PGQ Cooperative Consumers

Dimitri Fontaine & Marko Kreen

Database processing oriented batches

If you’re managing an *OLTP* system, you probably have out of line processing to get done, and probably are using cron batches and home made *daemons*.

Example

```python
while True:
    // what a nice daemon!
```

Of course you want them

- reliable, easy to monitor and control (logs)
- out of a developer screen session
- easy to stop & restart
- to reuse existing models
If you’re managing an *OLTP* system, you probably have out of line processing to get done, and probably are using cron batches and home made *daemons*.

**Example**

```python
while True:
    # what a nice daemon!
```

Of course you want them
- reliable, easy to monitor and control (logs)
- out of a developer screen session
- easy to stop & restart
- to reuse existing models
If you’re managing an *OLTP* system, you probably have out of line processing to get done, and probably are using cron batches and home made *daemons*.

**Example**

```python
while True:
    // what a nice daemon!
```

Of course you want them

- reliable, easy to monitor and control (logs)
- out of a developer screen session
- easy to stop & restart
- to reuse existing models
Queueing with *PGQ*

*Off-line* processing is better done with PGQ

- Mainly written in PLpgSQL (and C)
- Client *API* for python
- and PHP
- some work is happening for Java
- **Cooperative Worker** (Skytools 3)

PGQ: Stable, Reliable, Easy to monitor
PGQ SQL API

PGQ is bundled as a PostgreSQL Extension, with a nice API for you to use. Let’s produce some events:

- CREATE EXTENSION pgq;
- select pgq.create_queue('LogEvent');
- select pgq.insert_event('LogEvent', 'data', 'DataFor123');

Don’t forget to run the ticker daemon: pgqd
PGQ SQL API

PGQ is bundled as a PostgreSQL Extension, with a nice API for you to use. Let’s produce some events:

- CREATE EXTENSION pgq;
- select pgq.create_queue('LogEvent');
- select pgq.insert_event('LogEvent', 'data', 'DataFor123');

Don’t forget to run the ticker daemon: pgqd
**PGQ Consumer**: each consumer sees all the batches.

- `select pgq.register_consumer('LogEvent', 'TestConsumer');`
- `select pgq.next_batch('LogEvent', 'TestConsumer'); [into batch_id]`
- `select * from pgq.get_batch_events(batch_id);`
- `select pgq.finish_batch(batch_id)`
**PGQ Consumer**: each consumer sees all the batches.

- `select pgq.register_consumer('LogEvent', 'TestConsumer');`
- `select pgq.next_batch('LogEvent', 'TestConsumer'); [into batch_id]`
- `select * from pgq.get_batch_events(batch_id);`
- `select pgq.finish_batch(batch_id)`
Monitoring PGQ Consumer

- `consumer_acte_access_accept lag`
  - Current: 687.21m
  - Minimum: 215.71m
  - Average: 184.06
  - Maximum: 5.78k

- `consumer_acte_access_accept last seen`
  - Current: 379.19m
  - Minimum: 31.09m
  - Average: 165.16
  - Maximum: 5.77k

Last update: Tue Oct 7 23:35:06 2008
PGQ Cooperative Consumers share the load in between them.

- `select pgq_coop.register_subconsumer('testqueue', 'maincons', 'subcons1');`
- `select pgq_coop.next_batch('testqueue', 'maincons', 'subcons1');`
- `select * from pgq.get_batch_events(batch_id);`
- `select pgq_coop.finish_batch(batch_id)`
PGQ Cooperative Consumers share the load in between them.

- `select pgq_coop.register_subconsumer('testqueue', 'maincons', 'subcons1');`
- `select pgq_coop.next_batch('testqueue', 'maincons', 'subcons1');`
- `select * from pgq.get_batch_events(batch_id);`
- `select pgq_coop.finish_batch(batch_id)`
Monitoring Cooperative Consumers

![Graph showing PgQ fotolog q_photo consumer lag - by day]

- pgq_photo lag
  - Cur: 27.40
  - Min: 0.00
  - Avg: 14.24
  - Max: 44.38
- pgq_photo last seen
  - Cur: 26.82
  - Min: 0.00
  - Avg: 12.23
  - Max: 26.45
- pgq_photo_worker_photo_1 lag
  - Cur: 30.08
  - Min: 0.00
  - Avg: 20.70
  - Max: 125.73
- pgq_photo_worker_photo_1 last seen
  - Cur: 2.06
  - Min: 0.00
  - Avg: 1.09
  - Max: 2.06
- pgq_photo_worker_photo_2 lag
  - Cur: 27.38
  - Min: 0.00
  - Avg: 1.95
  - Max: 21.26
  - 127.25
- pgq_photo_worker_photo_2 last seen
  - Cur: 1.05
  - Min: 0.00
  - Avg: 2.64
  - Max: 4.69
- pgq_photo_worker_photo_3 lag
  - Cur: 30.07
  - Min: 0.00
  - Avg: 1.74
  - Max: 20.22
  - 107.43
- pgq_photo_worker_photo_3 last seen
  - Cur: -nan
  - Min: 684.95m
  - Avg: 684.95m
  - Max: 684.95m
- pgq_photo_worker_photo_4 lag
  - Cur: 56.82
  - Min: 0.00
  - Avg: 22.45
  - Max: 161.86
- pgq_photo_worker_photo_4 last seen
  - Cur: 1.45
  - Min: 0.00
  - Avg: 1.45
  - Max: 2.02
  - 2.58

Last update: Sun May 13 20:25:29 2012

Munin 1.4.5