Working with open source Greenplum distributing data to scale out

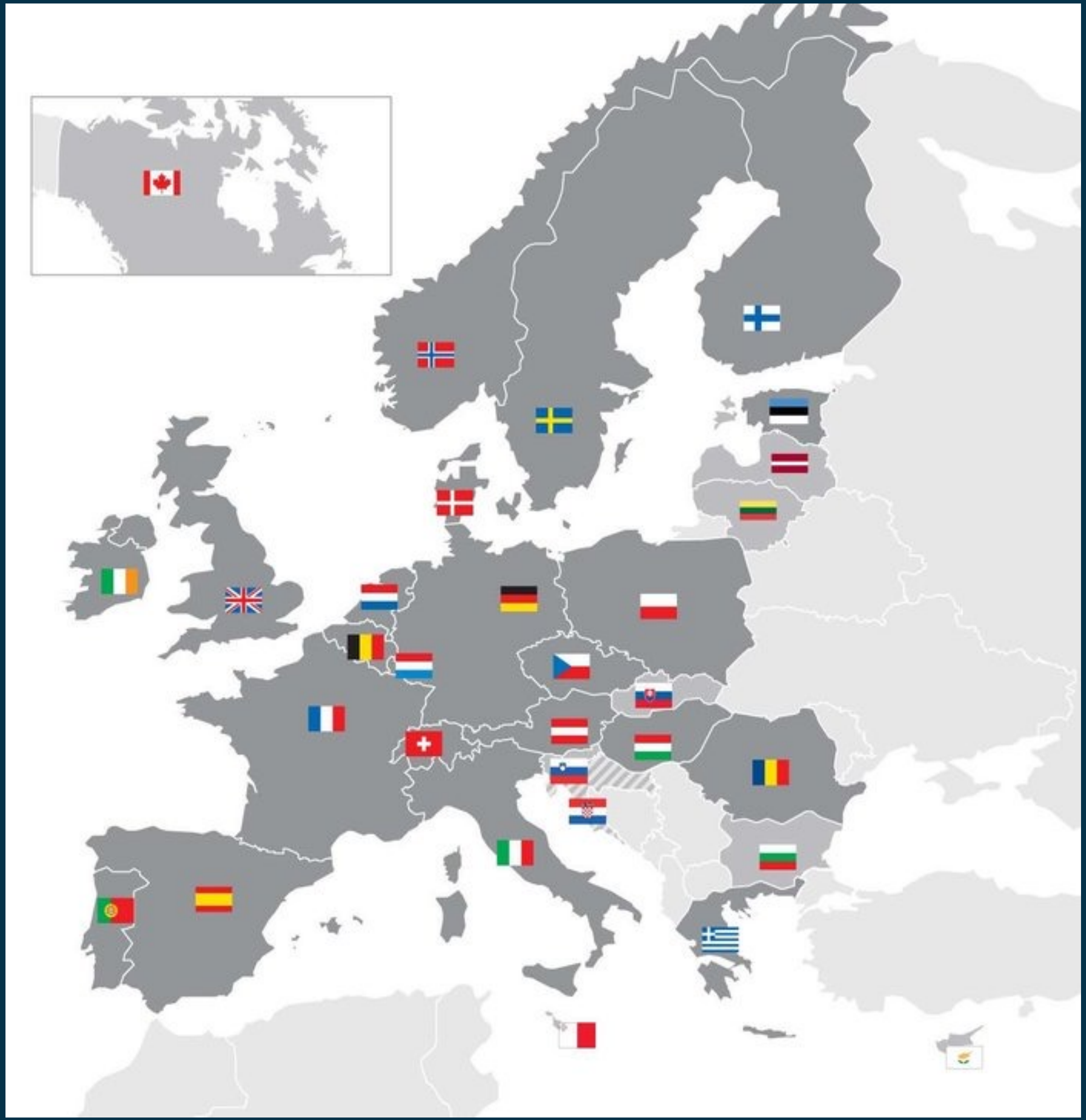
Pilar de Teodoro, Joaquim Oliveira
European Space Astronomy Centre, Madrid, Spain

pgConf'23, NYC, 04/10/2023

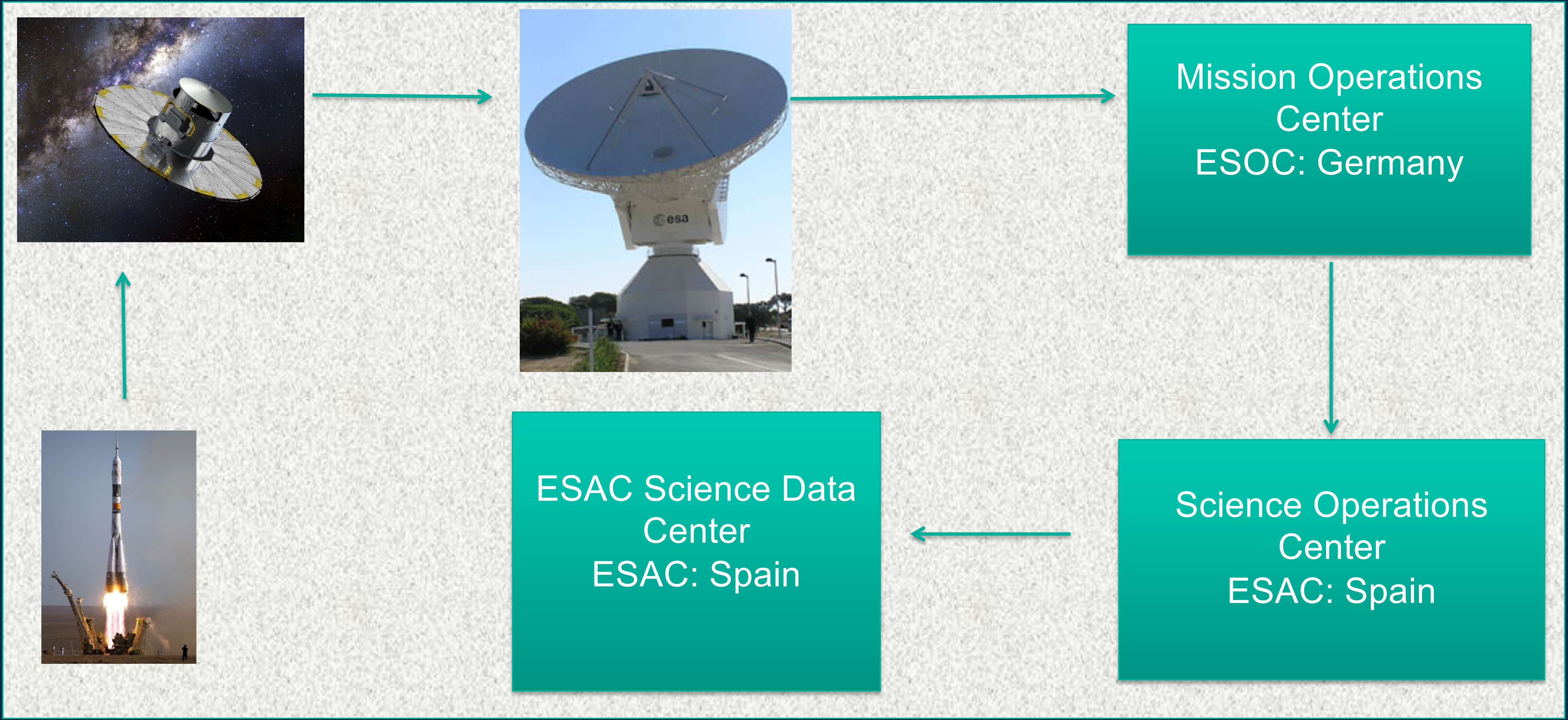
Outlines

- What is ESA and ESDC
- Why scaling out
- Tests performed
- Lessons learnt and what is missing
- Next steps

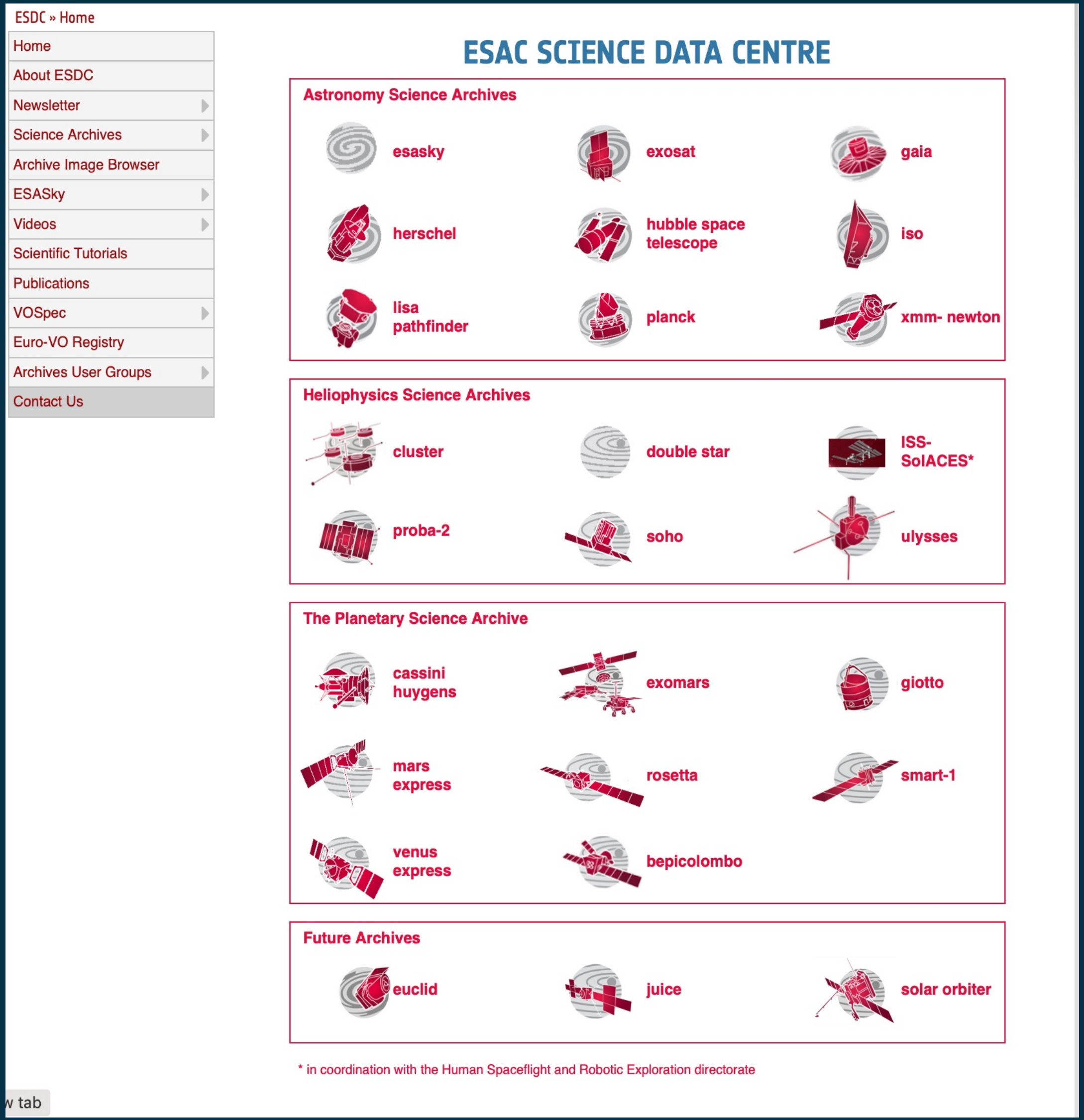
European Space Agency



Data Flow


Mission Operations Center
ESOC: Germany
Science Operations Center
ESAC: Spain
ESAC Science Data Center
ESAC: Spain

http://archives.esac.esa.int



ESDC » Home

- Home
- About ESDC
- Newsletter
- Science Archives
- Archive Image Browser
- ESASky
- Videos
- Scientific Tutorials
- Publications
- VOSpec
- Euro-VO Registry
- Archives User Groups
- Contact Us

ESAC SCIENCE DATA CENTRE

Astronomy Science Archives

esasky, exosat, gaia, herschel, hubble space telescope, iso, lisa pathfinder, planck, xmm- newton

Heliophysics Science Archives

cluster, double star, ISS-SolACES*, proba-2, soho, ulysses

The Planetary Science Archive

cassini huygens, exomars, giotto, mars express, rosetta, smart-1, venus express, bepicolombo

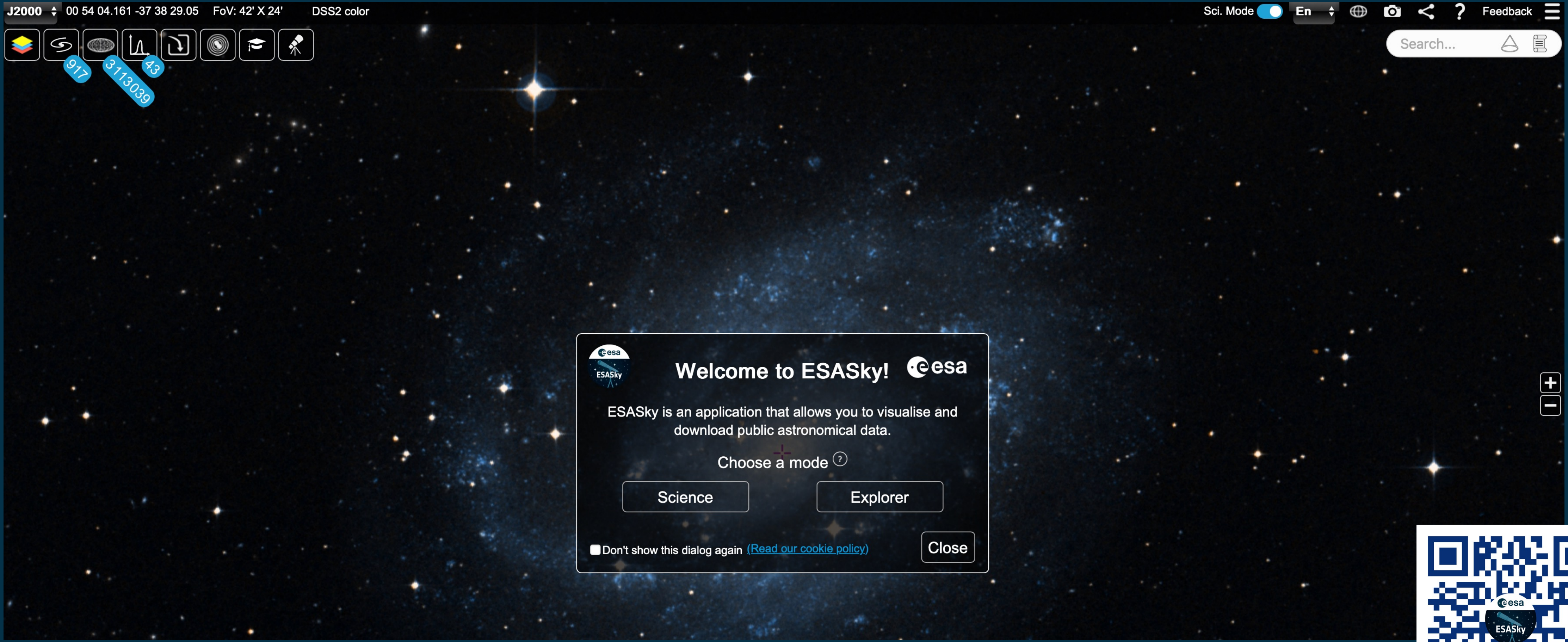
Future Archives

euclid, juice, solar orbiter

* in coordination with the Human Spaceflight and Robotic Exploration directorate

w tab

ESA SKY. http://sky.esa.int



Welcome to ESASky!

ESASky is an application that allows you to visualise and download public astronomical data.

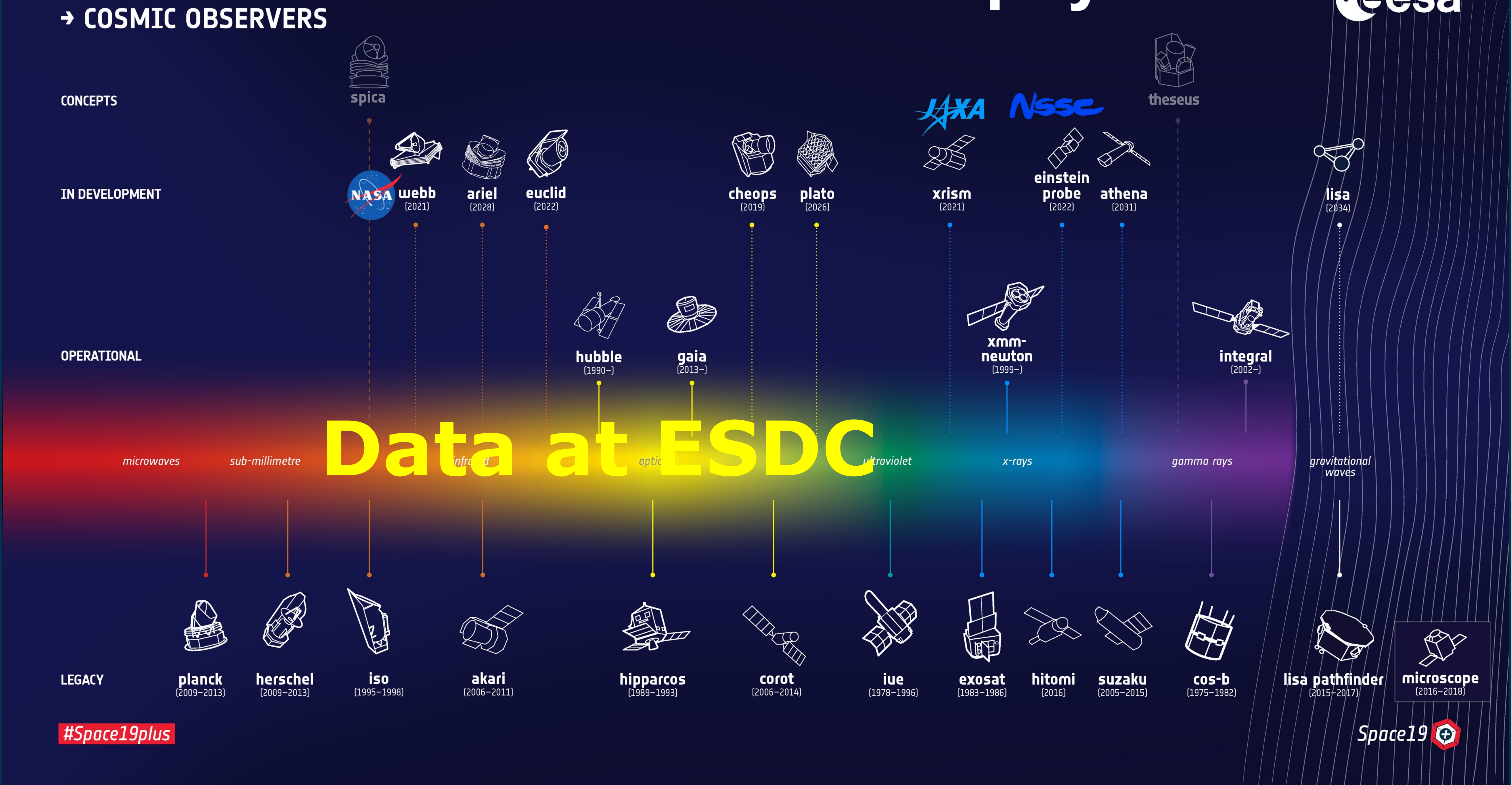
Choose a mode

Science Explorer

Don't show this dialog again (Read our cookie policy)

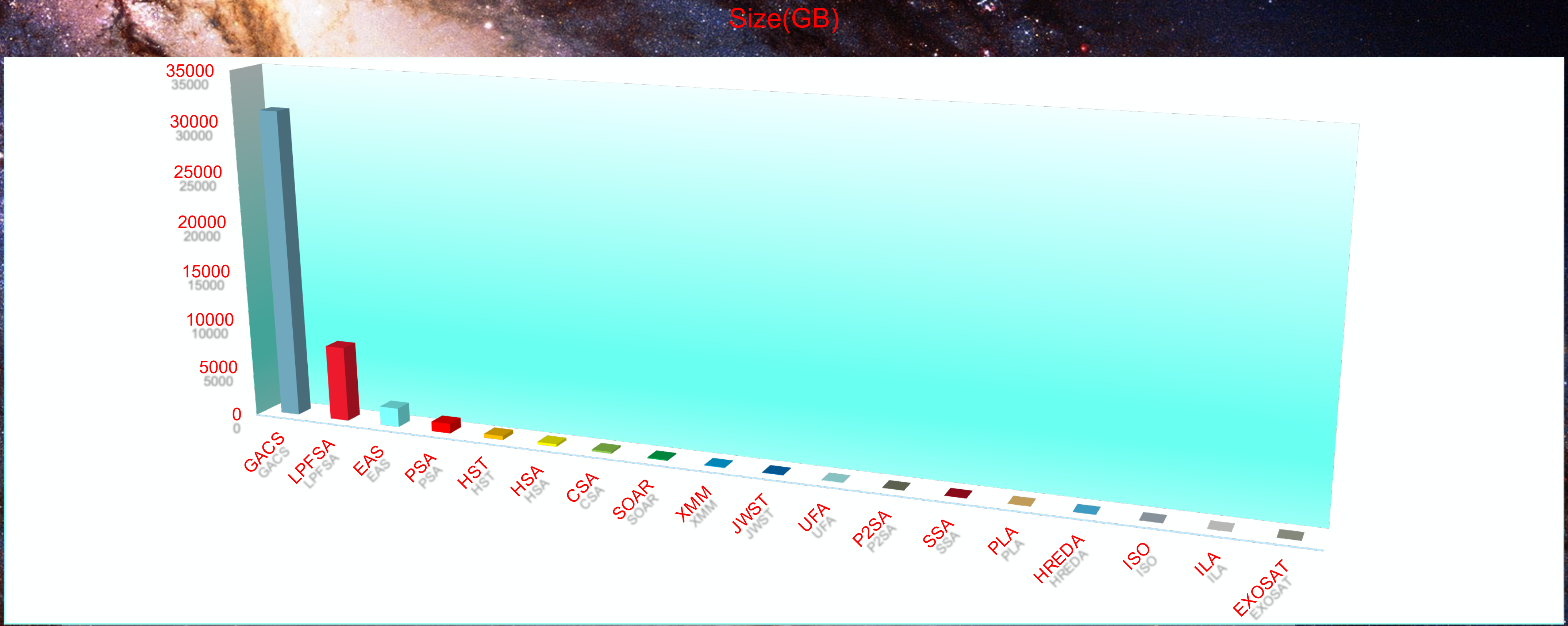
Close

The ESA Fleet for Astrophysics



Data at ESDC

Databases Size at ESDC (September 2023)



Database Systems at ESDC

Most archives: **PostgreSQL**: 9.4 to 15.3 (15 operational databases)

- Open Source
- Big community
- Spherical queries (pg_sphere, q3c, healpix, postgis)
- Professional companies support (that we do not use at our own risk)

ISO: **Sybase**

Euclid DPS: **Oracle**

ESASky, PSA (some functionalities): **ElasticSearch**

Euclid Prototype: Spark

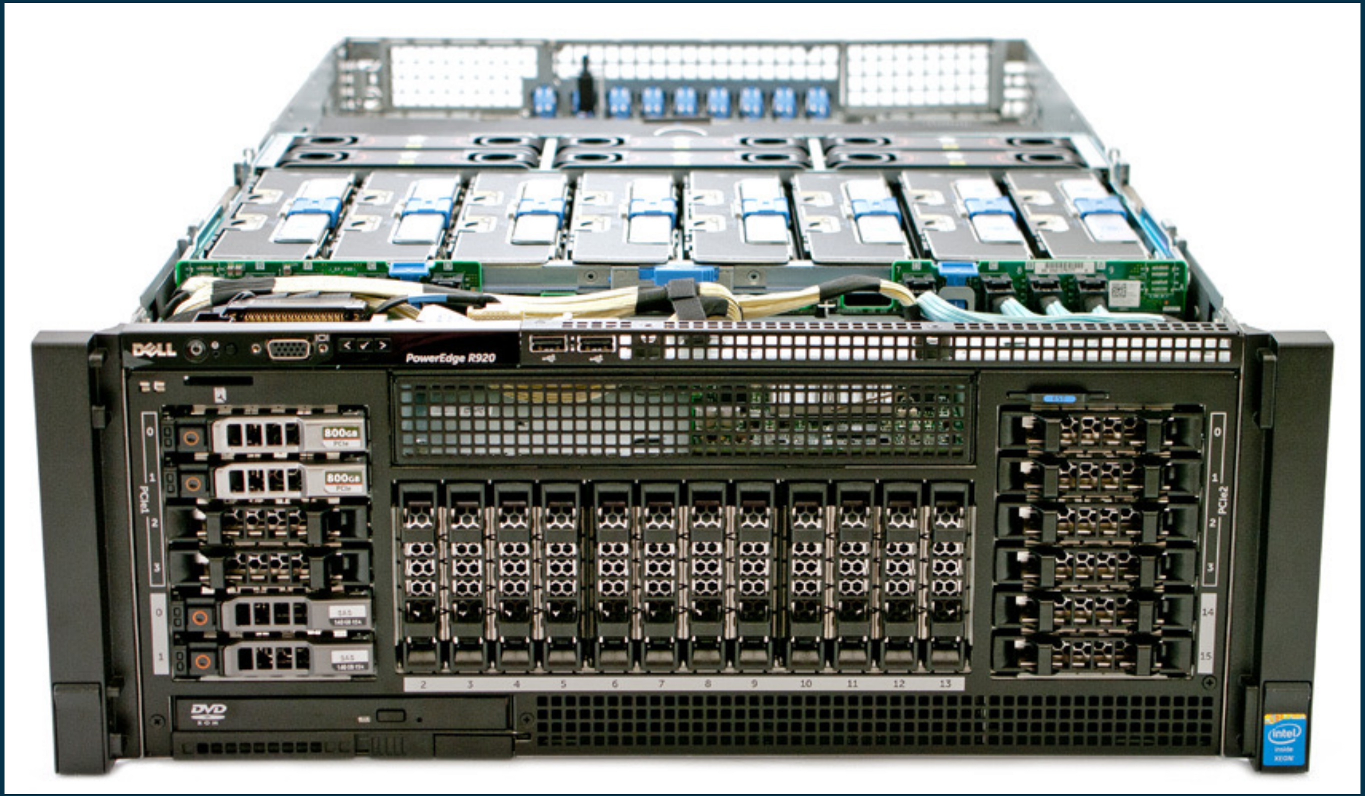
Import from Files, MySQL…

Databases HW

Chosen to scale up:

Largest (GACS):

- Dell PowerEdge R920
- 1.5TB RAM
- 96 cores
- 40 TB SSD



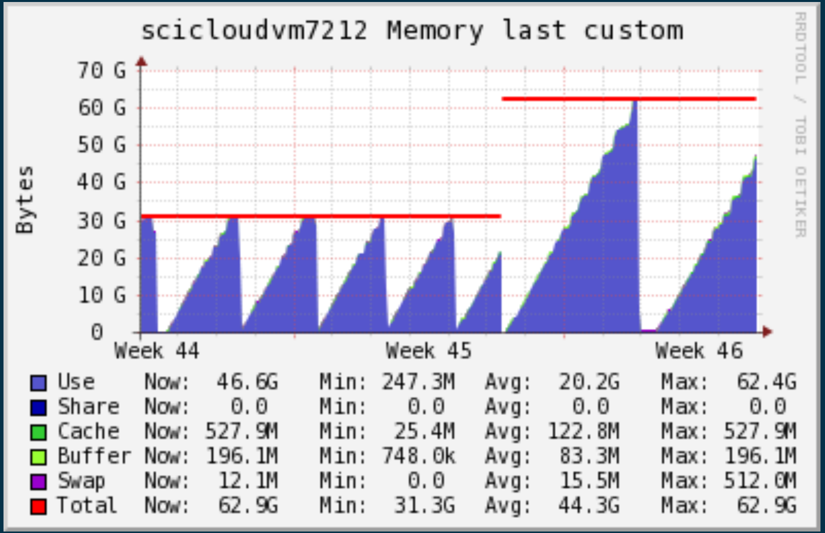
Plan for Scale-out Tests

Objective: Big Datasets queried in efficient times (Requirements). Ingestion performance.

Pre-Requisites: Resources, data to ingest (Catalogues, Relational tables)

Comparison: ingestion time, retrieving time, administration, problems arisen.

Scalability



```
🟦 Use     Now:  46.6G   Min: 247.3M   Avg:  20.2G   Max:  62.4G
🟦 Share   Now:   0.0    Min:   0.0    Avg:   0.0    Max:   0.0
🟩 Cache   Now: 527.9M   Min:  25.4M   Avg: 122.8M   Max: 527.9M
🟩 Buffer  Now: 196.1M   Min: 748.0k   Avg:  83.3M   Max: 196.1M
🟪 Swap    Now:  12.1M   Min:   0.0    Avg:  15.5M   Max: 512.0M
🟥 Total   Now:  62.9G   Min:  31.3G   Avg:  44.3G   Max:  62.9G
```

Distributed flavours tested

- Postrgres-XL
- Citus
- Greenplum

Greenplum / Postgres fast track

What is Postgres?

- PostgreSQL is a powerful, open-source object-relational database system.
- Database engine used in most of the ESDC Archives

What is Greenplum?

- Greenplum is a big data technology based on MPP architecture and the Postgres open-source database technology
- Database cluster used in Euclid (SAS) and Plato Archives with master and segments nodes (typically known as worker nodes)

What is Heap Storage?

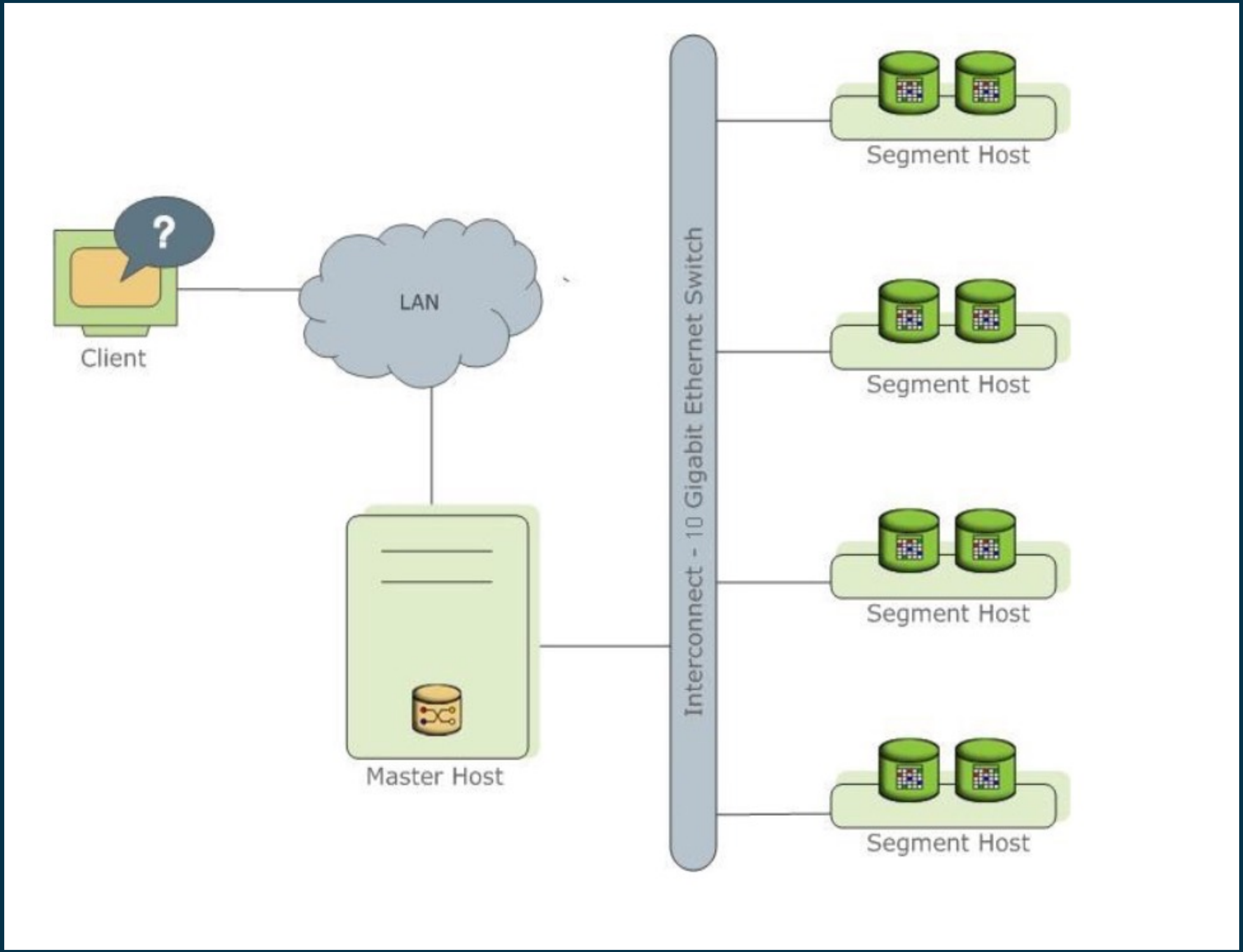
- By default, Greenplum Database uses the same heap storage model as PostgreSQL. Heap table storage works best with OLTP-type workloads where the data is often modified after it is initially loaded (inserting, deleting, or updating database data).

What is Column oriented storage?

- Data is stored in columns (rather than rows), good for data warehouse workloads, not often changed, allows more compression options and allows better performance on tables with many columns.

Sources: wikipedia and https://docs.vmware.com/en/VMware-Greenplum/6/

Greenplum Architecture



Greenplum Testing

- Ingestion
- Expansion
- Performance
- High Availability
- Resource Management
- Community edition vs Commercial edition

Greenplum Testing

Community edition does not include:

- Support from VMware Greenplum team
- Product packaging and installation script.
- Support for QuickLZ compression. QuickLZ compression is not provided in the open-source version of Greenplum Database due to licensing restrictions.
- Support for managing Greenplum Database using VMware Greenplum Command Center.
- Support for full text search and text analysis using VMware Greenplum Text.
- Support for data connectors:
    - Greenplum-Spark Connector
    - Greenplum-Informatica Connector
    - Greenplum-Kafka Integration
    - Gemfire-Greenplum Connector
- Data Direct ODBC/JDBC Drivers
- gpcopy utility for copying or migrating objects between Greenplum systems.

GP Testing: Ingestion

gpfdist: *nohup **gpfdist** -d /spv02/ -p 4000 > **gpfdist**_4000.log 2>&1 &*

Create external table flag_ext()

location ('gpfdist://172.25.4.244:4000/') FORMAT 'CSV';*

273*10GB csv files in 3 hours (NetApp) – 6 gpfdist servers

Disk write min bandwidth: 179,69(MB/s) [netapp server]

In Dell Limerick cluster: 1,2GB/s [local SSD]

The gpfdist protocol is used in a CREATE EXTERNAL TABLE SQL command to access external data served by the Greenplum Database gpfdist file server utility. When external data is served by gpfdist, all segments in the Greenplum Database system can read or write external table data in parallel.

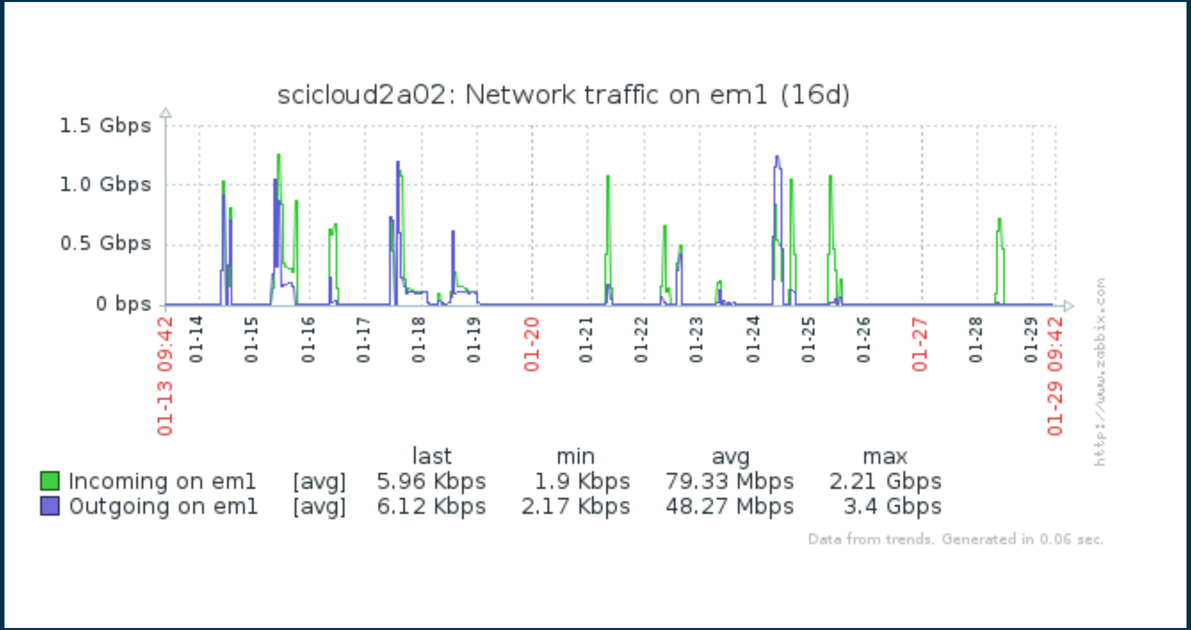
From: https://docs.vmware.com/en/VMware-Greenplum/7/greenplum-database/admin_guide-external-g-using-the-greenplum-parallel-file-server--gpfdist-.html

GP Testing: Expansion

GPExpand: Expanding the cluster to new datanodes or same ones increasing number of segments.

In general, adding nodes to a Greenplum cluster achieves a linear scaling of performance and storage capacity.

The expansion process uses standard Greenplum Database operations so it is transparent and easy for administrators to troubleshoot. Segment mirroring and any replication mechanisms in place remain active, so fault-tolerance is uncompromised and disaster recovery measures remain effective.



		last	min	avg	max
🟩 Incoming on em1	[avg]	5.96 Kbps	1.9 Kbps	79.33 Mbps	2.21 Gbps
🟦 Outgoing on em1	[avg]	6.12 Kbps	2.17 Kbps	48.27 Mbps	3.4 Gbps

GP Testing: First Performance Tests

Performance:

•Gaia query profiling tests: Catalogues XMatch

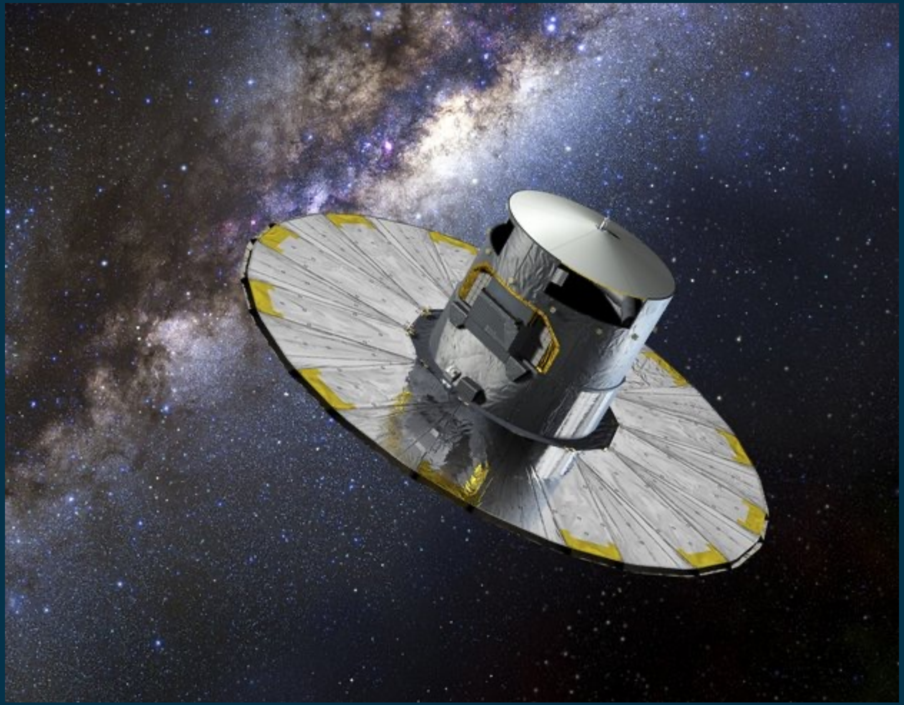
•Euclid query profiling tests: 1,3TB catalogue

What to test:

•Table formats

•Optimizers

•Scalability



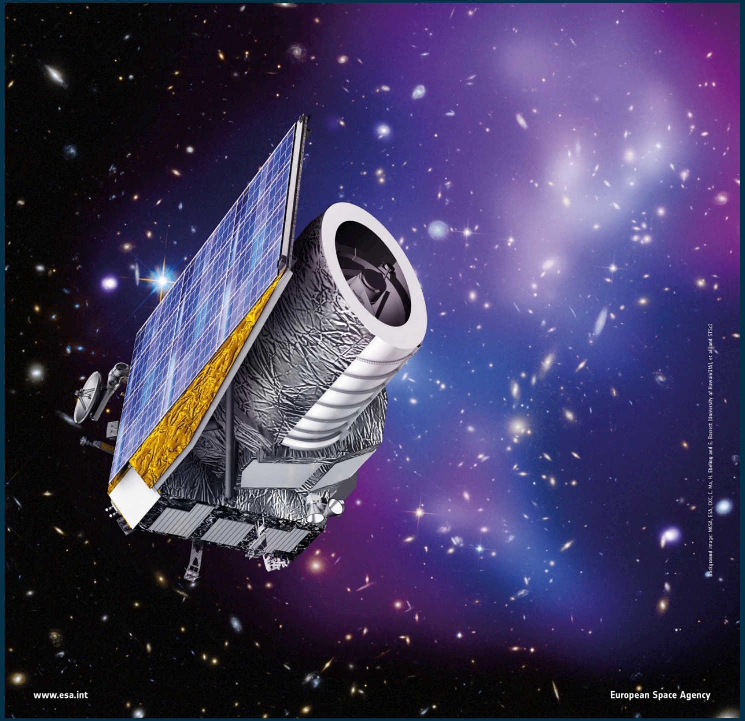


Table formats

- Heap table storage works best with OLTP-type workloads where the data is often modified afterit is initially loaded. By default, Greenplum Database uses the same heap storage model as PostgreSQL. Append-
- Optimized table storage works best with de-normalized fact tables in a data warehouse environment. Fact tables are usually loaded in batches and accessed by read-only queries. Compression is available for AO tables.
- Row oriented AO tables support ZLIB (levels 1-9) compression.
- Column oriented AO tables support the same, plus compression per column including Run-Length Encoding.

Optimizers

Optimizer: Pivotal Optimizer (GPORCA)

Optimizer: Postgres query optimizer

Enabling GPORCA for a System:

Set the server configuration parameter optimizer for the Greenplum Database system.

1. Log into the Greenplum Database coordinator host as gpadmin, the Greenplum Database administrator.
2. Set the values of the server configuration parameters. These Greenplum Database gpconfig utility commands sets the value of the parameters to on:
3. $ gpconfig -c optimizer -v on --coordinatoronly
4. Restart Greenplum Database. This Greenplum Database gpstop utility command reloads the postgresql.conf files of the coordinator and segments without shutting down Greenplum Database.
5. gpstop -u

Enabling GPORCA for a Database

Set the server configuration parameter optimizer for individual Greenplum databases with the ALTER DATABASE command. For example, this command enables GPORCA for the database *test_db*.

```
> ALTER DATABASE test_db SET OPTIMIZER = ON ;
```

Enabling GPORCA for a Session or a Query

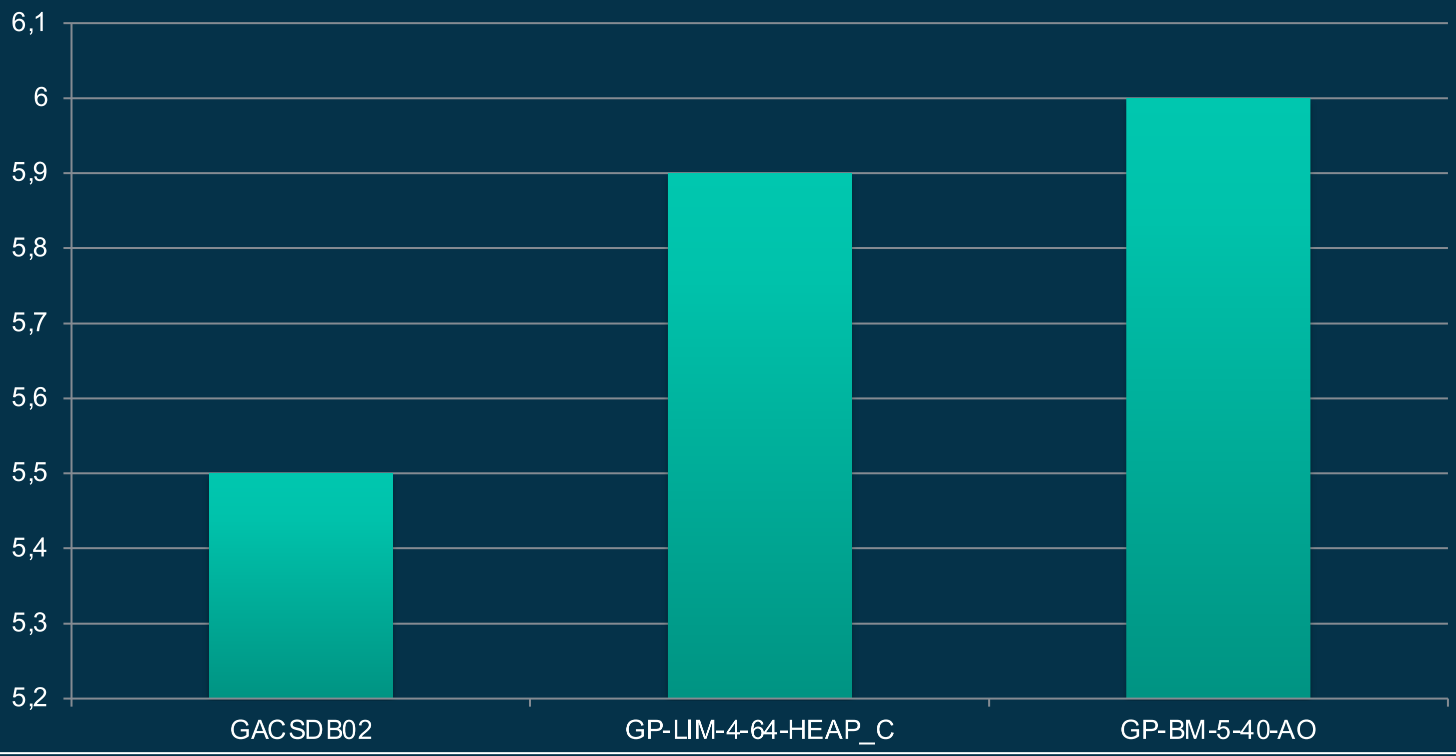
You can use the SET command to set optimizer server configuration parameter for a session. For example, after you use the psql utility to connect to Greenplum Database, this SET command enables GPORCA:

```
> set optimizer = on ;
```

GP testing: Xmatch Performance

Most relevant queries: Gaia query

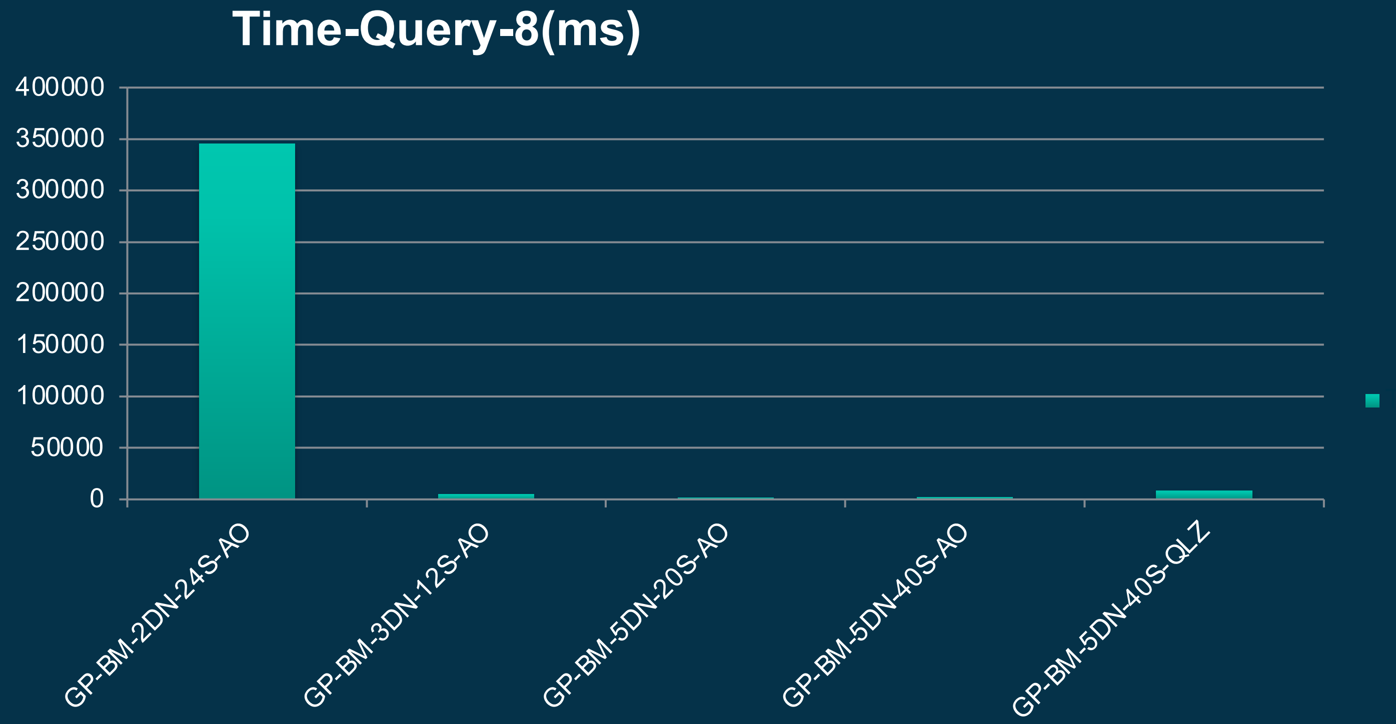
```
SELECT gdr1.source_id AS iddr1, gdr2.source_id AS iddr2
FROM gaiadr1.gaia_source AS gdr1, gaiadr2.gaia_source AS gdr2
WHERE q3c_join(gdr2.ra, gdr2.dec, gdr1.ra, gdr1.dec, 0.5/3600)
```



GP Testing: Cone Search Performance

Euclid query:

```
SELECT q3c_dist(euclid.flag."ra_gal",euclid.flag."dec_gal",30,30)*3600 AS dist , euclid.flag."ra_gal" AS ra_gal , euclid.flag."dec_gal" AS dec_gal , euclid.flag."euclid_vis" AS euclid_vis , euclid.flag."euclid_nisp_y" AS euclid_nisp_y , euclid.flag."euclid_nisp_j" AS euclid_nisp_j , euclid.flag."euclid_nisp_h" AS euclid_nisp_h , euclid.flag."observed_redshift_gal" AS observed_redshift_gal
        FROM euclid.flag
        WHERE (q3c_join(30,30,euclid.flag."ra_gal",euclid.flag."dec_gal",0.01677)) = '1'
        ORDER BY dest ASC;
```



GP Testing: Performance

Select count(*) from gaiadr2.gaia_source: 1692919135 rows

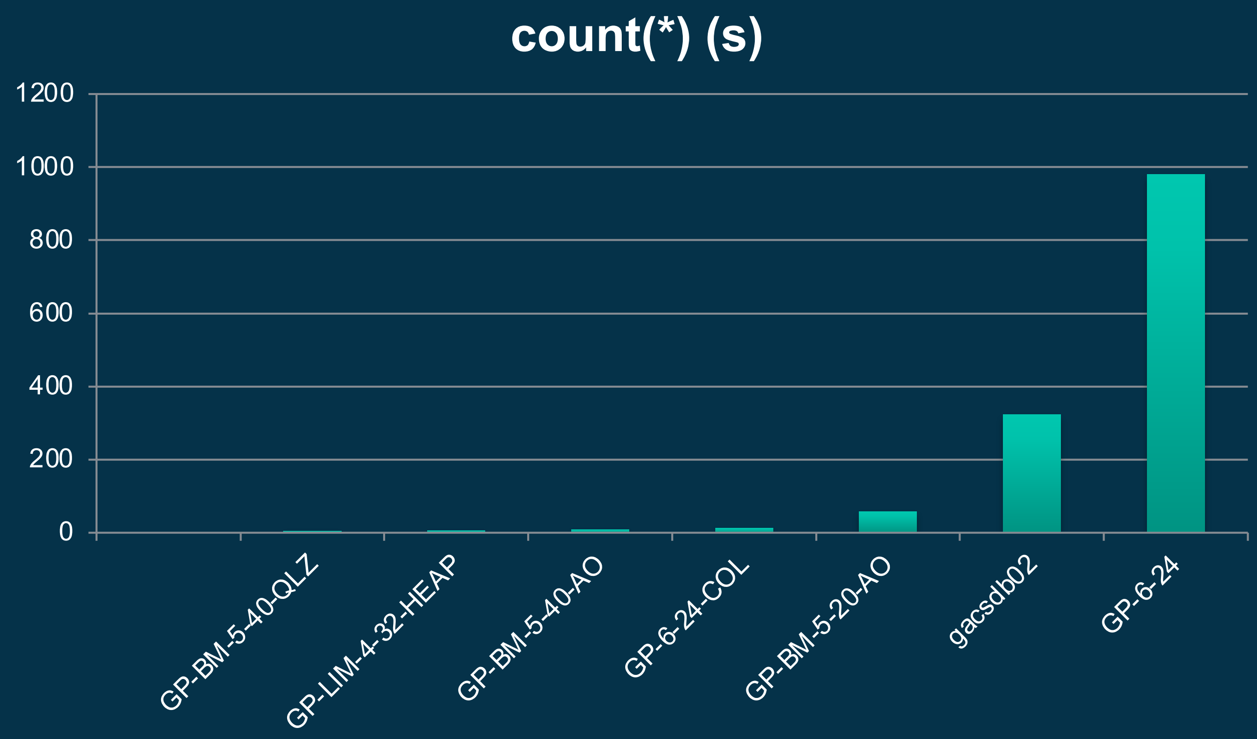
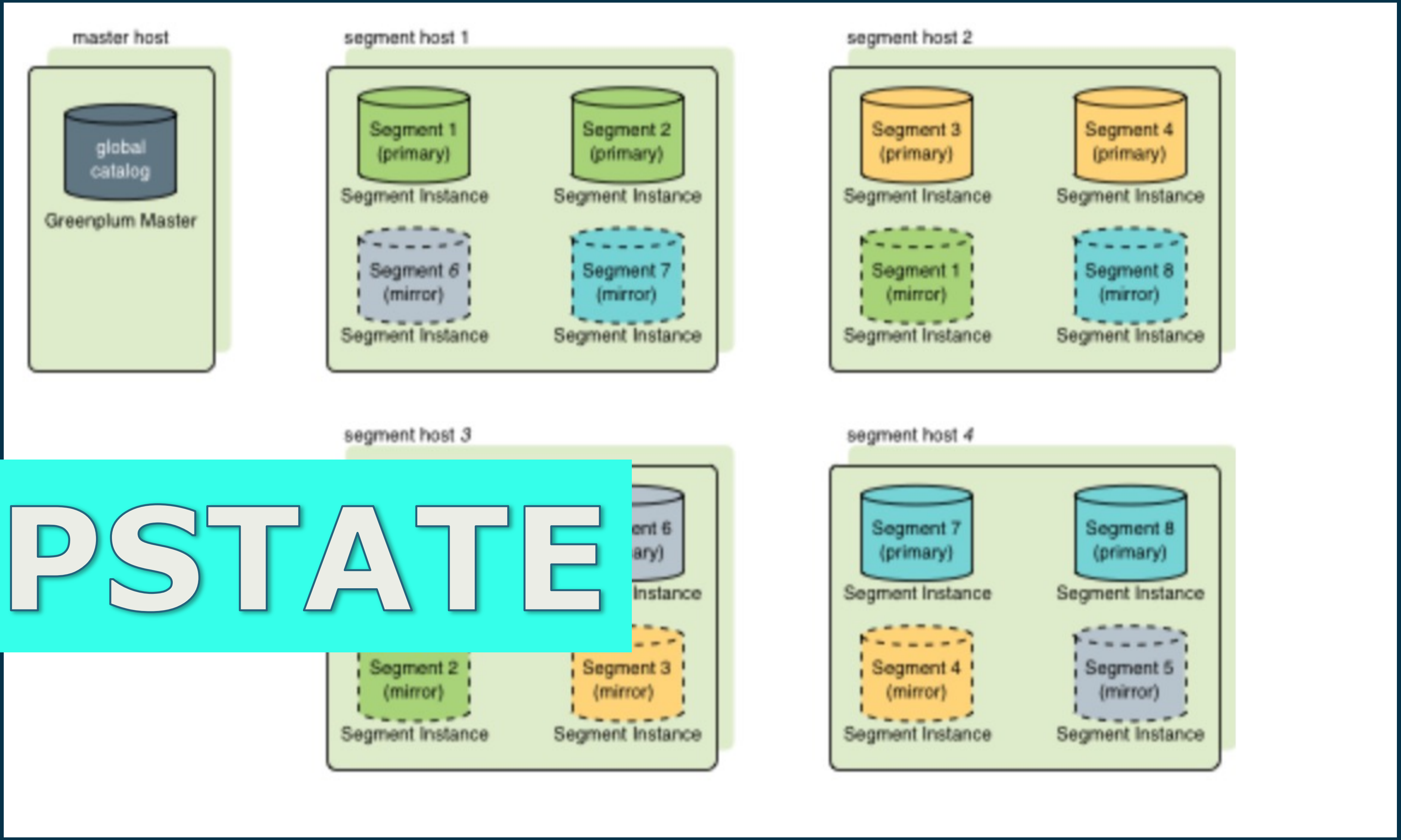


Table format	Time(s)
GP-BM-5-40-QLZ	6
GP-LIM-4-32-HEAP	7
GP-BM-5-40-AO	10
GP-6-24-COL	14
GP-BM-5-20-AO	59
gacsdb02	323
GP-6-24	980

GP Testing: High Availability



GPSTATE

GP Testing: Managing Resources

- Resource groups: set and enforce CPU, memory, and concurrent transaction limits in Greenplum Database. Use Linux Control Groups (cgroups) to manage CPU resources.
- Resource queues: manage the degree of concurrency in a Greenplum Database system.
    - manage the number of active queries that may execute concurrently,
    - the amount of memory each type of query is allocated,
    - priority of queries.

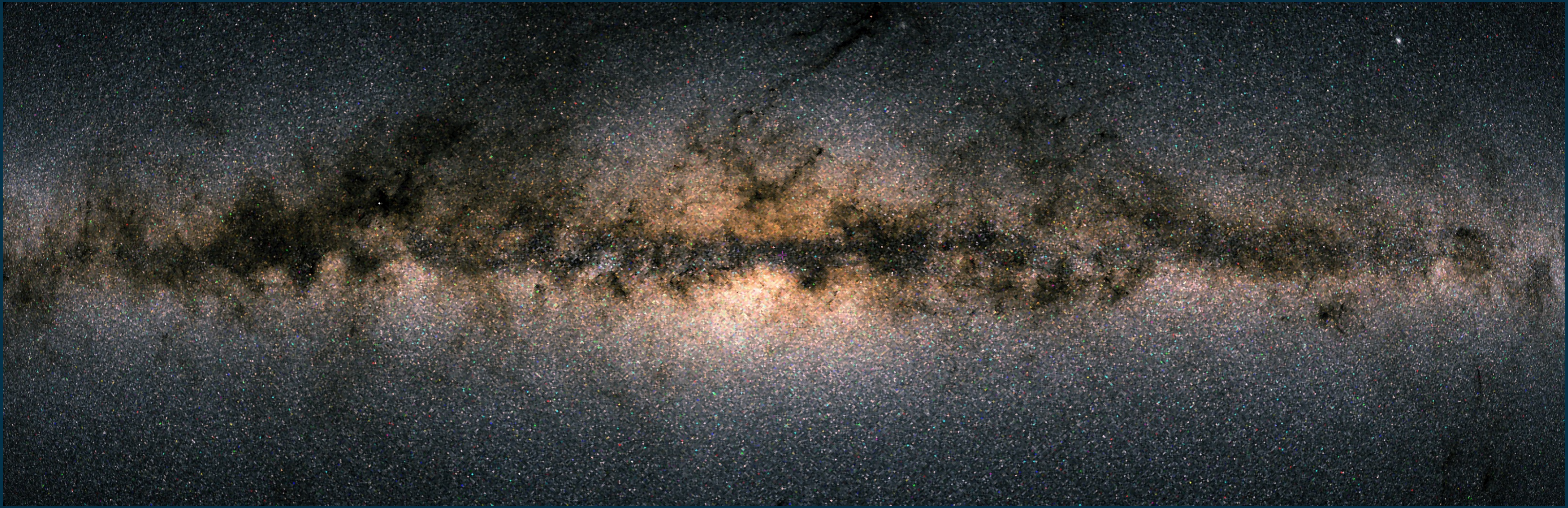
GP Testing: Resource management

```
esdc=#  CREATE RESOURCE QUEUE adhoc WITH (ACTIVE_STATEMENTS=3);
CREATE QUEUE
esdc=# ALTER RESOURCE QUEUE adhoc WITH (PRIORITY=LOW);
ALTER QUEUE
```

Current Testing – GACS

Greenplum 6 cluster

Gaia Data Release 3 catalogue on 13 June 2022



Current testing

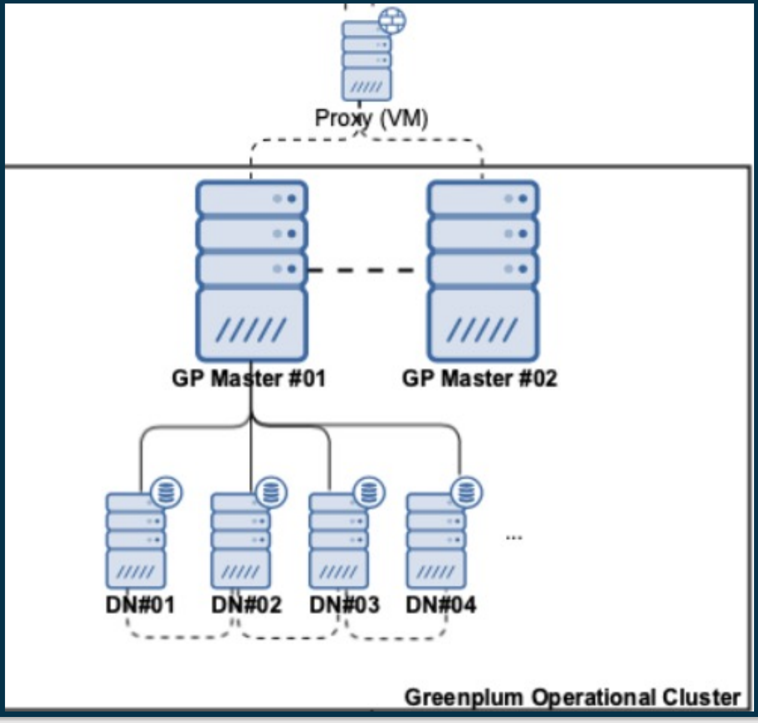
- New hardware

- Test with real data, no synthetic data

- Test real use cases

- Learn from working with the data
    - Table format options
    - Query plan optimizers
    - Indexes
    - Data distribution
    - …

HW requested



Component	Mem	CPU	Storage (Local)	Storage (Shared)
		Cores: 40	SSD: 12 x 4TB	HDD: TBD
		Sockets: 4	Optimized: mixed used (**12Gbps**)	For user DB quota
			48000 IOPS (32KB)	
Greenplum Operational Cluster- Data Nodes	512 GB (ea)			
		Cores: 40	SSD: 3 x 1.92TB	HDD: TBD
		Sockets: 4	Optimized: combined use (**6Gbps**)	For user DB quota
			24000 IOPS (32KB)	
Greenplum Operational Cluster- Master Nodes	256 GB (ea)			

Reference tables

Table name	# columns	Heap size (GB)	Columnar size with compression (GB)	Compression %
gaiadr3.gaia_source_lite	51	342	180	48%
gaiadr3.gaia_source	152	877	509	42%
user_dr3int6.gaia_source	243	1230	751	39%

```
Distributed by: (source_id)
Options: appendonly=true, orientation=column, compresstype=zlib,
compresslevel=7
```

#rows: 1.811.709.771

GP Testing: count(*)

Query	# columns	Postgres Heap format	Greenplum Heap format	Greenplum Heap format GPORCA	Greenplum Columnar format	Greenplum Columnar format GPORCA
Q1	51	77,9	22,3	21,7	15,9	15,6
Q1	152	63,6	40,9	42,3	11,9	11,9
Q1	243	64,4	55,2	58,5	11,9	12
Q2	51	75,8	45,9	26,9	7,1	11,4
Q2	152	68,3	127,6	66,7	9,5	11,6
Q2	243	82,9	93,9	84,8	9,8	14,1
Q3	51	87,4	39,1	23,2	9,9	15,2
Q3	152	73,3	94,6	53,7	11,4	15,5
Q3	243	192,7	62,7	67	13	15,3

Execution times in seconds

```
Q1=# SELECT COUNT(*) FROM <table>;
     count
---------------
 1.811.709.771
(1 row)
```

```
Q2=# SELECT COUNT(*)
Q2=# FROM <table>
Q2=# WHERE phot_g_mean_mag <= 18.25
Q2=# AND has_mcmc_msc = 'true';
    count
-------------
 348.630.727
(1 row)
```

```
Q3=# SELECT COUNT(*)
Q3=# FROM <table>
Q3=# WHERE parallax IS NOT NULL
Q3=# AND phot_g_mean_mag <= 18.25;
    count
-------------
 353.992.031
(1 row)
```

GP Testing: selected columns

Query	# columns	selected columns	Postgres Heap format	Greenplum Heap format	Greenplum Heap format GPORCA	Greenplum Columnar format	Greenplum Columnar format GPORCA
Q1	51	*	347,5	500,2	507,3	466,2	468,7
Q1	152	*	1.141,20	881,9	883,3	800,7	806,5
Q1	243	*	918,2	1153,1	1.138,50	1247,1	1.244,50
Q2	51	*	44,4	11,5	8,5	85,1	85,5
Q2	152	*	238,9	17,9	22	244,8	244
Q2	243	*	365,8	93,9	76,4	370,1	373
Q2	51	7	12,6	7,8	1,2	25,8	25,8
Q2	152	7	82,4	12,4	2,1	26,9	26
Q2	243	7	13,2	80,8	63,9	25,9	25,9
Q3	51	*	656,6	110,6	105,2	150,4	150,7
Q3	152	*	3.398,90	208,4	194,1	435,4	462,7
Q3	243	*	8.896,20	246,4	248,3	704,8	753,5
Q3	51	7	669,3	71	64,6	69,7	69,9
Q3	152	7	1.441,40	101,6	70,4	67	73,2
Q3	243	7	5.536,40	69	70,4	66,4	70,7

Execution times in seconds

```
Q1=# SELECT * FROM <table>;
...
(1.811.709.771 rows)
```

```
Q2=# SELECT <select>
Q2=# FROM <table>
Q2=# WHERE phot_g_mean_mag <= 18.25
Q2=# AND has_mcmc_msc = 'true';
...
(348.630.727 rows)
```

```
Q3=# SELECT <select>
Q3=# FROM <table>
Q3=# WHERE parallax IS NOT NULL
Q3=# AND phot_g_mean_mag <= 18.25;
...
(353.992.031 rows)
```

GP Testing: Q3C[1] indexing

# columns	angle	Postgres Heap format	Greenplum Heap format	Greenplum Columnar format
51	0,5	308,8	0,2	0,2
51	1	306,6	1,9	1,9
51	2,5	470,7	5,8	4,8
152	0,5	12,6	7,5	0,2
152	1	498,1	8,6	9,8
152	2,5	648,2	14	13,2
243	0,5	12,1	90,3	0,2
243	1	529,7	85,7	9,8
243	2,5	701	86,3	13,2

Query optimizer will be chosen by Greenplum
Execution times in seconds

```
Q=# SELECT ...
Q=# FROM <table>
Q=# WHERE '1' = (q3c_join(270.89,-30.03,ra,dec,0.5))
Q=# AND ruwe < 1.4;
...
(358.976 rows)
```

```
Q=# SELECT ...
Q=# FROM <table>
Q=# WHERE '1' = (q3c_join(270.89,-30.03,ra,dec,1))
Q=# AND ruwe < 1.4;
...
(1.514.361 rows)
```

```
Q=# SELECT ...
Q=# FROM <table>
Q=# WHERE '1' = (q3c_join(270.89,-30.03,ra,dec,2.5))
Q=# AND ruwe < 1.4;
...
(10.125.050 rows)
```

[1] Q3C - PostgreSQL extension for spatial indexing on a sphere.

GP Testing: search in external tables

Using *gpfdist,* serve data files as external tables

```
Q=# SELECT source_id
Q=# FROM <table>
Q=# WHERE source_id = ${a_source_id};
```

Included file names	Execution time (s)
N/A (regular query to database table)	0,02
GaiaSource_0*	741,6
GaiaSource_0*, GaiaSource_1*	1.103,8
GaiaSource_0*, GaiaSource_1*, GaiaSource_0*	2.915,0
GaiaSource_0*, GaiaSource_1*, GaiaSource_2*, GaiaSource_3*	3.556,2
GaiaSource_0*, GaiaSource_1*, GaiaSource_2*, GaiaSource_3*, GaiaSource_4*	7.805,0
GaiaSource_0*, GaiaSource_1*, GaiaSource_2*, GaiaSource_3*, GaiaSource_4*, GaiaSource_5*	…
GaiaSource_0*, GaiaSource_1*, GaiaSource_2*, GaiaSource_3*, GaiaSource_4*, GaiaSource_5*, GaiaSource_6*	…
GaiaSource_0*, GaiaSource_1*, GaiaSource_2*, GaiaSource_3*, GaiaSource_4*, GaiaSource_5*, GaiaSource_6*, GaiaSource_7*	…
…	…

Things not tested

- Index compression
- More fine tunning of the tables

Benchmark conclusions

- Greenplum shows an overall improvement over Postgres single-instance

- The number of columns shouldn't be determinant for the performance of the queries
  - We are using column format to store the data; tables can have as many columns as determined
  - Selecting more columns has more impact thought (more data is transferred)
  - Columnar format shows consistent results across different table sizes, while Heap format shows promising results in some conditions.
- Q3c indexing:
  - The number of columns is not completely determinant for the performance of the q3c queries.
  - Always uses the legacy optimizer, never uses the GPORCA optimizer.
- External tables: Using external tables to search for a source is not an option with csv files. There is no index, and the files are large. The system needs to open all files to check where the row is.
- Best solution will be to have the data in both formats and make the application decide where to get the data and with what optimizer.

Lessons learnt – Data distribution

Data Distribution

- All the segments should contain equal portions of data for best performance. If the data is unbalanced or skewed, overall performance is affected.
- Table distribution in Greenplum Database physically divides a table across the Greenplum segments to enable parallel query processing

Distribution Key

- Ensure an even distribution of data, you want to choose a distribution key that is unique for each record
- if that is not possible, then choose DISTRIBUTED RANDOMLY.
- Avoid default first column distribution with many NULL values, Greenplum will choose the first column if none is specified
- You should choose how to distribute your data

Replication: replicate small tables used in JOINS, instead of distributing them across segments

Lessons learnt – Keys

Unique Constraints: To enforced it, the table must be hash-distributed (not DISTRIBUTED RANDOMLY)

- columns must be the same as (or a superset of) the table's distribution key columns.

Index / No Index: Generally, start with no indexes at all.

- Data distribution can somehow be a form of indexing itself. Create indexes as needed.

Partitioning: divide big tables to improve query performance and facilitate data warehouse maintenance tasks

- partitioning does not change the physical distribution
- avoid unnecessary partitioning - small tables for instance
- partitioned tables are also distributed across Greenplum

Foreign Keys: Foreign key constraints are not supported in Greenplum Database.

Lessons learnt – DBA operations

VACUUM:

- it is necessary to do VACUUM periodically, especially on frequently-updated table

ANALYZE:

- help the query planner to choose the most appropriate query plan

VACUUM and **ANALYZE** once a day during a low-usage time of day

Avoid **FULL VACCUM** unless needed

Avoid **CTAS** for large tables

- Try to use transactions to split the tasks
- CTAS can consume all your available resources and may freeze the Greenplum database

Lessons learnt – Table orientation

Row-oriented tables:

- for frequent write data operations
- suited for wide tables, where all or most columns are in the SELECT

Column-oriented tables:

- not optimized for write operations
- suited for wide tables, where you typically only access a small subset of columns in your queries
- can be compressed

Compressing column-oriented tables

- CPU usage
- Compression ratio/disk size
- Speed of compression
- Speed of decompression/scan rate

CONS of using Greenplum

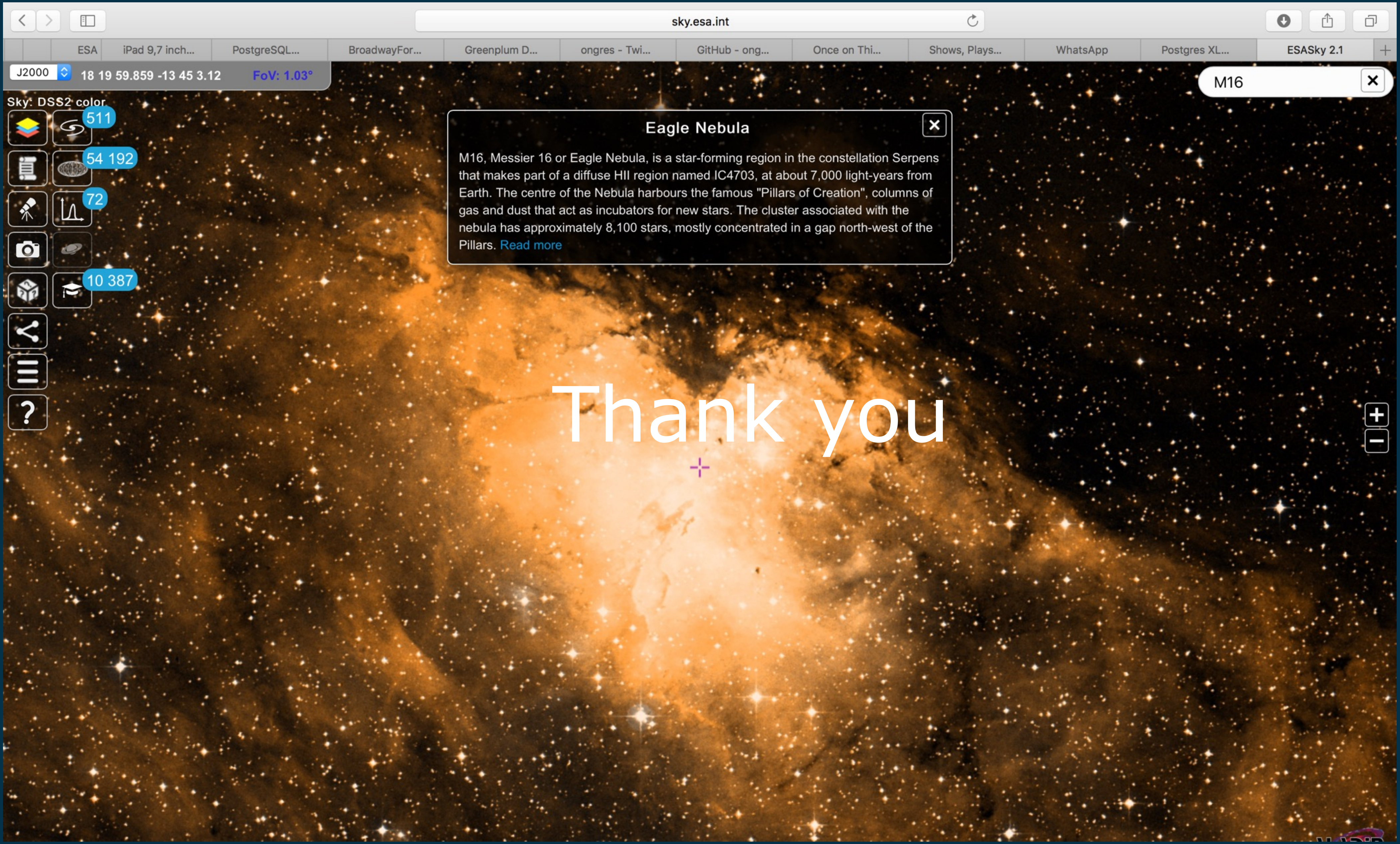
- Change of data model: Foreign keys needs to be removed
- Reserved words: Postgres 9.4 (Greenplum 6)
- Q3C needs to be reviewed for GP

Books and documentation

- Data Warehousing with Greenplum (Marshall Presser)-o’Reilly
- Tutorials: https://greenplum.org/tutorials/
- Official documentation: https://docs.vmware.com/en/VMware-Greenplum/index.html
- Github releases: https://github.com/greenplum-db/gpdb/releases
- Lessons: http://www.dbaref.com/greenplum-database-best-practice---part1#

Open Questions

- Upgrade to Greenplum 7
- Query Caching: Heimdall? Pgpool-II? memcached? Redis
- Backups strategy
- All mission data in 1 cluster?



Eagle Nebula

M16, Messier 16 or Eagle Nebula, is a star-forming region in the constellation Serpens that makes part of a diffuse HII region named IC4703, at about 7,000 light-years from Earth. The centre of the Nebula harbours the famous "Pillars of Creation", columns of gas and dust that act as incubators for new stars. The cluster associated with the nebula has approximately 8,100 stars, mostly concentrated in a gap north-west of the Pillars. Read more

Thank you