- Réplication
- E-maj
- Foreign Data Wrapper
- PostGIS
- PostgreSQL-f

Réplications, disponibilités, durabilités

Evolution dans la base de données :

Postgres 8.2 : warm standby
Postgres 9.0 : hot stanby
Postgres 9.1 : réplication synchrone (activable par transaction)
Postgres 9.3 : réplication bi-directionnelle

De nombreuses possibilités externes :
Exemples :
ETL : kettle (Gnu GPL), pgbouncer (pool de conection PostgreSQL)

Note : hot standby – mode synchrone praticable en grande echelle
Ex : ~2000 requêtes secondes entre Amsterdam-NewYork

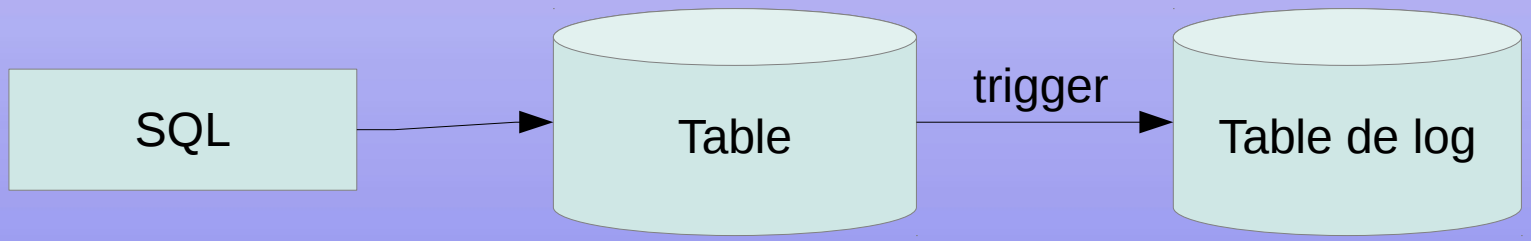
E-maj : Enregistrement des mise à jour

Principe :

- Reprise à partir de marques : i.e. : points de reprises (sans utilisation de pg_dump/pg_restore)
- Groupe de tables, permettant d'appliquer des commit/rollback sur l'ensemble

Caractéristiques :

- OpenSource
- Surtout au niveau des logs : toutes les requêtes de mises à jours sont loggés dans une table de log : le volume généré est important (30%)
- Ne gère pas les triggers qui doivent être désactivés
- DDL non gérées : Emaj est adapté aux bases dont le schémas n'évolue pas ou peu (alter table, drop...)



Foreign Data Wrapper

Inclure une table « virtuelle » , hébergée sur une autre base de données

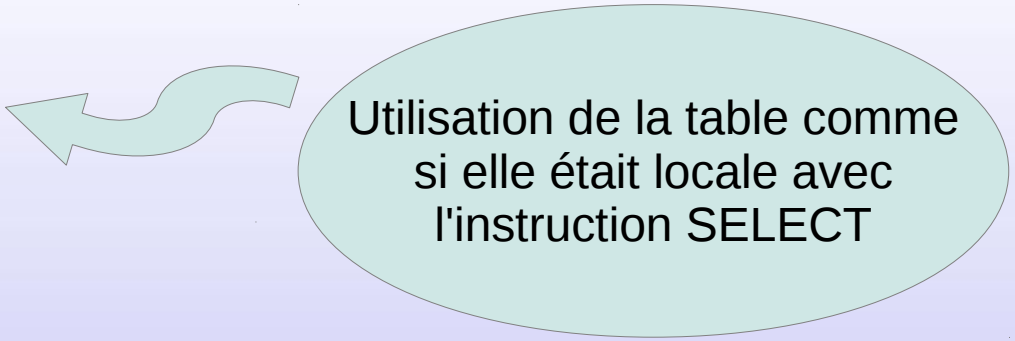
Utilisation :
- Migration de données
- Peut remplacer l'usage d'un ETL
- Peut servir « d'API » pour relier plusieurs bases entre elles

Caractéristiques :
- Suite à dblink
- Plugins (ou **wrapper**) permettant de liés des données de différents types de SGBD : PostgreSQL, MySQL, Oracle, CSV, twitter, …
  Possibilité de créé son propre wrapper :
  - API C ou python à l'aide du wrapper « multicorn »

Exemples d'utilisations :
- Linké un fichier CSS, (parfile?)

Syntaxe SQL

```
CREATE TABLE matableetrangere {
        field,
        ….,
} SERVER …. OPTIONS {
        filename= …,
        format=csv,
        ….
}
```



Limitations :

- Impossible d'écrire sur les tables étrangères
- Jointures :
  peut être lent en nécessitant la COPY (commande copy PostgreSQL) de la table étrangère. Possibilité de coder un wrapper en donnant des statistiques.
  La clause WHERE peut être envoyé à la table distante.
- Impossible de créer un index sur une table étrangère
- Lance un nouveau processus dans le cas d'une table étrangère hébergé sur un serveur Postgres
- Pas encore de couches d'abstraction style JDBC, ODBC => nécessite l'écriture de wrapper (~200lignes de C au minimum)

Postgis : base de données de géolocalisation

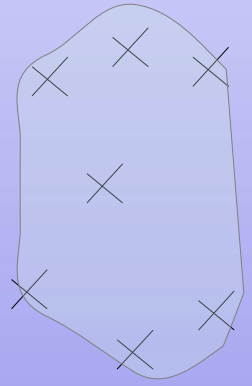
« 80% de l'information est spaciale »

Caractéristiques :

- Forte charge, volumétries
- Utilisateurs : public (ex : Collectivité territoriale Grenoble, IGN) + privé (France Télécom)
- Dimension 2D, 3D (distance)
- Index spatiaux basés sur un arbre Rtree (comme PgSphere)

Possibilité de stocker différentes géométries :

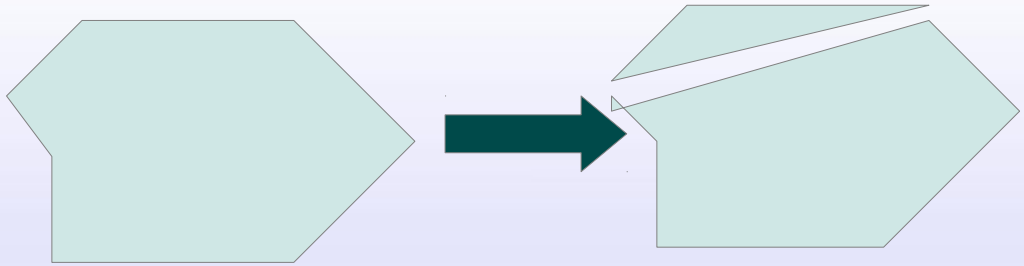
- Le point (comme PgSphère),le Multi-point
- Le polygone
- Des formes plus complexes: enveloppes et courbes définies par un ensemble de points



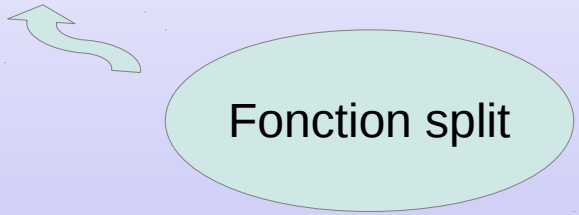
Syntaxe SQL :

```
CREATE TABLE matable (
        ….,
        geom geometry(....) ) ;
CREATE INDEX … ON matable USING GIST(geom) ;
```

Fonctions géométriques :
- Jointure : TS_Union, TS_Intersect,
- Distance : TS_distance, CENTROID, ...
- Split



Opérateurs (similaire à PgSphere) :
- ex : opérateur de distance ↔ (les centres des box)
  ou <#> (les bords des box)



Pourrait permettre de stocker des zones comme délimiter des nuages, nébuleuses, galaxies ????

Avenir et ressentis sur Postgis :
- Un intérêt certain et une communauté dynamique
- Plus lourd que PostgreSQL
- Version (future) Postgis3.0 : amélioration des performances :
  création des index (x3), lecture

PostgreSQL_f : imprécision et médiation des SGBD

Projet Pilgrim (Brest) – départements de mathématiques appliqués.
« théories des possibilités », ou « logique floue »

Idée :

- Recherche selon un critère non binaire (contrairement au SQL) et pouvant être paramétré
- Exemple : rechercher des valeur « proche « d'une constante
- Notion d'ordre dans le résultat

Exemple d'utilisation de la fonction between :

Between SQL

Between PostgreSQL-f « floue »

Caractéristiques :

- Plus proche du langage parlé
- Développement d'un extension PostgreSQL pour les **types numériques**

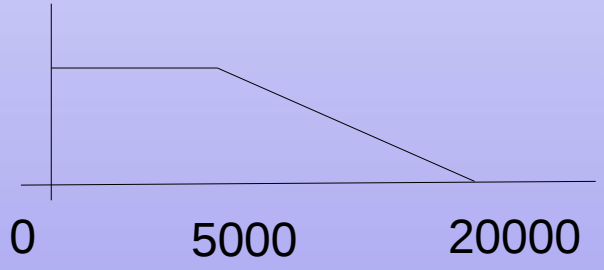
PostgreSQL-f : imprécision et médiation des SGBD

Implantation PostgreSQL :

- Fonction d'incertitude de type trapézoïdale
- Transformation de la logique floue [0,1] → {0,1}
- Création de fonction (ex ci dessous)
- Création d'opérateur de distance (~), IN, BETWEEN, ...
- Création d'index.

Syntaxe SQL :

```
/* definition de la notion cheap */
SELECT create_predicate('cheap', 0,0,5000,20000) ;
SELECT  * FROM table WHERE colonne IS 'cheap' ;
SELECT  * FROM table WHERE lapluspart(.....) ;
```



Avancement du projet :

En développement (version alpha disponible)