336 346,89	YT	19 135 368,36	mem_mb_x_sec_quant_95
mem_MB_x_sec_quant_999	3 851 558,69	YT	23 776 479,98	mem_mb_x_sec_quant_999
sec_per_1000_row_min	0,05	SPARK	0,01	sec_per_1000_row_min
sec_per_1000_row_max	3,32	SPARK	0,33	sec_per_1000_row_max
sec_per_1000_row_quant_95	0,16	SPARK	0,09	sec_per_1000_row_quant_95
sec_per_1000_row_quant_999	3,32	SPARK	0,26	sec_per_1000_row_quant_999
mbsec_per_1000_row_min	43,17	YT	128,12	mbsec_per_1000_row_min
mbsec_per_1000_row_max	1 812,15	YT	16 113,89	mbsec_per_1000_row_max
mbsec_per_1000_row_quant_95	568,52	YT	4 717,20	mbsec_per_1000_row_quant_95
mbsec_per_1000_row_quant_999	1 812,15	YT	15 608,72	mbsec_per_1000_row_quant_999

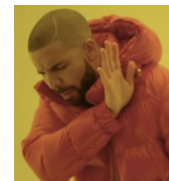
быстрее

меньше
RAM

быстрее

меньше
RAM

По 95 перцентилю Spark x2,56 быстрее и x8 прожорливее



**Silver -
обновить
детальный слой**

Silver - дедупликация детального слоя + merge инкремента

```
select row_number() over (partition by id order by col14 desc, col5 desc) as rn_col14
from (
  select case
    when (next_col1 in (terminal_states) and coalesce(next_col5 - col5, 1) = 1) then 1
    when col1 in (sale_states) and prev_col1 in (sale_states) then -1
    else 2
  end as col14,
  t8.* without col14
  from (
    select t7.*,
      lead(col1) over w2 as next_col1,
      lag(col1) over w2 as prev_col1
    from (
      select t6.*
      from (
        select lead(col1) over w1 as next_col1_dup,
              t5.*
        from (
          select case when t4.col5 is null then t4.col3 else t4.col12 end as col13,
                 t4.* without col13
          from (
            select t3.*
            from (
              select t.*,
                min(t.col6) over (partition by id) as col7
              from tmp t
            ) t2
            where t2.col8 = 1 or coalesce(t2.col7, 0) != -2
          ) t3
            where not (t3.col8 = 1 and coalesce(t3.col9_orig, 0) != 0)
          ) t4
            window w1 as ( partition by id order by col5, col13, col4, col3, col1, col12)
          ) t5
        ) t6
        where col5 != coalesce(next_col5_dup, -1) or (col5 is null and next_col5_dup is null
          or col3 != coalesce(next_col13_dup, timestamp(0)))
        ) t7
        window w2 as ( partition by id order by col5, col13, col4, col3, col1, col12)
      ) t8
    ) t9
```

1

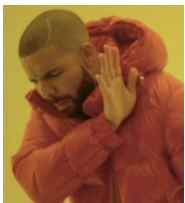
1

2

2

Silver - batch дедупликация детального слоя + merge инкремента

YT 25.1 (SPYT 2.7.5 Spark 3.5.4)						Hadoop 3.3.6 / Spark 3.5.1
Метрика	шардами YQL	шардами SPYT	без шардов YQL	без шардов SPYT	ВЫИГРАЛ	шардами в параллель
rows_sum	57 345 352 305					61 754 537 534
duration_sec (wall time)	192 000 (~ 52 часов)	105 000 (~ 29 часов)	> 24h	20 880 (~ 5,8 часа)	SPARK	9 360 (~ 2,6 часа)
mem_MB_x_sec (x10 ¹⁰)	12,73	36,5	не собирали так как wall time критически не подходит	16,79	YT	4,70
sec_per_1mln_row	3,348	1,831		0,364	SPARK	0,152
mbsec_per_1mln_row	12 370	365 000		167 900	YT	47 040
CPU_sec	N/A	17 750 000		11 080 000	SPARK	10 782 720
CPU_sec_per_1mln_row	N/A	309,528		193,215	SPARK	174,606



Silver - batch дедупликация детального слоя + merge инкремента





Код - as is как написали мы



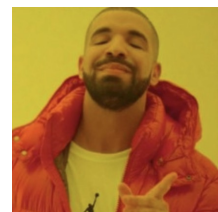
3 MapReduce YQL vs 1 Spark



До конфигурации кластера

Silver - batch дедупликация детального слоя + merge инкремента

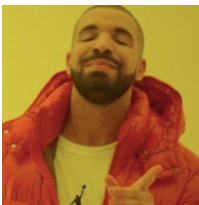
YT 25.3			Hadoop 3.3.6 / Spark 3.5.1
Метрика	без шардов YQL + python	ВЫИГРАЛ	full - шардами в параллель
rows_sum	57 345 352 305		61 754 537 534
duration_sec (wall time)	7 980 (~ 2,2 часа)	YT	9 360 (~ 2,6 часа)
mem_MB_x_sec (x10 ¹⁰)	3,02	YT	4,70
sec_per_1mln_row	0,14	YT	0,152
mbsec_per_1mln_row	30 234	YT	47 040
CPU_sec	9 805 824	YT	10 782 720
CPU_sec_per_1mln_row	171,01	YT	174,606



Silver - merge инкремента to dynamic table

YT 25.3		Hadoop 3.3.6 / Spark 3.5.1	
Метрика	increment dyn tables + yql	полный пересчёт шардами в параллель	increment iceberg paimon
rows_sum	29 434 533	61 754 537 534	В теории возможно, но не реализовывалось
duration_sec (wall time)	1 440 (~24 мин)	9 360 (~ 2,6 часа)	
mem_MB_x_sec ($\times 10^{10}$)	0,24	4,70	
sec_per_1mln_row	48,92	0,152	
mbsec_per_1mln_row	2 359	47 040	
CPU_sec	663 552	10 782 720	
CPU_sec_per_1mln_row	22 543	174	

dyn-таблицы требуют содержания tablet-nod (20 штук 16 Gb 1 CPU), но эти ноды обслуживают множество таблиц и на батче ресурсы не едят



Gold - относительно простая агрегация

YT 25.1			Hadoop 3.3.6 / Spark 3.5.1
Метрика	полный пересчёт шардами YQL	ВЫИГРАЛ	полный пересчёт
rows_sum	6 476 672 839		8 077 658 648
duration_sec (wall time)	1 140 (19 минут)	SPARK	960 (16 минут)
mem_MB_x_sec ($\times 10^{10}$)	0,064	YT	0,259
sec_per_1mln_row	0,176	SPARK	0,119
mbsec_per_1mln_row	98 816,16	YT	321 771,18
CPU_sec	~ 395 808	SPARK	~ 188 006
CPU_sec_per_1mln_row	61,11	SPARK	23,275



Мнение от DevOps

- Легкость развертывания
- Масштабирование



Hadoop,
HDFS, YARN,
Hive,
Zookeeper,
Ranger, Solr,
Livy, Spark,
Ansible

YTsaurus,
K8s



- **Неполная/неточная документация**
- **Трудности при обновлении**
- **Ограниченное конфигурирование**
- **Прикладной мониторинг (стало лучше с Odin)**

Pro и Contra

Зачем можно [не]хотеть внедрить YT?

Почему да?

- Более эффективное хранение холодных данных (EC)
- Понятный scaling storage/compute
- В некоторых кейсах – ресурсов меньше
- Есть настоящие транзакции
- Продукт сейчас развивается активнее Hadoop.
- KV с любыми движками доступа к нему, возможности для инкрементов и разного доступа.
- Удобство развёртывания - k8s, один pipeline деплоя, есть api. Меньше зоопарк и версий.
- Хороший интерфейс.
- Для SPYT есть изолированный shuffle-tracker.

Почему нет?

- Код/трейсы/ошибки на ++. Экспертиза.
- Может быть нужен низкоуровневый код.
- Скорость YQL у нас пока ниже. Нетривиальная оптимизация. Мало гайдов, кейсов, информации, практики по оптимизации YQL-запросов.
- Документация.
- SPYT не совсем Spark.
- YQL - SQL, но не совсем.
- Нет большого сообщества/базы знаний (1 чат, 1400 чел.).
- Есть мнения, что без support в прод не стоит

Выводы/советы

Выводы и советы

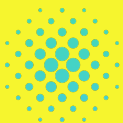
- Система работает и работает во многом хорошо
- Система сложная, с ноги не залетишь
- Иногда - ничего не понятно. Начните разбираться-и разберётесь.
- Мониторинг – сразу
- Ресурсы - сразу
- 1 vCPU - 4 Gb RAM
- Дока – пока не идеальная
- Есть недокументированное



**Пожелания к
продукту**

- **Wizard - сайзер, конфигуратор, валидатор**
- **Архитектурное описание компонентов**
- **Метастор**
- **Отдельные Spark-сессии из ноутбуков**
- **Все возможные крутилки – в доку**

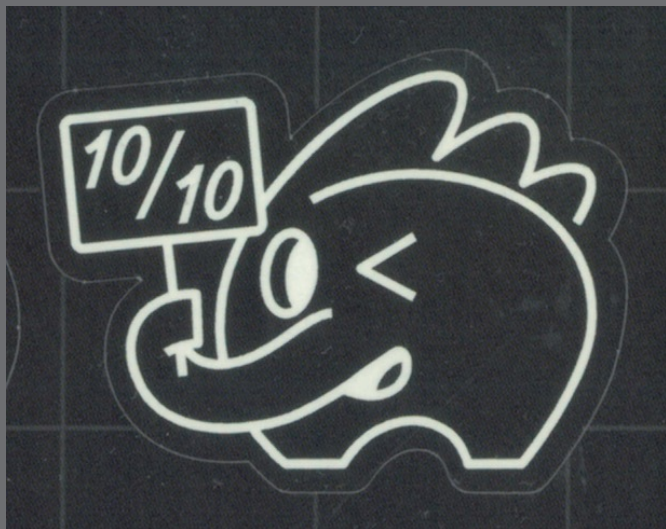
Вопросы



SmartData

2026

**СПАСИБО
ЗА ВНИМАНИЕ!**



Никита Благодарный
telegram @nblagodarnyy



центр развития
перспективных технологий

