

Skytools3

PostgreSQL trigger-based replication

Dimitri Fontaine `dimitri@2ndQuadrant.fr`

July, 18 2013

2ndQuadrant France PostgreSQL Major Contributor

- ✓ pgloader, prefix, skytools, ...
- ✓ apt.postgresql.org
- ✓ CREATE EXTENSION
- ✓ CREATE EVENT TRIGGER
- ✓ MySQL migration tool, new pgloader version



2ndQuadrant France PostgreSQL Major Contributor

- ✓ pgloader, prefix, skytools, ...
- ✓ apt.postgresql.org
- ✓ CREATE EXTENSION
- ✓ CREATE EVENT TRIGGER
- ✓ MySQL migration tool, new pgloader version



2ndQuadrant France PostgreSQL Major Contributor

- ✓ pgloader, prefix, skytools, ...
- ✓ apt.postgresql.org
- ✓ CREATE EXTENSION
- ✓ CREATE EVENT TRIGGER
- ✓ MySQL migration tool, new pgloader version



Skype Tools for Replication

- ✓ PGQ
- ✓ *londiste*
- ✓ walmgr



PGQ is organized into 3 components

Producer, Consumer, Ticker

Some things did change in PGQ *version 3*

- ✓ The ticker is now `pgqd`
- ✓ The network topology is now known `pgq_nodes`
- ✓ And we have *Cooperative Consumers* `pgq-coop`



Londiste3 concepts

Londiste relies on PGQ nodes

- ✓ Root
- ✓ Branch
- ✓ Leaf
- ✓ Leaf -merge='qname'



Operating londiste

Basic commands

- ✓ `status`
- ✓ `members`
- ✓ `change-provider`
- ✓ `takeover -all -dead`



Londiste Add Table



londiste add-table

Plenty new options in add-table

- ✓ --wait-sync
- ✓ --dest-table
- ✓ --skip-truncate
- ✓ --create, --create-full
- ✓ --trigger-flags
- ✓ --trigger-arg
- ✓ --no-triggers
- ✓ --copy-node,
--copy-condition
- ✓ --merge-all, --no-merge
- ✓ --max-parallel-copy



DDL Handling

- ✓ `londiste conf.ini execute`
- ✓ EXECUTE queue events
- ✓ *System Table* `londiste.applied_execute`
- ✓ SQL meta-data attributes



execute Meta-Data attributes

```
--*--  
--*-- Local-Table: mytable, othertable,  
--*-- thirdtable  
--*-- Local-Sequence: thisseq  
--*--
```

- ✓ *Local-Table* Table must be added to local node with add-table.
- ✓ *Need-Table* Physical table must exist in database. It does not matter if it is replicated or not.

DDL and table renaming

```
--*-- Local-Table: mytable  
ALTER TABLE @mytable@ ...;
```

Londiste Handlers

It's possible to register handlers to deal with specific needs

- ✓ `add --handler [--handler-arg ...]`
- ✓ `show-handlers`
- ✓ `pl/proxy` sharding, re-sharding



Sharding plugging



Handler: shard

- ✓ Event filtering by hash, for partitioned databases.
- ✓ `key=COLUMN` column name to use for hashing
- ✓ `hashfunc=FUNCNAME` function to use for hashing
- ✓ (default: `partconf.get_hash_raw`)
- ✓ `ev_extra3='hash=' || partconf.get_hash_raw(key_column)`

Preparing for part: hashlib

```
> ... add-table pgbench_accounts \  
> --handler=part --handler-arg=key=aid
```

```
psql rootdb < /usr/share/postgresql/8.4/contrib/hashlib.sql  
psql sharddb_0 < /usr/share/postgresql/8.4/contrib/hashlib.sql  
psql sharddb_1 < /usr/share/postgresql/8.4/contrib/hashlib.sql
```

```
> psql rootdb -c 'create extension hashlib;'  
> psql sharddb_0 -c 'create extension hashlib;'  
> psql sharddb_1 -c 'create extension hashlib;'
```



Preparing for part: setup

```
CREATE SCHEMA partconf;
```

```
CREATE TABLE partconf.conf (  
    part_nr integer,  
    max_part integer,  
    db_code bigint,  
    is_primary boolean,  
    max_slot integer,  
    cluster_name text  
);
```



Preparing for part: get_hash_raw

```
CREATE FUNCTION partconf.get_hash_raw  
  ( i_input integer)  
  RETURNS integer  
  LANGUAGE sql
```

```
AS $$
```

```
-- used to wrap hashtext so that we can replace it in 8.4  
-- with older implementation to keep compatibility
```

```
select hash_string(\$1::text, 'lookup2');
```

```
$$;
```

Preparing for part: local_part

```
> psql rootdb < partconf.sql
> psql shardedb_0 < partconf.sql
> psql shardedb_1 < partconf.sql

> psql shardedb_0
=> insert into partconf.conf(part_nr, max_part)
      values (0,1);

> psql shardedb_1
=> insert into partconf.conf(part_nr, max_part)
      values (1,1);
```



Preparing for part: add-table

- > `londiste3 st3partsplit/st3_rootdb.ini`
`add-table pgbench_accounts`
`--handler=part --handler-arg=key=aid`
- > `londiste3 st3partsplit/st3_sharddb_0.ini`
`add-table pgbench_accounts --create`
`--handler=part --handler-arg=key=aid`
- > `londiste3 st3partsplit/st3_sharddb_1.ini`
`add-table pgbench_accounts --create`
`--handler=part --handler-arg=key=aid`



Other Handlers

- ✓ applyfn
- ✓ bulk
- ✓ dispatch
- ✓ multimaster

- ✓ part
- ✓ qtable
- ✓ vtable

Handler: bulk

bulk loading with 3 options

- ✓ correct: COPY, COPY temp + UPDATE, COPY temp + DELETE
- ✓ delete: correct + update devient DELETE + COPY
- ✓ merged: merge insert rows with update rows



Handler: dispatch

`bulk_monthly_batch`

- ✓ bulk
- ✓ hourly daily monthly yearly
- ✓ event batch field time



Handler: dispatch

- ✓ `table_mode` part, direct, ignore
- ✓ `part_mode` batch_time, event, time, date_field, current_time
- ✓ `part_field` date_field
- ✓ `period` hour, day, month, year
- ✓ `row_mode` plain, keep_latest, keep_all
- ✓ `event_types` I,U,D
- ✓ `load_mode` direct, bulk
- ✓ `method` correct, delete, merged, insert
- ✓ `fields` field name mapping, no COPY support
- ✓ `skip_fields`
- ✓ `table`
- ✓ `pre_part`, `post_part`
- ✓ `encoding`
- ✓ `analyze`

Extras

- ✓ check
- ✓ fkeys
- ✓ compare
- ✓ repair

- ✓ wait-sync
- ✓ wait-provider
- ✓ wait-root



Questions?

Now is the time to ask!

