Intégrer des données dans PostgreSQL

PGDAY France 2014

Dimitri Fontaine dimitri@2ndQuadrant.fr
@tapoueh

6 juin 2014

Dimitri Fontaine

2ndQuadrant France
PostgreSQL Major Contributor

- `pgloader`
- `prefix`, `skytools`
- `apt.postgresql.org`
- `CREATE EXTENSION`
- `CREATE EVENT TRIGGER`
- *Bi-Directional Réplication*
- `pginstall`

pgloader : Intégrer des données dans PostgreSQL

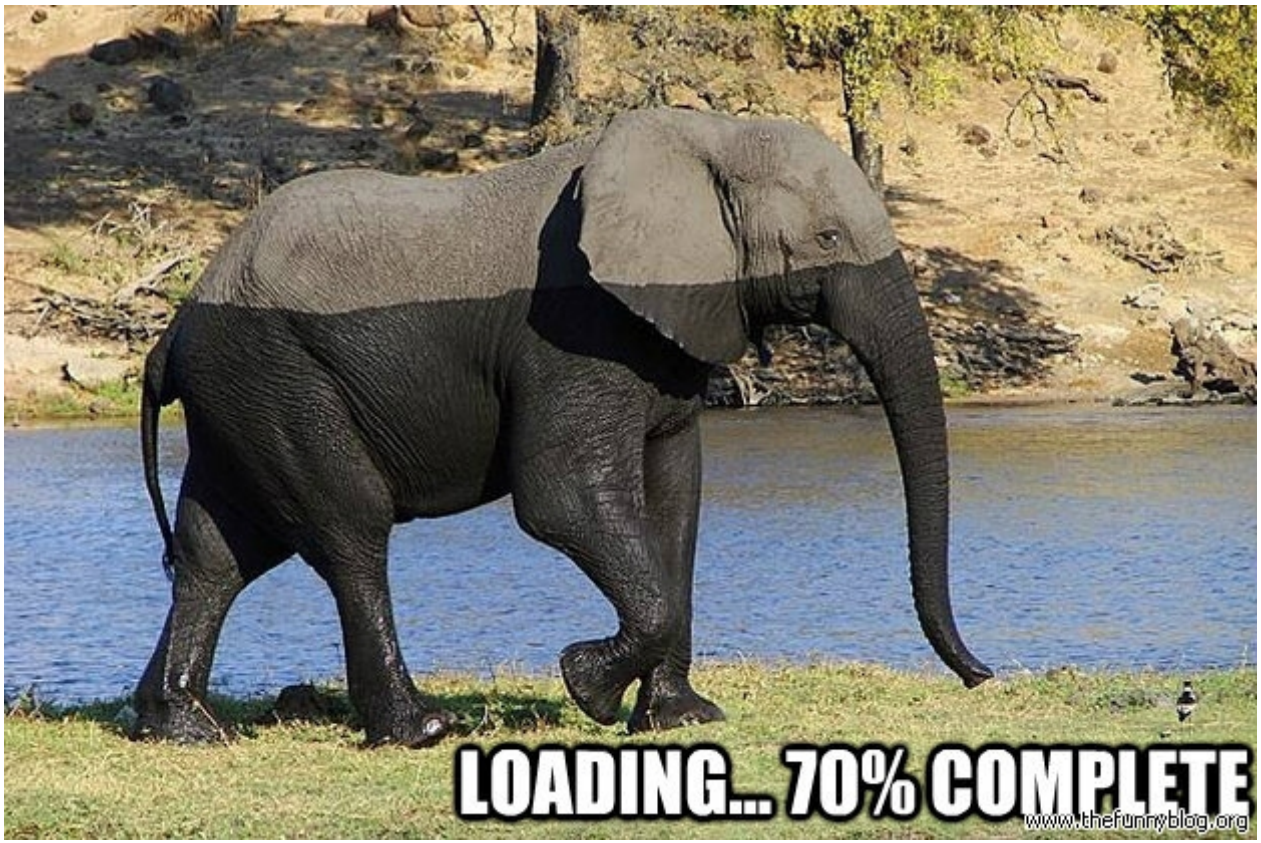
http://pgloader.io/

pgloader : Open Source, github

https://github.com/dimitri/pgloader

pgloader : Charger des données

http://pgloader.io/



Importer depuis des fichiers CSV

http://pgloader.io/howto/csv.html

Importer depuis des fichiers dBase III

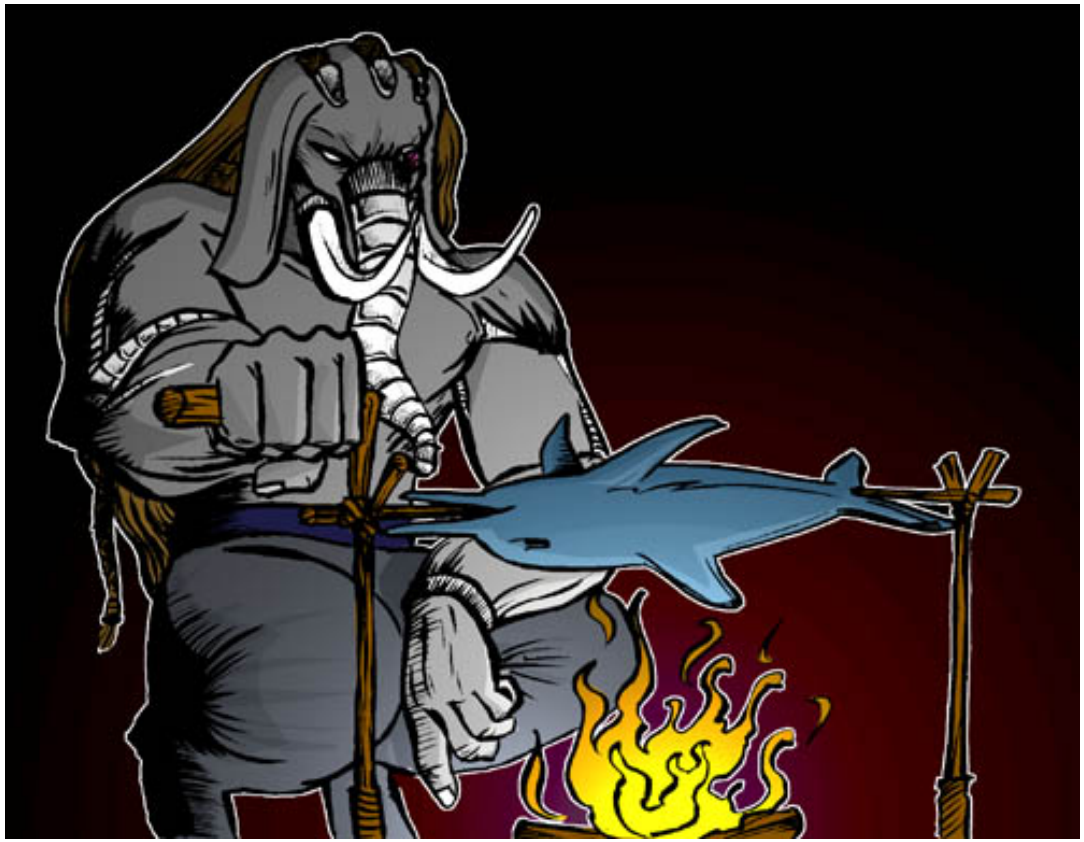
`http://pgloader.io/howto/dBase.html`

Importer depuis des fichiers SQLite

http://pgloader.io/howto/sqlite.html

Importer depuis une connection MySQL

http://pgloader.io/howto/mysql.html

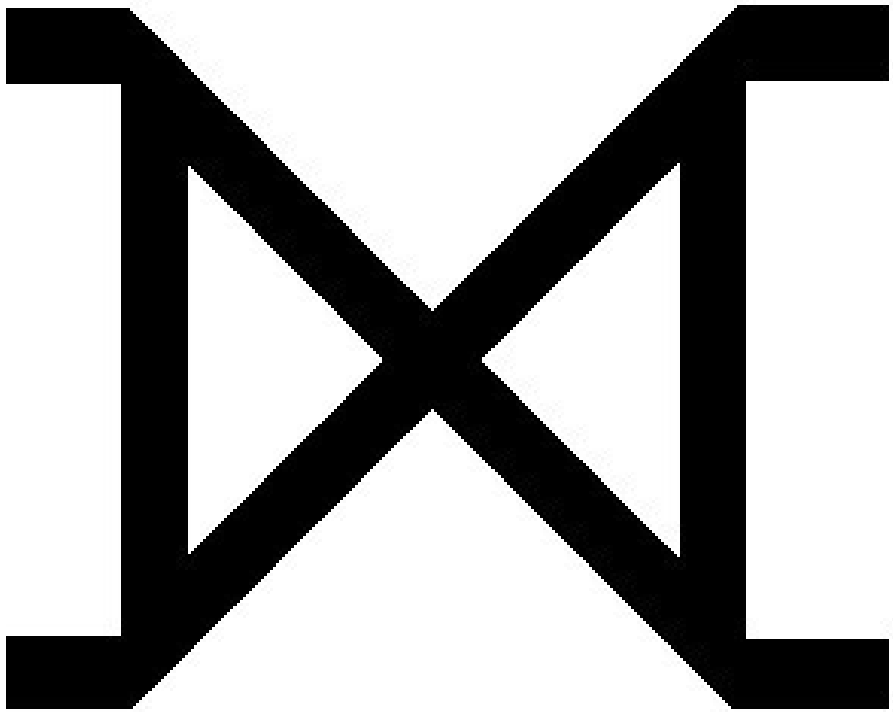


Importer depuis une connection MySQL

http://pgloader.io/howto/mysql.html

pgloader : Transformer les données à la volée

http://pgloader.io/



pgloader : version 3

Version 1 en **TCL**, version 2 en **Python**, version 3 en **Common Lisp**

pgloader : charger des données **rapidement**

Retour d'utilisation, v2 contre v3

```
 select rows, v2, v3,
        round((  extract(epoch from v2)
        / extract(epoch from v3))::numeric, 2)
        as speedup
   from timing;
```

```
  rows   |         v2          |        v3         | speedup
---------+---------------------+-------------------+---------
 4768765 | @ 37 mins 10.878    | @ 1 min 26.917    |   25.67
 3115880 | @ 36 mins 5.881     | @ 1 min 10.994    |   30.51
 3865750 | @ 33 mins 40.233    | @ 1 min 15.33     |   26.82
 3994483 | @ 29 mins 30.028    | @ 1 min 18.484    |   22.55
(4 rows)
```

COPY

La commande copy sera toujours plus rapide que pgloader, mais sa gestion des échecs est limitée.

Fonctionnalités de pgloader

pgloader propose bien plus que copy

- Gestion des erreurs avec des fichiers de rejet de données
- Transformation des données à la volée
- Language de commande évolué pour spécifier le travail à réaliser
- Architecture parallèle, IO asynchrones
- Support de nombreux formats

On passe du fichier de configuration

```
[pgsql]
base = pgloader
client_encoding = 'latin1'
pg_option_standard_conforming_strings = on
null          = ""
empty_string = "\ "

[csv]
table             = csv
format            = csv
filename          = csv/csv.data
field_size_limit = 512kB
field_sep         = ,
quotechar         = "
columns           = x, y, a, b, d:6, c:5
only_cols         = 3-6
skip_head_lines   = 1
```

À la commande

```
LOAD CSV
     FROM inline (x, y, a, b, c, d)
     INTO postgresql:///pgloader?csv (a, b, d, c)

     WITH truncate,
          skip header = 1,
          fields optionally enclosed by '"',
          fields escaped by double-quote,
          fields terminated by ','

      SET client_encoding to 'latin1',
          work_mem to '12MB',
          standard_conforming_strings to 'on'
```

À la commande

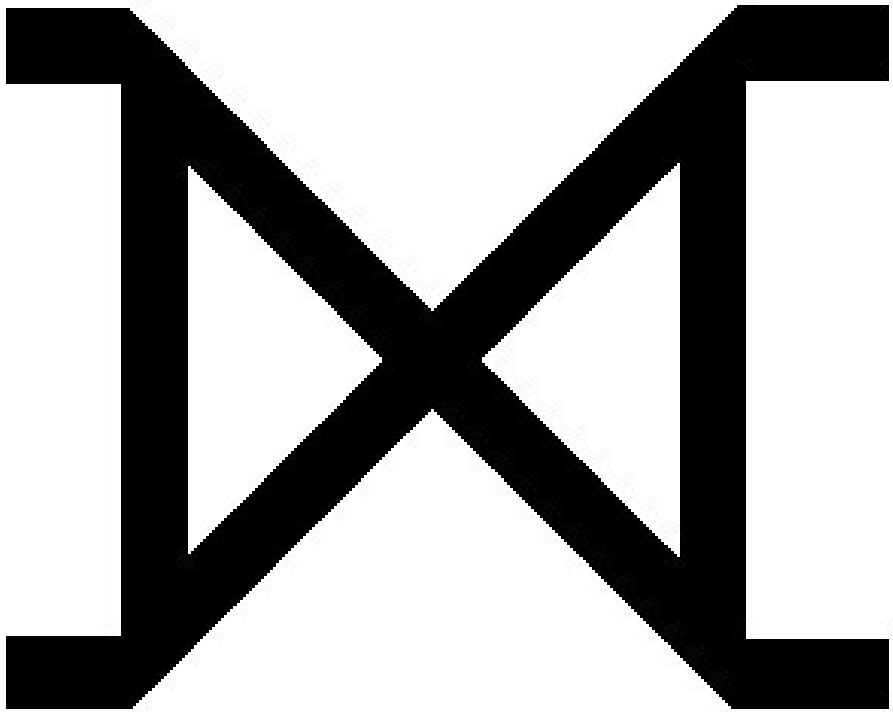
```
BEFORE LOAD DO
 $$ drop table if exists csv; $$,
 $$ create table csv (
     a bigint,
     b bigint,
     c char(2),
     d text
    );
 $$;
```

Exemples de sources de données

```
FROM stdin
FROM inline (a, b, c)
FROM data/2013_Gaz_113CDs_national.txt

FROM FILENAME MATCHING ~/GeoLiteCity-Location.csv/
FROM ALL FILENAMES MATCHING ~/ALIOR/
FROM ALL FILENAMES MATCHING ~/F[A-Z]{4}1[45]|OZ20/

FROM http://www.census.gov/geo/maps-data/
            data/docs/gazetteer/places2k.zip

FROM http://www.insee.fr/fr/methodes/nomenclatures/
                cog/telechargement/2013/dbf/historiq2013.zip
```

pgloader : Transformer les données à la volée

http://pgloader.io/



Transformation de données

```
FROM FILENAME MATCHING ~/GeoLiteCity-Blocks.csv/
     WITH ENCODING iso-8859-1
     (
        startIpNum, endIpNum, locId
     )
INTO postgresql:///ip4r?geolite.blocks
     (
        iprange ip4r using (ip-range startIpNum endIpNum),
        locId
     )
```

Transformation de données

```
FROM FILENAME MATCHING ~/GeoLiteCity-Location.csv/
     (
        locId, country,
        region     null if blanks,
        city       null if blanks,
        postalCode null if blanks,
        latitude, longitude,
        metroCode  null if blanks,
        areaCode   null if blanks
     )
INTO postgresql:///ip4r?geolite.location
     (
        locid, country, region, city, postalCode,
        location point
           using (format nil "(~a,~a)" longitude latitude),
        metroCode, areaCode
     )
```

Migration complète de MySQL à PostgreSQL

http://www.galaxya.fr/

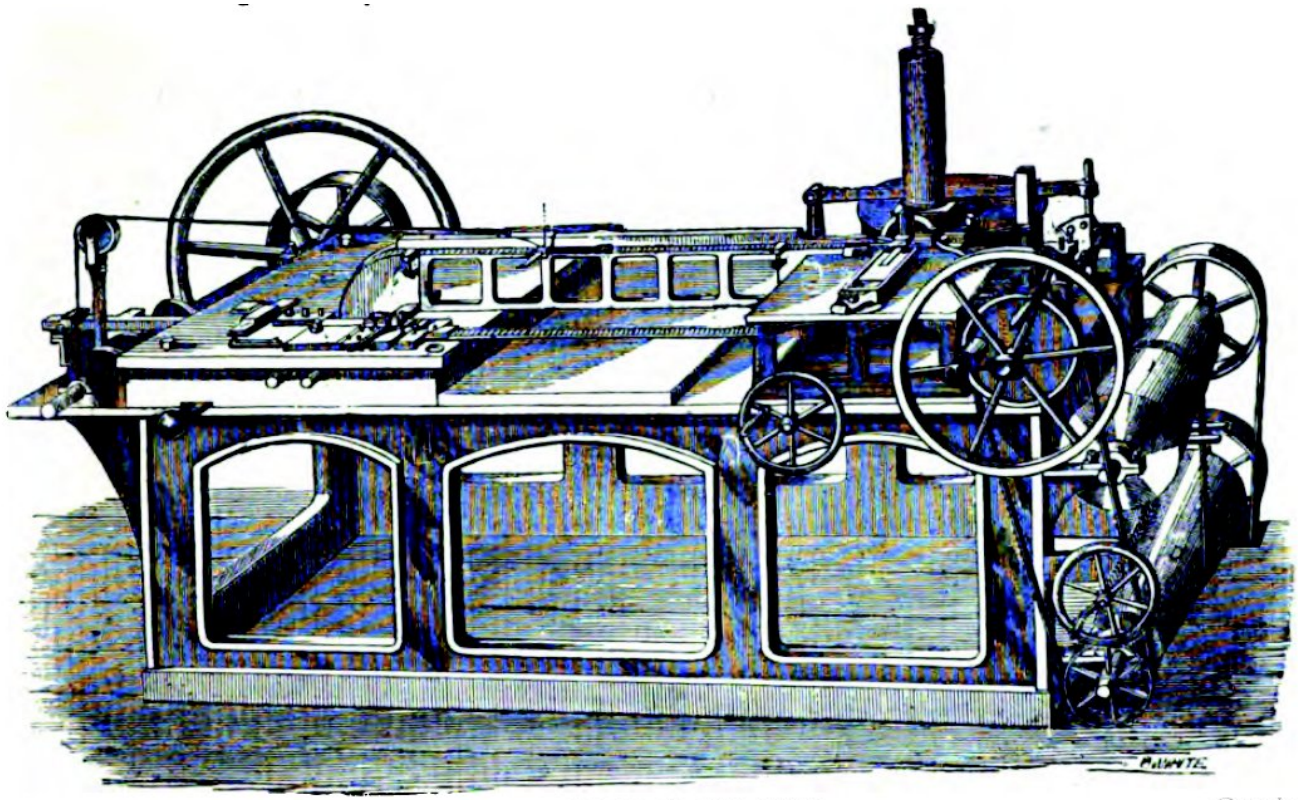
Pour Galaxya, utilisation des outils habituels

Mais les vrais problèmes ne sont pas résolus...

- `mysql2pgsql`, puis édition manuelle du schema
- `SELECT INTO OUTFILE` sur le server, puis `COPY`
- Le client MySQL prétend donner du CSV avec une redirection
- On ajoute toujours une étape `awk` ou `sed`
- On trouve quelques scripts en Python et en Ruby

MySQL et les types de données

Text vide ou NULL, valeurs par défaut, dates à `0000-00-00`, `int(11)`, `float(20,2)`, `tinyint` mais pas de `boolean`, `sets`, encodage, ...



JOHNSON AND ATKINSON'S TYPE CASTER.

MySQL : édition des règles de transtypages

```
LOAD DATABASE
     FROM      mysql://root@unix:/tmp/mysql.sock:/goeuro
     INTO postgresql://dim@unix:/tmp:/godollar

CAST type datetime to timestamptz
                drop default drop not null
               using zero-dates-to-null,

     column bools.a to boolean drop typemod
                using tinyint-to-boolean,

     type char when (= precision 1)
       to char keep typemod,

     column enumerate.foo
      using empty-string-to-null
```

MySQL : MATERIALIZE VIEWS

Changer de schéma pendant la migration, c'est possible

```
MATERIALIZE VIEWS foo,
  d as $$
      select cast(d as date) as d, count(*) as n
        from plop
       where d > '2013-10-02'
    group by cast(d as date);
  $$
```

Limites de pgloader

Il reste beaucoup de travail

- Vues (dialectes SQL différents)
- Triggers
- Procédures Stockées
- Types de données géométriques

Les autres bases de données supportées

À suivre

De nouvelles sources de données

Chargement avec éventuelle *normalisation* des données

De nouvelles connections


FREE
OUR
DATA

Questions?

Now is the time to ask!