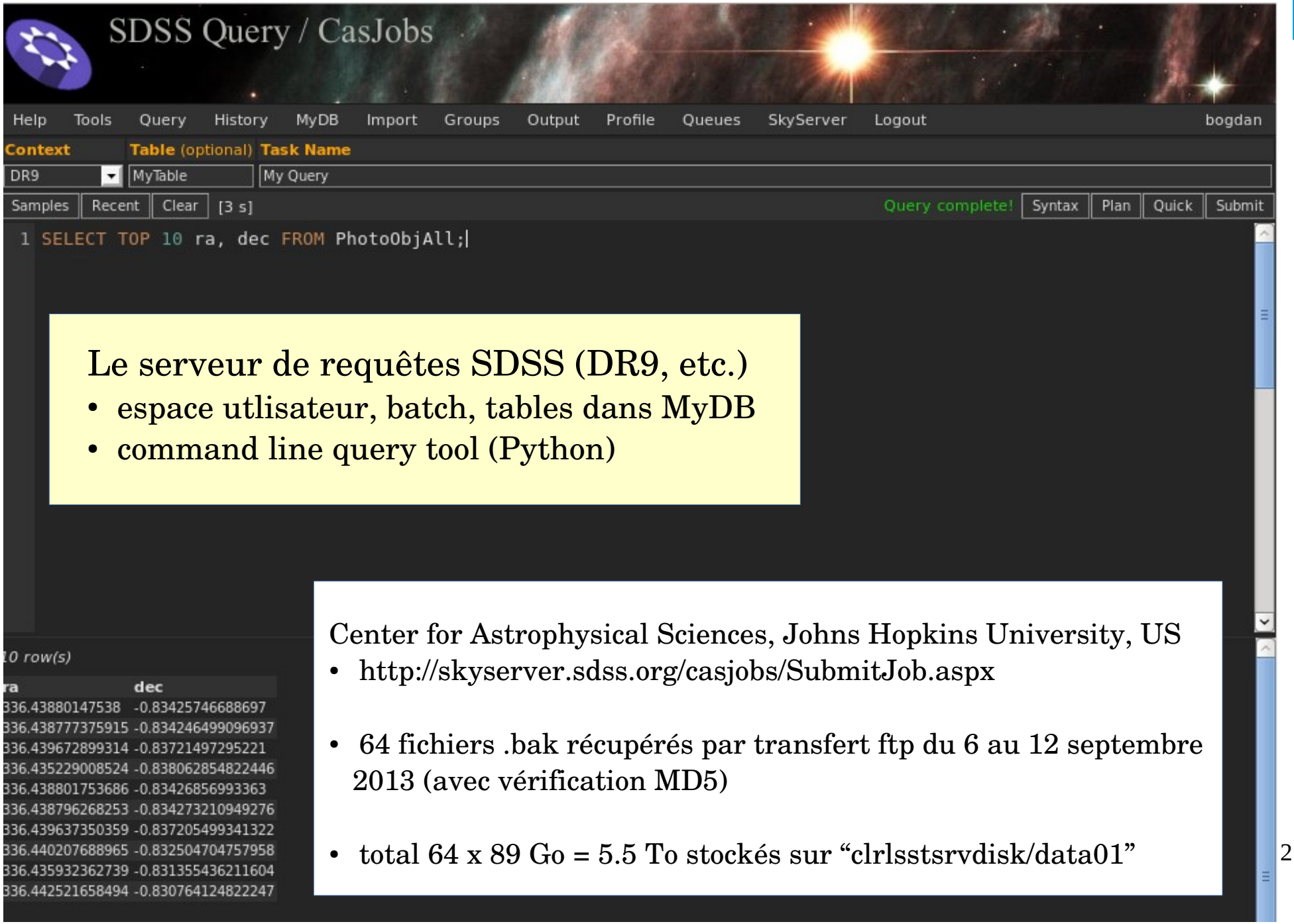
Installation de la SDSS DR9 sur un serveur Linux avec MariaDB (LPC)

LSST-France
7-8/12/2015

(Session Computing 7/12/2015)

Bogdan Vulpescu
Laboratoire de Physique Corpusculaire
Clermont-Ferrand



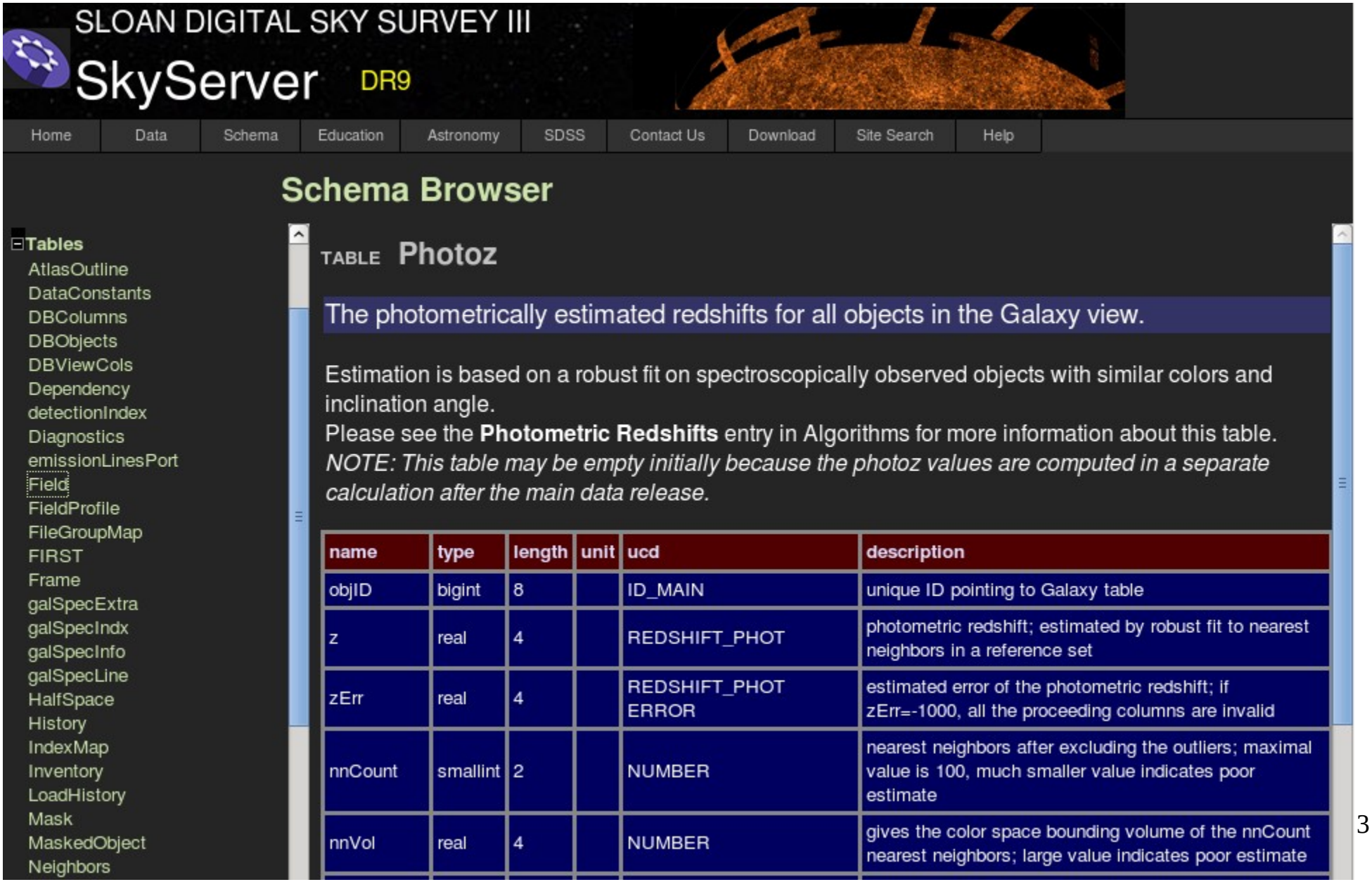
Le serveur de requêtes SDSS (DR9, etc.)

- espace utlisateur, batch, tables dans MyDB
- command line query tool (Python)

Center for Astrophysical Sciences, Johns Hopkins University, US

- http://skyserver.sdss.org/casjobs/SubmitJob.aspx
- 64 fichiers .bak récupérés par transfert ftp du 6 au 12 septembre 2013 (avec vérification MD5)
- total 64 x 89 Go = 5.5 To stockés sur “clrlsstsrvdisk/data01”

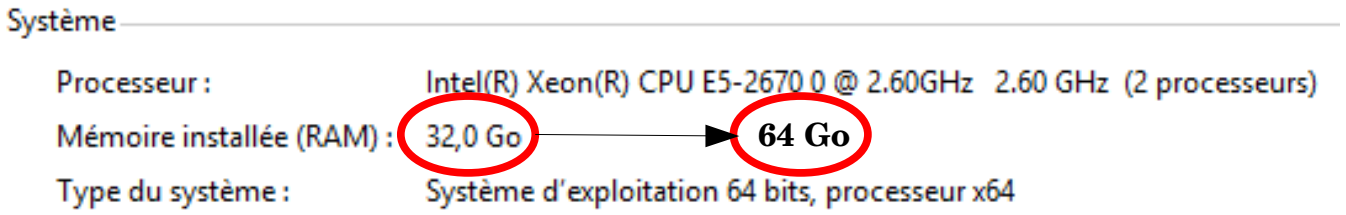
Les tables & views dans la schéma DR9



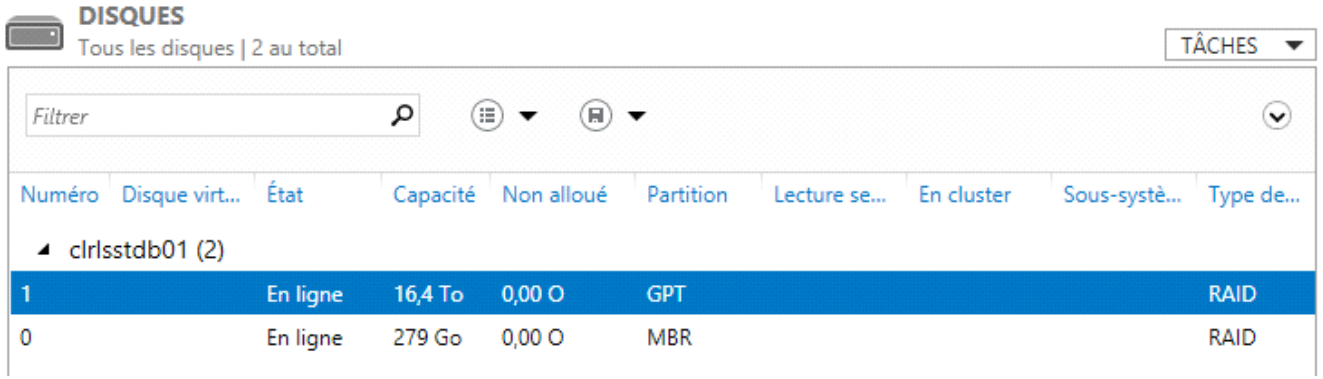
name	type	length	unit	ucd	description
objID	bigint	8		ID_MAIN	unique ID pointing to Galaxy table
z	real	4		REDSHIFT_PHOT	photometric redshift; estimated by robust fit to nearest neighbors in a reference set
zErr	real	4		REDSHIFT_PHOT ERROR	estimated error of the photometric redshift; if zErr=-1000, all the proceeding columns are invalid
nnCount	smallint	2		NUMBER	nearest neighbors after excluding the outliers; maximal value is 100, much smaller value indicates poor estimate
nnVol	real	4		NUMBER	gives the color space bounding volume of the nnCount nearest neighbors; large value indicates poor estimate

Les ressources hardware

- Dell PowerEdge R720xd
- 32 Go, 2.6 GHz
- 2 x 8 cores

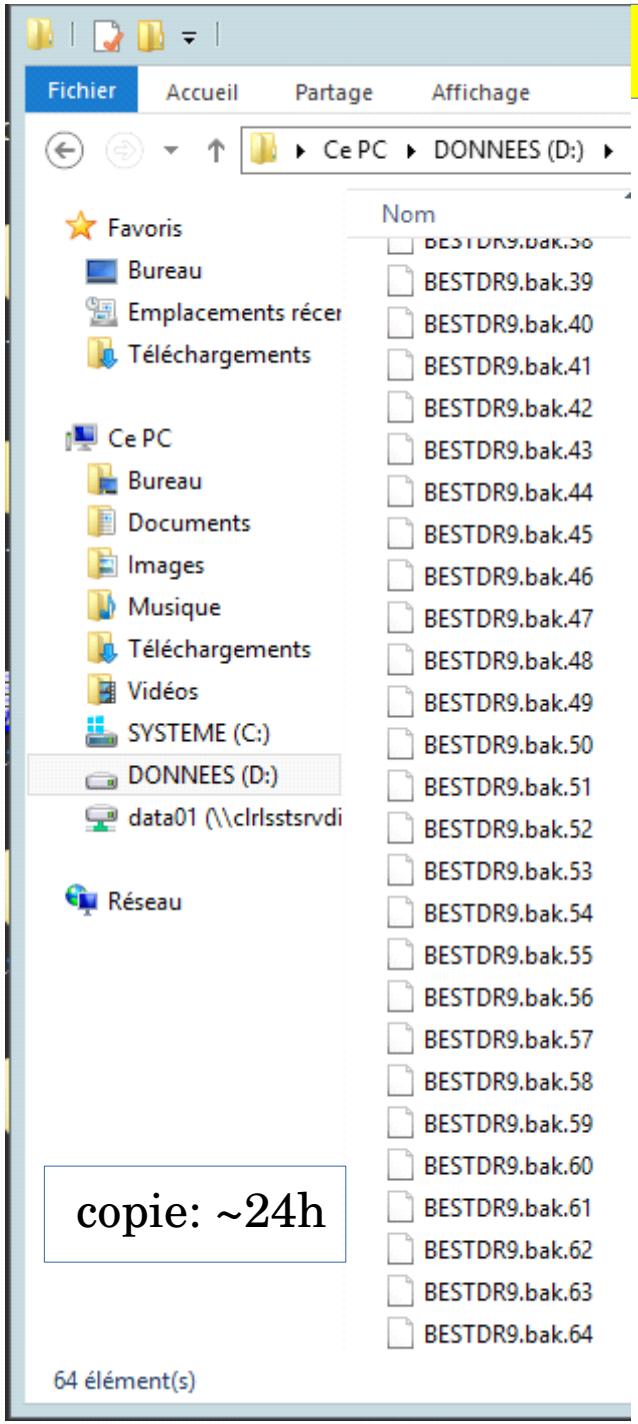


- espace disque en local:
  279 Go + 16.4 To (RAID)



- après la restauration:

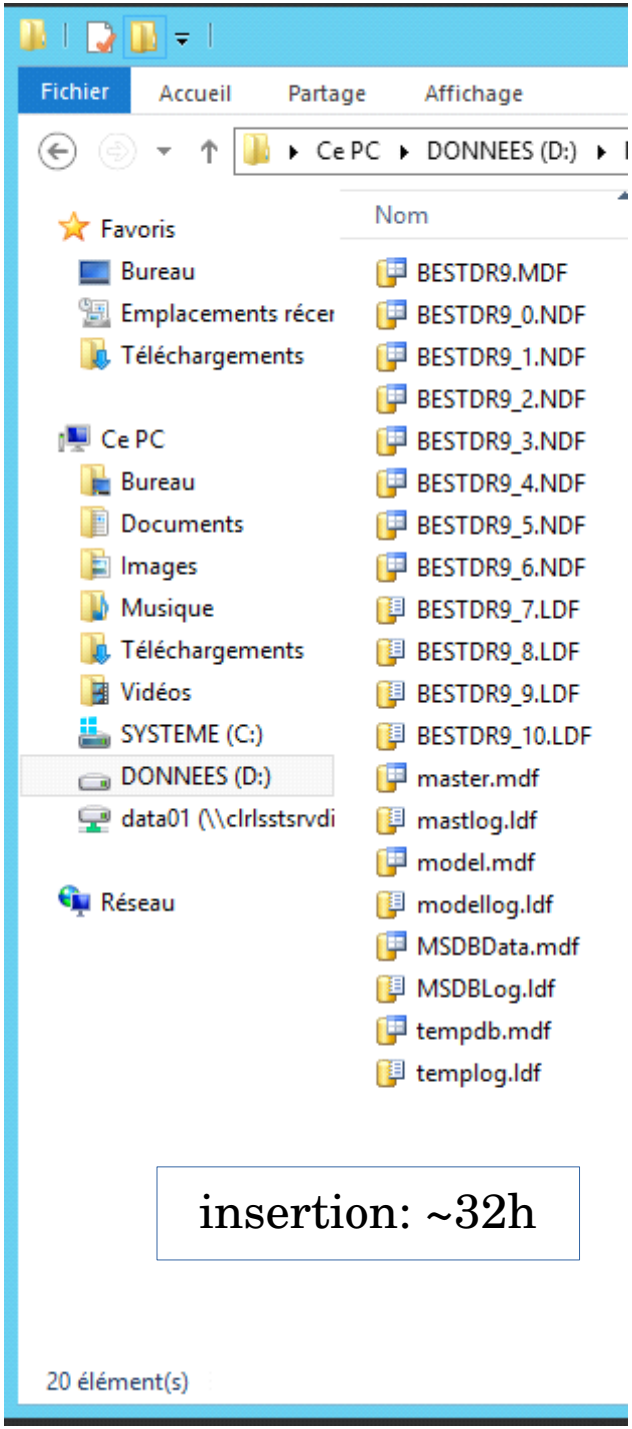
- Serveur Windows 2012 R2 Standard 64Bit 2 proc.
- Serveur SQL 2012 Standard
- Microsoft SQL Server Management Studio



5.50 To → 9.86 To

Restauration de la DB (MSSQL)

copie: ~24h



insertion: ~32h

L'extraction de données en format CSV pour la table principale *PhotoObjAll* (> 1.2 milliards)

Quelle partition des données?

PhotoObjAll contient une clé primaire “objID” (nombre entier très long):

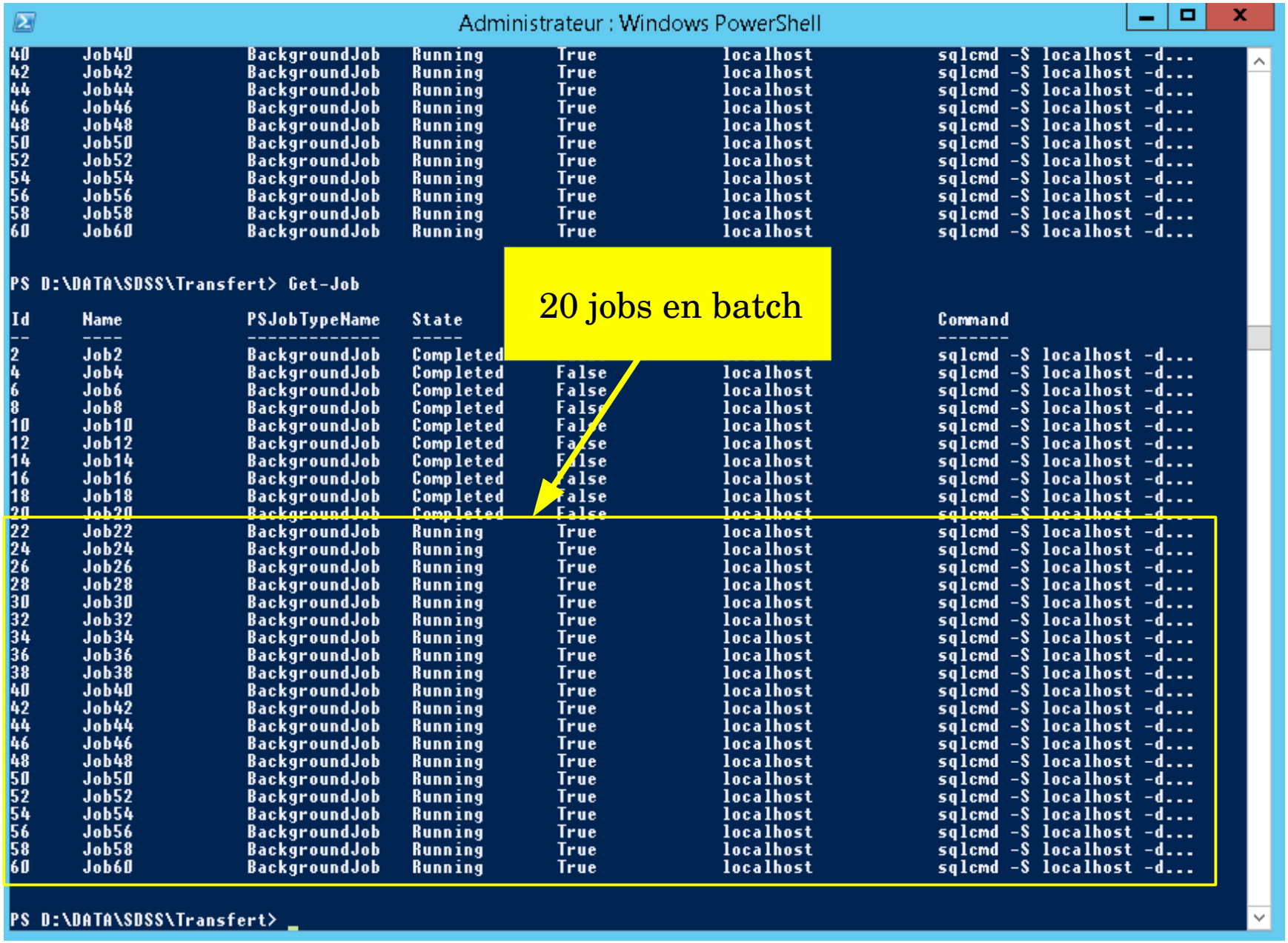
- indexée: unique mais pas continue
- domaine: 1237645876861272065 . . . 1237680531356583125
- domaine divisé en 100 paquets: delta(objID) = 541476489235

Intervalle en “objID” pour chaque page (paquet):

```
  1-page: 1237645876861272065                    . . . 1237645876861272065+1*541476489235
  2-page: 1237645876861272065+1*541476489235     . . . 1237645876861272065+2*541476489235
  3-page: 1237645876861272065+2*541476489235     . . . 1237645876861272065+3*541476489235
  . . .
  . . .
100-page: 1237645876861272065+99*541476489235   . . . 1237645876861272065+100*541476489235
```

extraction en parallèle ---> 5 jours, 5.8 To CSV

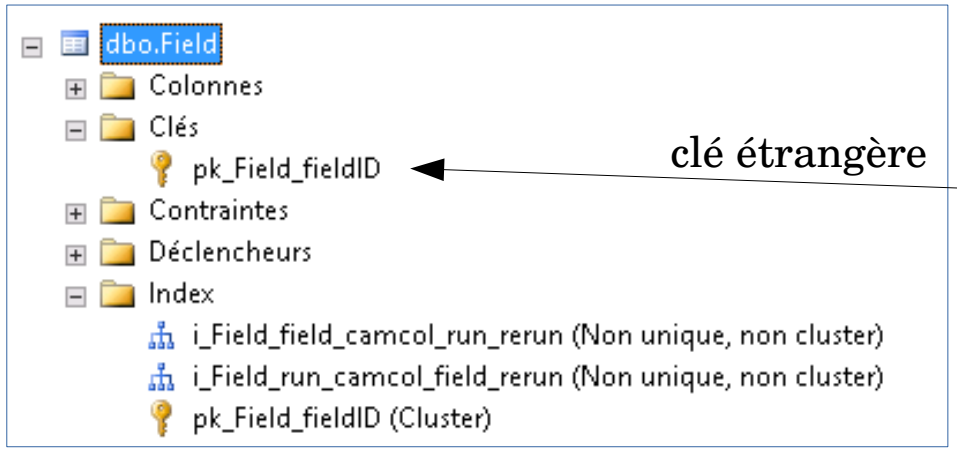
Extraction en batch (Windows Powershell 3.0)



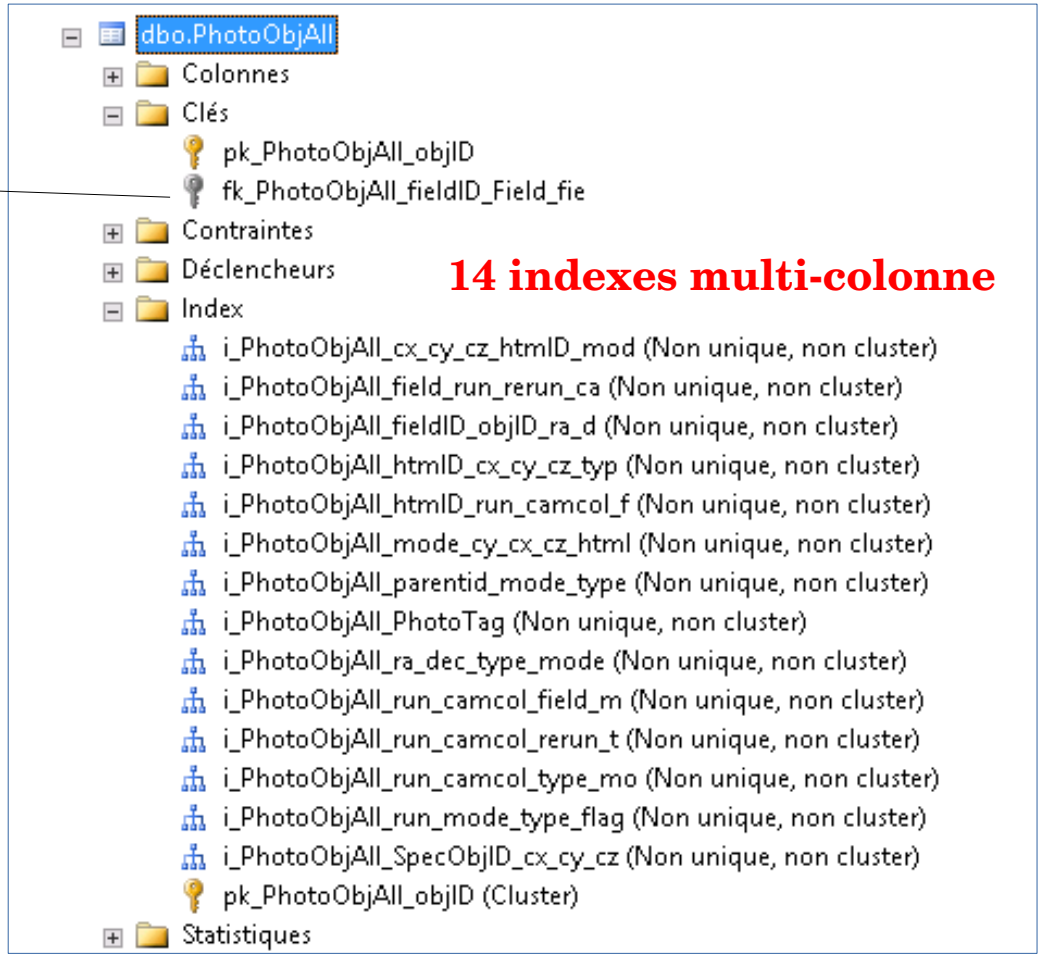
Le serveur MySQL/MariaDB

- “clrlsstwn04” (même type de machine que “clrlsstdb01”)
- OS = Scientific Linux 6.6
- espace disque local pour les données: 7.3 To
- pas dans le domaine NIS du laboratoire (utilisateur locaux)
- montage NFS du disque de données “clrlsstsrvdisk:/data01” avec les fichiers CSV
- MySQL Workbench 6.3 GUI pour l'administration de la DB
- deux utilisateurs sur la DB: un pour “lire/écrire” et un deuxième pour “lire”

Les clés et les indexes de la table *PhotoObjAll*



clé étrangère



14 indexes multi-colonne

Choix du motéur de stockage: MyISAM

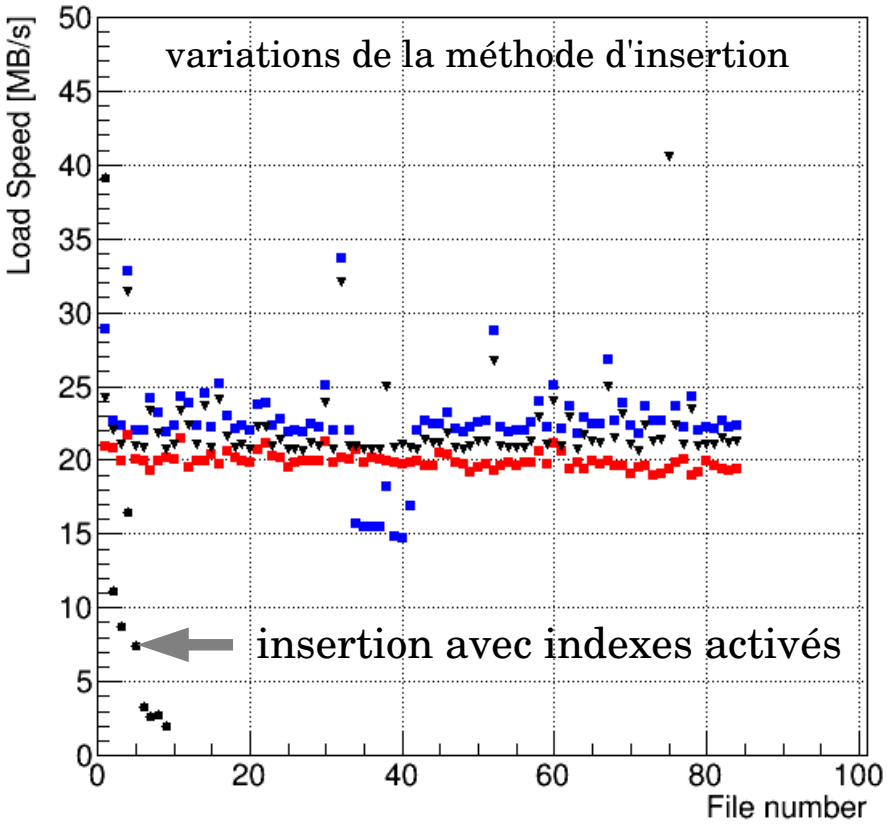
- MySQL par défaut dans qserv
- chargement rapide
- ne traite pas les clés étrangères

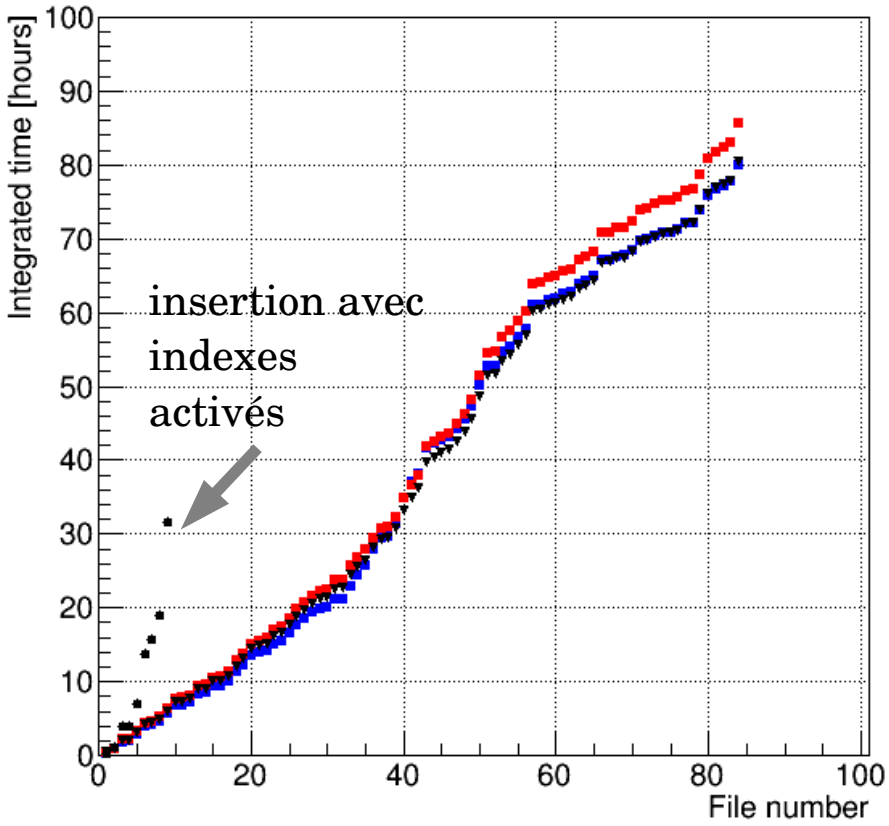
Exemple ~430000 entrées du *PhotoObjAll*

	Field		PhotoObjAll	
	LOAD DATA	CREATE INDEX	LOAD DATA	CREATE INDEX
InnoDB	240 sec	50 sec	140 sec	230 sec
MyISAM	70 sec	22 sec	40 sec	185 sec

Le chargement des fichiers CSV (LOAD DATA)

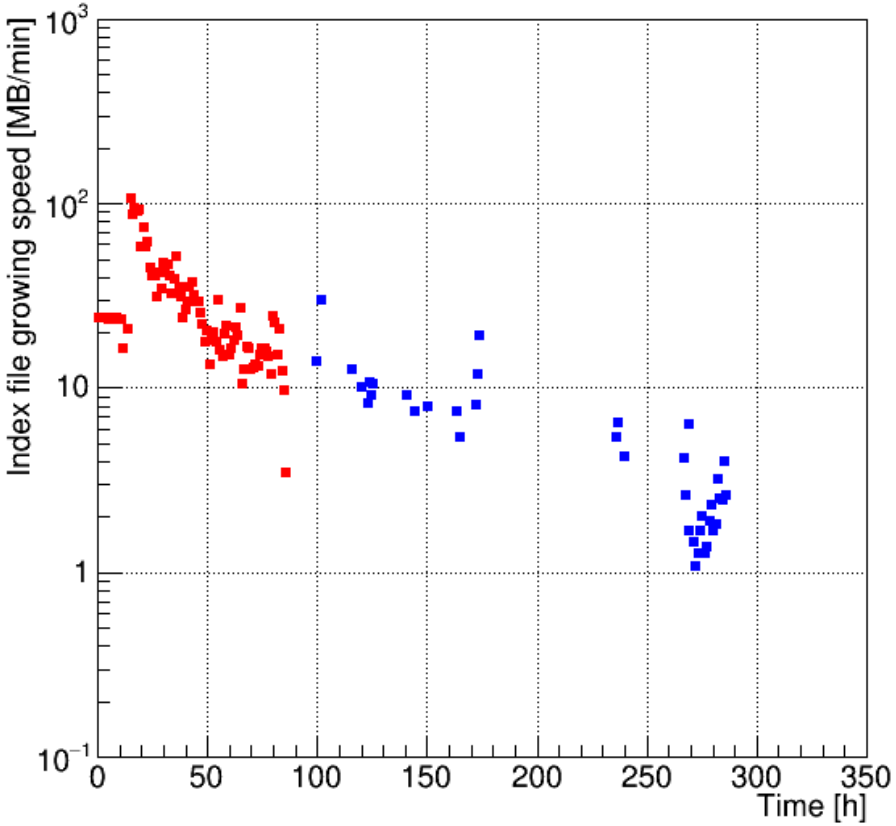
- insertion des données avec les indexes désactivés
- vitesse d'insertion ~ 21 Mo/s, temps total ~ 3.3 jours

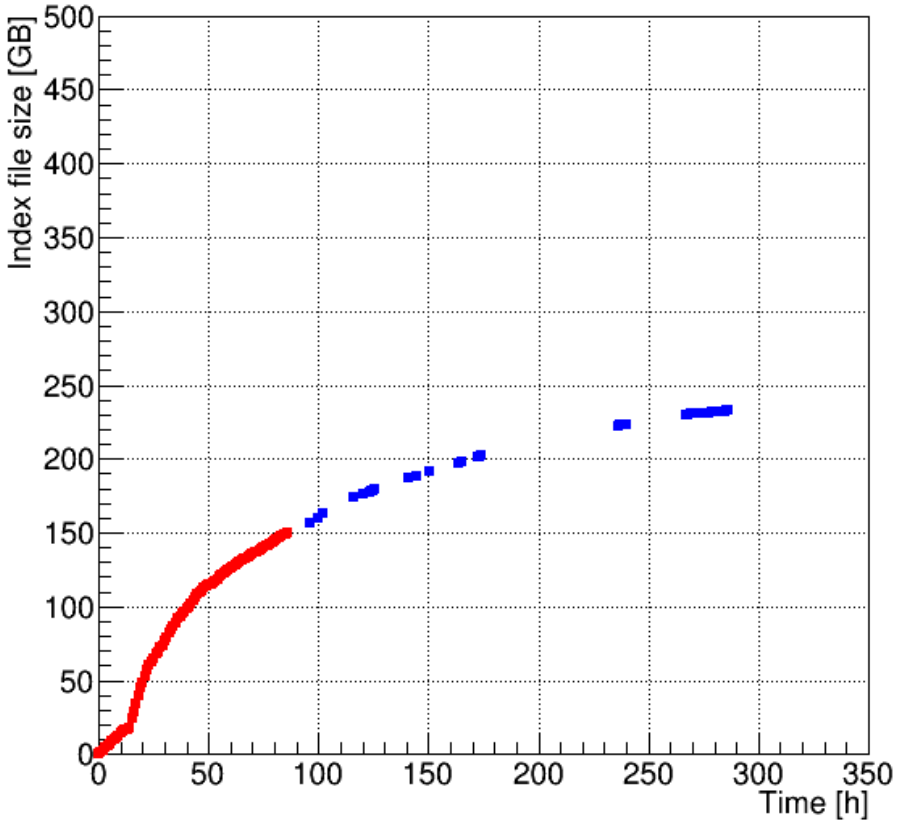




Création des indexes après l'insertion des données

- test d'ajustement de différents paramètres
- création d'une réplique temporaire de la table: 13 heures
- estimation taille fichier d'indexes > 1 To
- 240 Go en 300 heures ! . . .





L'importance des indexes dans un exemple

- section de la table *PhotoObjAll* avec 10^6 entrées, serveur MySQL sur une station de travail
- LOAD DATA avec indexes activés: ~ 5 minutes (3.6 Go données, 1.4 Go indexes)
- LOAD DATA avec indexes désactivés: ~ 2 minutes
- requête:

```
SELECT ra, dec_ FROM PhotoObjAll WHERE type_ = 3 AND ra BETWEEN 49.0 and 50.0
```
(5872 rows)

  - avec les indexes désactivés: 27 secondes
  - avec les indexes activés: 0.68 secondes
  - avec les indexes re-crées par la procedure suivante: 0.08 secondes (optimisation?)

Solution pour la création des indexes sur la table *PhotoObjAll*

- testes pour la production Winter13 (Fabrice J.) sur la table RunDeepForcedSource
  - 2.5 To, 3.87 milliard lignes, 87 colonnes, 5 indexes à colonne unique
  - MariaDB 5.5 + patch + “myisamchk”
  - 125h 21m 58s mono-thread (270 GB fichier index)
  - 11h 30m multi-thread 8 cores (258 GB fichier index)
- **réparation de la table avec “myisamchk” (utilisation différente de Winter13)**
  - **4.3 To, 1.23 milliard lignes, 500 colonnes, 14 indexes multi-colonne**
  - **61h mono-thread (1.2 To fichier index)**
  - pas de multi-thread . . .
  - attention aux fichiers temporaires . . .

Continuation avec les autres tables de la DR9

Problèmes rencontrées:

- chaque table traitée séparément, pas de procédure automatique
- mots réservés dans MariaDB utilisés comme noms de colonne dans MSSQL
- séparateur des champs (colonnes): “,” , “;” , “&” ou “@”
- `varchar(max)` n'a pas d'équivalent dans MariaDB
- `binary(8)` ---> `binary(24)` dû au passage par le fichier CSV
- images (jpeg) dans la tables *Frame*: le formatage de l'extraction/insertion est plus difficile (essai solution trouvée sur le web)
- . . . autres . . .

Conclusions

- SDSS DR9 (septembre 2013) installée sur un serveur MariaDB 5.5
- 317 heures compactes pour le transfert de la table principale *PhotoObjAll* du serveur MSSQL vers le serveur MariaDB et la création des indexes
- solution pour la création (réparation) rapide (mono-thread) des indexes; comment faire multi-thread ?
- dimensionnement de l'espace de stockage nécessaire
- cherche solution pour l'extraction/insertion des images (jpeg); pas d'images dans les tables *Frame* et *SpecObjAll*
- continue avec les autres tables
- hier: *AtlasOutline, DataConstants, DBColumns, DBObjects, DBViewCols, Dependency, detectionIndex, Diagnostics, emissionLinesPort, Field, FieldProfile, FileGroupMap, FIRST, Frame, galSpecExtra, galSpecIndx, galSpecInfo, galSpecLine, PhotoObjAll, Photoz*, *SpecObjAll* = 6.0 To