HYBRID STORAGE

DEVELOPMENT STATE

GOLOSOVA MARINA

LABORATORY OF BIG DATA TECHNOLOGIES
FOR MEGA-SCIENCE PROJECTS

29/01/2015

REVIEW

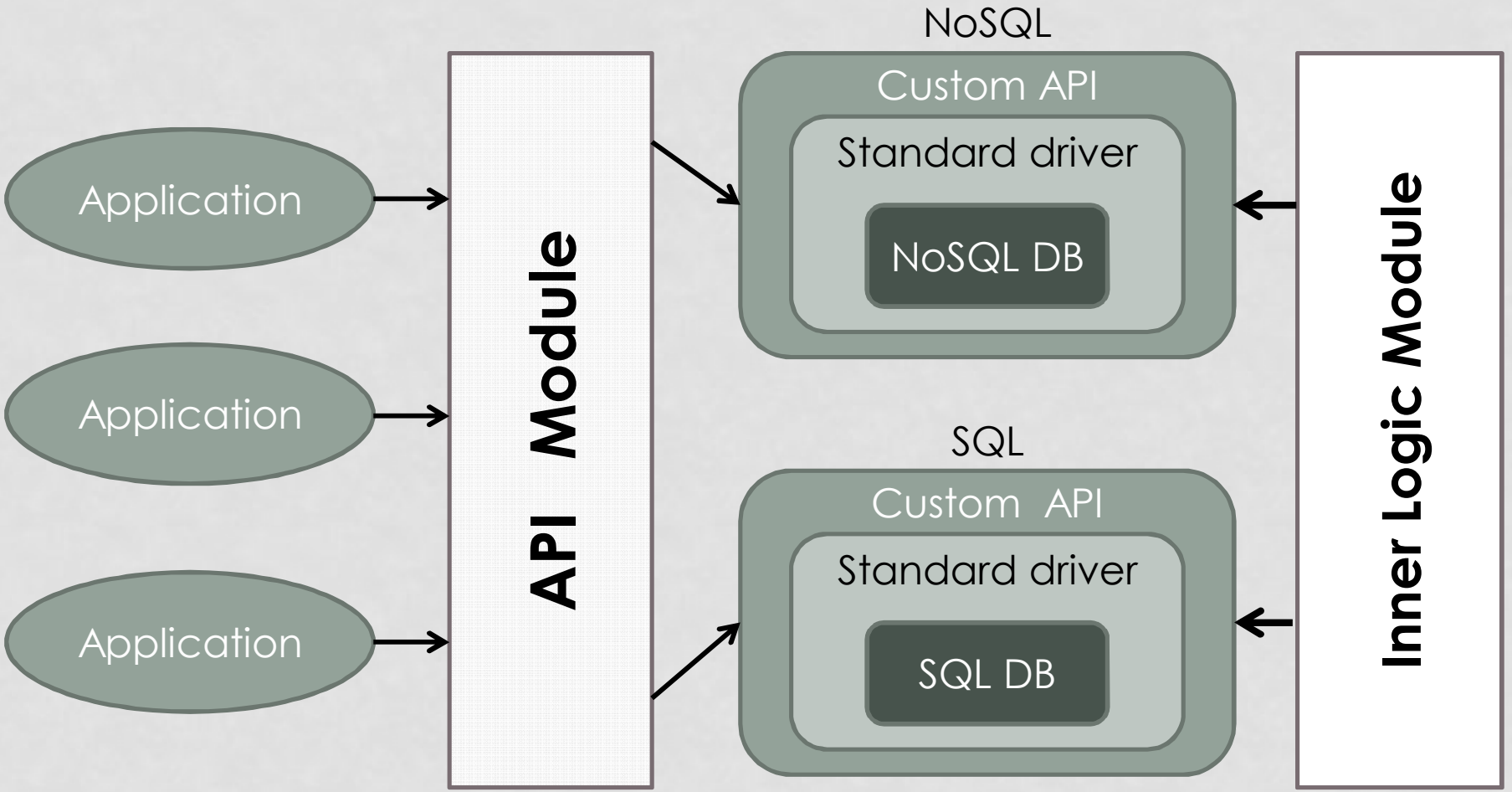
Main development topics

- Inner logic module
- Storage API

Testing

- NoSQL (Cassandra) vs. ATLAS PanDA Archive (Oracle)

HYBRID STORAGE MODEL



Part 1: INTERNAL LOGIC

INTERNAL LOGIC

Storage management inner tasks

- Synchronization: provide consistency of SQL and NoSQL parts
- Inner synchronization: provide inner data consistensy within NoSQL
- *[clean SQL from archive records]*

Synchronization main points

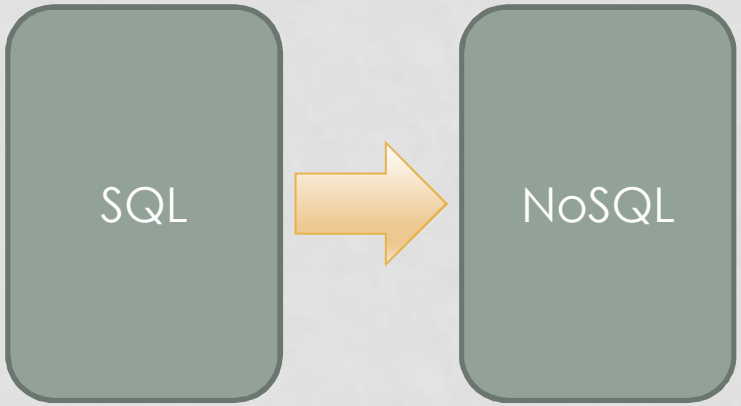
- detect unsynchronized data
- synchronize data
- do not affect the whole system performance

SYNCHRONIZATION

- ✓ Step 1. Simple copy.
  Works for tests, not good at all for real life.

- ➢ Step 2. Criteria:
  - modification time
  - not copied yet *(row count)*

- ○ Step 3. Policy:
  - when
  - how much



➢ CRITERIA

✓ Modification time

Copy data from Date1 to Date2

Date1: 2014-01-27

Date2: 2014-01-29 (today)

➢ Not copied yet?

Nobody would like to check every job, right?

So check time interval: day or N hrs

Service tables in NoSQL:

- [datetime – number of jobs]

SQL

to_date(modificationtime)	count
2015-01-26	…
✓ 2014-01-27	1096887
➢ 2014-01-28	2210772
➢ 2014-01-29	75862

NoSQL

date	count	state
2015-01-26	…	Ok
2014-01-27	1096887	Ok
2014-01-28	1023529	?
~~2014-01-29~~	~~0~~	~~?~~

POLICY

- **When?**
  Daily.

- **How much?**
  If something went wrong, we don't want to synchronize too much at a time
  - jobs limit: 5M
  - days limit: 3 days

INNER (NOSQL) SYNCHRONIZATION

- ✓ Step 1. Simple copy.
  From main table (Jobs) into dependent tables.

- ➢ Step 2. Criteria:
  - modification time
  - inner consistency broken *(count)*

- Step 3. Policy:
  - when
  - how much

➢ CRITERIA

- Modification time
  Consistency interval: day.
- Inner consistensy
  Service table in NoSQL

To be fixed →

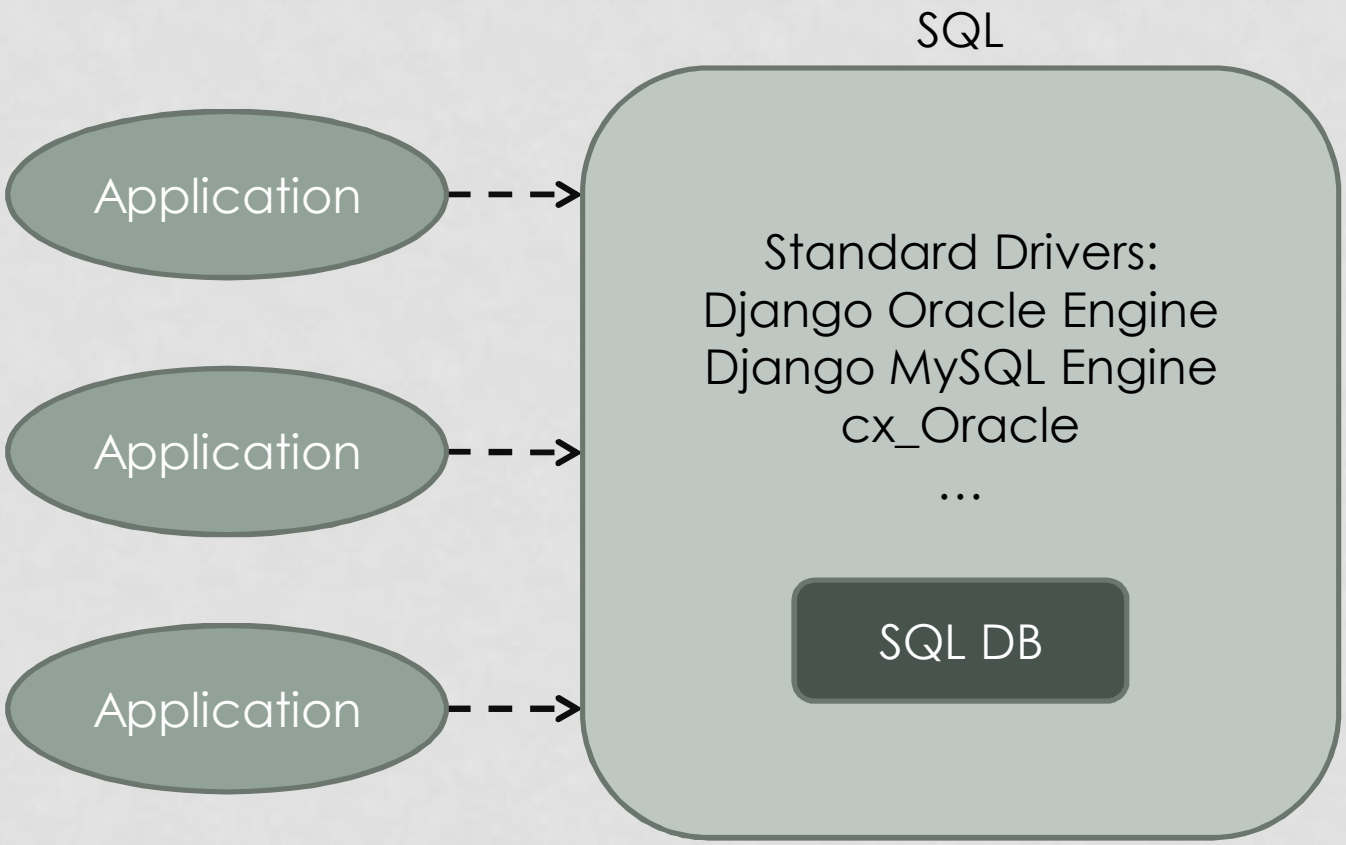
Table	Date	Key	Value	Count	State
Jobs	2015-01-27	TaskId	1516868	201	-
Task	2015-01-27	TaskId	1516868	201	Ok
Jobs	2015-01-27	TaskId	1516869	723	-
Task	2015-01-27	TaskId	1516869	512	?
Jobs	2015-01-27	Cloud	UK	692	-
Cloud	…	…	…	…	…

POLICY (inner synchronization)

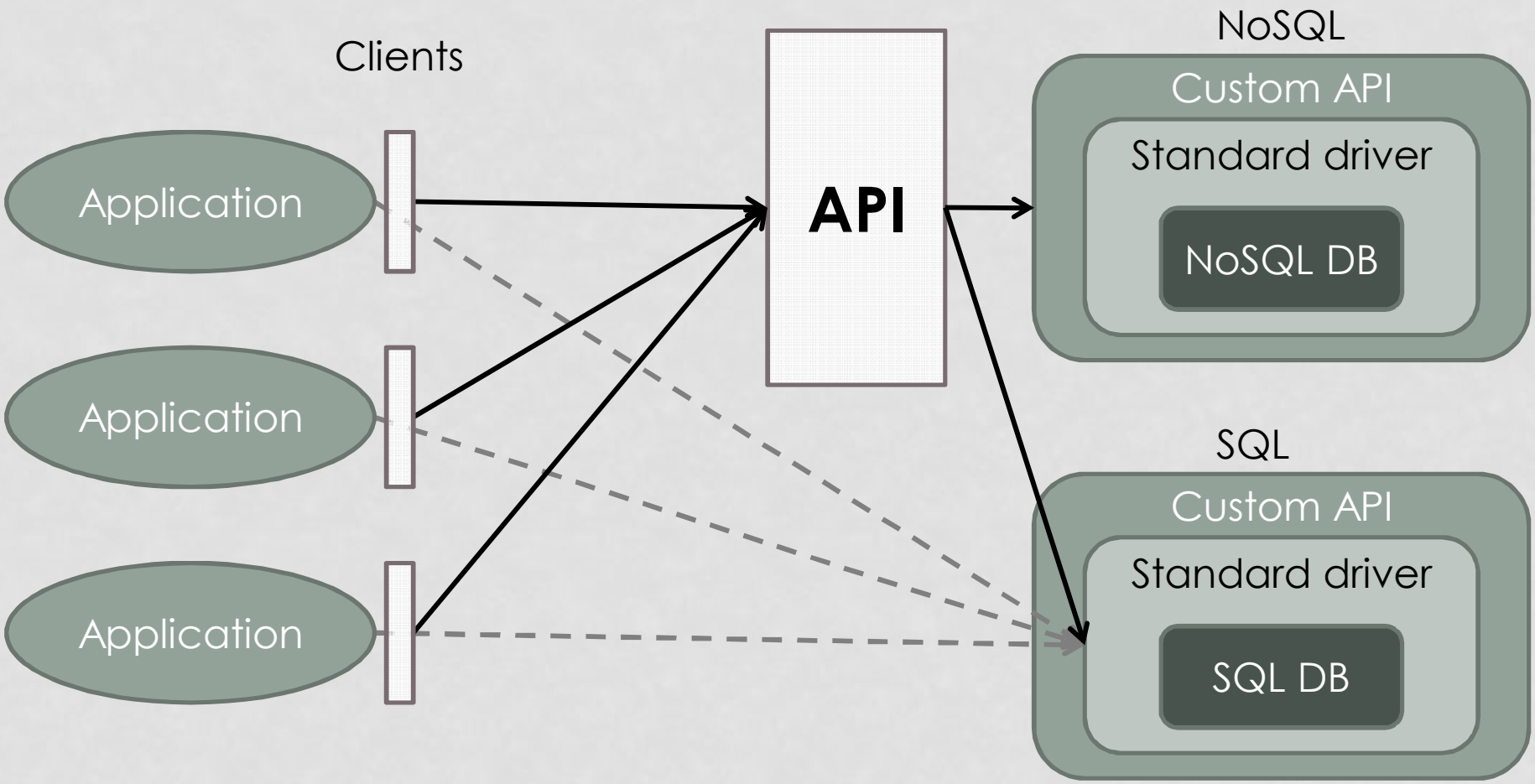
- **When?**
  - *initial synchronization: with writes into main table*
  - *service data update and check: daily*
  - *fixing synchronization: manual*
- **How much?**
  - jobs limit: 10M
  - days limit: 3 days

Part 2: STORAGE API

SQL ONLY


SQL
Application
Application
Application
Standard Drivers:
Django Oracle Engine
Django MySQL Engine
cx_Oracle
…
SQL DB

API MODULE


Clients
Application
Application
Application
API
NoSQL
Custom API
Standard driver
NoSQL DB
SQL
Custom API
Standard driver
SQL DB

PROS AND CONS

Disadvantages of unified API

- System gets less flexible
  (every request is to be implemented in advance)

Advantages

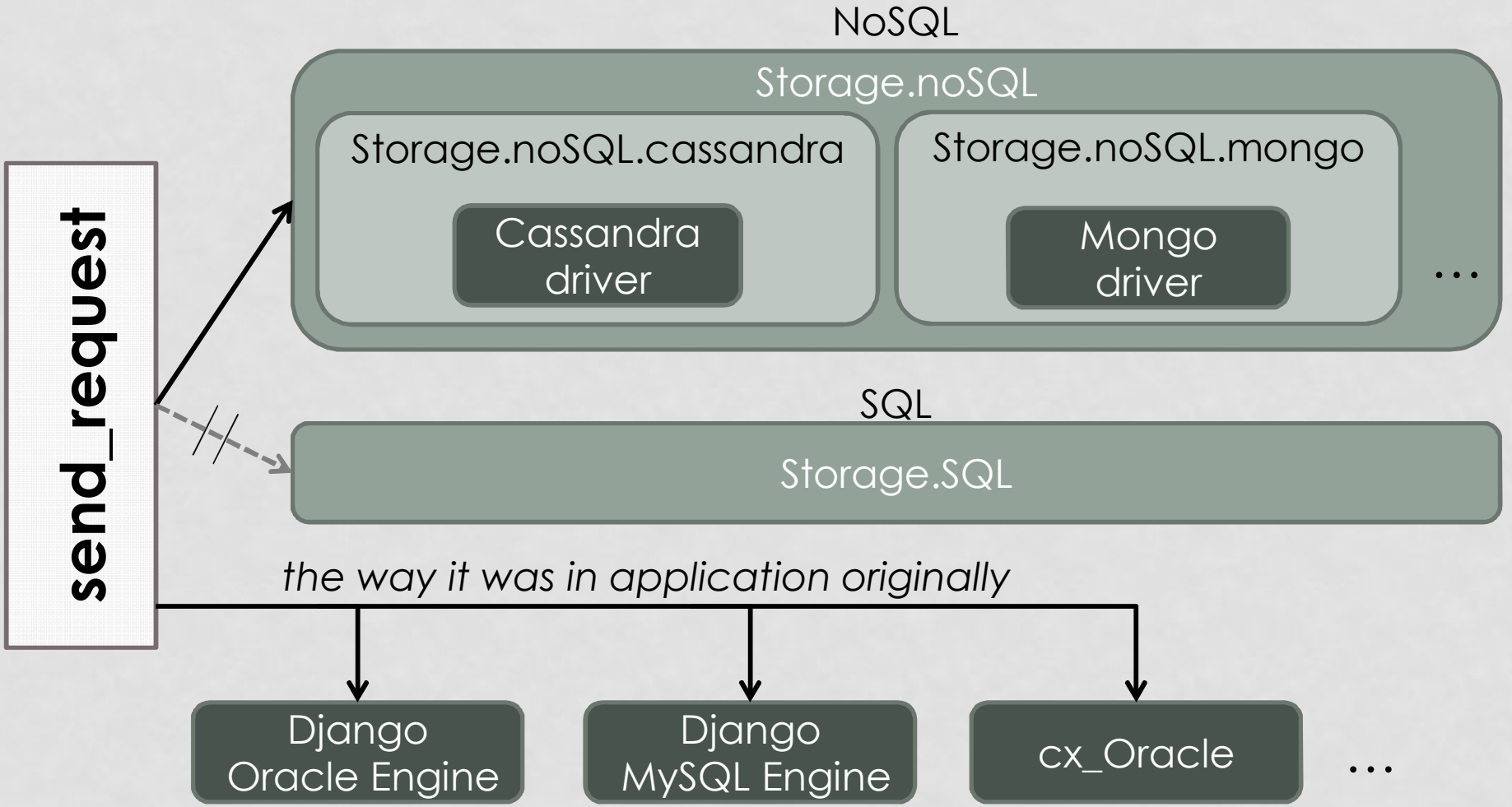
+ Isolation:
  + *no need to make **an explicit request** to every DB (or to **determine destination** DB by request parameters)*
  + ***no changes in application code** if storage was switched from one DB engine to another (e.g. from Cassandra to HBase or from Oracle to MySQL)*

API MODULE

General idea

1. Recieve requests from external applications
2. Determine location of the requested data (SQL or NoSQ)
3. Transfer the request to a proper database
4. Get a result
5. Return the result

QUERY DISTRIBUTION



Part 3: TESTS

CASSANDRA VS. ORACLE

Test	Cassandra	Oracle
Job by ***PandaID***	0,007 s	0,097 s
Jobs by ***TaskID***	0,012 s	0,063 s
Jobs by ***JediTaskID***	0,004 s	0,065 s
Jobs by ***TaskID*** with given ***JobStatus*** within given *interval of* ***ModificationTime***	0,008 s	0,069 s
Jobs by ***TaskID*** within given *interval of* ***ModificationTime***	0,023 s	0,065 s

Measured time is a **query execution time** (without data transfer);
Each request was send **100 times** with **different query parameters**;
Presented in table value is a **mean value** for these 100 requests;

Oracle: 1 year archive; partitions.
Cassandra: 2 weeks archive; request-specific tables.

THANK YOU

API: REQUESTS

storage.api.requests – determined set of functions:

- JobByID(caller, parameters, fields, SQLOnly=True)
- TaskJobList(caller, parameters, fields, SQLOnly=True)
- …

Parameters:

Name	Value
caller	"monitor" | "server" | …
parameters	{"pandaid" = <pandaid>, …}
fields	[<field1>, <field2>, …]
SQLOnly	True | False

API: ROUTING

storage.api.routing - to determine SQL or NoSQL:

- routing(caller, parameters, SQLOnly=False)

Caller:

- "server" → SQL
- "monitor" → depends on parameters

Parameters:

Name	Condition	Result
ModificationTime	< (today – 3 days)	SQL
JobStatus	not in <*FINISHED*>	SQL
…		

API: TRANSFER

storage.api.send_request

- *send_request(destination, request_type, parameters, fields, waitAll*=False)

Parameters

Name	Value
destination	storage.SQLdb | storage.noSQLdb | NULL
request_type	JobByID | …
parameters	{"pandaid" = <pandaid>, …}
fields	[<field1>, <field2>, …]
waitAll	False | True