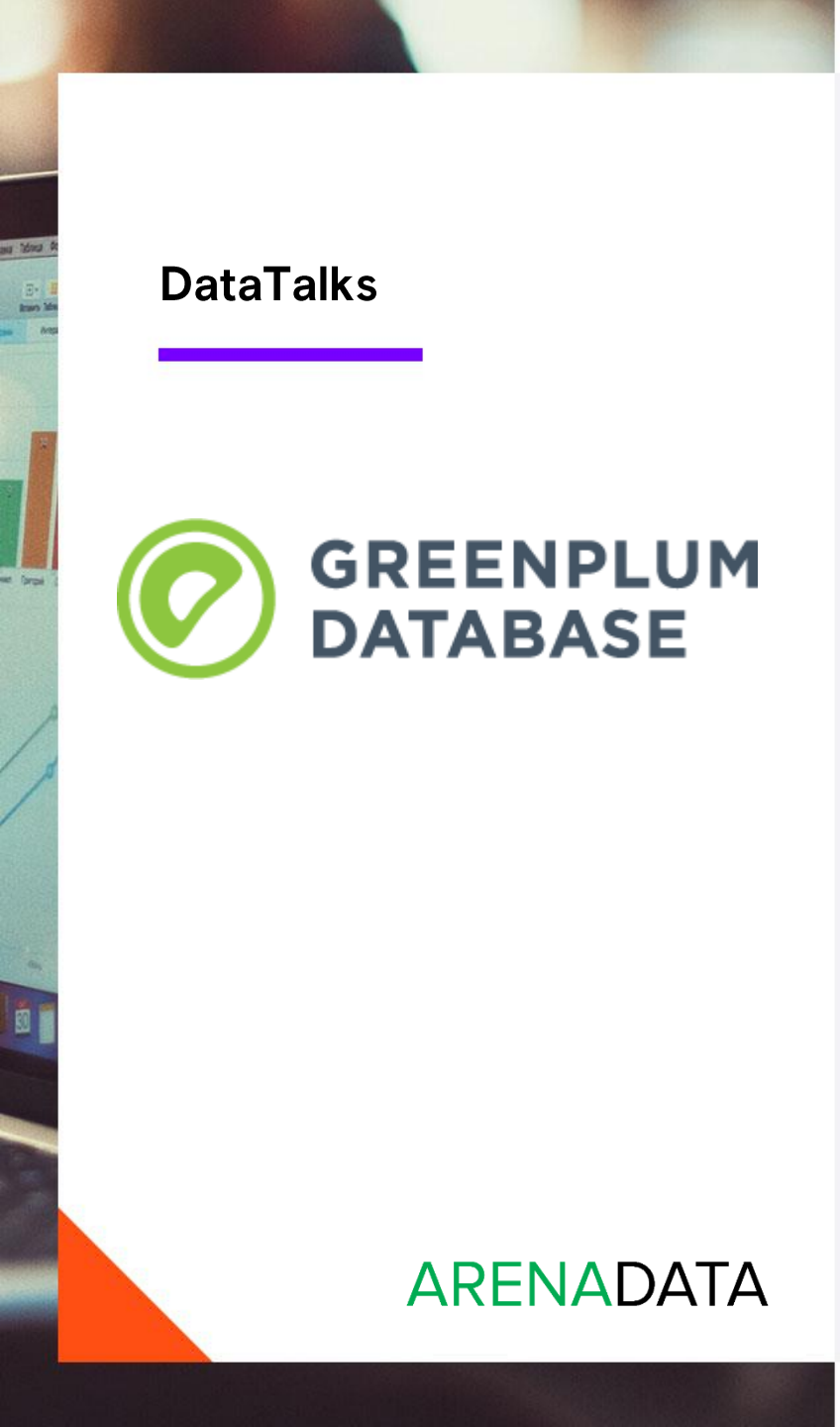


DataTalks

GREENPLUM DATABASE

Архитектура MPP на примере СУБД Greenplum

Управление современным IT-ориентированным предприятием немыслимо без данных – на основе данных принимаются решения, данные зарабатывают деньги, данные помогают быть одним компаниям сильнее, чем другие.

В этой лекции мы разберём, как современные компании справляются с ситуацией, когда данных слишком много:

- Как обрабатывать терабайты данных в реальном времени или близко к этому?
- Как спроектировать систему для обработки данных, не зная заранее объёма этих данных?
- Как устроена современная инфраструктура хранилища данных?

Мы рассмотрим эти вопросы на примере Greenplum – открытой, гибкой и мощной системы для параллельной обработки данных.

ARENADATA

Ростелеком

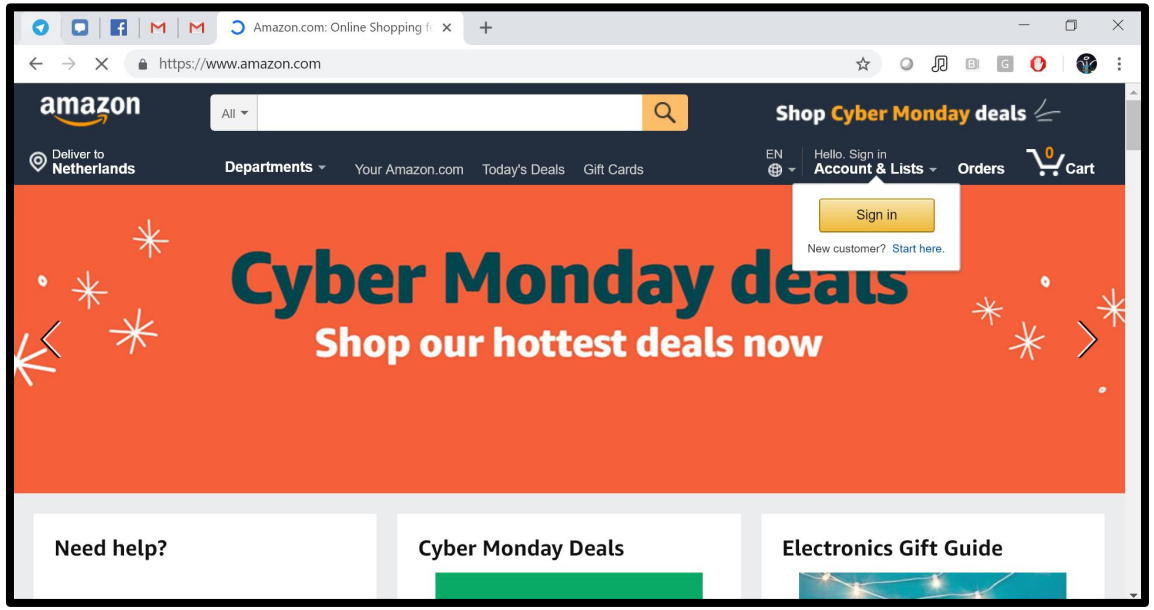
Архитектура MPP
на примере СУБД Greenplum

ARENADATA

Дмитрий Павлов

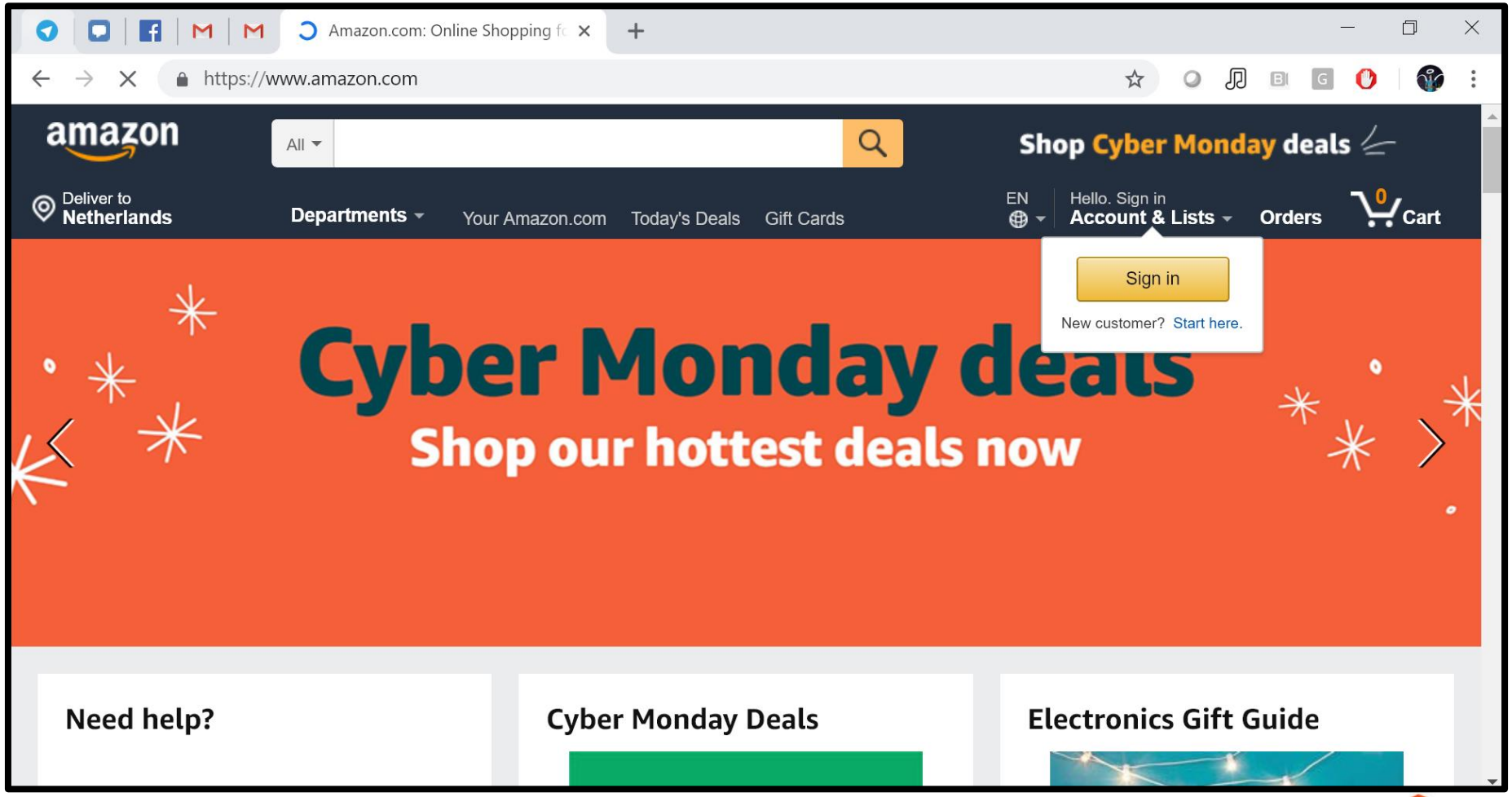
Product Owner
Arenadata.io

Ростелеком



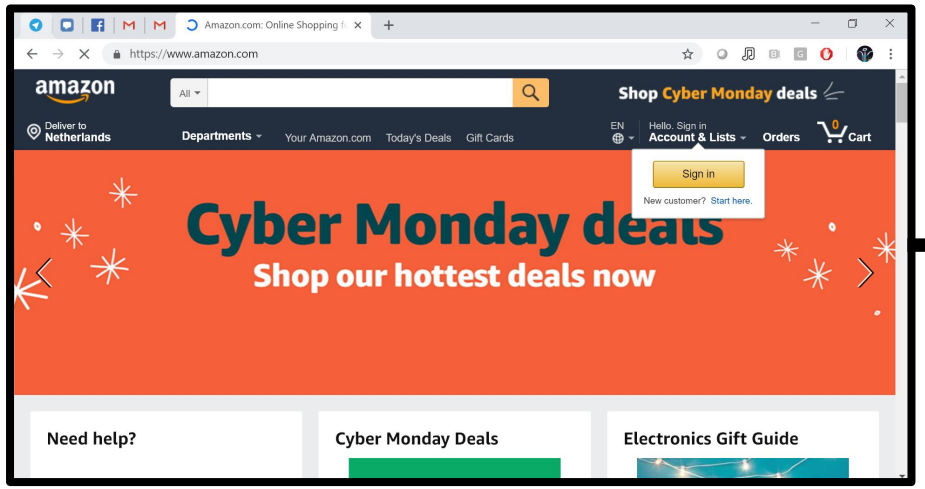
ARENADATA

Ростелеком



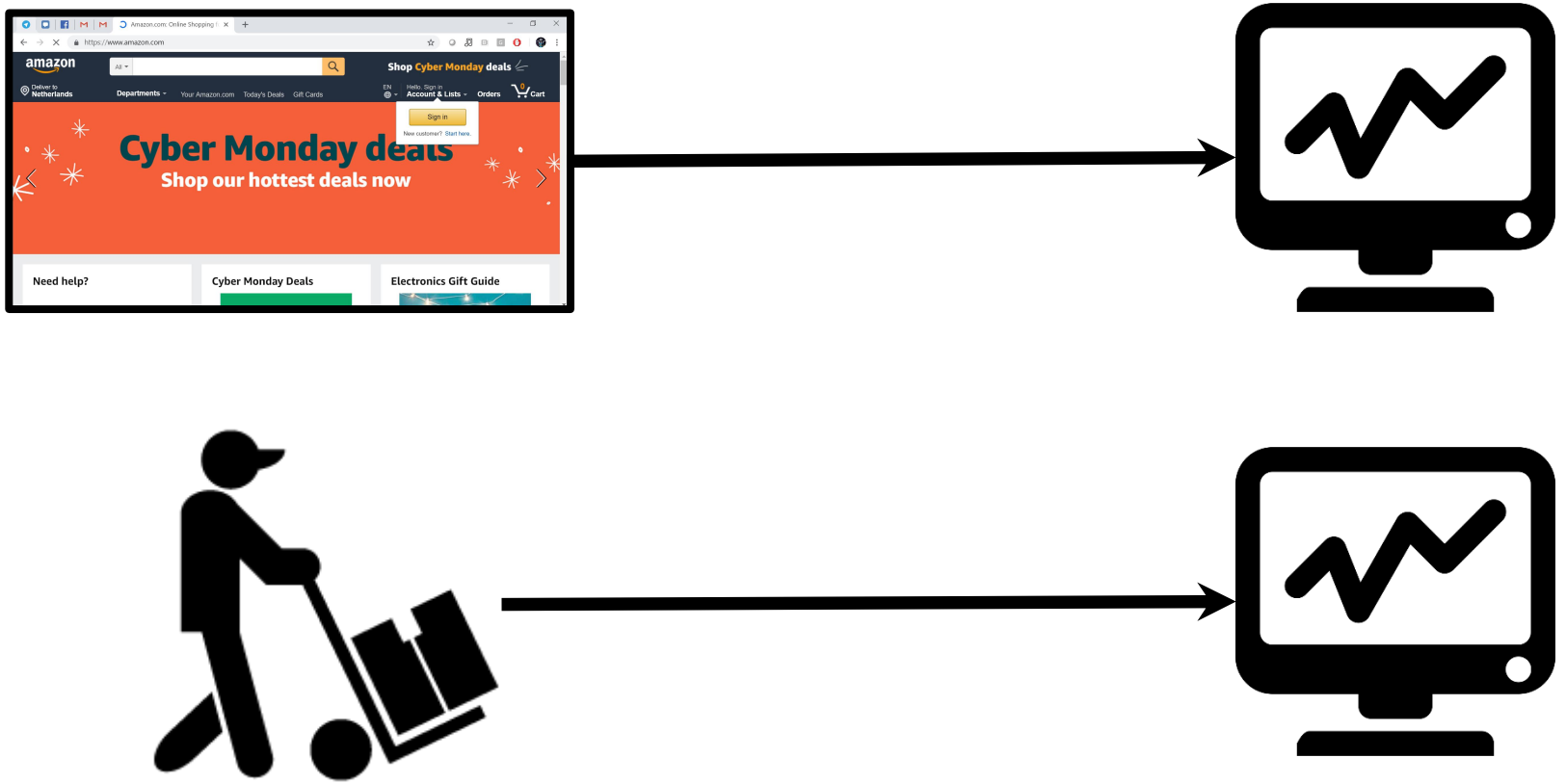
ARENADATA

Ростелеком



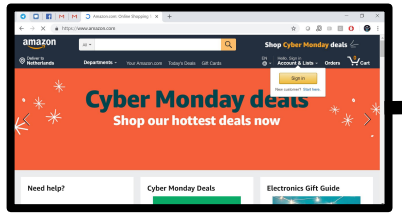
ARENADATA

Ростелеком



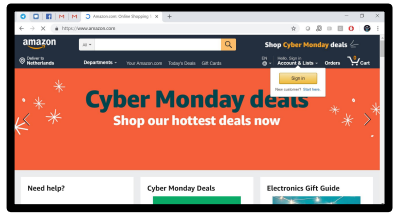
ARENADATA

Ростелеком



ARENADATA

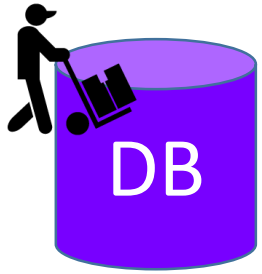
Ростелеком

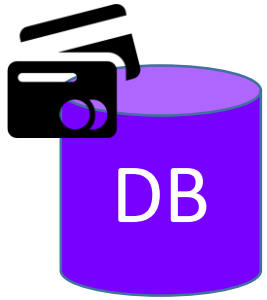


– источники данных

ARENADATA

Ростелеком

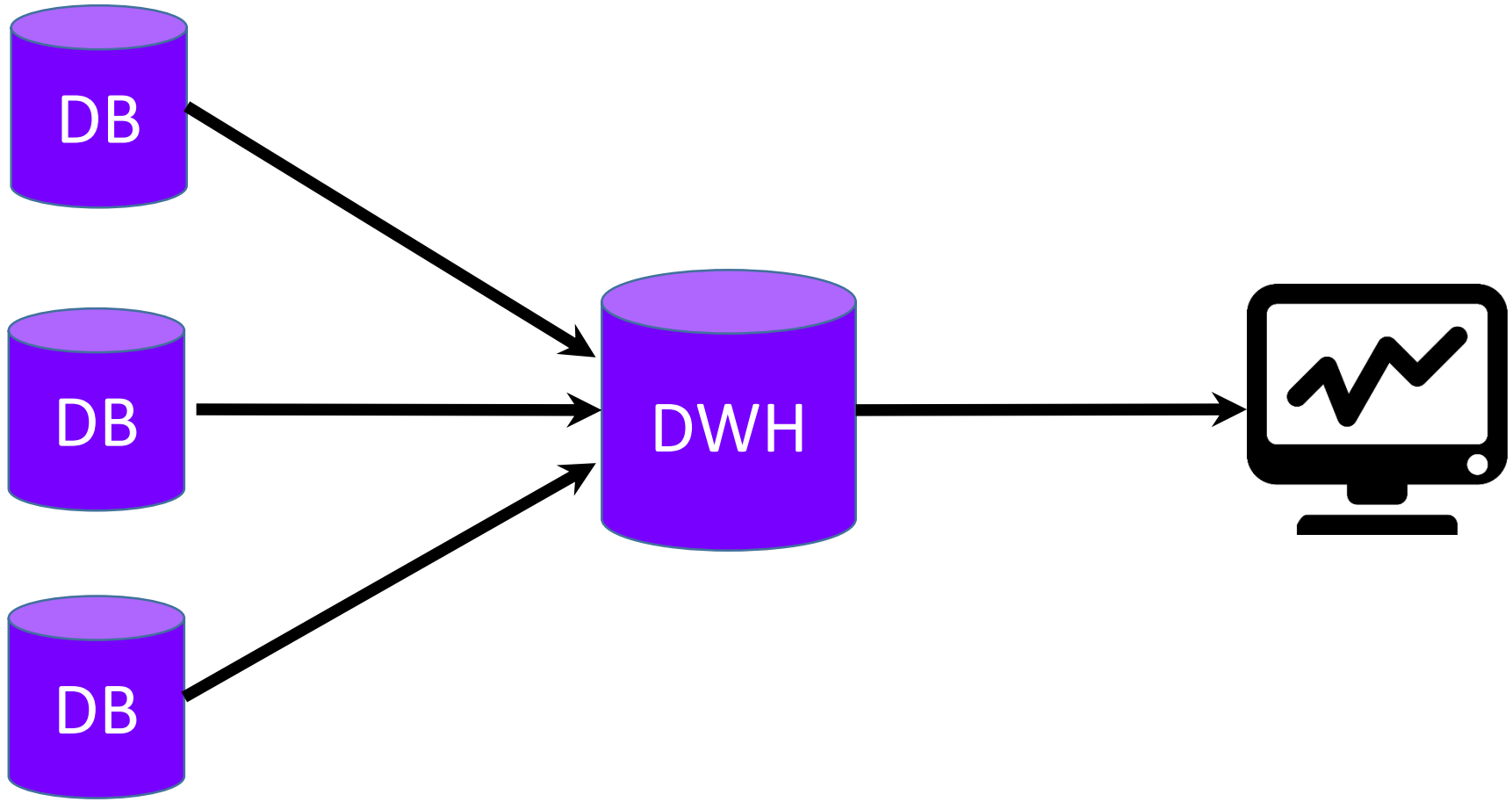




Чаще всего источники – это базы данных конкретных систем на предприятии.

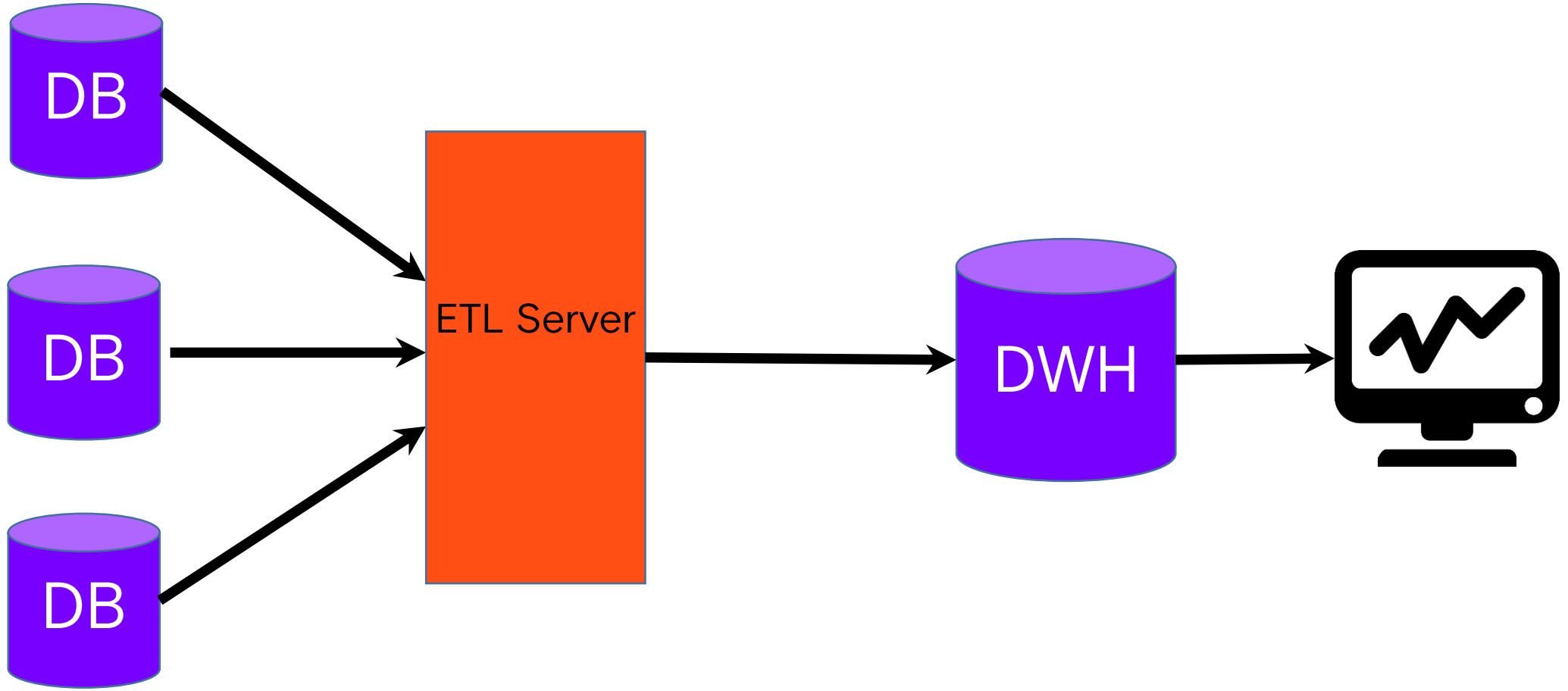
Это – святое. Их нагружать аналитикой нельзя.

Ростелеком

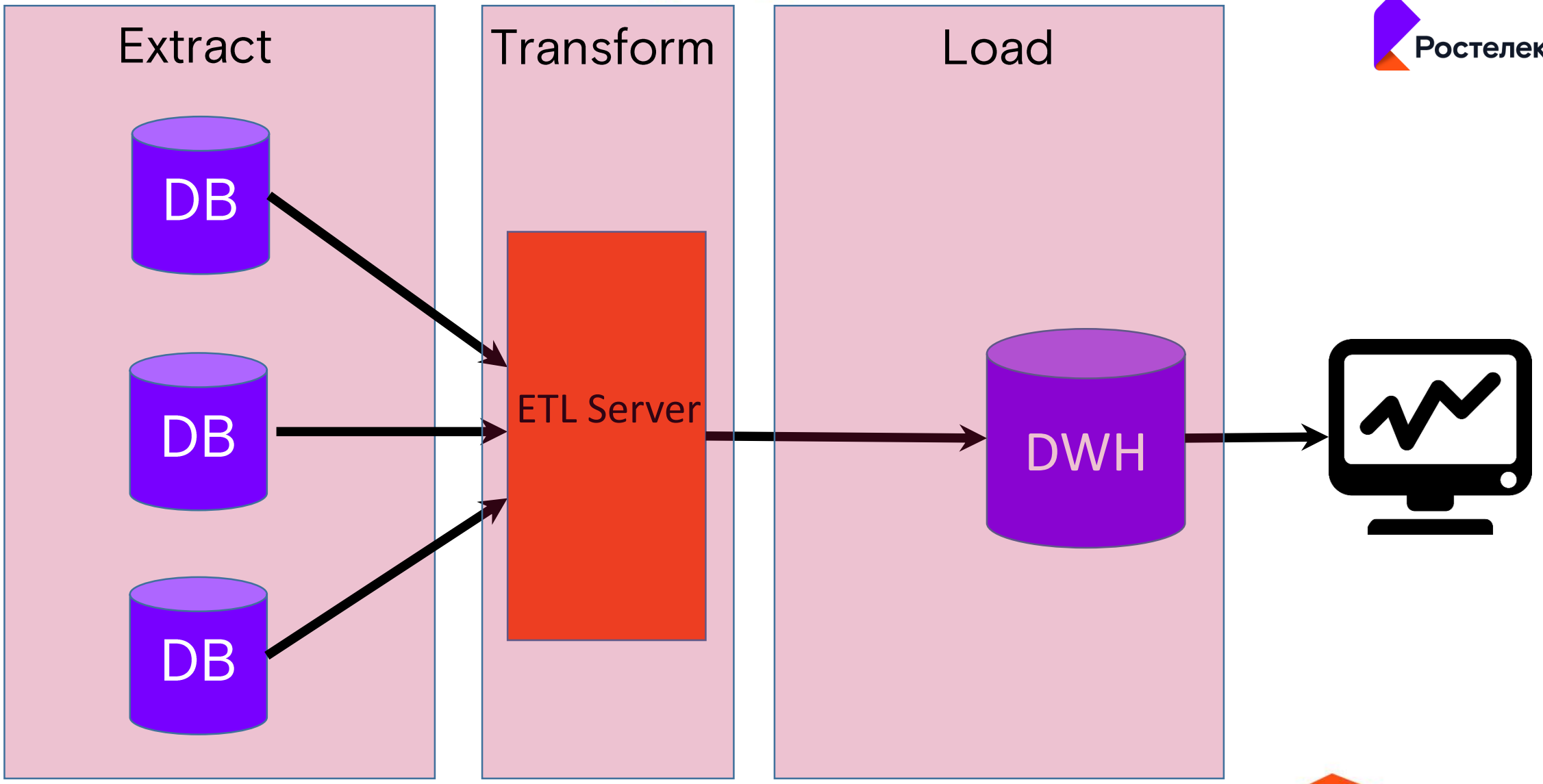


ARENADATA

Ростелеком

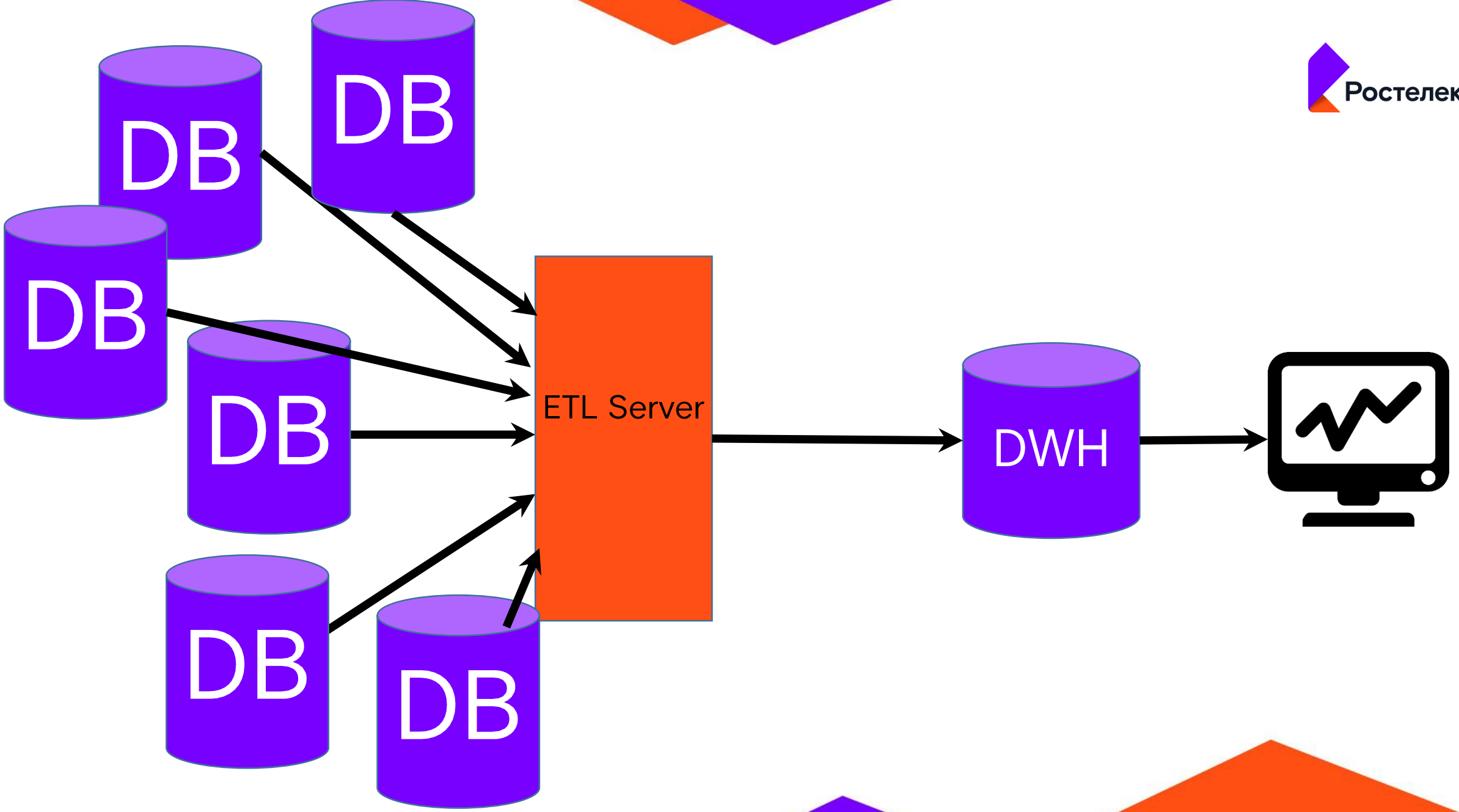


ARENADATA



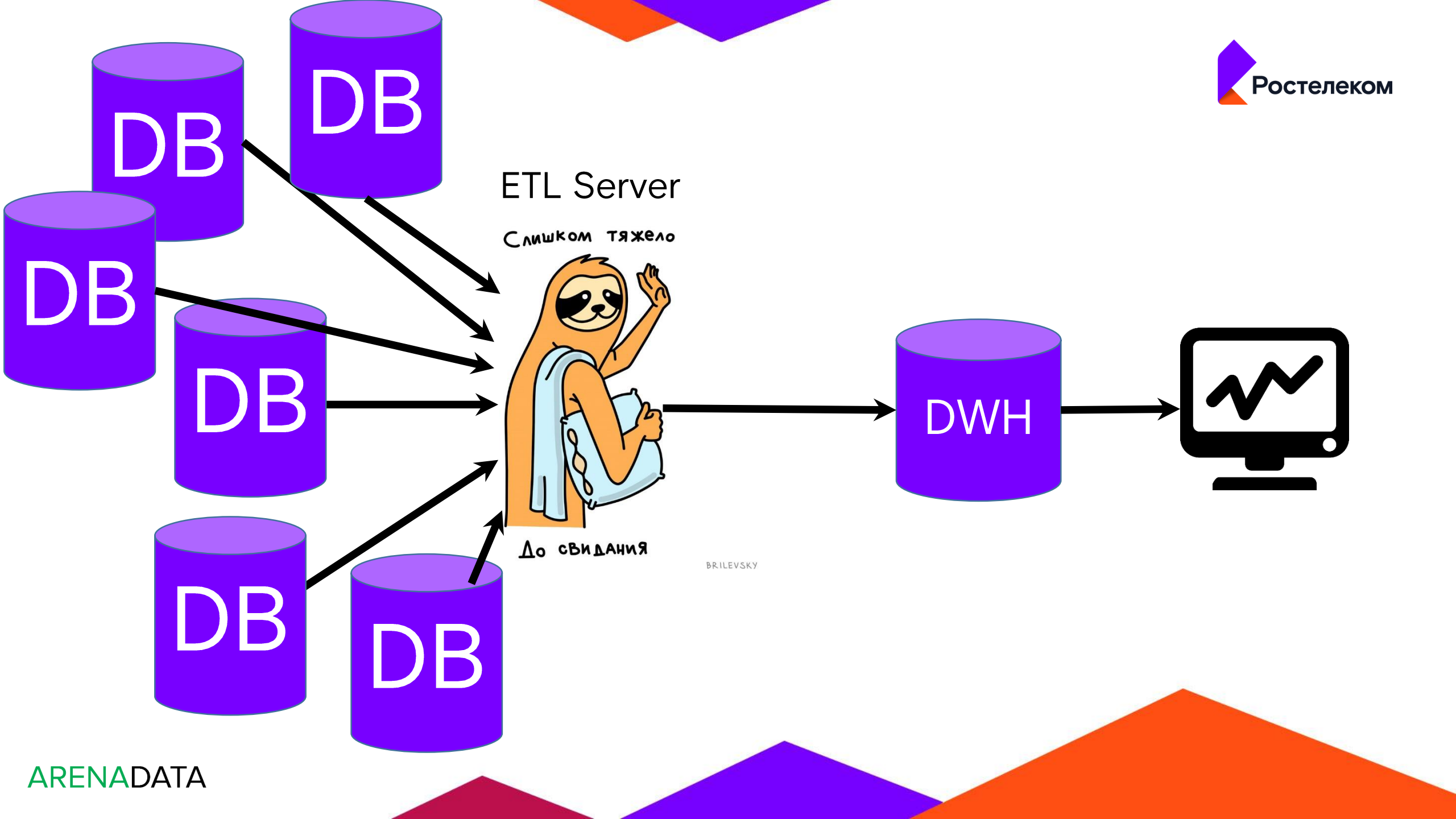
Ростелеком

ARENADATA



Ростелеком

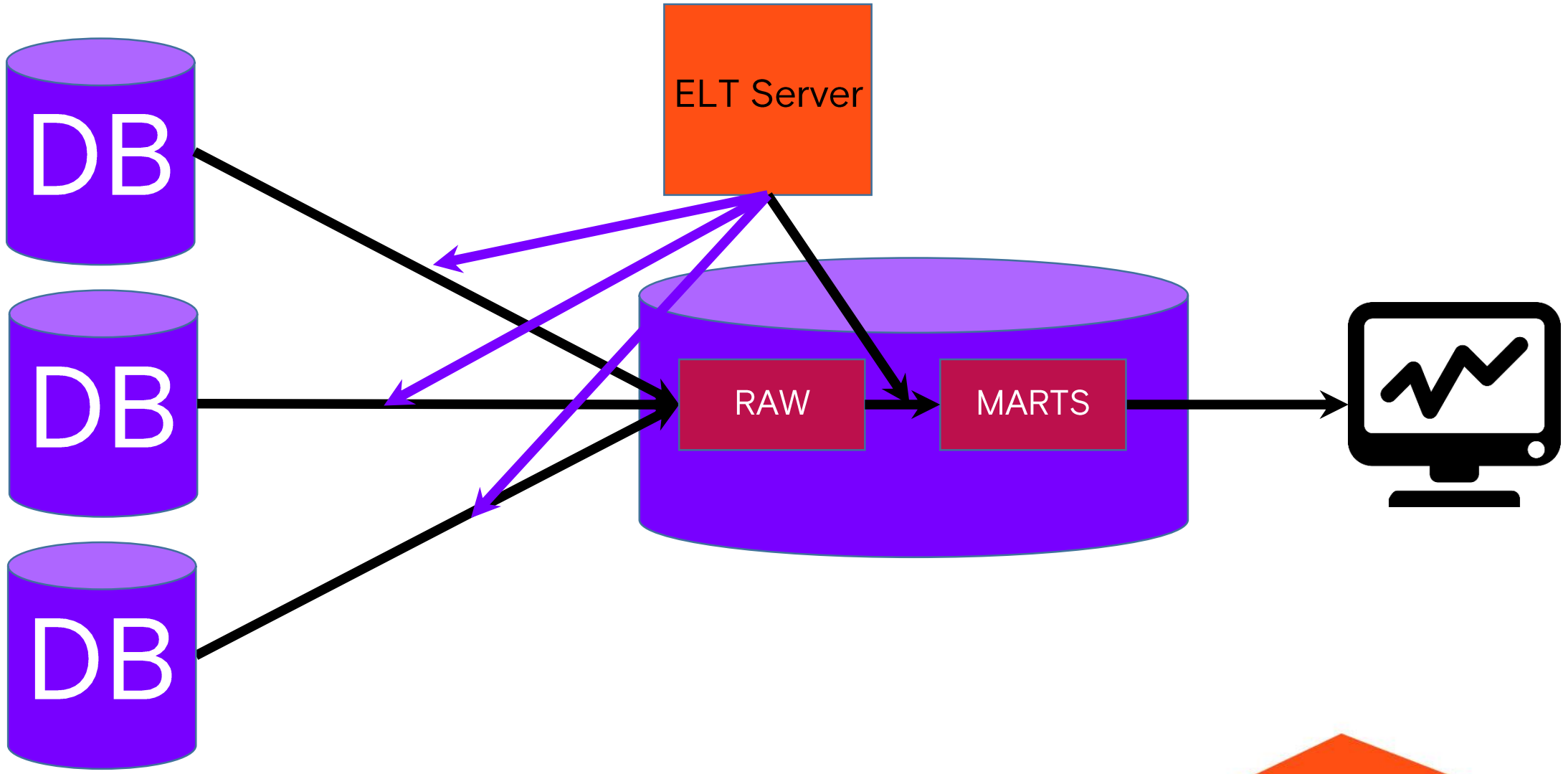
ARENADATA



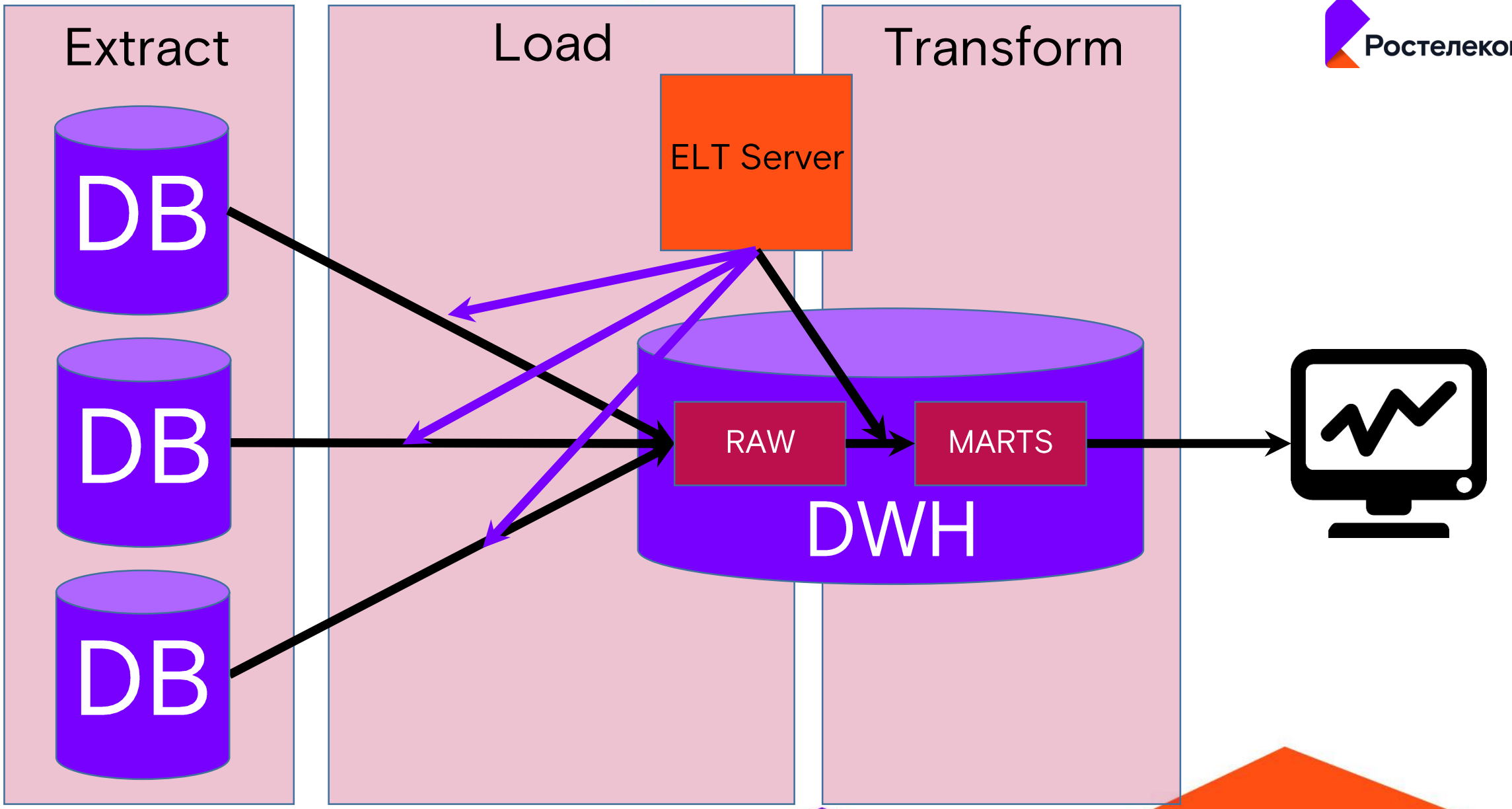
Ростелеком

ARENADATA

Ростелеком

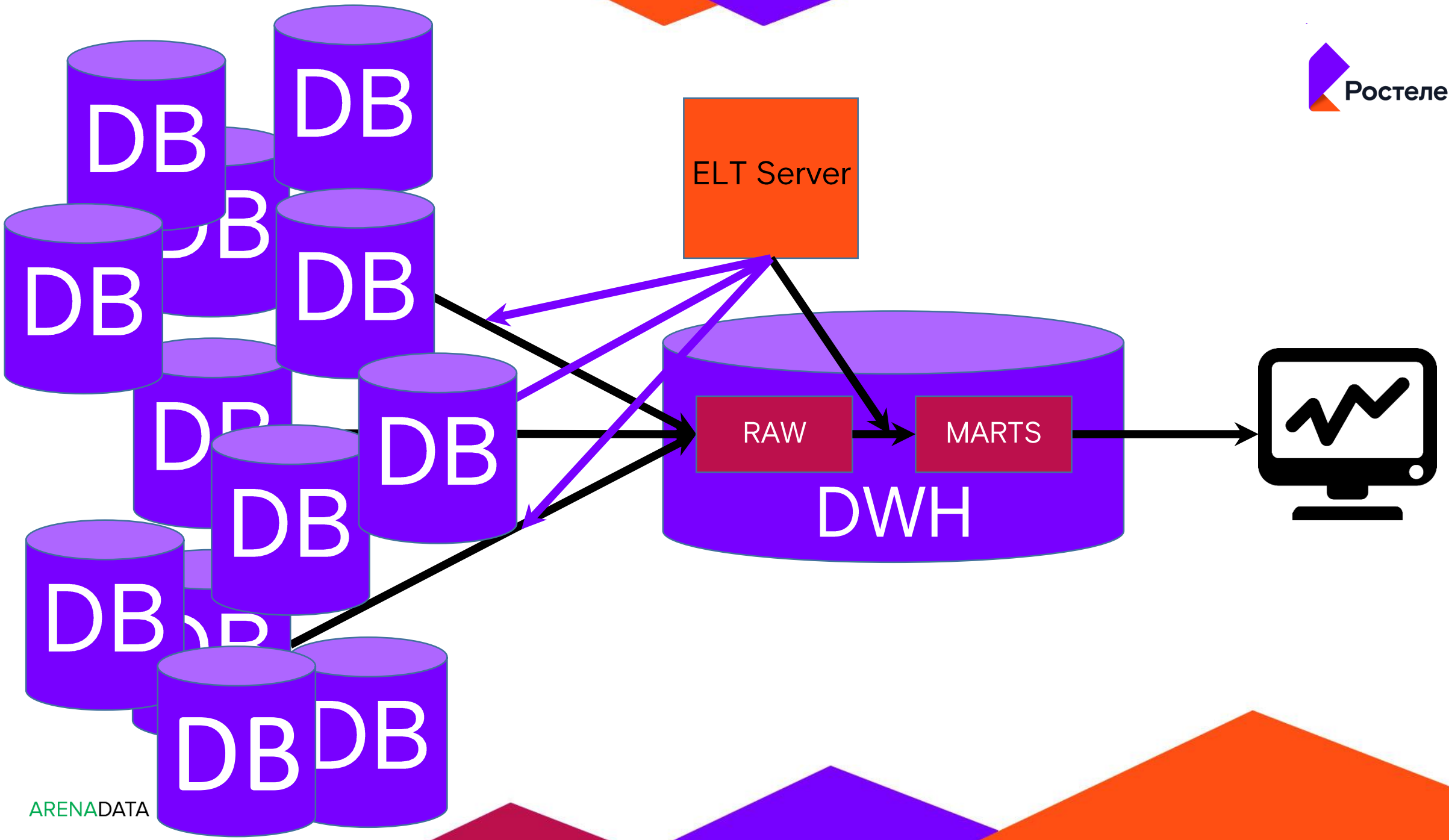


ARENADATA



Ростелеком

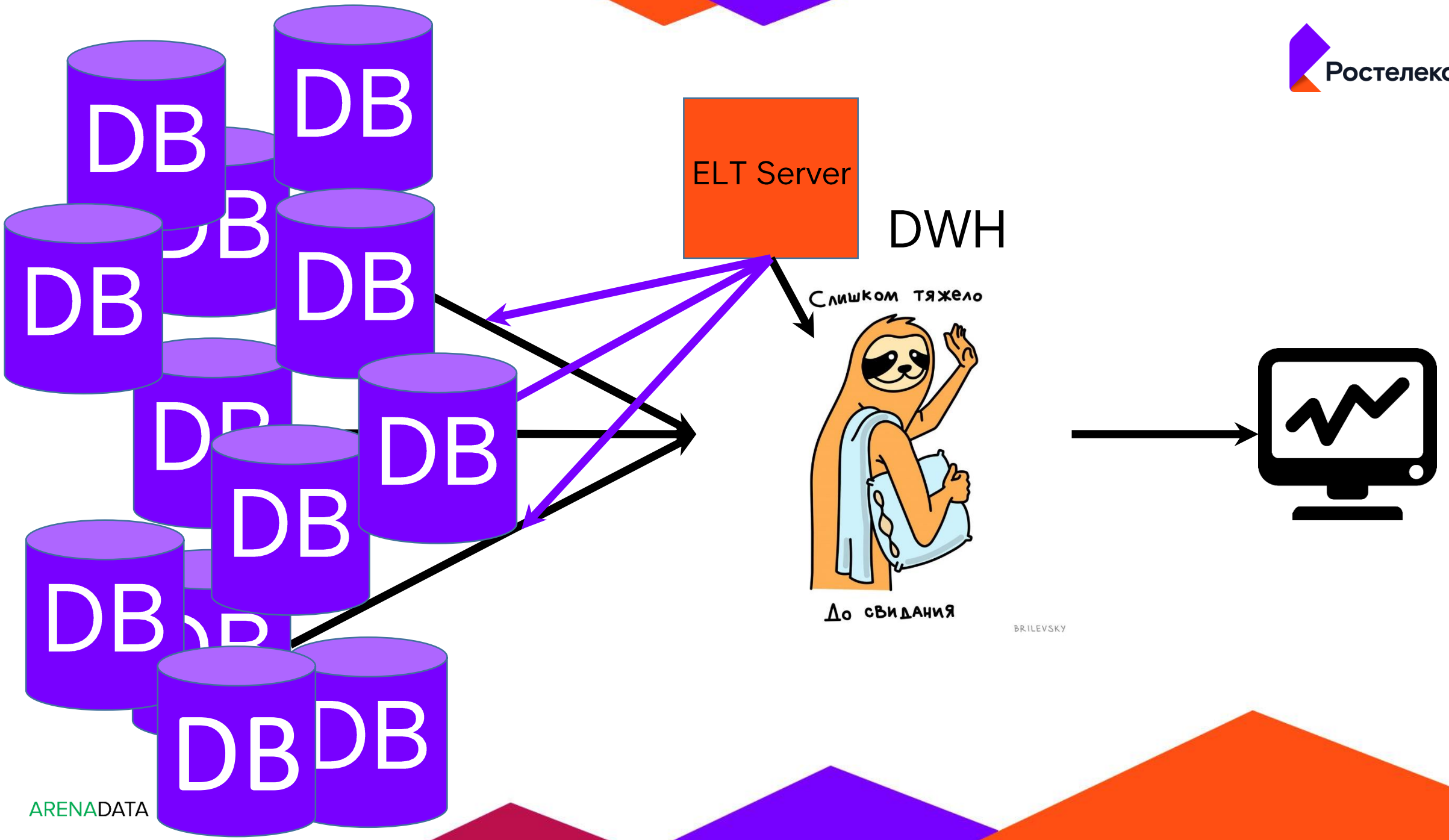
ARENADATA

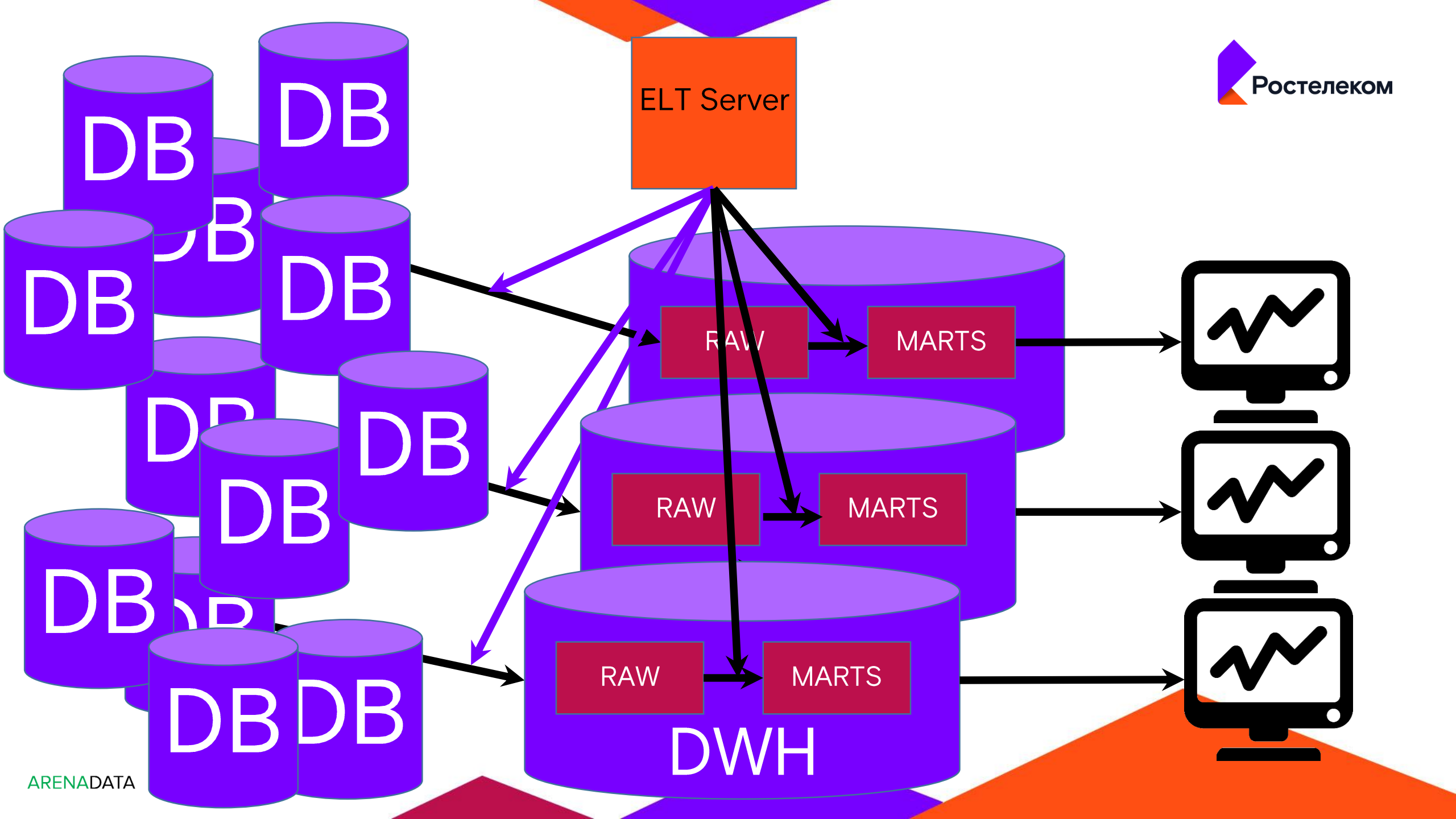


Ростелеком

ARENADATA

Ростелеком


DB
DB
DB
DB
DB
DB
DB
DB
DB
DB
DB
ELT Server
DWH
Слишком тяжело
До свидания
BRILEVSKY


ELT Server
DB
RAW
MARTS
RAW
MARTS
RAW
MARTS
DWH
Ростелеком
ARENADATA

Ростелеком

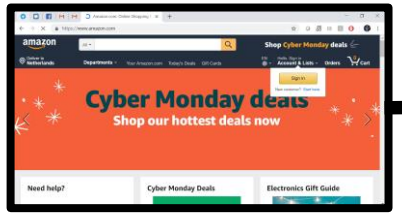
ELT Server

DB

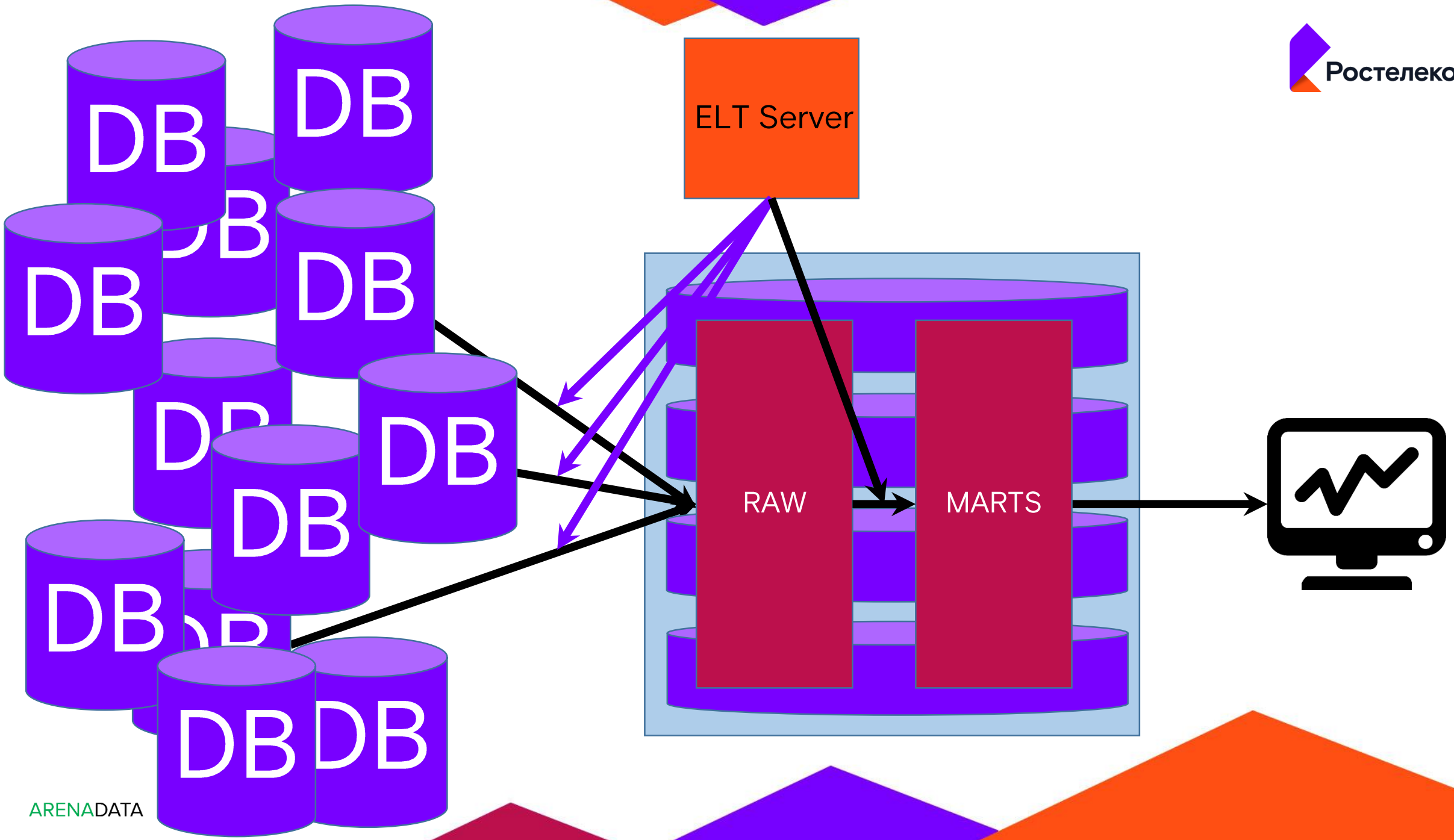
DWH

ARENADATA

Ростелеком

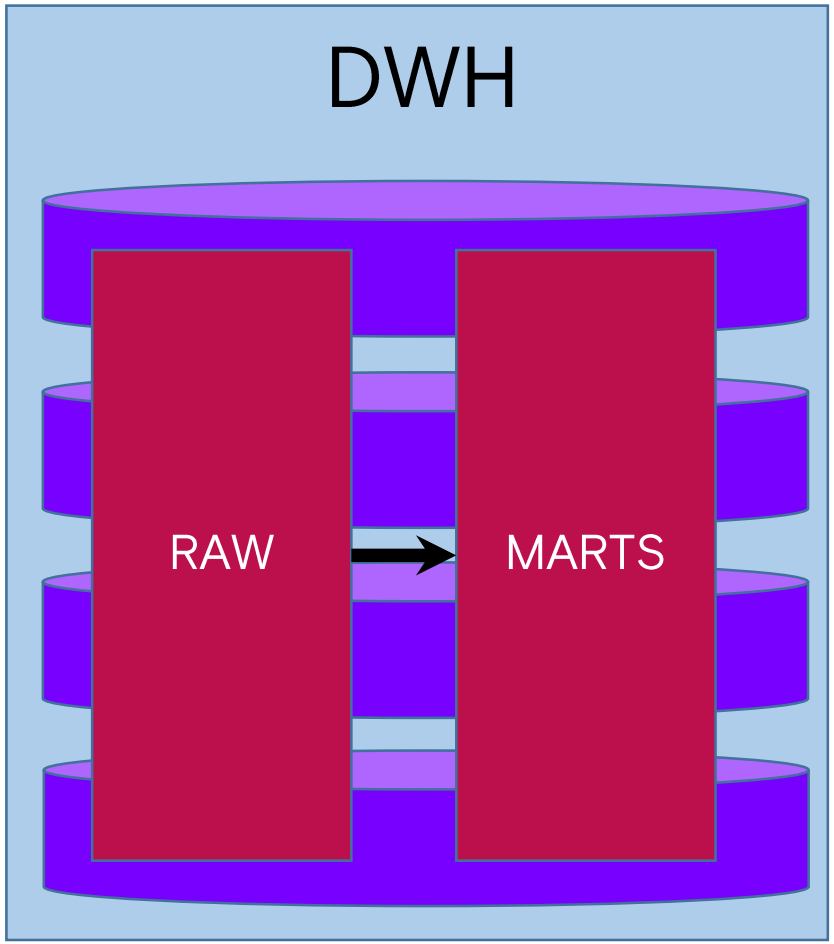


ARENADATA



Ростелеком

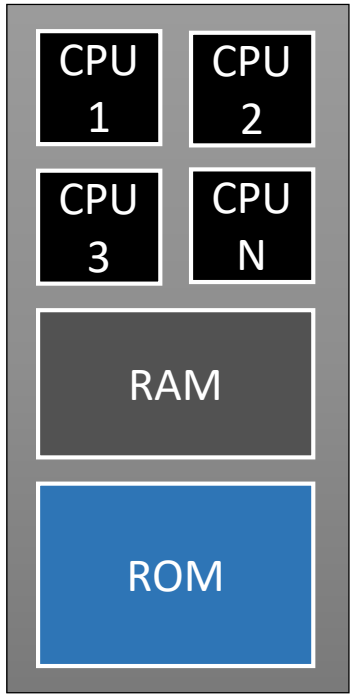
ARENADATA



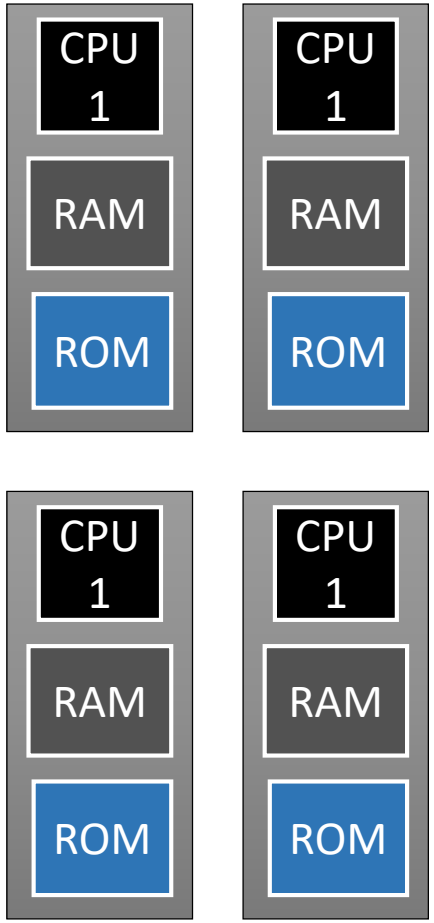
Massively Parallel Processing

- Единый набор данных для всех пользователей
- Масштабируемость с ростом нагрузки

SMP
(symmetric multiprocessing)

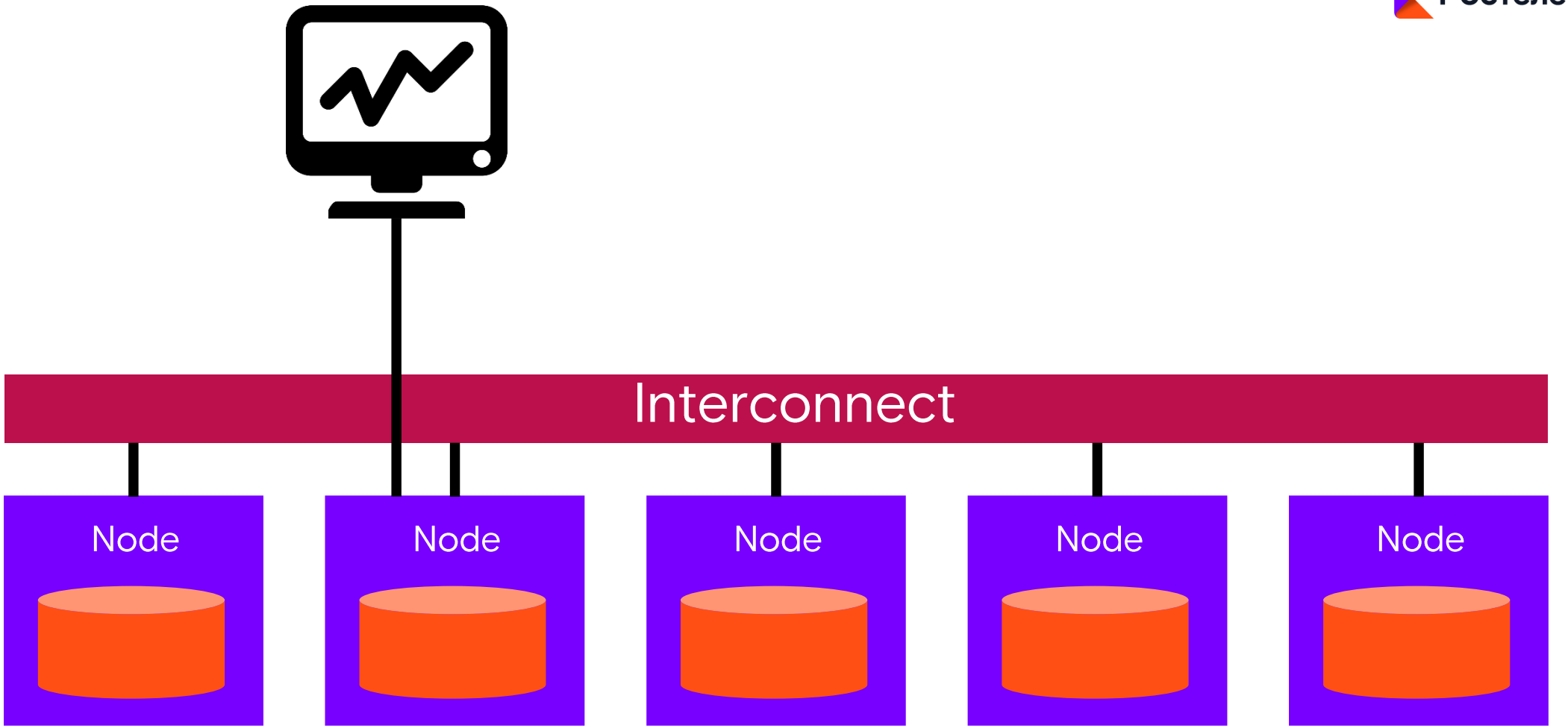


MPP
(massive parallel processing)



- Горизонтальная масштабируемость
- Линейный рост производительности
- Отказоустойчивость
- Таблицы в десятки миллиарды записей
- Сотни Тб данных
- Сотни серверов

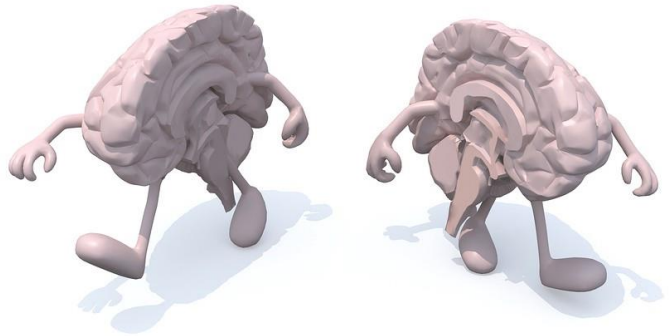
Ростелеком

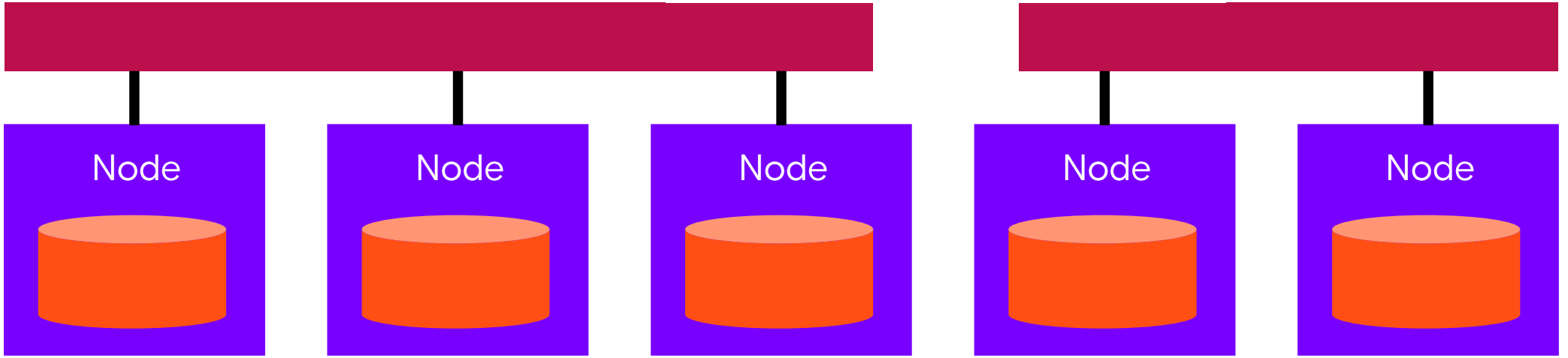


ARENADATA

Ростелеком

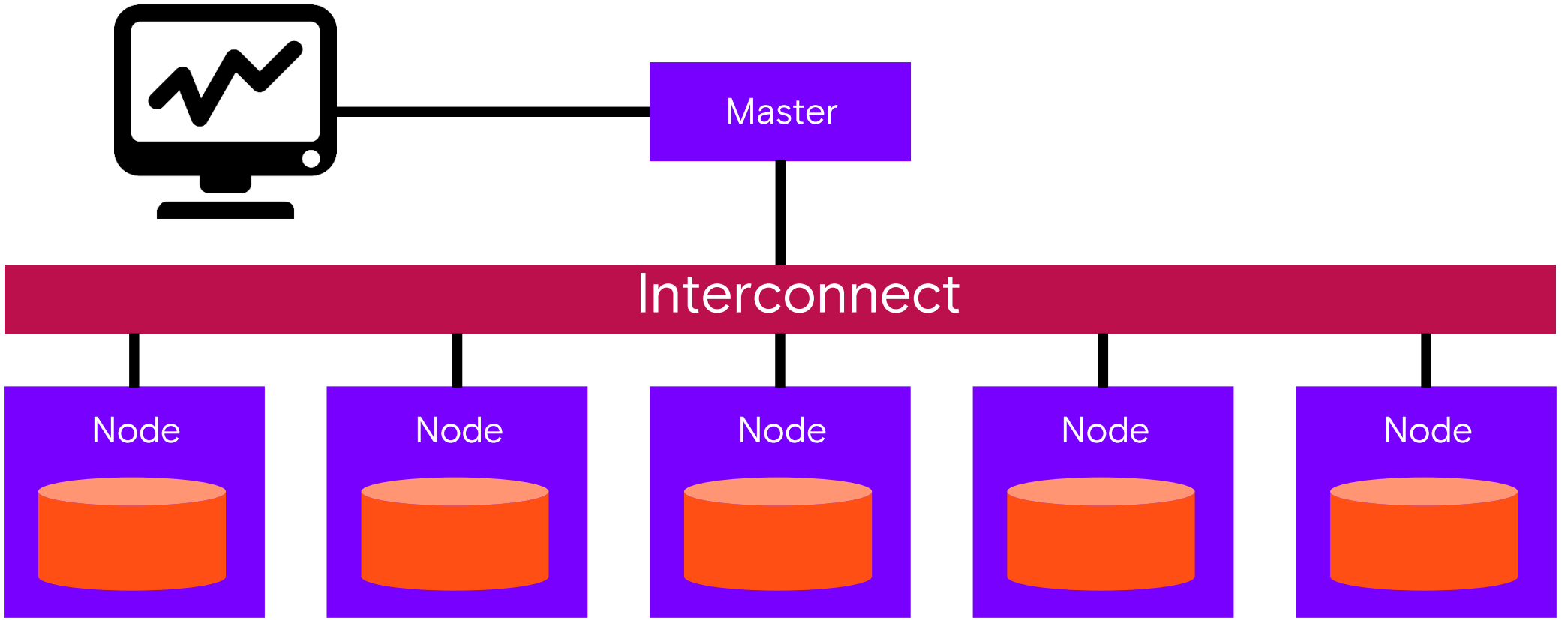
Split Brain



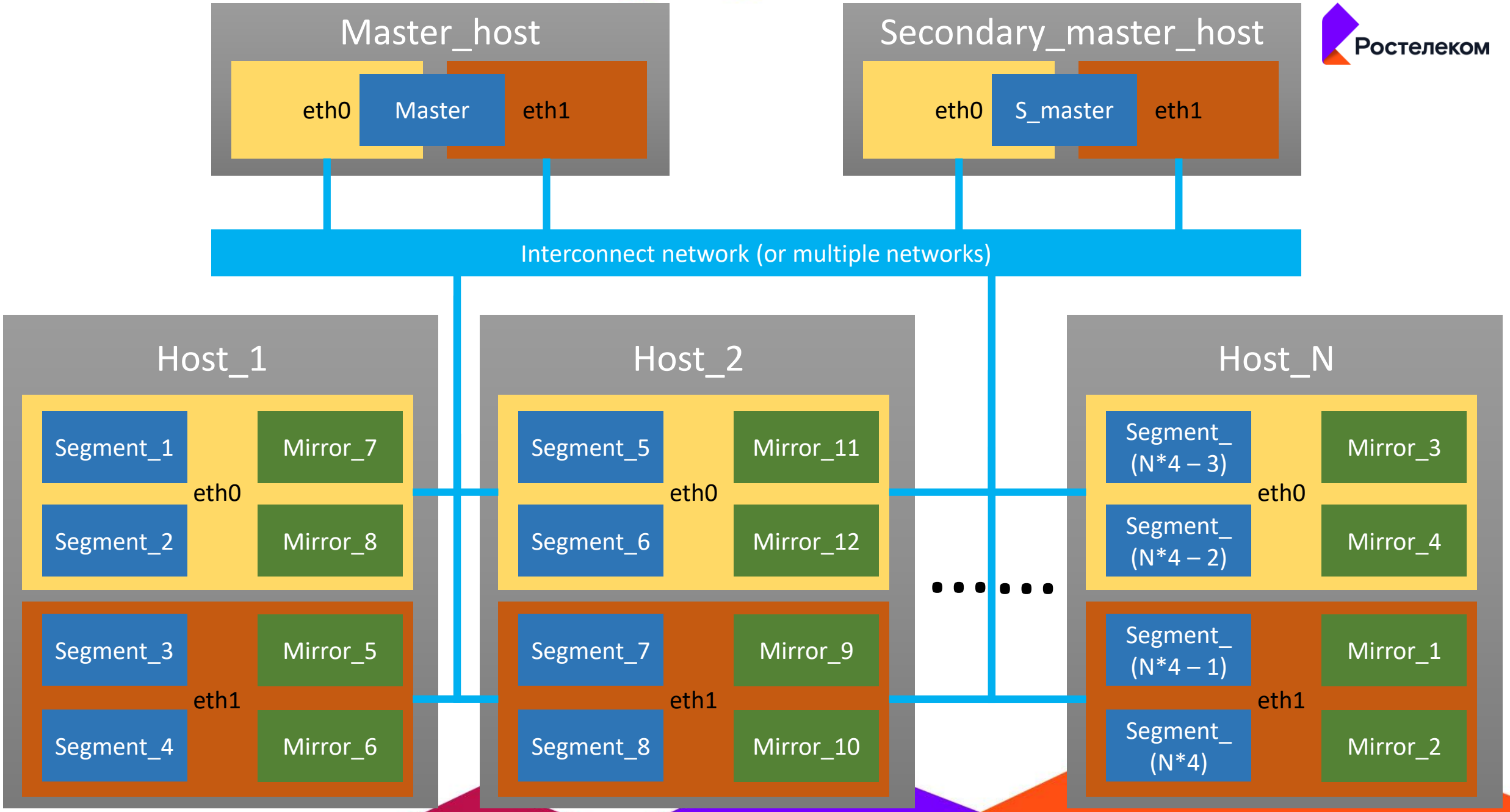


ARENADATA

Ростелеком



ARENADATA


Master_host
eth0
Master
eth1
Secondary_master_host
eth0
S_master
eth1
Interconnect network (or multiple networks)
Host_1
Segment_1
Segment_2
eth0
Mirror_7
Mirror_8
Segment_3
Segment_4
eth1
Mirror_5
Mirror_6
Host_2
Segment_5
Segment_6
eth0
Mirror_11
Mirror_12
Segment_7
Segment_8
eth1
Mirror_9
Mirror_10
Host_N
Segment_ (N*4 – 3)
Segment_ (N*4 – 2)
eth0
Mirror_3
Mirror_4
Segment_ (N*4 – 1)
Segment_ (N*4)
eth1
Mirror_1
Mirror_2

Ростелеком

- Каждая таблица делится на шарды согласно ключу распределения или рандомно
- Каждая запись сохраняется на одной или нескольких шардах

```
CREATE TABLE foo (a int, b text)
DISTRIBUTED BY (a);

adb=# insert into foo values (1,'raz'),(2,'dva');
INSERT 0 2
adb=# select gp_segment_id,* from foo;
 gp_segment_id | a |  b
             2 | 1 | raz
             7 | 2 | dva
```

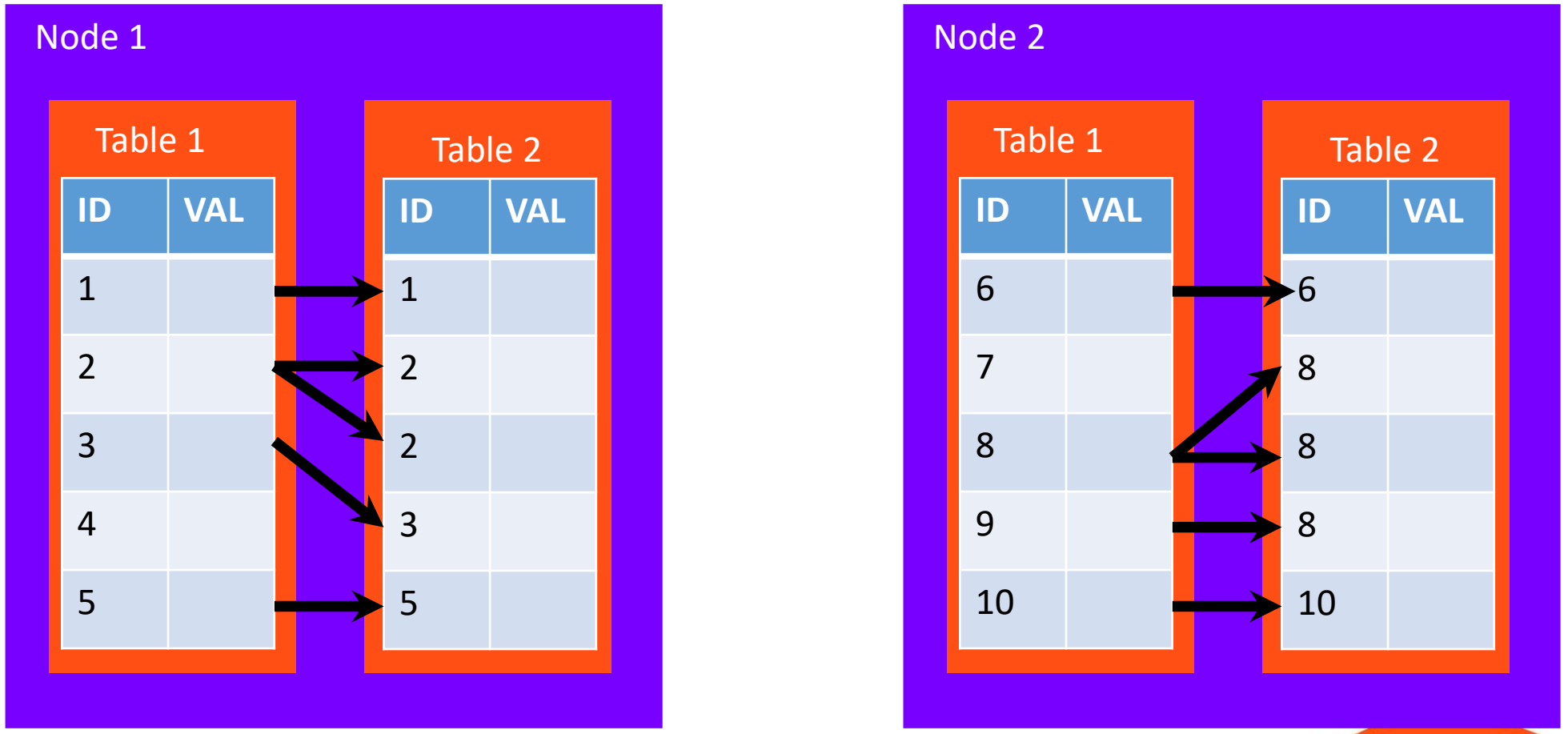
Ростелеком

- SKEW – дисбаланс в распределении данных по ключу распределения

```
adb=# CREATE TABLE foo (a int, b text)
adb-# DISTRIBUTED BY (a);
CREATE TABLE
adb=# insert into foo values (1,'raz'),(1,'dva');
INSERT 0 2
adb=# select gp_segment_id,* from foo;
 gp_segment_id | a |  b
---------------+---+-----
             2 | 1 | raz
             2 | 1 | dva
```

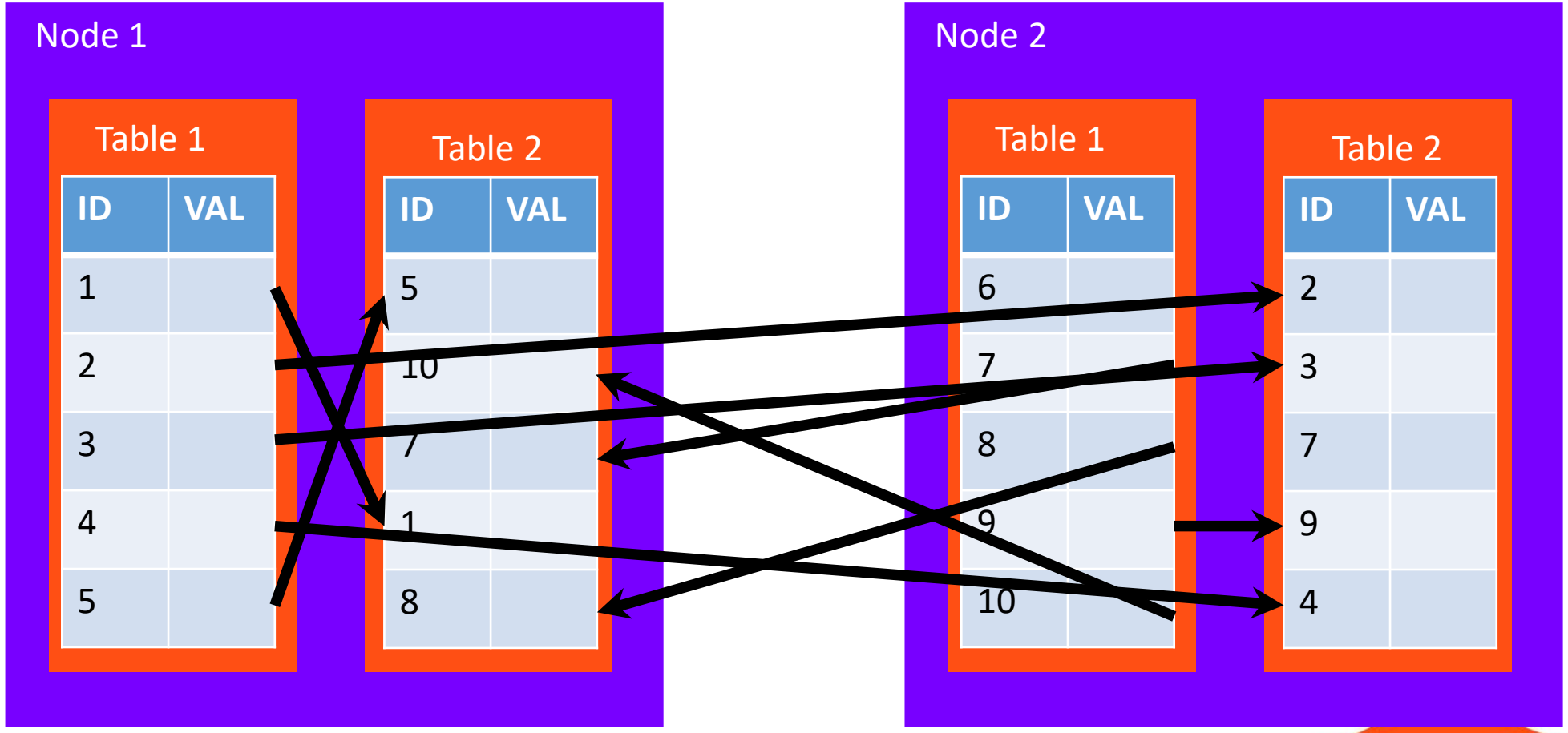
- Путь к успеху – **равномерное распределение данных по сегментам, отсутствие SKEW**;
- **Не выбирайте для ключей дистрибуции поля, записи в которых распределены сильно не равномерно**:
  - Не выбирайте даты;
  - Не выбирайте поля, где может быть большое число NULLs;
  - Не выбирайте поля, в последующем распределении которых вы не уверены (пример: номер телефона и колл-центр);

Local Join (DISTRIBUTED BY ID)



Ростелеком

Distributed Join (DISTRIBUTED RANDOMLY)



```
adb=# CREATE TABLE bar (a int, b text)
adb-# DISTRIBUTED BY (a);
CREATE TABLE
adb=#
adb=# insert into bar values (1,'raz'),(2,'dva');
INSERT 0 2
adb=# explain select * from foo f join bar b on f.a=b.a;
                                  QUERY PLAN
-------------------------------------------------------------------------------
 Gather Motion 8:1  (slice1; segments: 8)  (cost=0.00..862.00 rows=2 width=16)
   ->  Hash Join  (cost=0.00..862.00 rows=1 width=16)
         Hash Cond: foo.a = bar.a
         ->  Table Scan on foo  (cost=0.00..431.00 rows=1 width=8)
         ->  Hash  (cost=431.00..431.00 rows=1 width=8)
               ->  Table Scan on bar  (cost=0.00..431.00 rows=1 width=8)
```

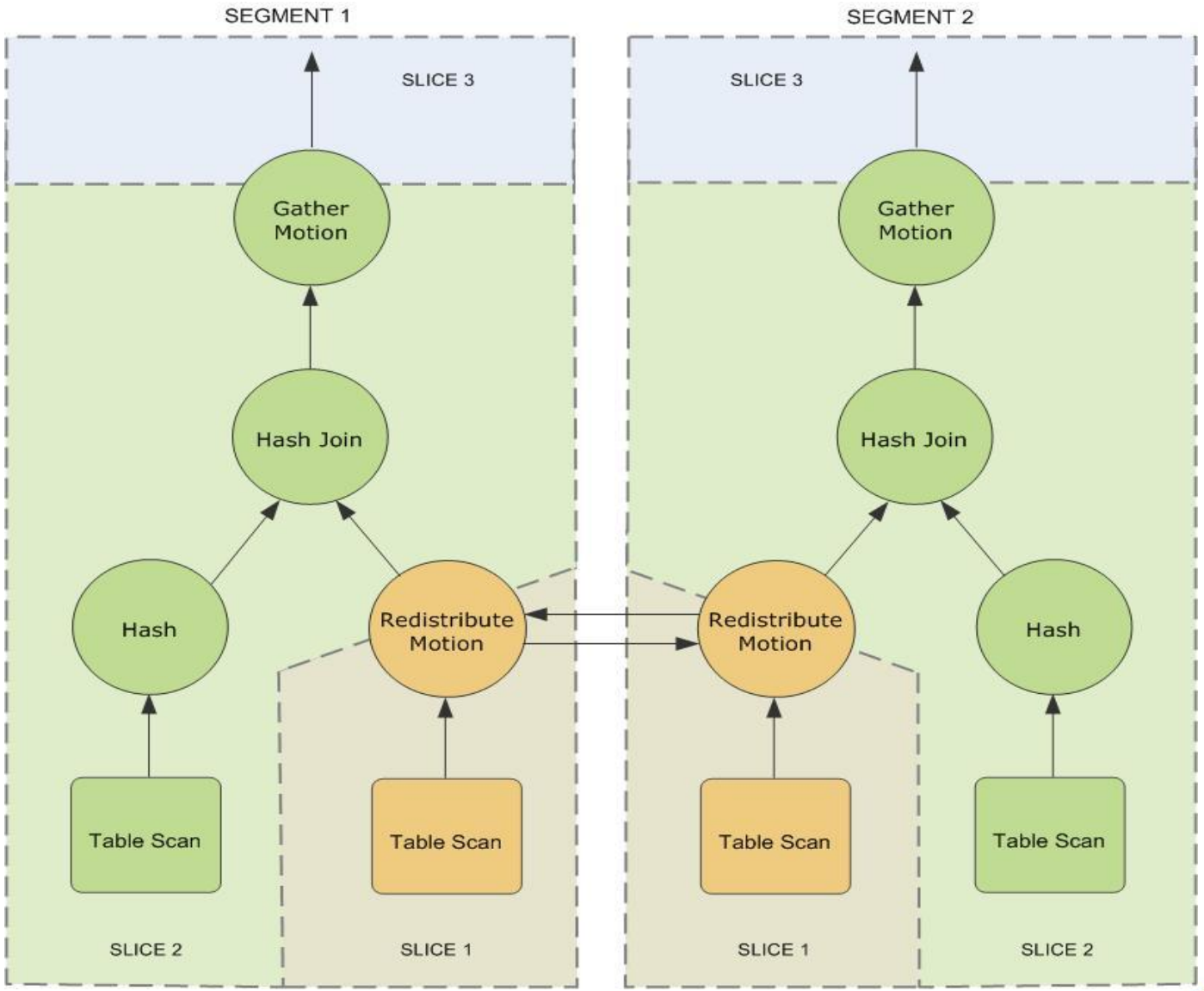
```
adb=# CREATE TABLE bar (a int, b text)
adb-# DISTRIBUTED BY (b);
CREATE TABLE
adb=# insert into bar values (1,'raz'),(2,'dva');
INSERT 0 2
adb=# explain select * from foo f join bar b on f.a=b.a;
                                                    QUERY PLAN
-------------------------------------------------------------------------------------------------------------------
 Gather Motion 8:1  (slice2; segments: 8)  (cost=0.00..862.00 rows=2 width=16)
   ->  Hash Join  (cost=0.00..862.00 rows=1 width=16)
         Hash Cond: foo.a = bar.a
         ->  Table Scan on foo  (cost=0.00..431.00 rows=1 width=8)
         ->  Hash  (cost=431.00..431.00 rows=1 width=8)
               ->  Redistribute Motion 8:8  (slice1; segments: 8)  (cost=0.00..431.00 rows=1 width=8)
                     Hash Key: bar.a
                     ->  Table Scan on bar  (cost=0.00..431.00 rows=1 width=8)
```

- Джойны по ключам распределения – самые быстрые;
- Джойны не по ключам распределения:
  - BROADCAST – репликация малой таблицы по всем сегментам кластера;
  - REDISTRIBUTE – перераспределение таблицы в памяти или в спилл-файлах. Опасность SKEW в памяти сохраняется;

Ростелеком

Резюме

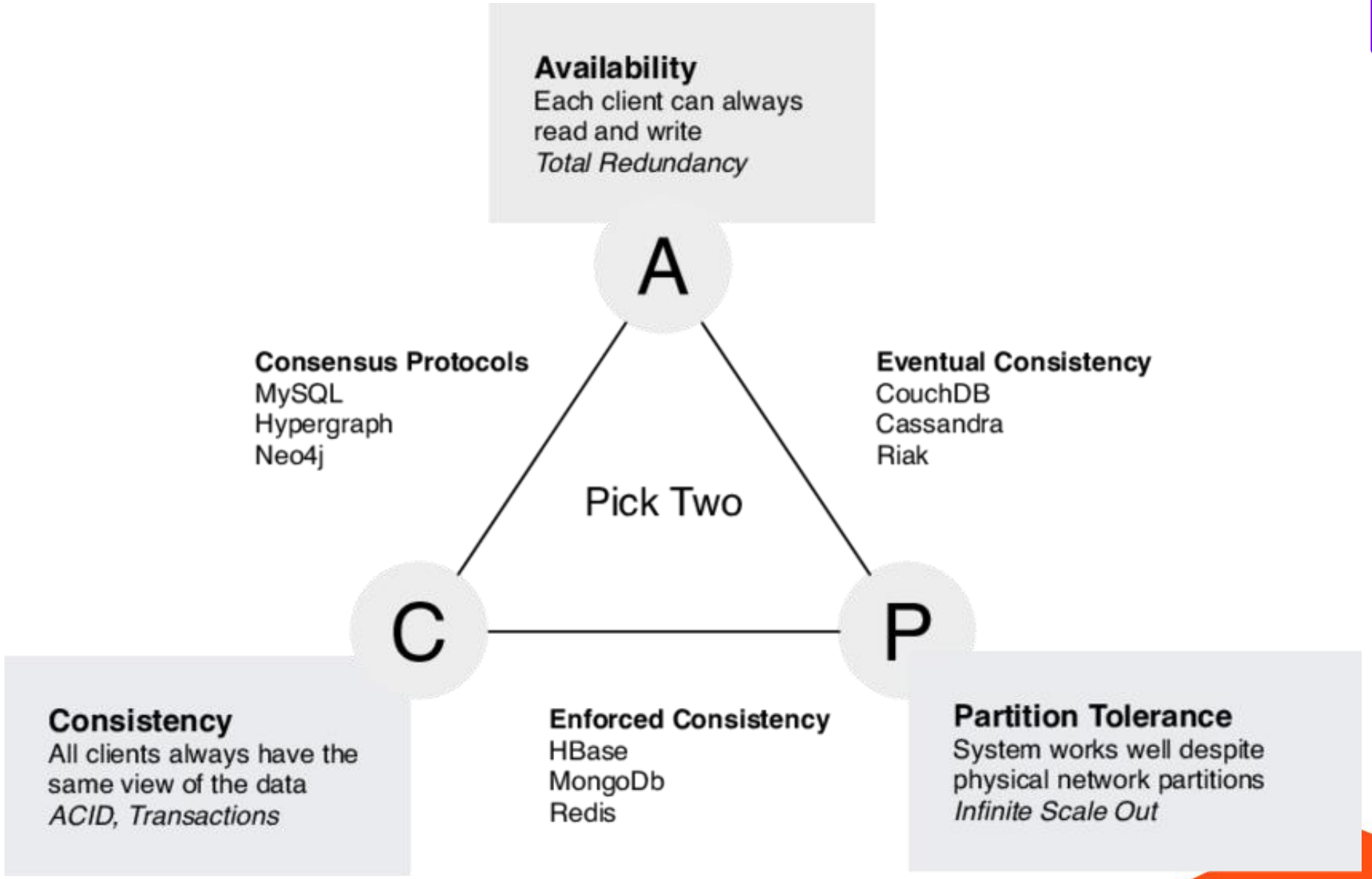
- В MPP системах есть специфика при проектировании схемы хранения данных
- Выбирайте правильные ключи
- Соединяйте таблицы по правильным полям



Ростелеком

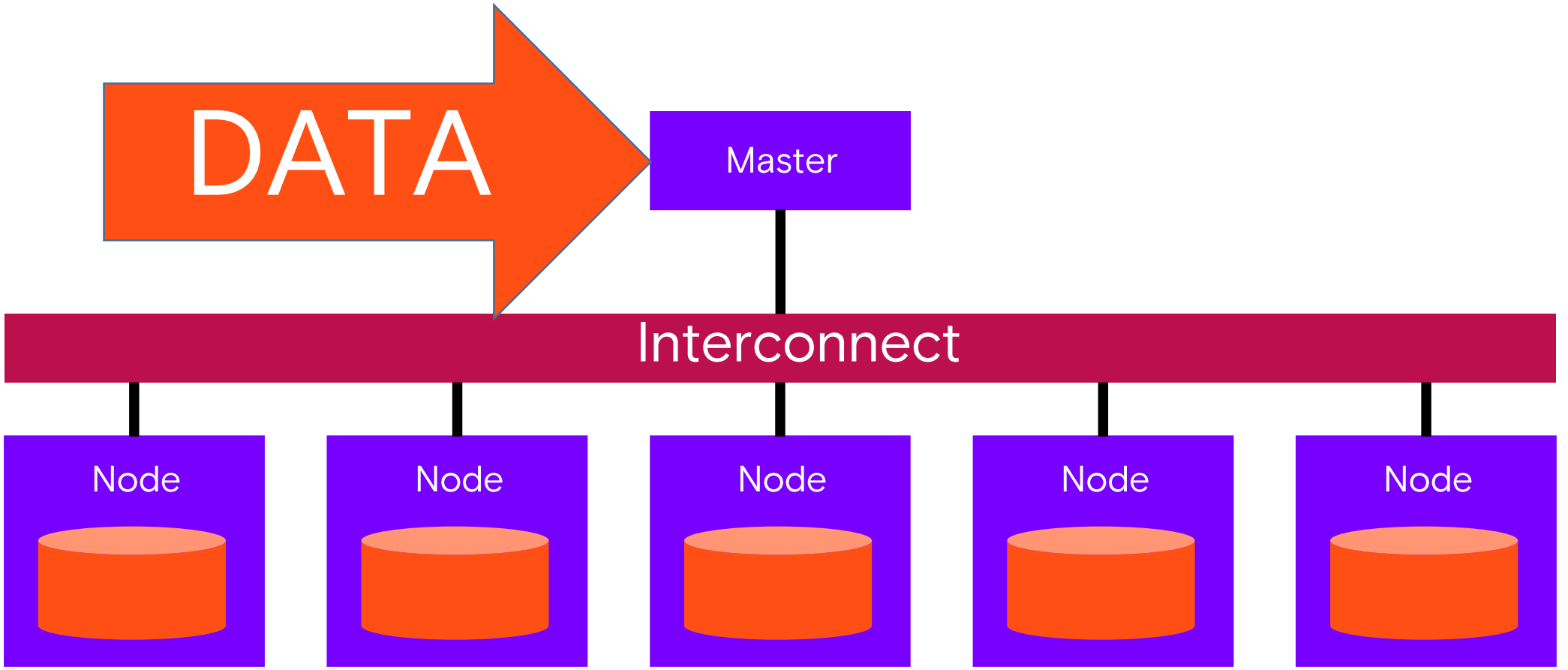
ARENADATA

Ростелеком


Availability
Each client can always read and write
Total Redundancy
A
Consensus Protocols
MySQL
Hypergraph
Neo4j
Eventual Consistency
CouchDB
Cassandra
Riak
Pick Two
C
P
Consistency
All clients always have the same view of the data
ACID, Transactions
Enforced Consistency
HBase
MongoDb
Redis
Partition Tolerance
System works well despite physical network partitions
Infinite Scale Out

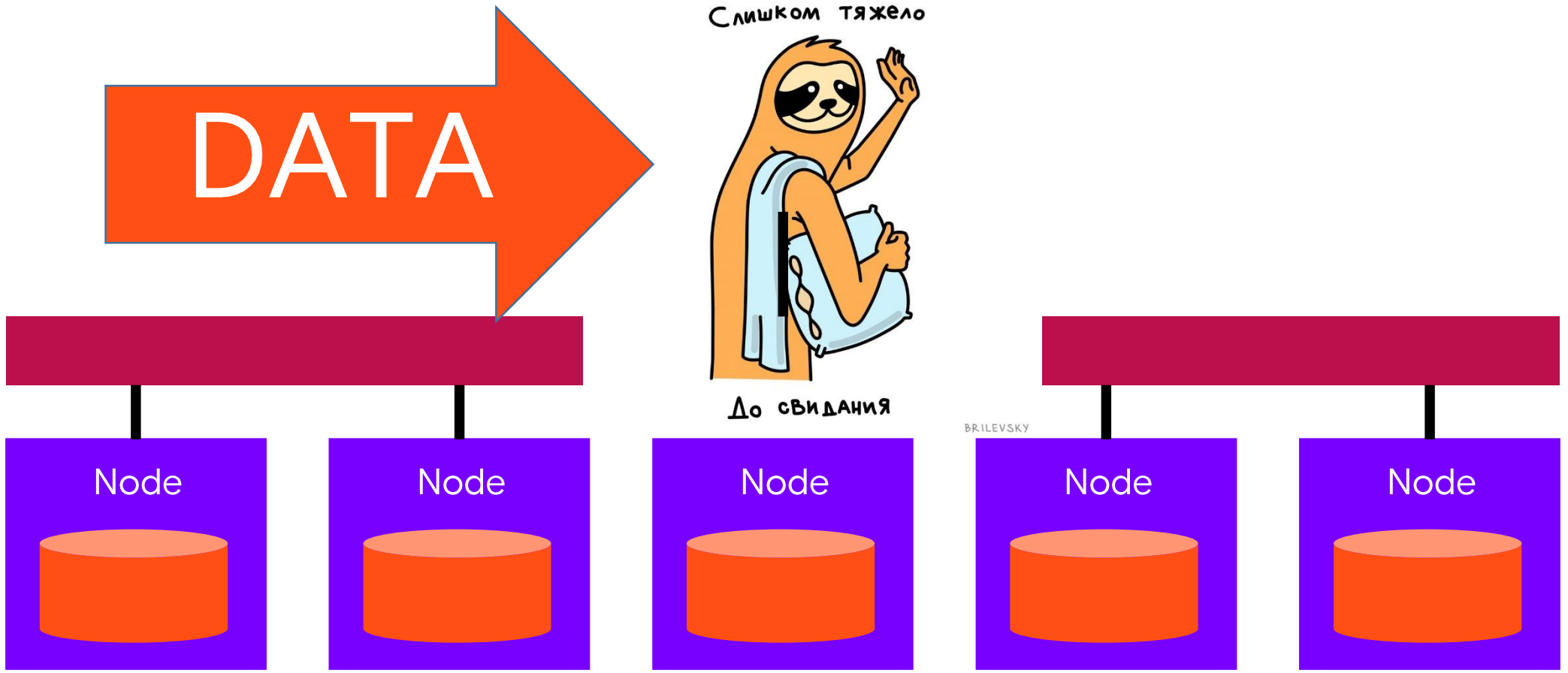
Загрузка больших объёмов данных

Ростелеком



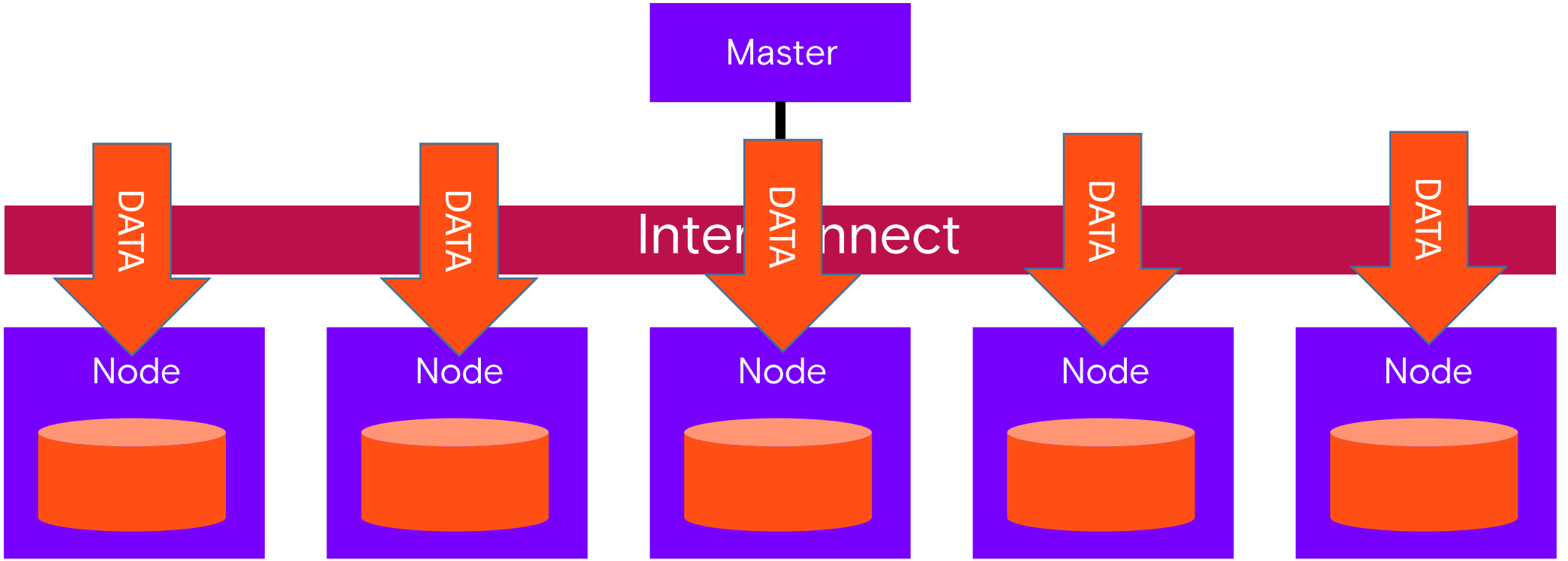
ARENADATA

Загрузка больших объёмов данных



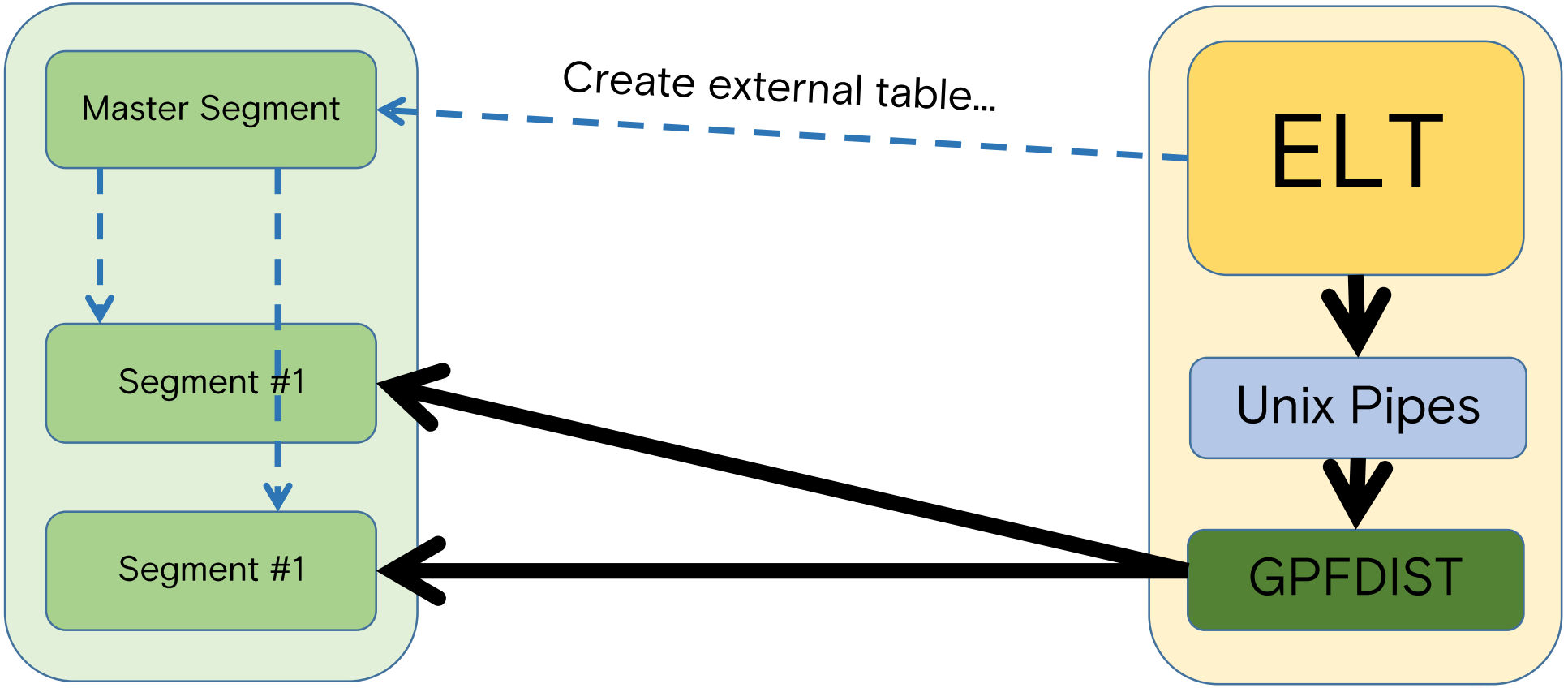
Ростелеком

Загрузка больших объёмов данных



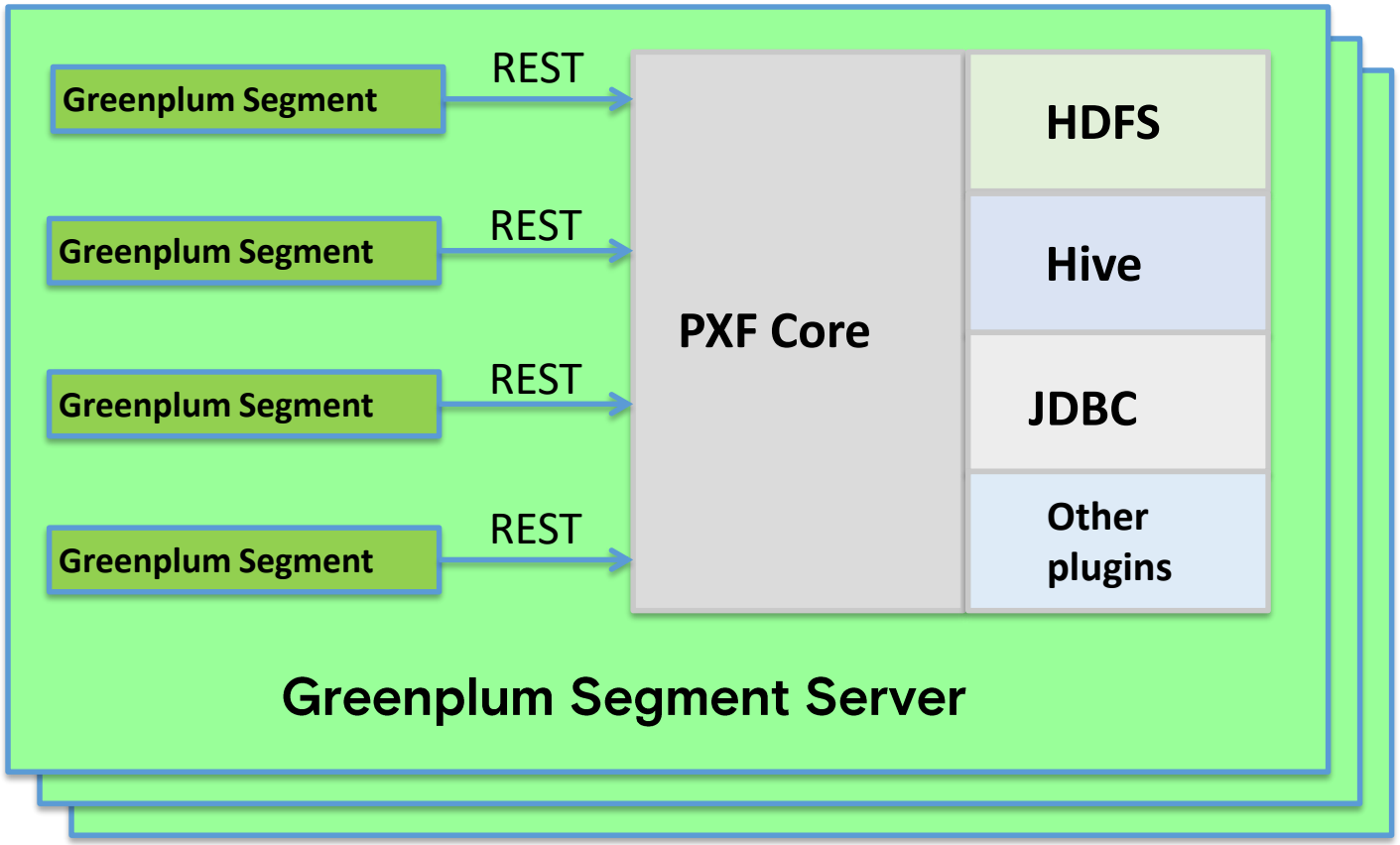
Загрузка больших объёмов данных

Ростелеком



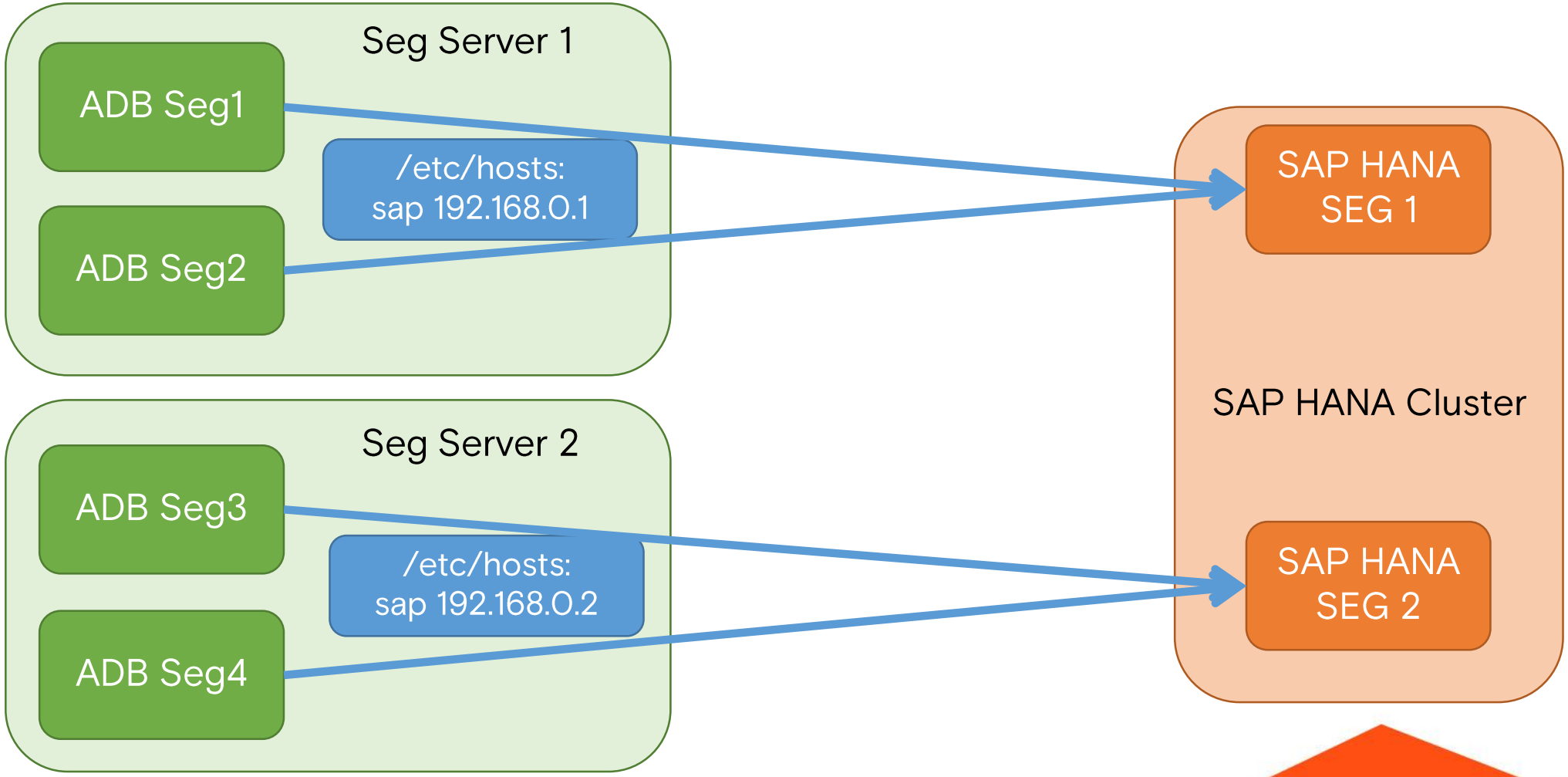
ARENADATA

Параллельная интеграция



Параллельная интеграция - пример

Ростелеком



Ростелеком

Отказоустойчивость

- В прод ландшафте используются все средства резервирования: RAID10, standby master, segment mirrorir
- Существует два крайних случая расположения зеркал

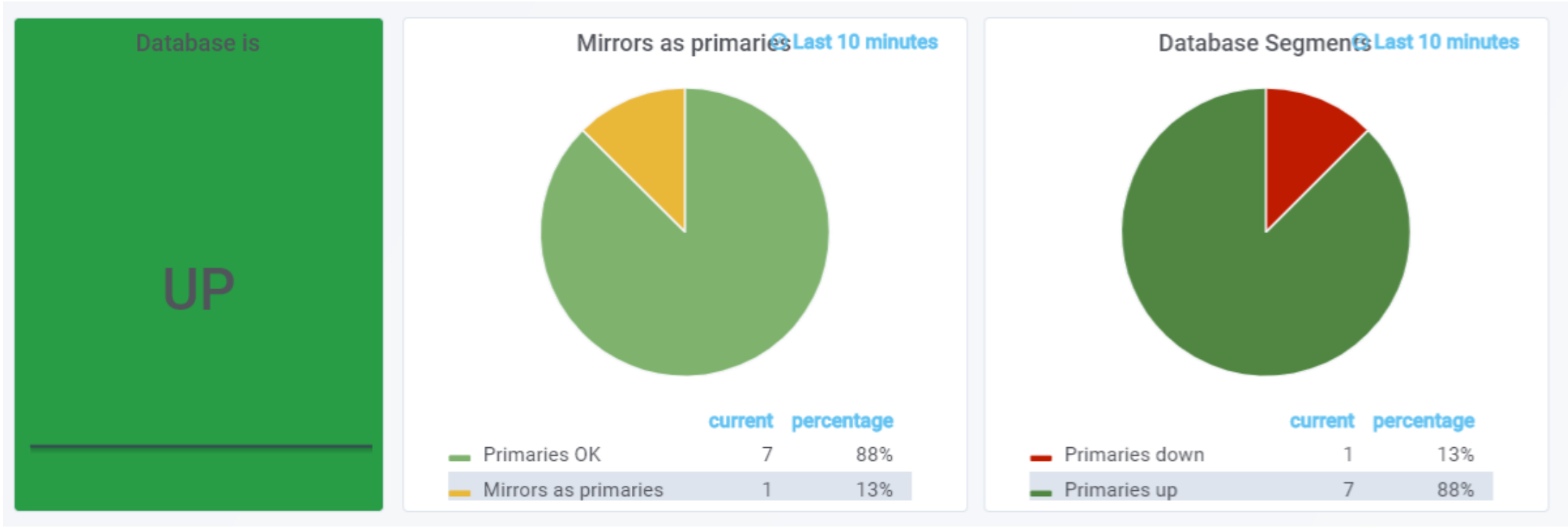
(N – число серверов в кластере, M – число сегментов на сервере):

Group mirroring	Spread mirroring
Зеркала сервера N располагаются на сервере N+1	Зеркала N распределяются по N - (N+M) серверам
При выходе из строя одного сервера уязвимым становится один сервер (обычно соседний)	При выходе из строя одного сервера уязвимыми становятся M серверов
При выходе из строя одного сервера производительность СУБД падает вдвое	При выходе из строя одного сервера производительность СУБД падает в 1/M раз
Возможен при любой конфигурации	Возможен при N>M+1

ARENADATA

Упали сегменты

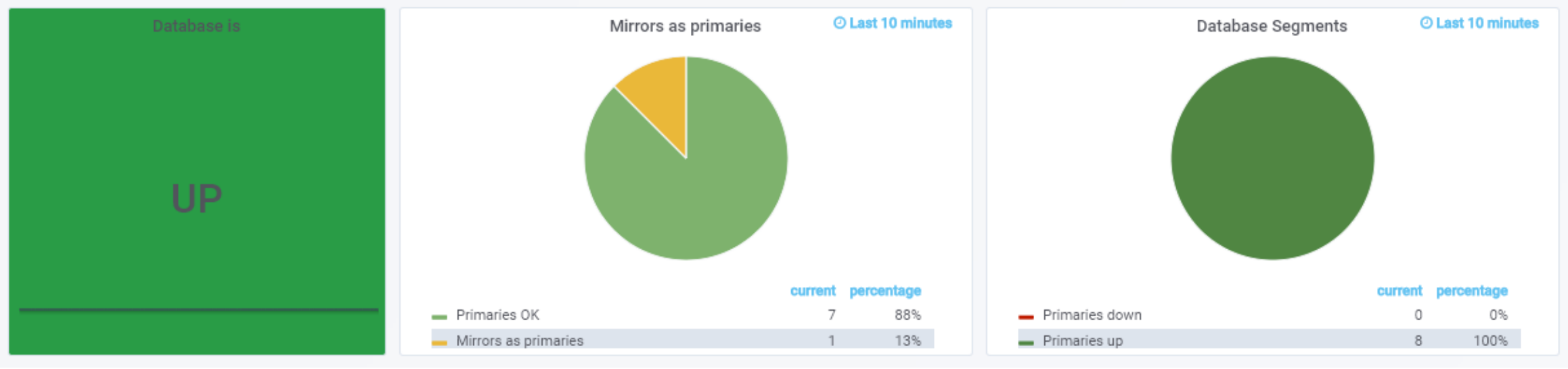
- Вам прилетел алерт. Если не прилетел, то настройте его;
- В мониторинге:



ARENADATA

Ростелеком

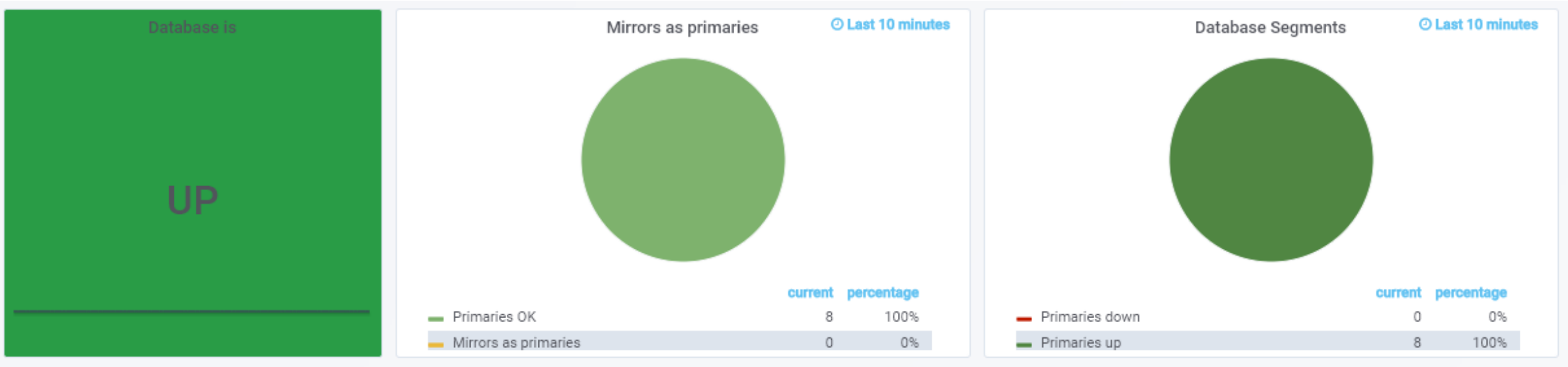
Починили сегмент, но работает зеркало



ARENADATA

Ростелеком

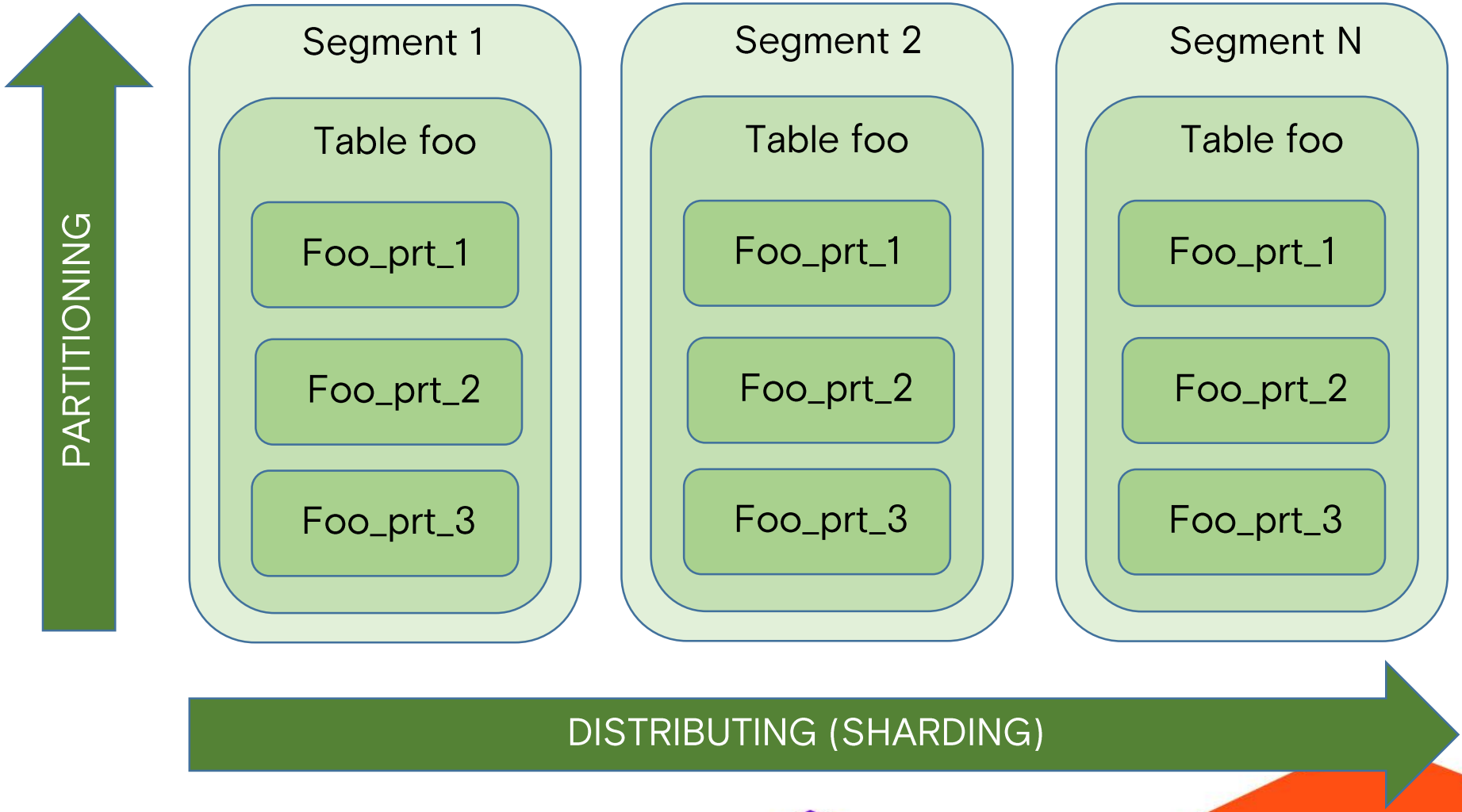
Выполнили ребаланс, ура



ARENADATA

Партиционирование vs шардирование

Ростелеком



ARENADATA

Процедурные языки

- User Defined Function – часть кода, написанная на одном из доступных языков, которую можно выполнять с помощью SQL-запроса
- Функции принимают параметры
- Функции возвращаю результат в виде одной или нескольких строк
- Функции могут работать на мастере и на сегментах:
  - Select function() – функция выполнится на мастере
  - Select function(field) from table – функция выполнится на сегментах

Процедурные языки

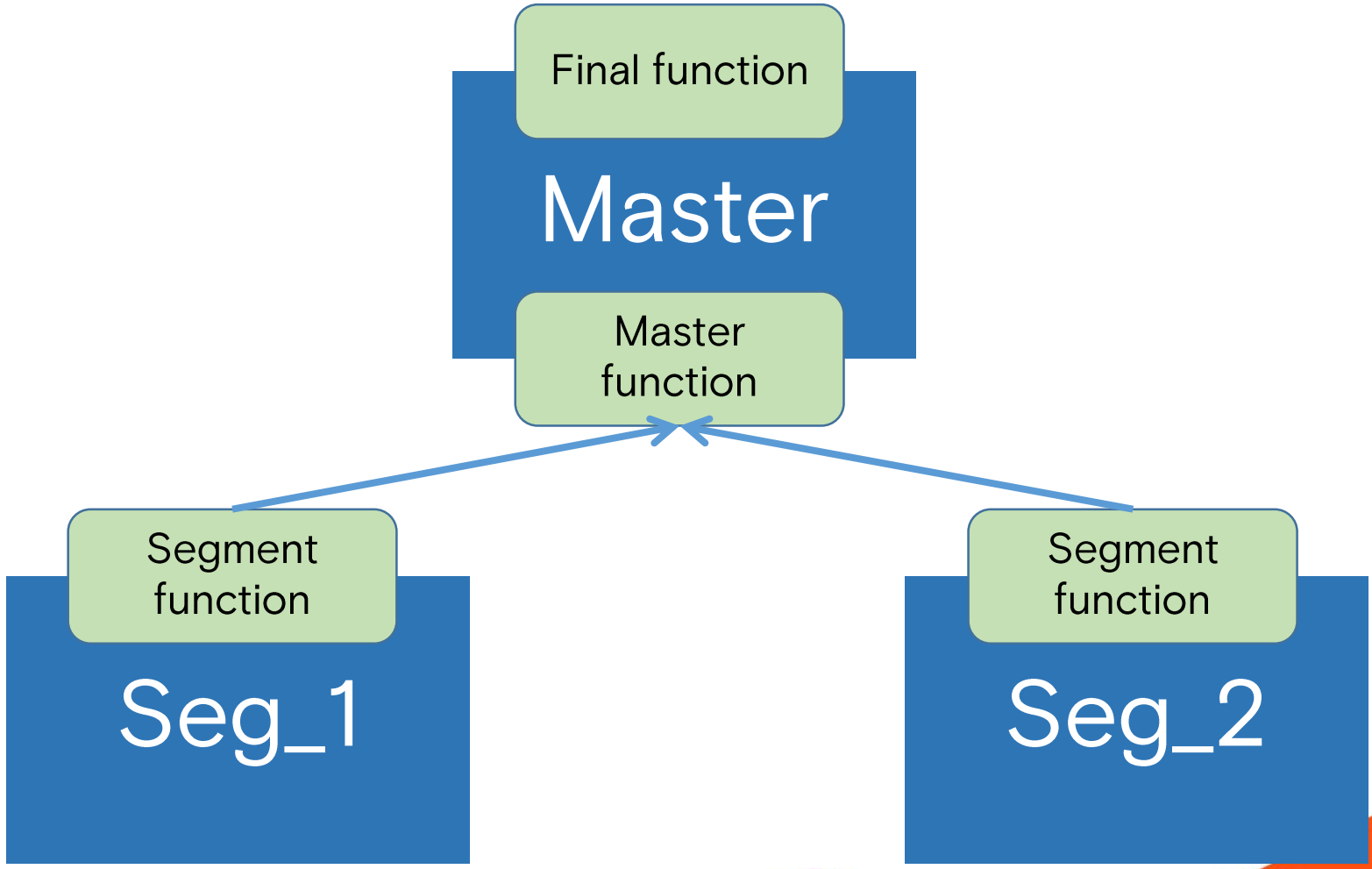
- Доступны следующие языки:
  - PL/pgSQL – trusted (установлен по-умолчанию)
  - PL/R – untrusted
  - PL/Python – untrusted (установлен по-умолчанию)
  - PL/Container (доступны Python и R) – trusted (установлен по-умолчанию)
  - PL/Java – trusted и untrusted
  - PL/Perl – trusted и untrusted
  - PL/C – trusted

Процедурные языки – PL/Python

```
create or replace function extract_host_simple(url text)
returns text
as $$
       import urlparse
       return urlparse.urlparse(url).netloc
$$
language plpythonu;
```

Ростелеком

Процедурные языки – аггрегативные функции

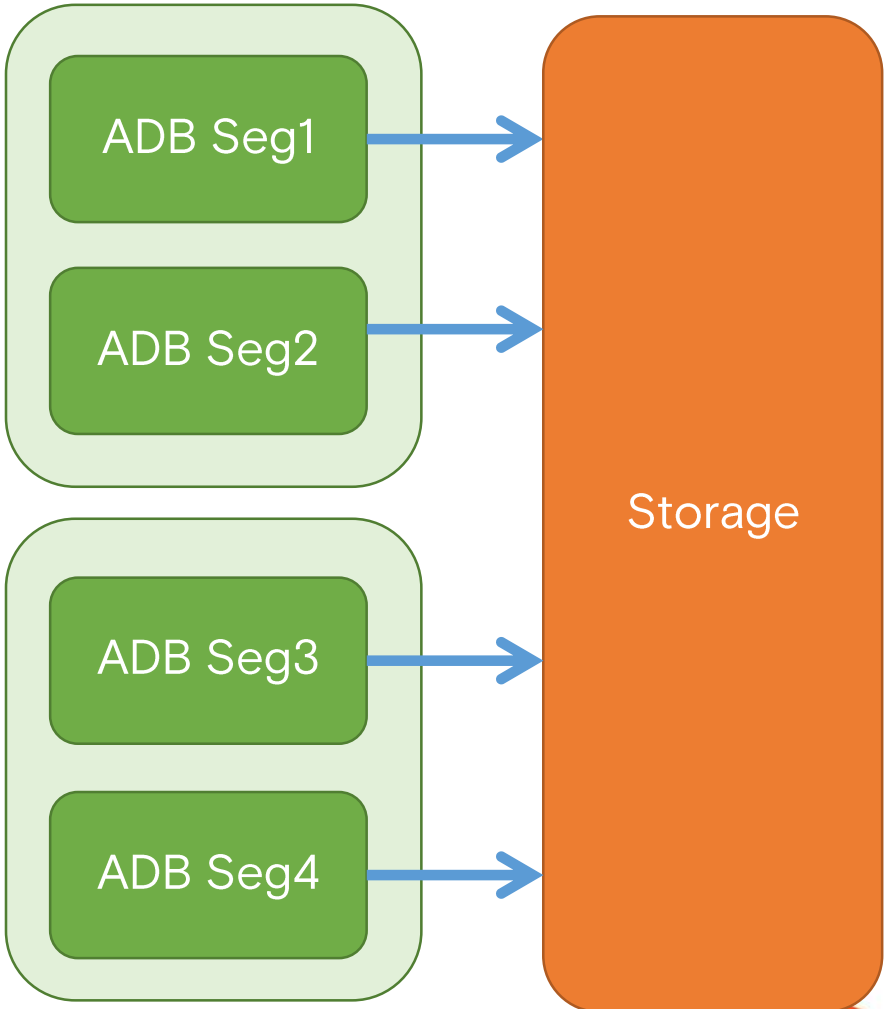


Бекапы

Ростелеком

Администраторы делятся на два типа:

- те, кто не делает бэкапы
- те, кто уже делает



Ростелеком

Главное правило массивно-параллельных систем:

Если что-то дешевле сделать не используя массивно-параллельные системы, не используйте массивно-параллельные системы

Вопросы?