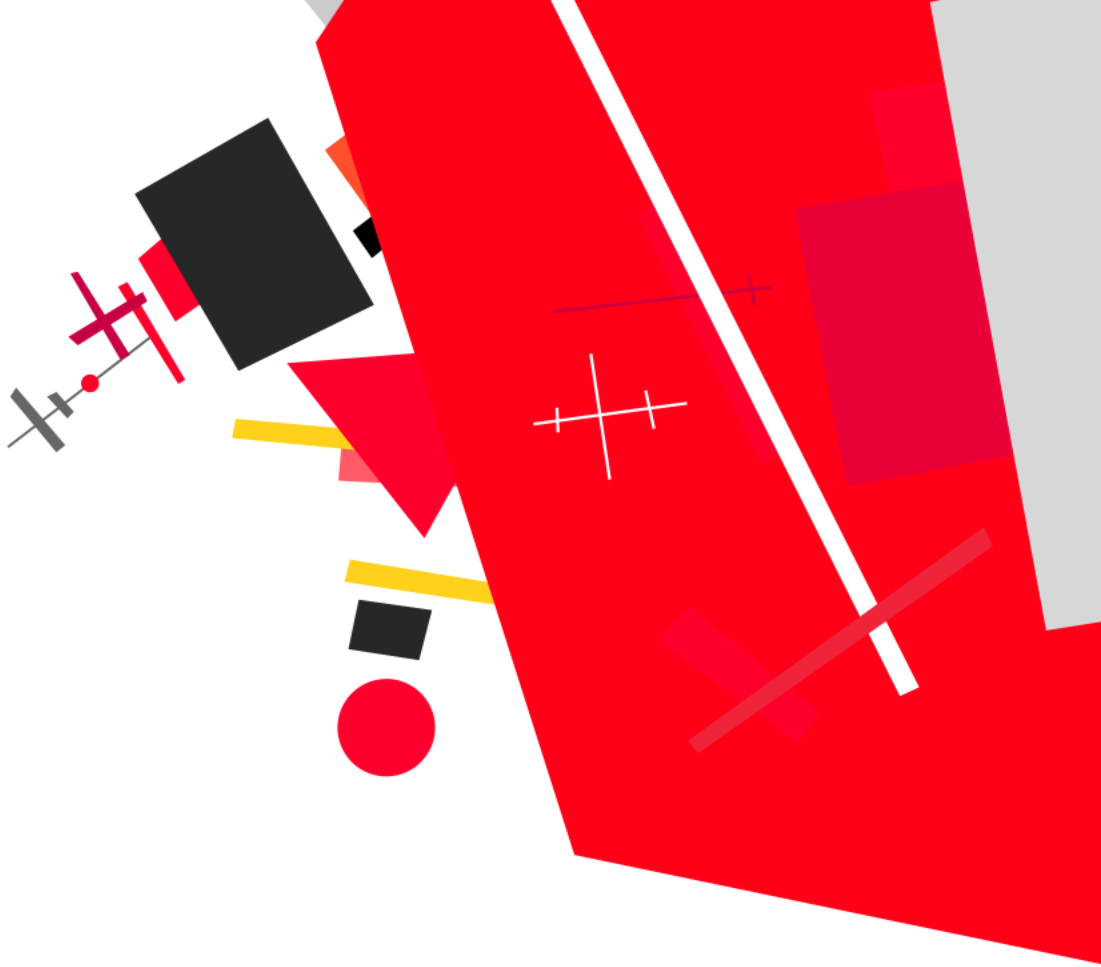
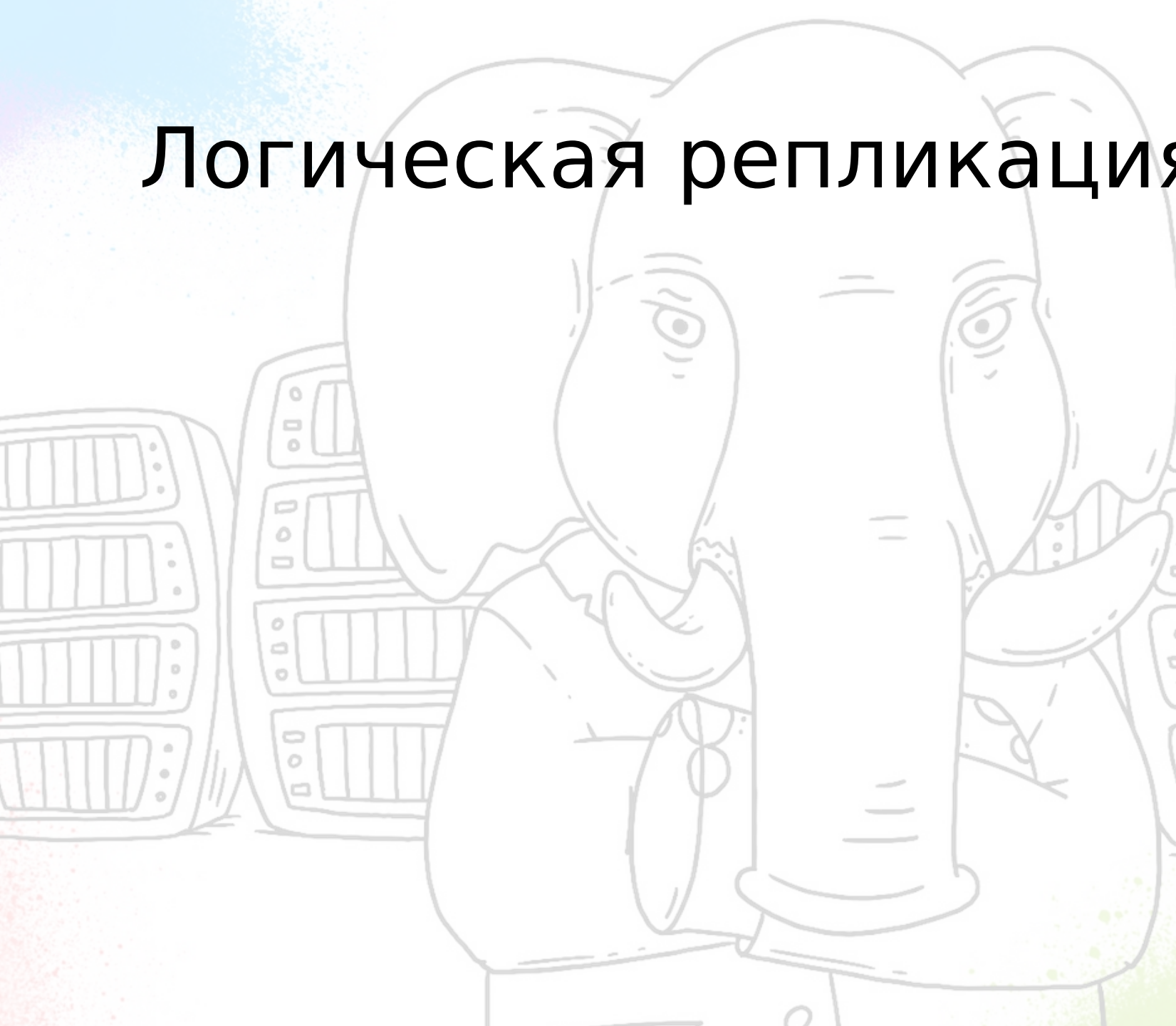
Логическая репликация и Avito

Костантин Евтеев
Михаил Тюрин
Сергей Бурладян



HighLoad++
Профессиональная конференция разработчиков высоконагруженных систем

Логическая репликация



Гибкость хранения

Event tracking

Управление доступом

Масштабирование

Интеграция

Обновление

Логическая репликация в Авито

Building data streams (PGDAY 2016) goo.gl/1ZT4Bb

Поток данных в Авито (PgConf.Russia 2016) goo.gl/mwhwya

Доставка справочников

Распределение нагрузки

Partial replication в сервисы

Доставка данных в поисковые системы

Persistent queue

Межсервисное взаимодействие

История

1999: PostgreSQL 6.5.3 MVCC

История

1999: PostgreSQL 6.5.3 MVCC

2000: Rep script

История

1999: PostgreSQL 6.5.3 MVCC

2000: Rep script

2001: PostgreSQL 7.1: write-ahead log

История

1999: PostgreSQL 6.5.3 MVCC

2000: Rep script

2001: PostgreSQL 7.1: write-ahead log

2004: Slony

История

1999: PostgreSQL 6.5.3 MVCC

2000: Rep script

2001: PostgreSQL 7.1: write-ahead log

2004: Slony

2005: PostgreSQL 8.0: point-in-time recovery

История

1999: PostgreSQL 6.5.3 MVCC

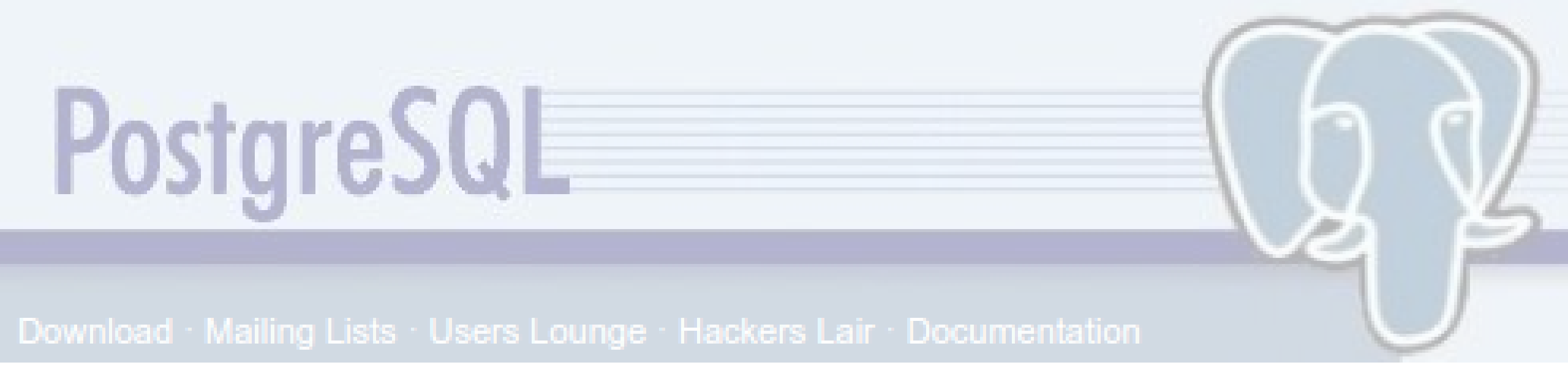
2000: Rep script

2001: PostgreSQL 7.1: write-ahead log

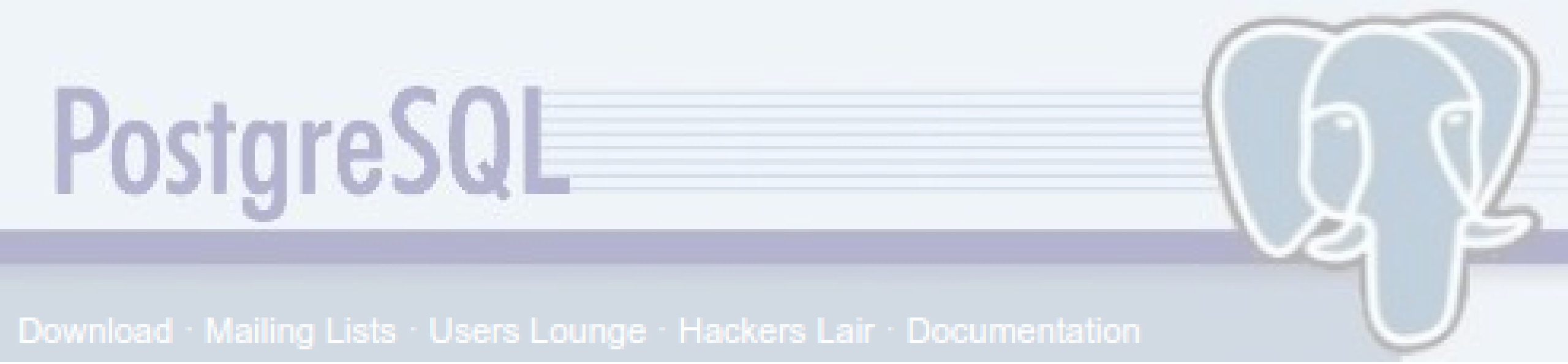
2004: Slony

2005: PostgreSQL 8.0: point-in-time recovery

2007 PgQ, Londiste

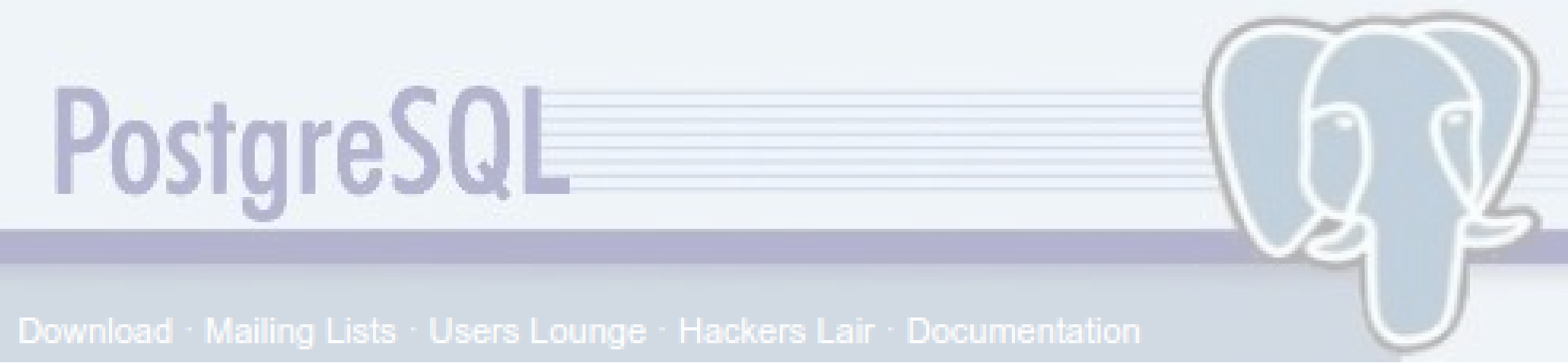


2008: 8.3: standby



2008: 8.3: standby

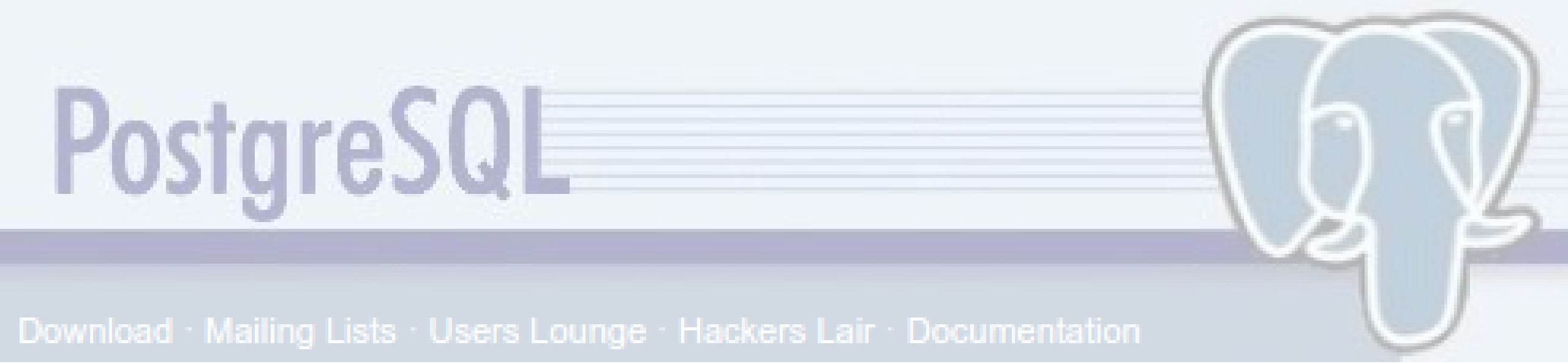
2010: 9.0: hot standby, streaming replication



2008: 8.3: standby

2010: 9.0: hot standby, streaming replication

2013: 9.3: sb can follow timeline switch

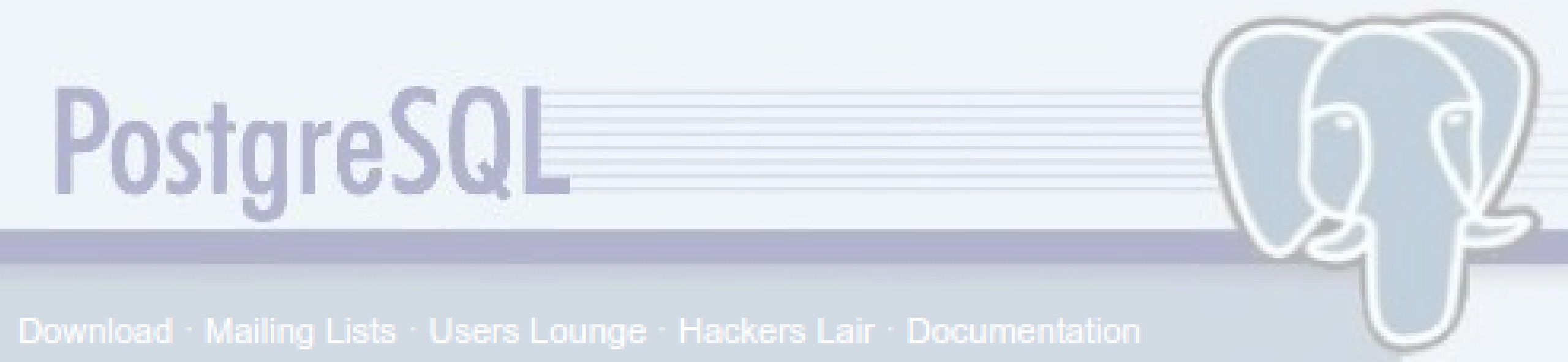


2008: 8.3: standby

2010: 9.0: hot standby, streaming replication

2013: 9.3: sb can follow timeline switch

2014: 9.4: replication slots, logical decoding



2008: 8.3: standby

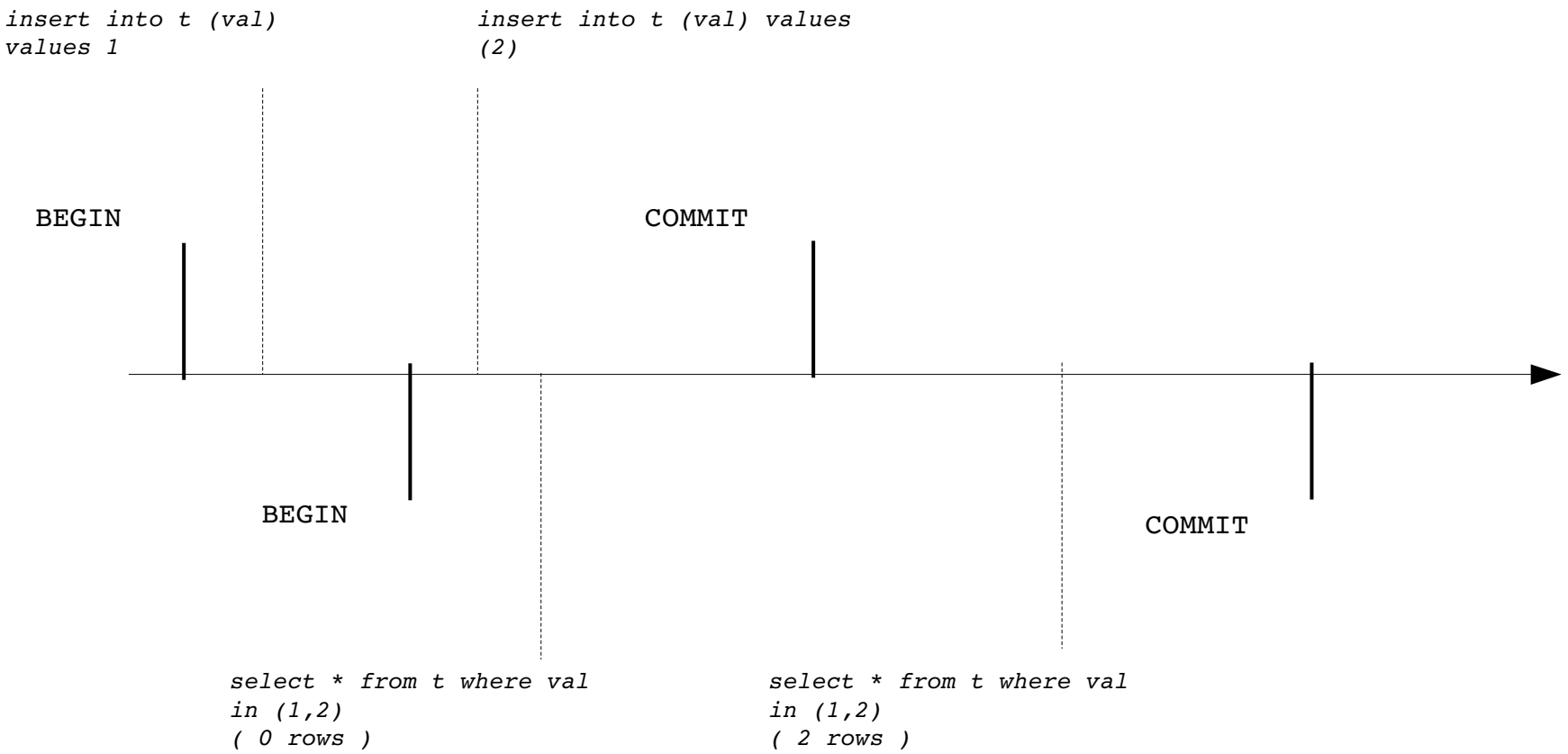
2010: 9.0: hot standby, streaming replication

2013: 9.3: sb can follow timeline switch

2014: 9.4: replication slots, logical decoding

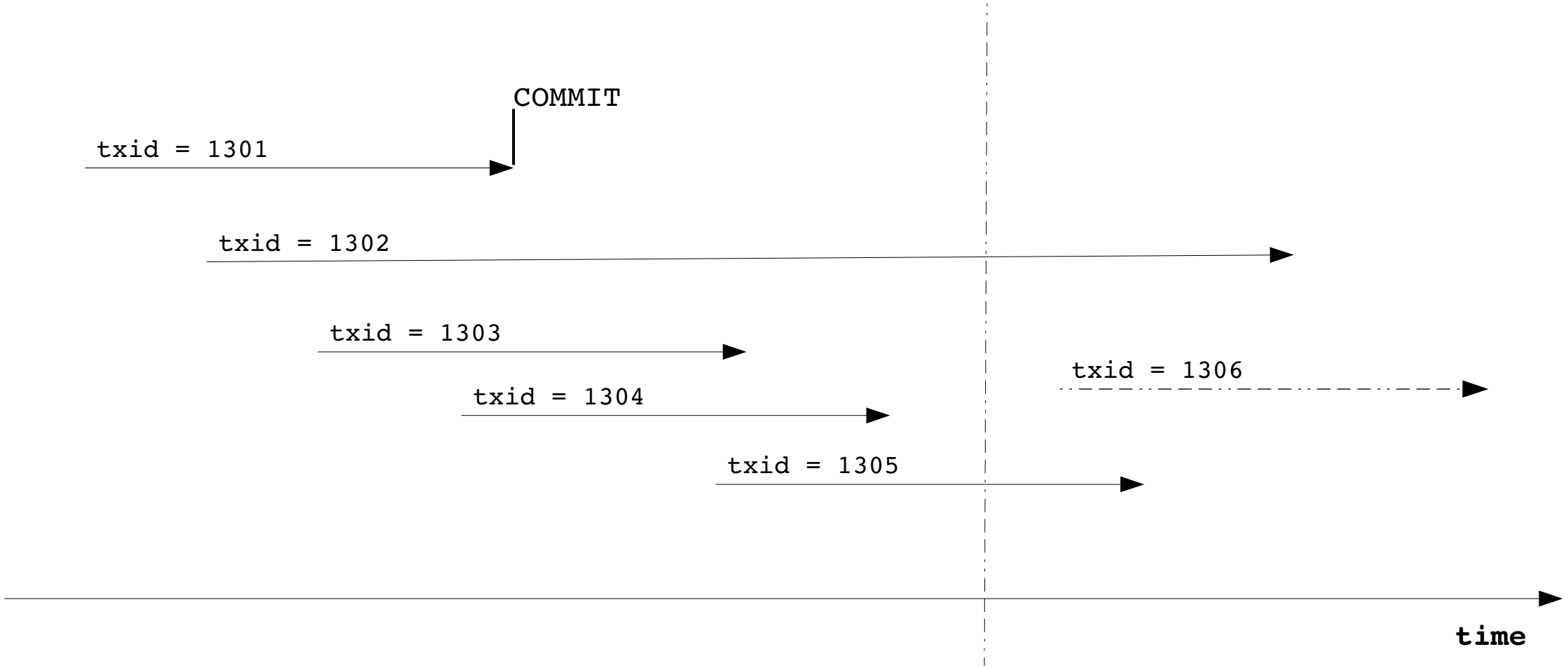
2017: 10: logical replication

MVCC



MVCC

Snapshot
xmin = 1302
xmax = 1306 (1305 +1)
xip = xip = [1302, 1305]



Rep script Вадима Михеева

```
snprintf(sql, 8192, "insert into _RSERV_SYNC_ "
          "(server, syncid, synctime, status, minid, maxid, active) "
  "values (%u, currval('_rserv_sync_seq_'), now(), 0, %d, %d, '%s')",
          server, SerializableSnapshot->xmin, SerializableSnapshot->xmax, active);
```

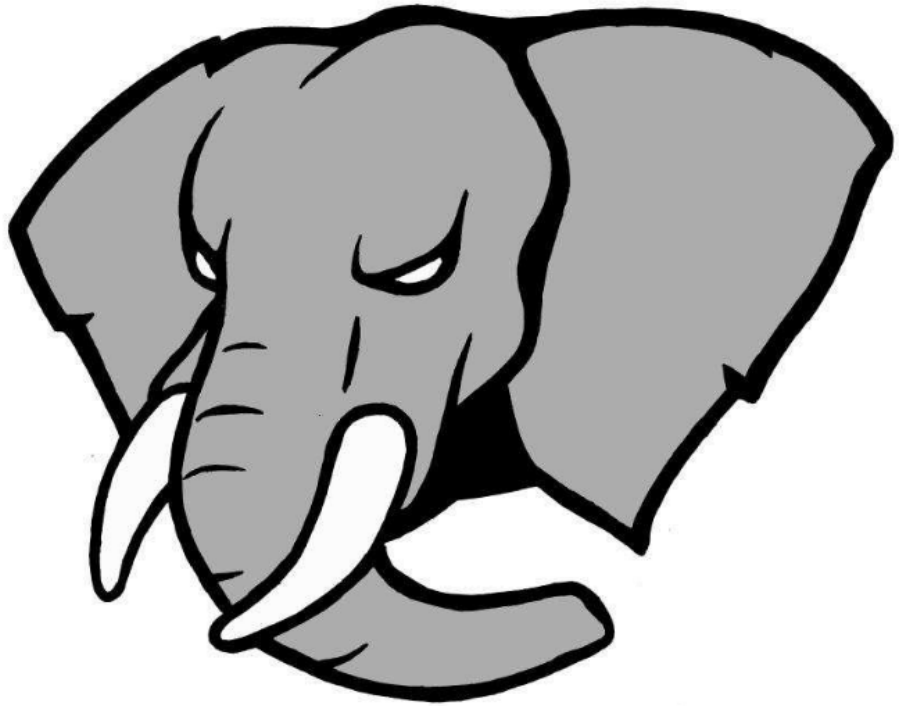
```
# Read last succeeded sync
my $sql = "select syncid, synctime, minid, maxid, active from _RSERV_SYNC_" .
" where server = $server AND syncid = (select max(syncid) from" .
" _RSERV_SYNC_ where server = $server AND status > 0)";
```

```
my $sinfo = "";
if (@lastsync && $lastsync[3] ne '')    # sync info
{
$sinfo = "and (l.logid >= $lastsync[3]";
$sinfo .= " or l.logid in ($lastsync[4])" if $lastsync[4] ne '';
$sinfo .= ")";
}

my $havedeal = 0;

# DELETED rows
$sql = "select l.reloid, l.key from _RSERV_LOG_ l" .
" where l.delete = 1 $sinfo order by l.reloid";
```

Slony



Slonik

Slon

Slony Schema

- конфигурация и текущее состояние в таблицах бд
- Процедуры, которые вызывают slon/slonik

Triggers запись в очередь

Идеи

Vadim Mikheev (rserv)

We can export internal Postgres visibility info (trancaction id /snapshot).

Идеи

Верхний ряд: Thomas Lockhart, Jan Wieck, Tom Lane, Marc Fournier

Нижний ряд: Vadim Mikheev, Bruce Momjian

Vadim Mikheev (rserv)

We can export internal Postgres visibility info (trancaction id /snapshot).

https://www.postgresql.org/files/developer/history.pdf

Идеи

Jan Wieck (Slony-I)

If we have 2 snapshots, we can query events that happened between them.

“Agreeable order” - order taken from sequence in AFTER trigger

Идеи

Верхний ряд: Thomas Lockhart, Jan Wieck, Tom Lane, Marc Fournier

Нижний ряд: Vadim Mikheev, Bruce Momjian

Jan Wieck (Slony-I)

If we have 2 snapshots, we can query events that happened between them.

“Agreeable order” - order taken from sequence in AFTER trigger

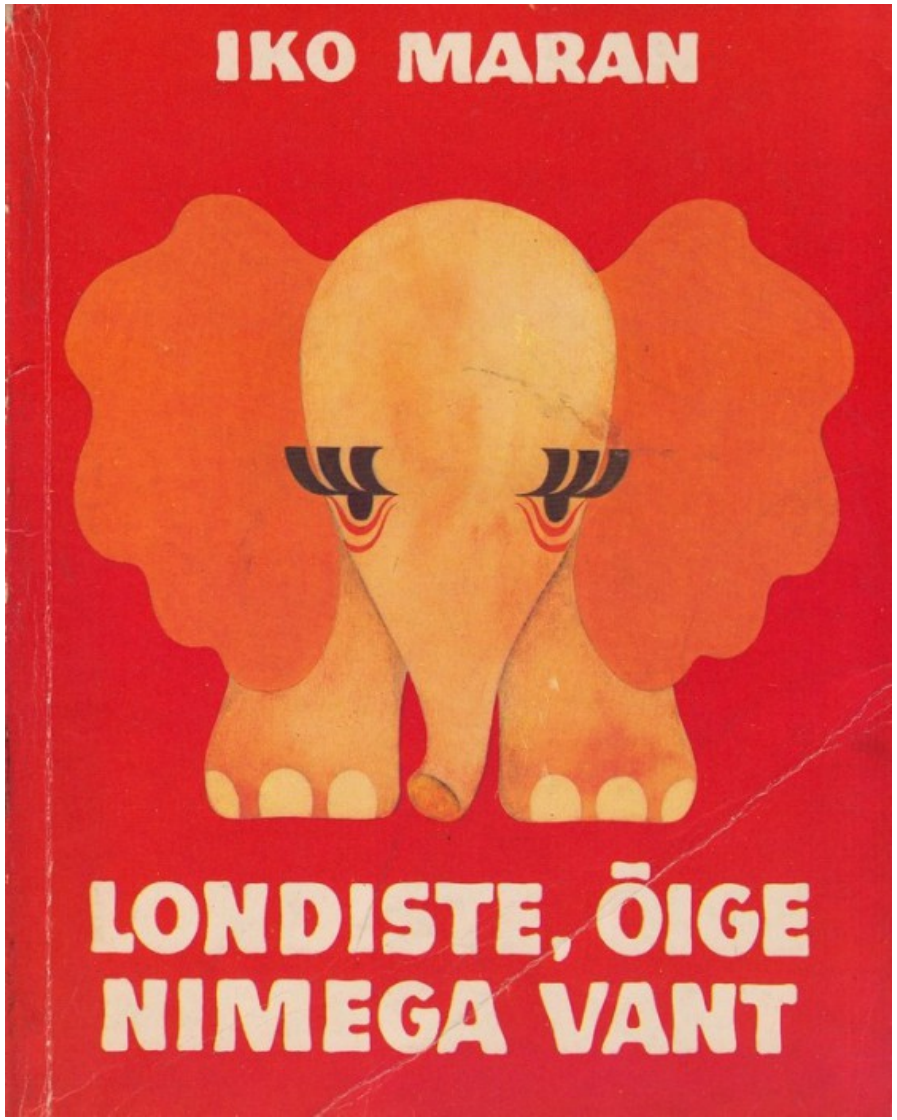
https://www.postgresql.org/files/developer/history.pdf

Postgres специфичное решение, на основе MVCC

Transaction IDs (txid) назначаются последовательно

Транзакции могут быть открыты различное время, и их результаты должны быть невидимы в этот момент

Snapshot представляет собой точку во времени и он разделяет txids, результаты которых доступны и те, которые продолжают выполняться

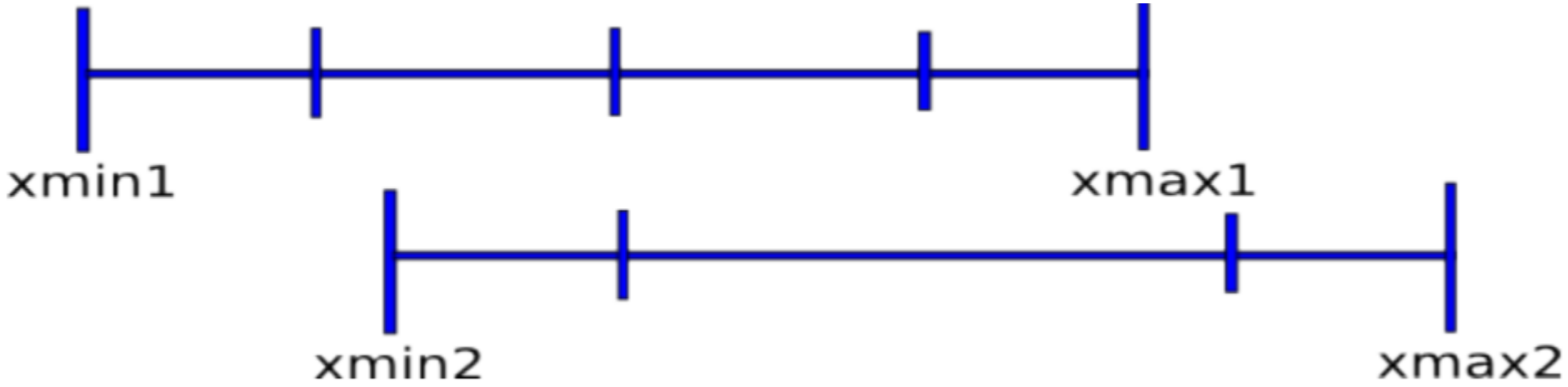


PGQ, Londiste

Xmin – lowest transaction ID in progress
Xmax – first unassigned transaction ID
Xip – list of transaction Ids in progress

txid_visible_in_snapshot(txid, snap) =
txid < snap.xmin
OR
(txid < snap.xmax AND txid NOT IN (snap.xip))

https://wiki.postgresql.org/wiki/SkyTools

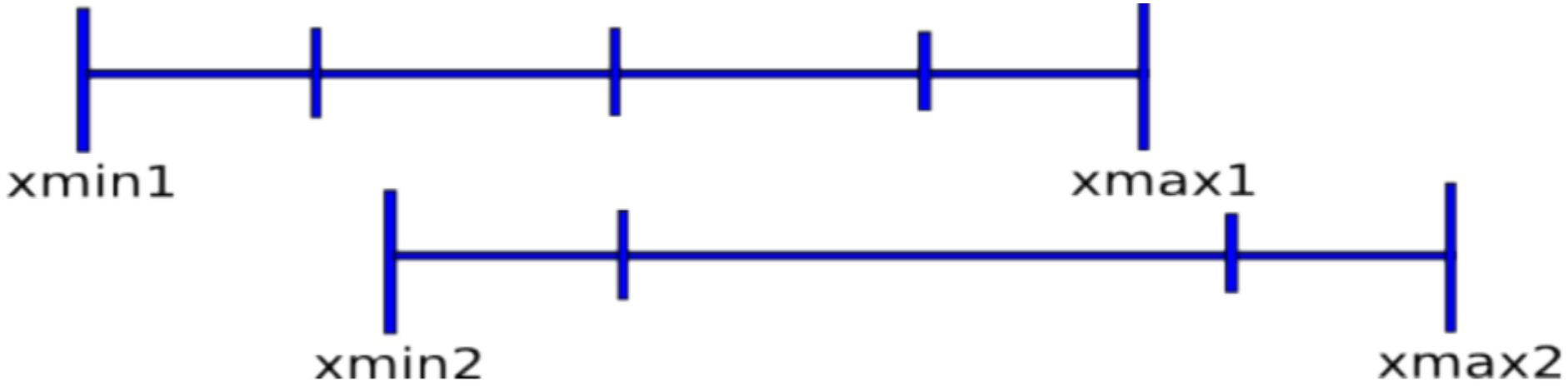


Snapshot 1: xmin1, xmax1, xip1

Snapshot 2: xmin2, xmax2, xip2

```
select * from queue q
where ... and
   not is_visible(q.ev_txid, snap1)
   and is_visible(q.ev_txid, snap2)
```

https://wiki.postgresql.org/wiki/SkyTools



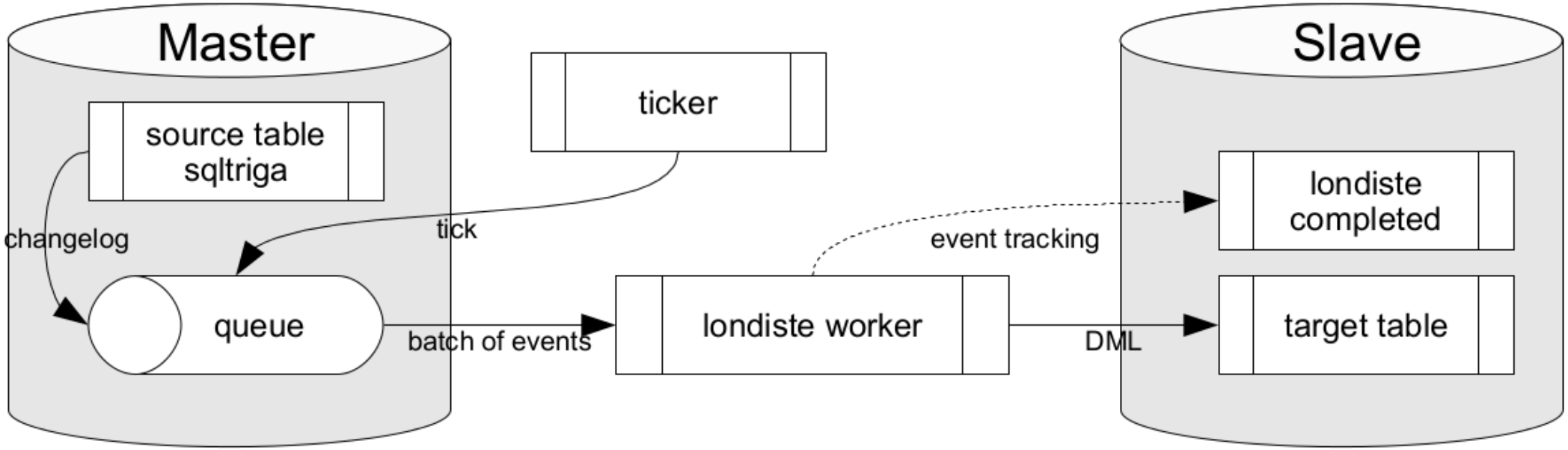
Snapshot 1: xmin1, xmax1, xip1

Snapshot 2: xmin2, xmax2, xip2

```
select * from queue q
where q.ev_txid between xmin1 and xmax2 and
  not is_visible(q.ev_txid, snap1)
   and is_visible(q.ev_txid, snap2)
```

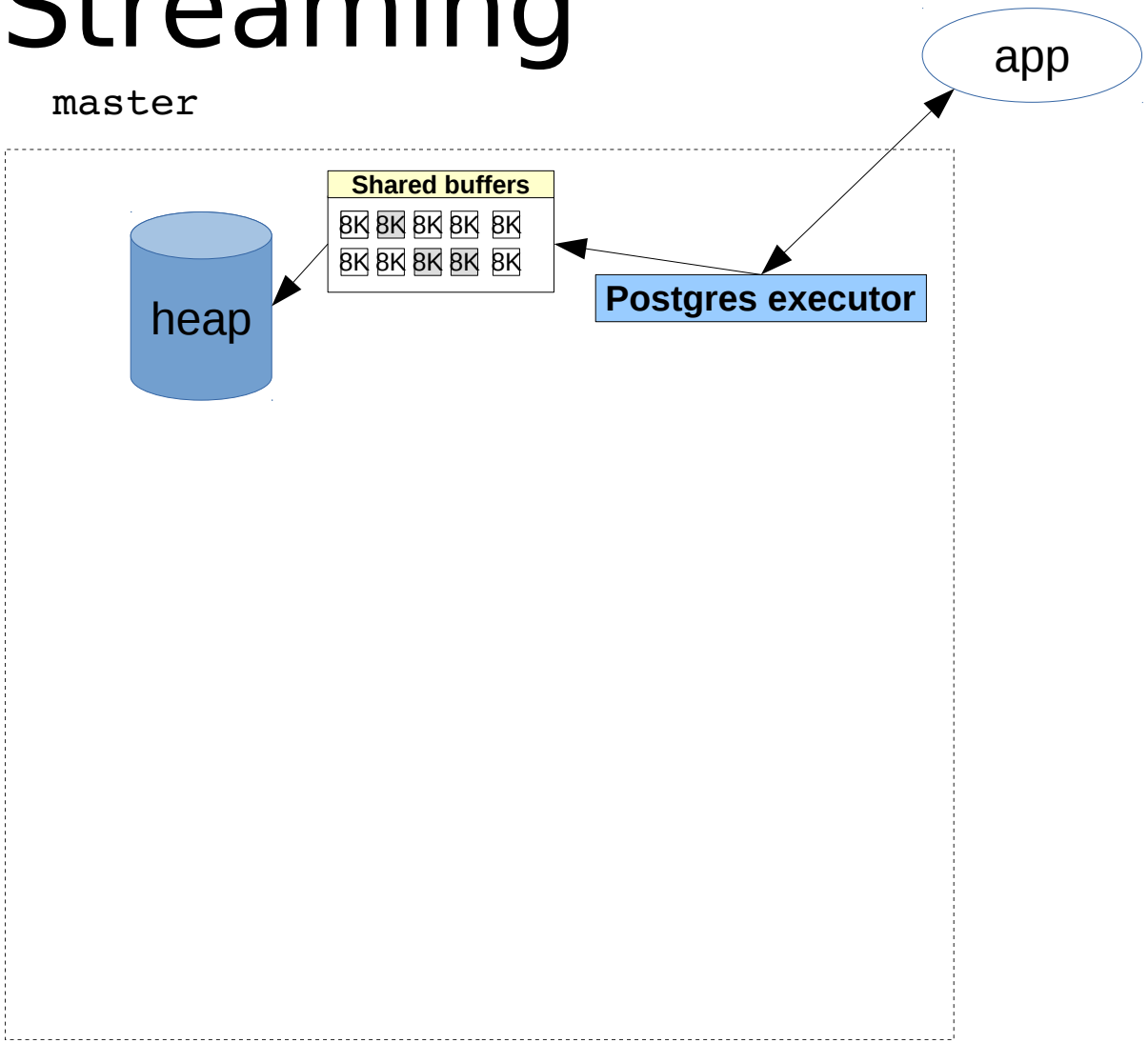
Londiste

https://www.pgcon.org/2009/schedule/attachments/91_pgq.pdf

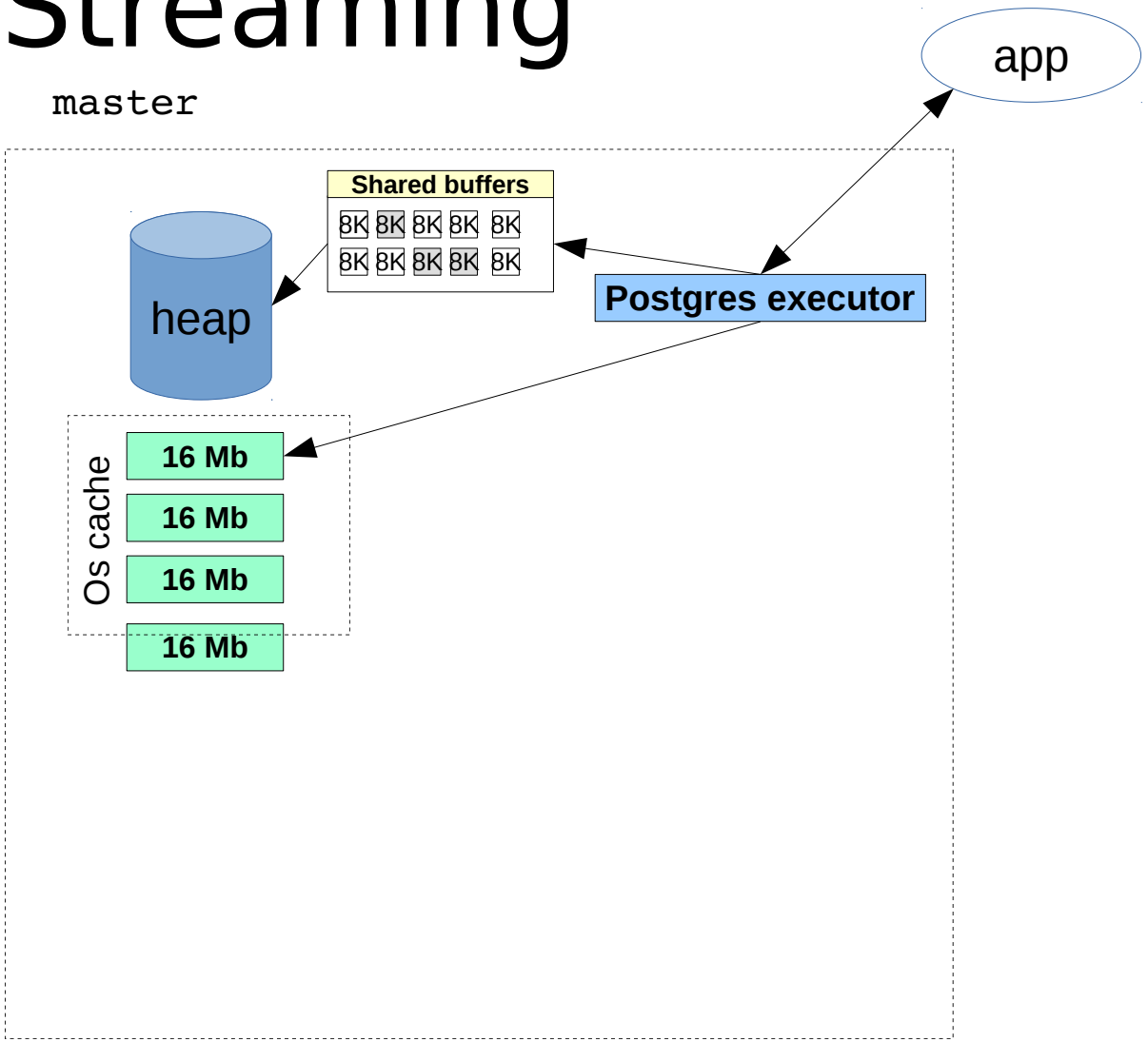


Streaming

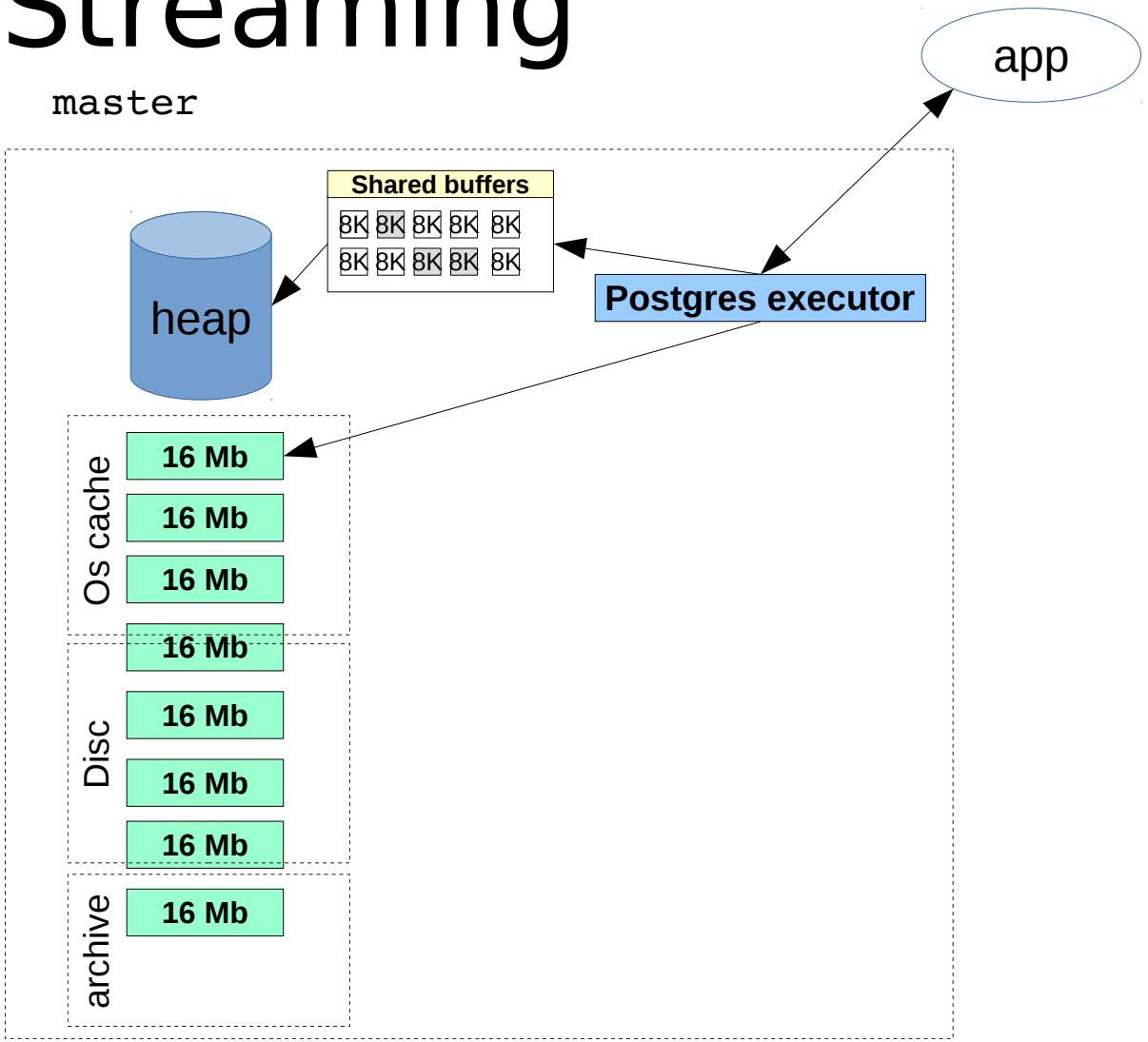
master



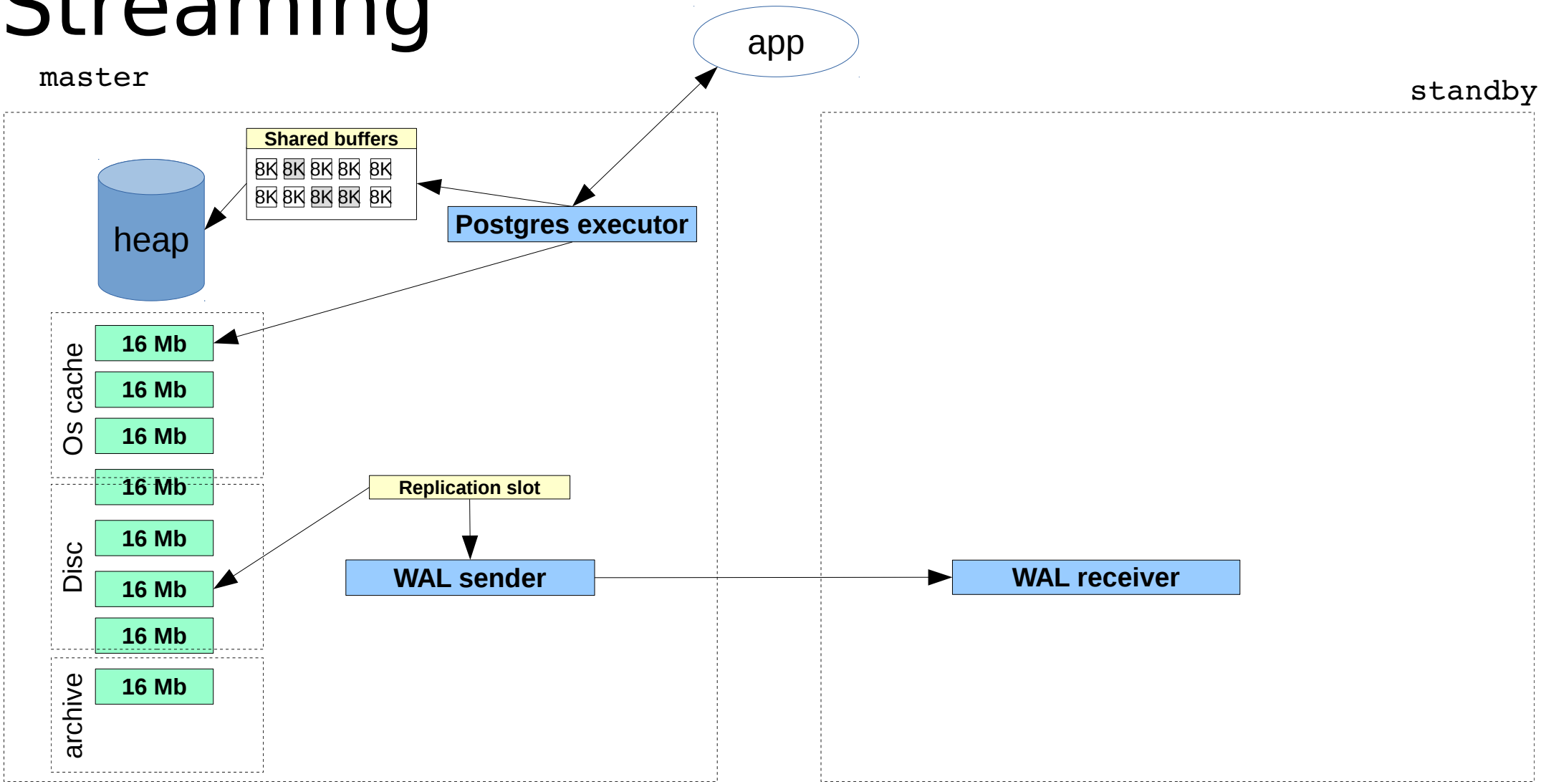
Streaming



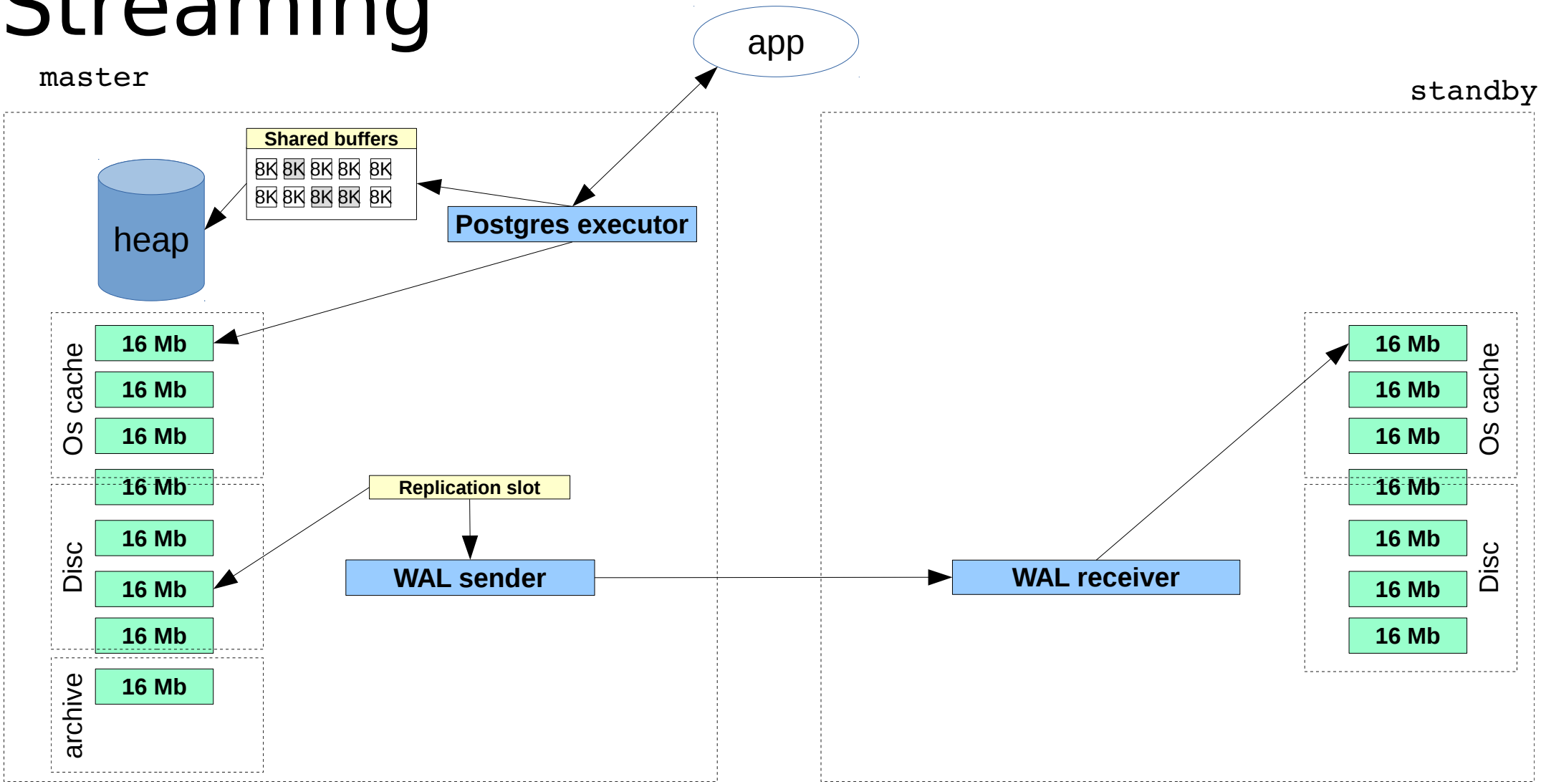
Streaming



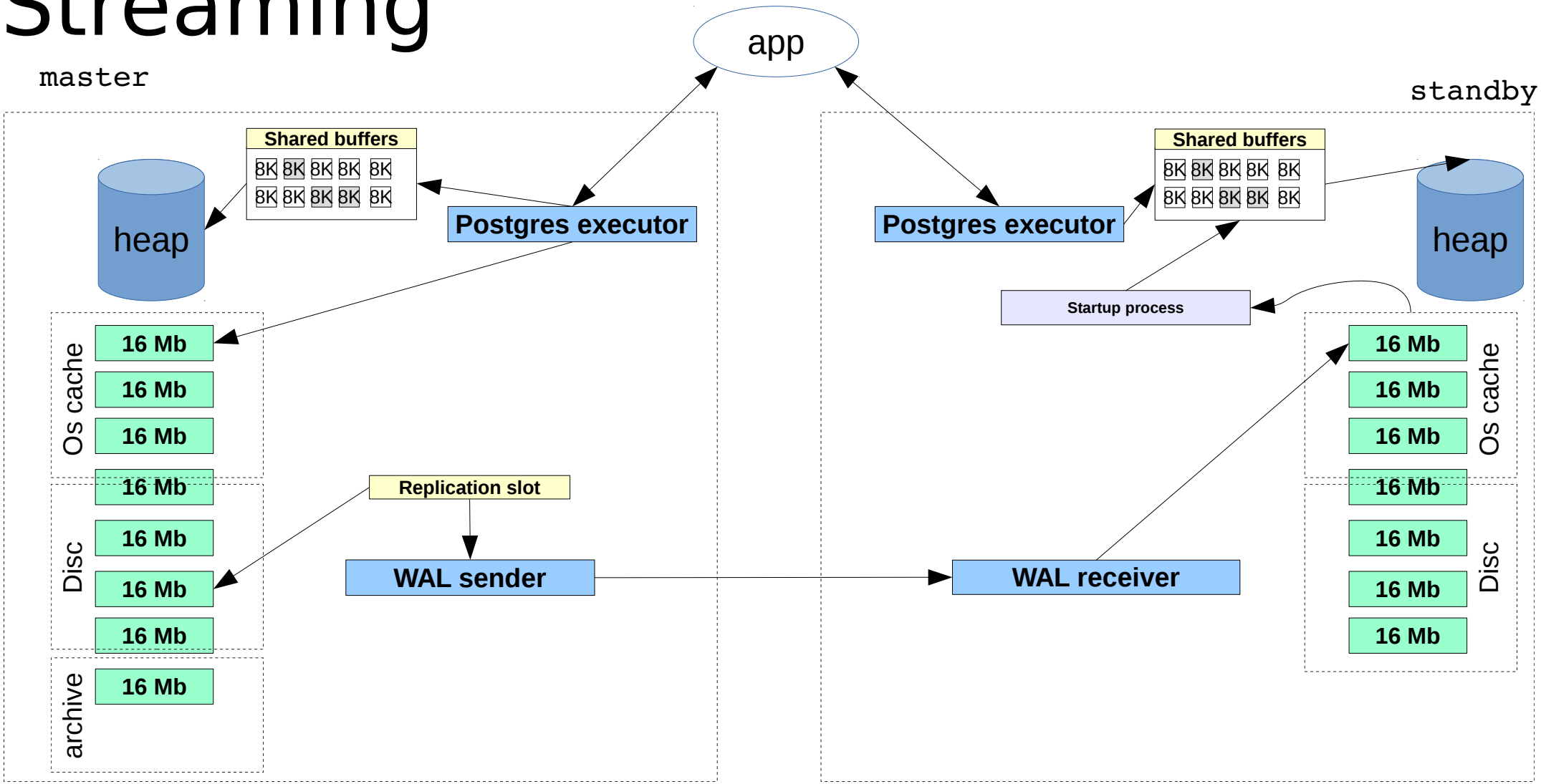
Streaming



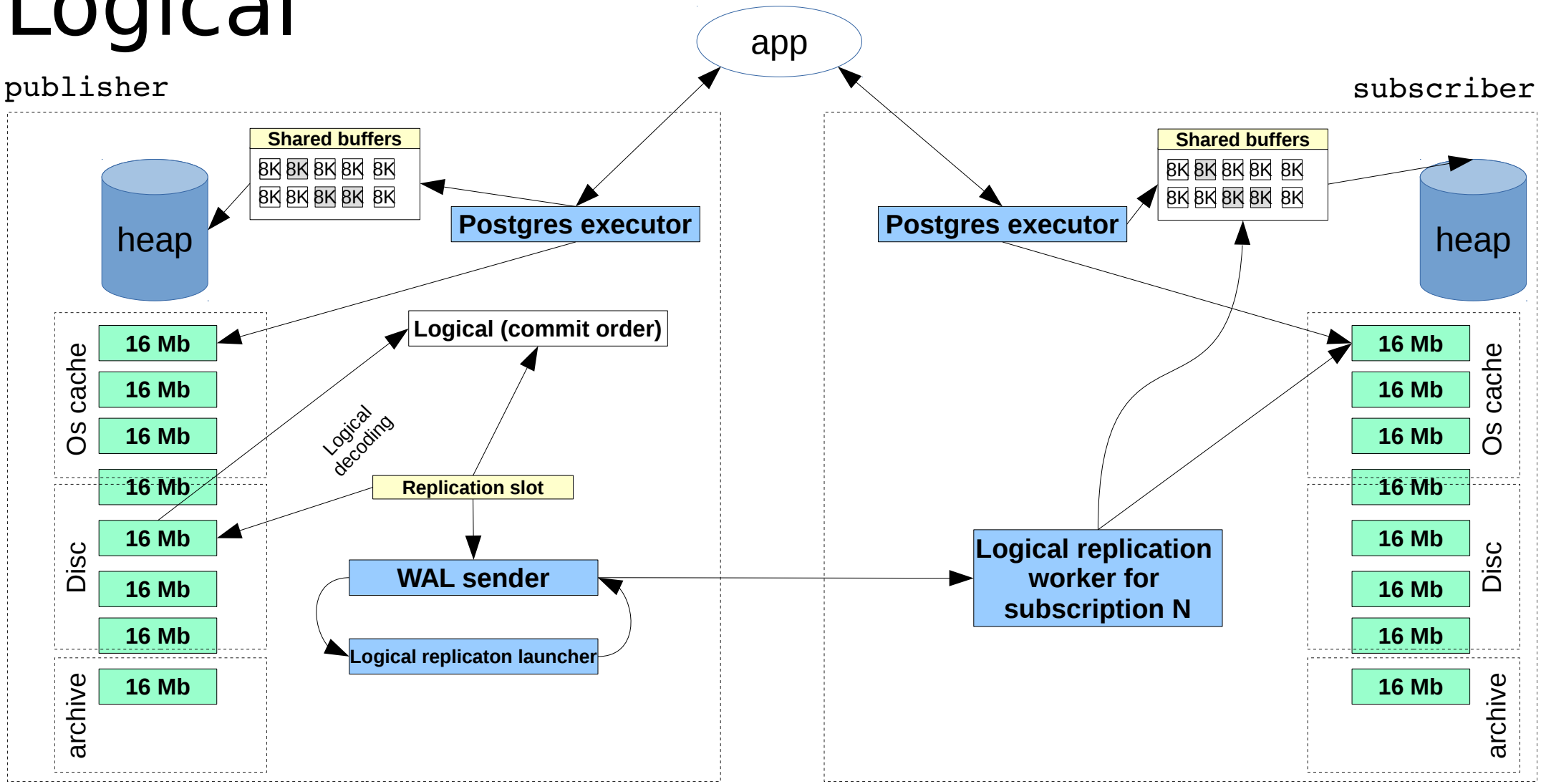
Streaming



Streaming

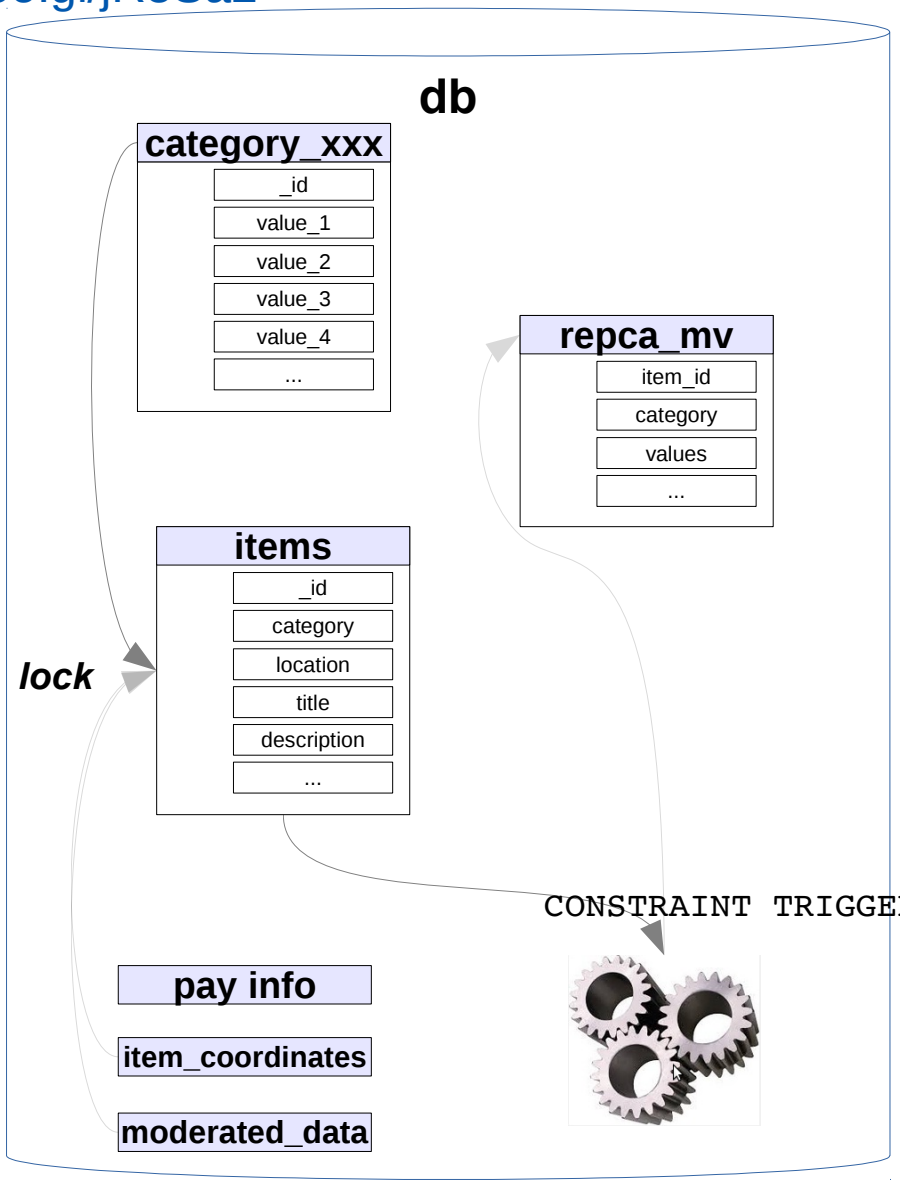


Logical

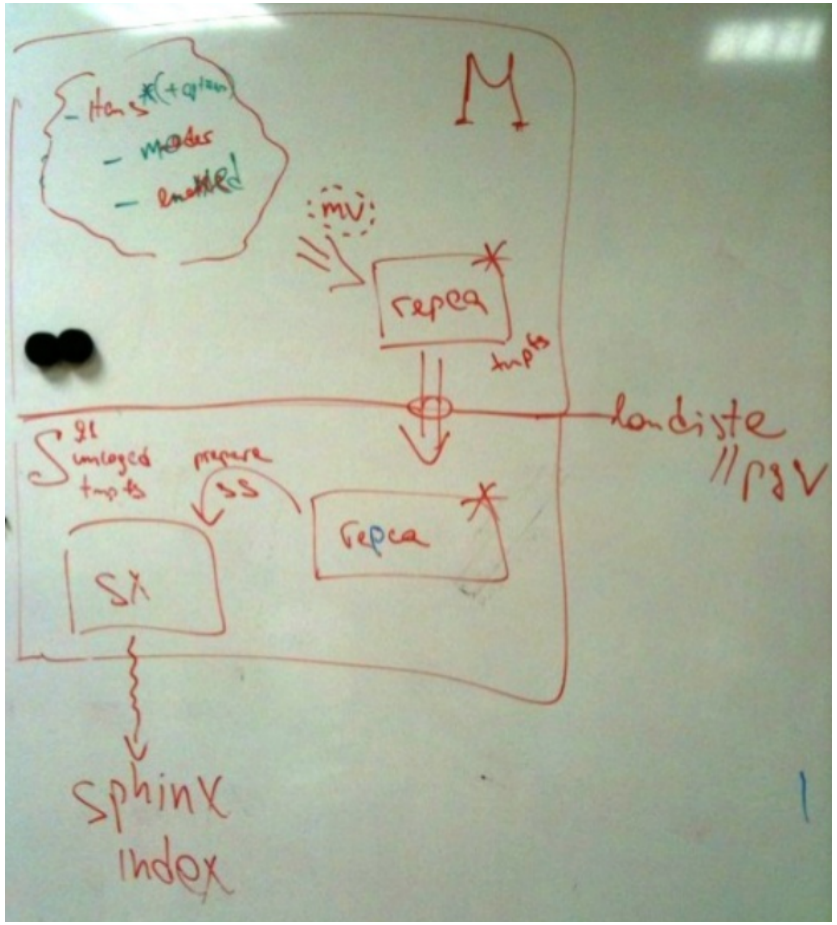


MatView

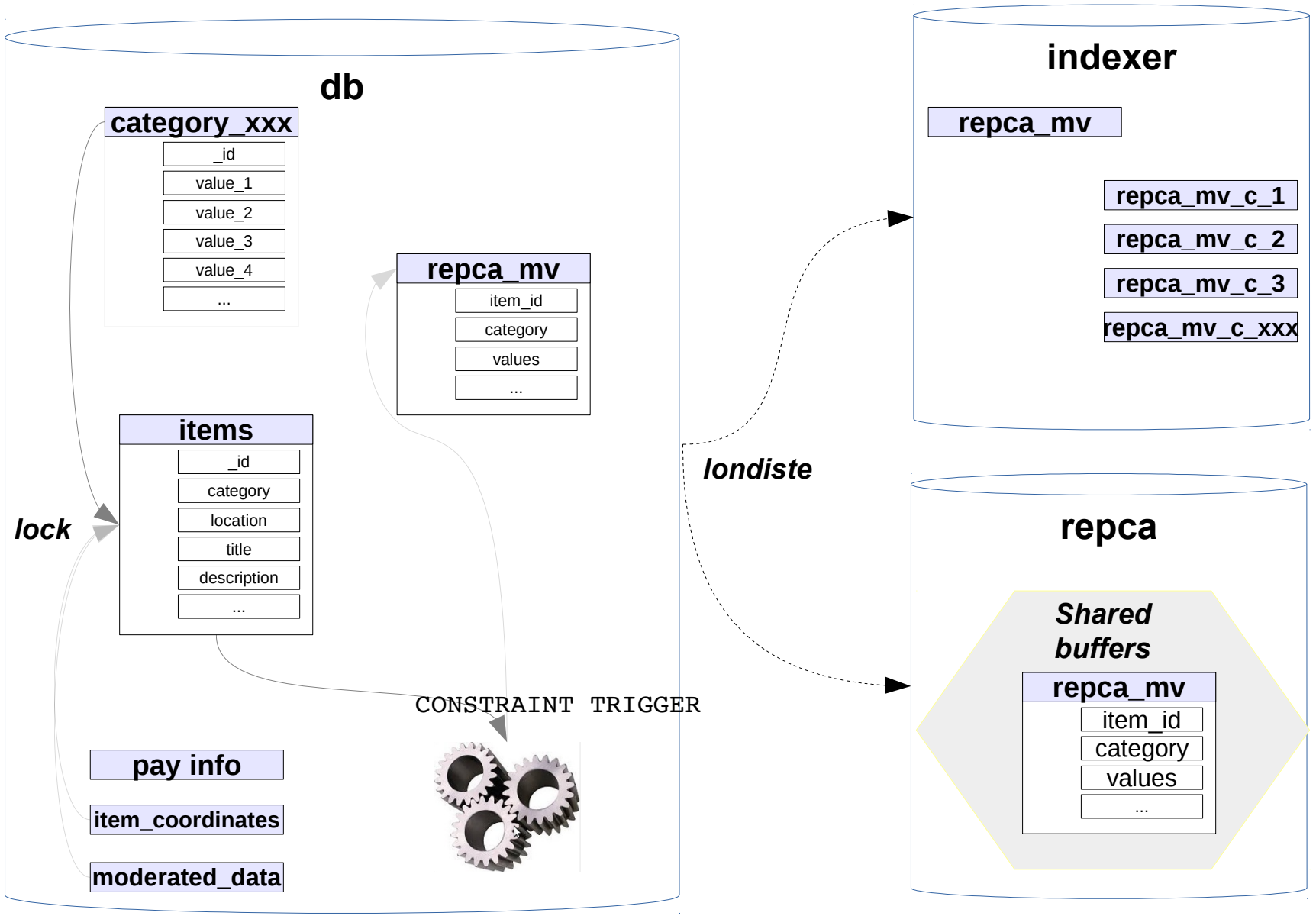
https://goo.gl/a6Z3gE
https://goo.gl/jR5Sa2



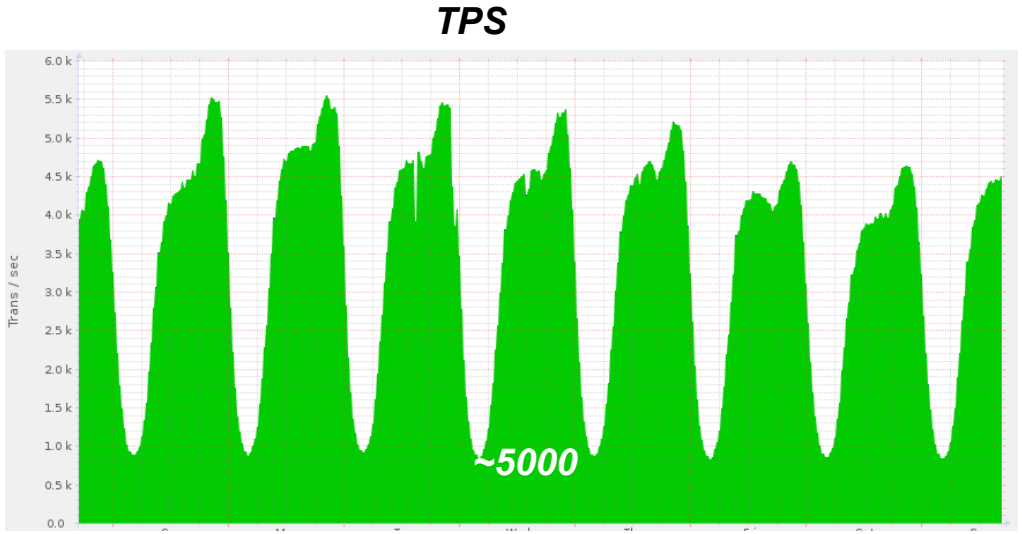
https://goo.gl/sXABqK (19 слайд)
https://pgconf.ru/media2015c/tyurin.pdf (11 слайд)

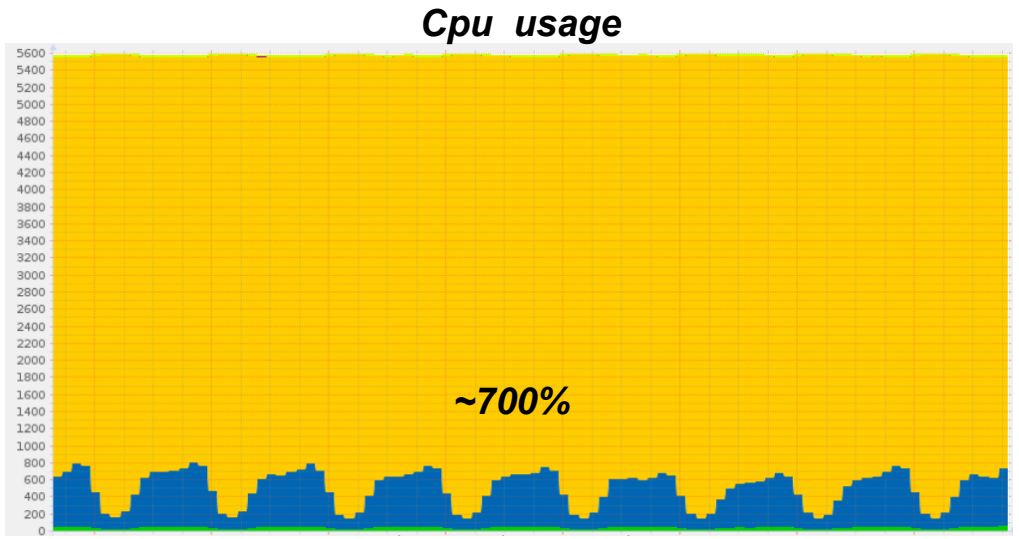


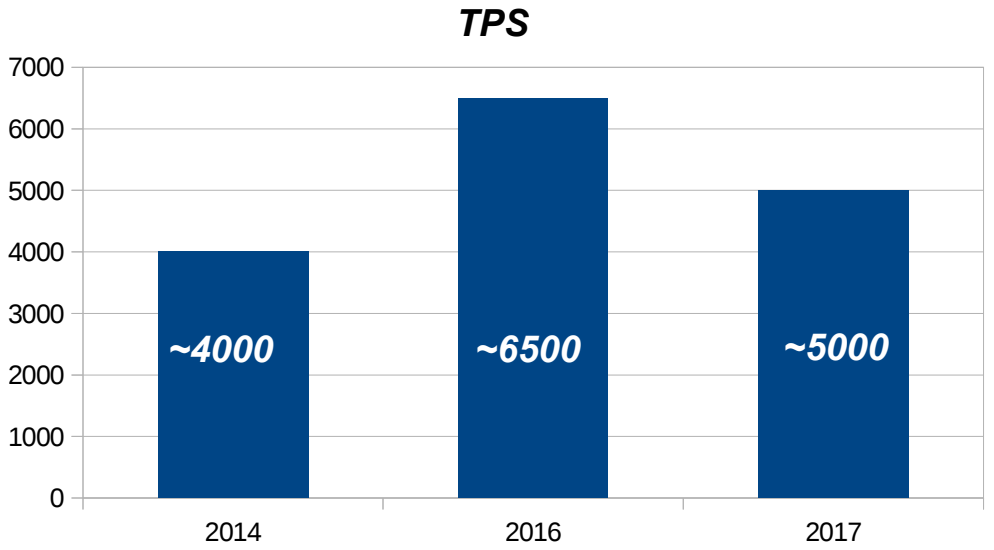
MatView + Londiste

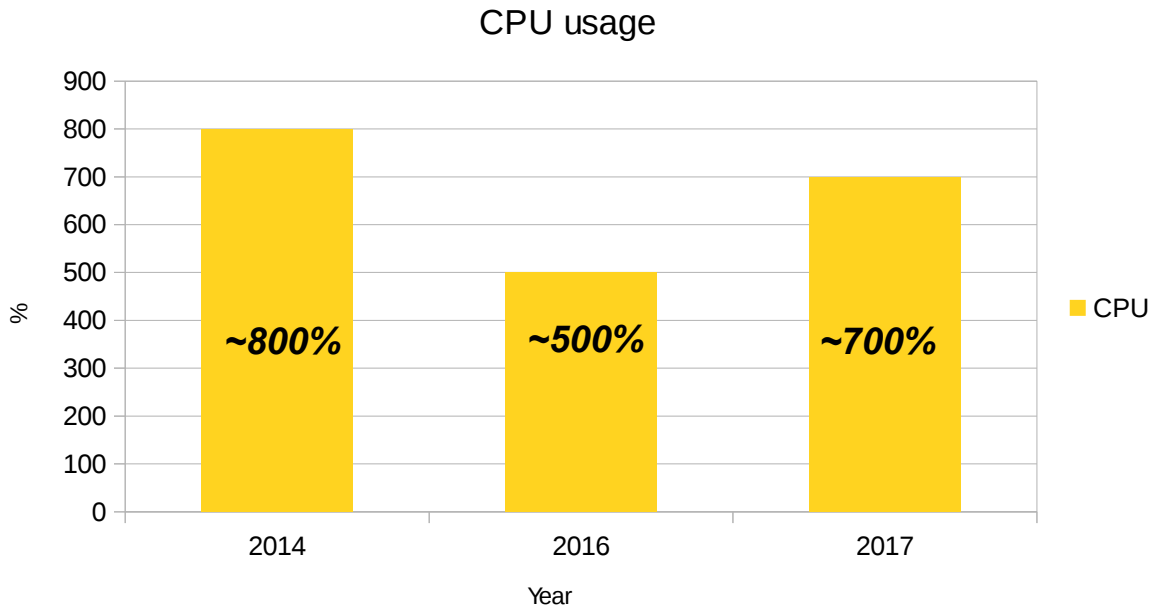


Repca

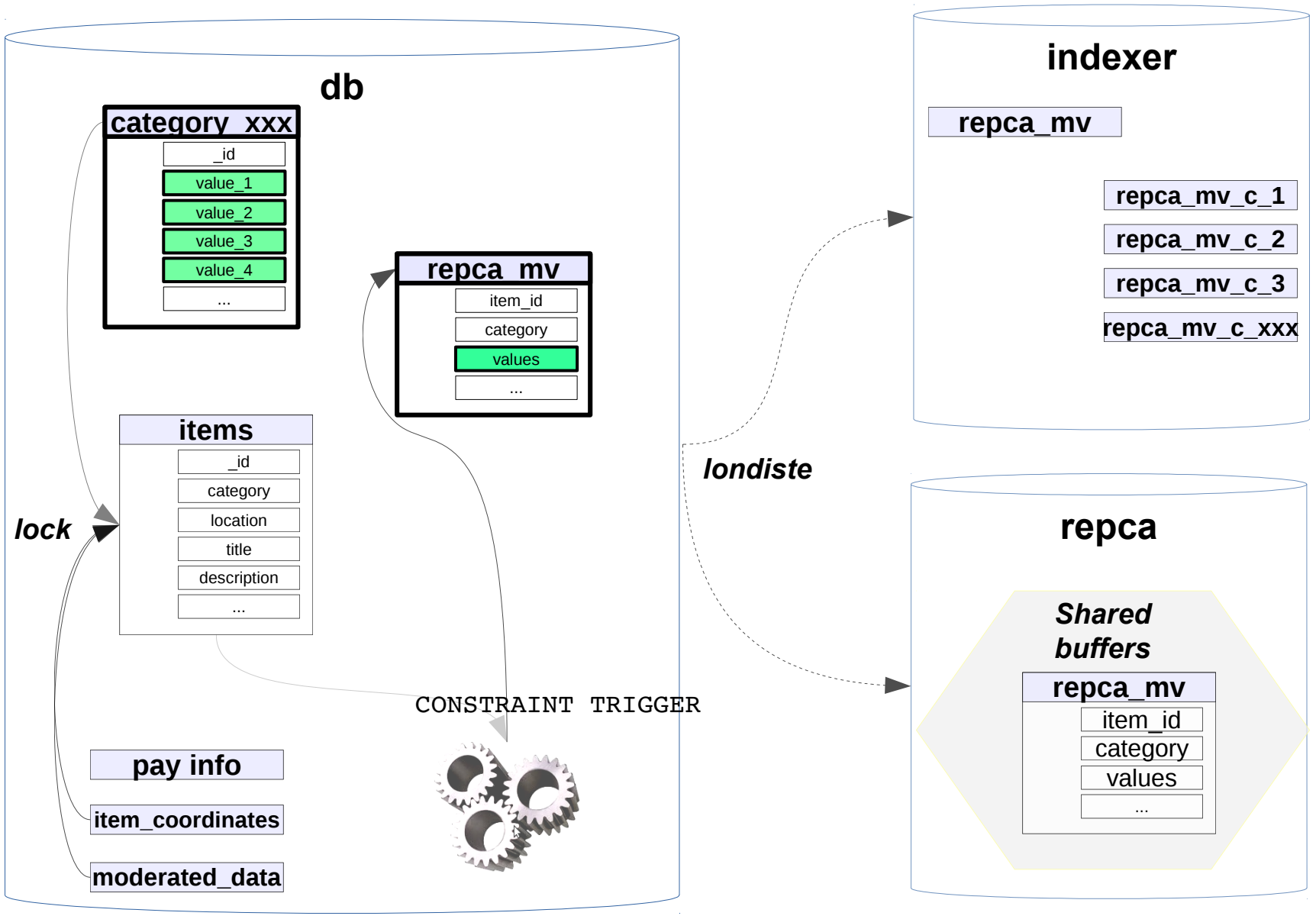




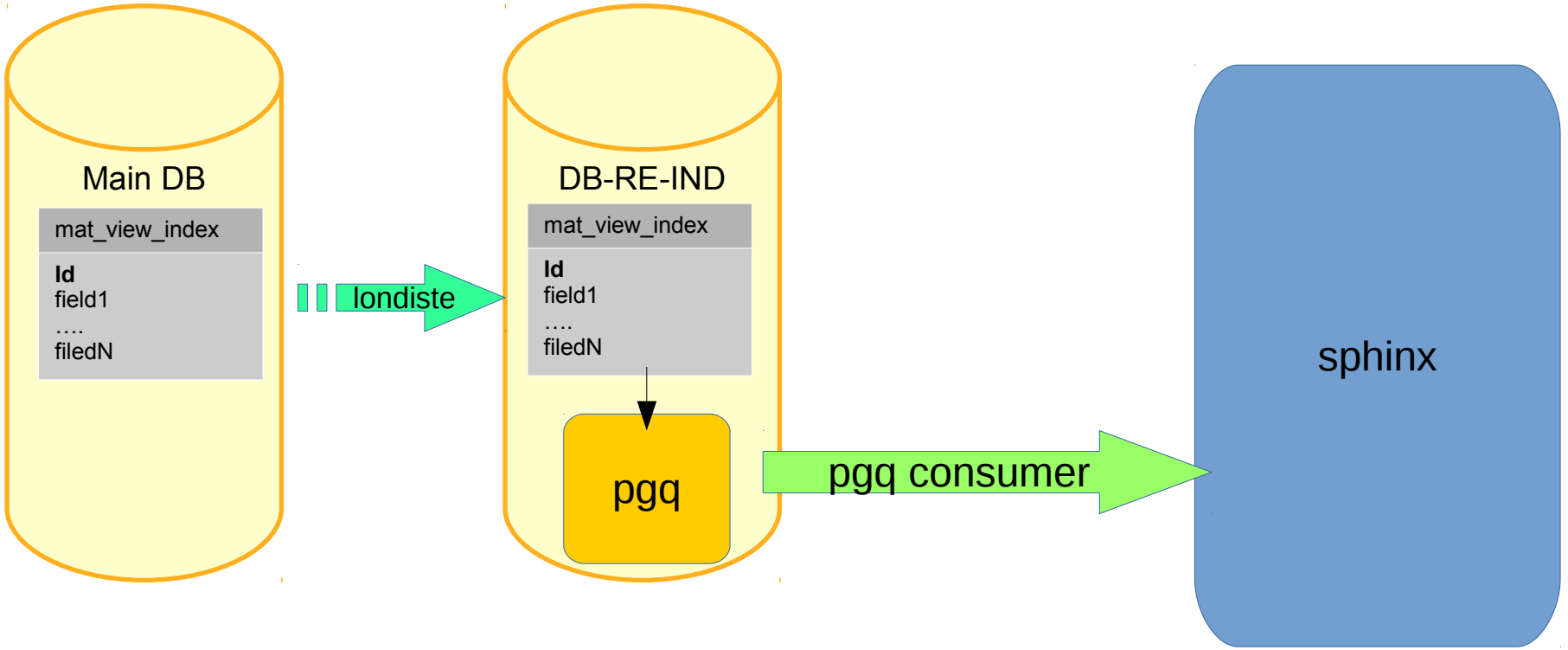




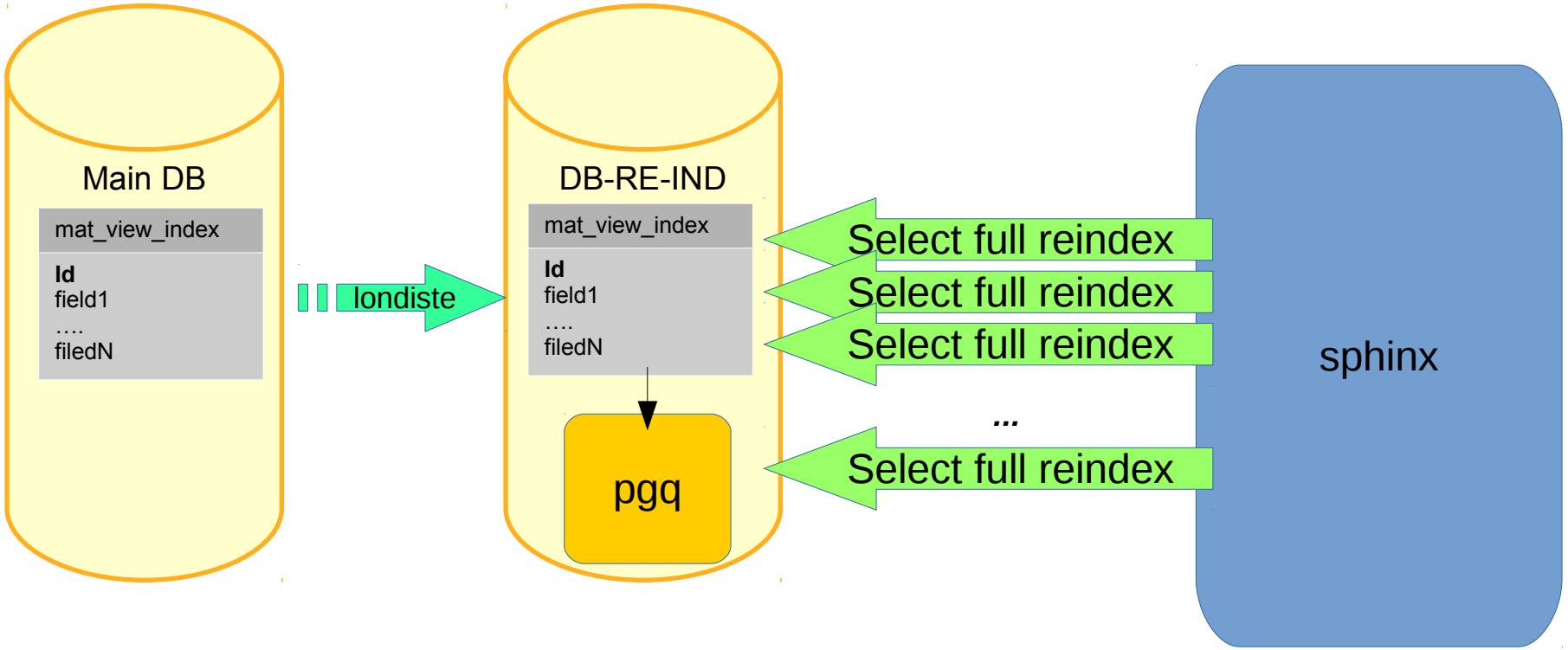
MatView + Londiste



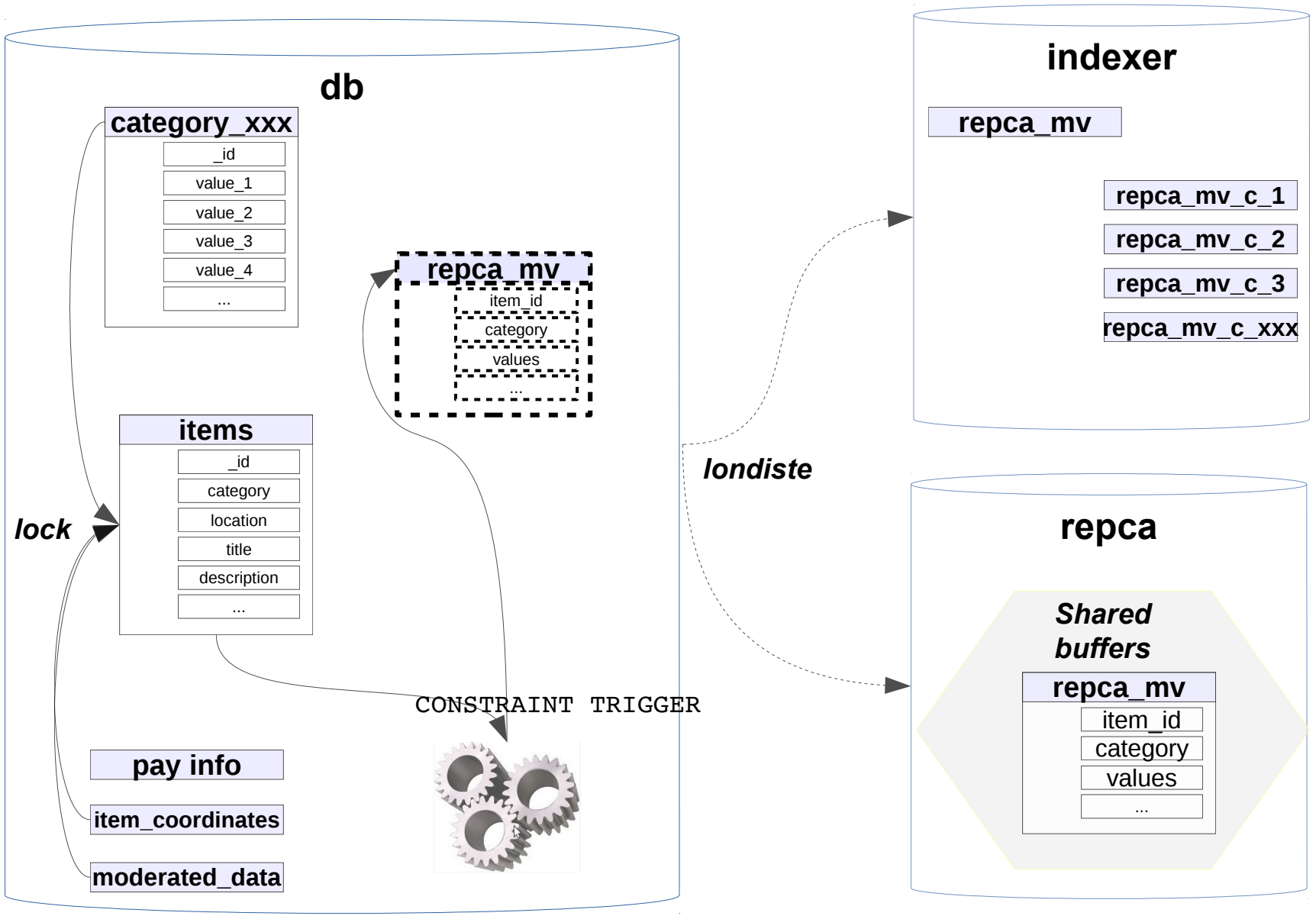
Real time Sphinx index
3 секунды!!!



Sphinx reindex



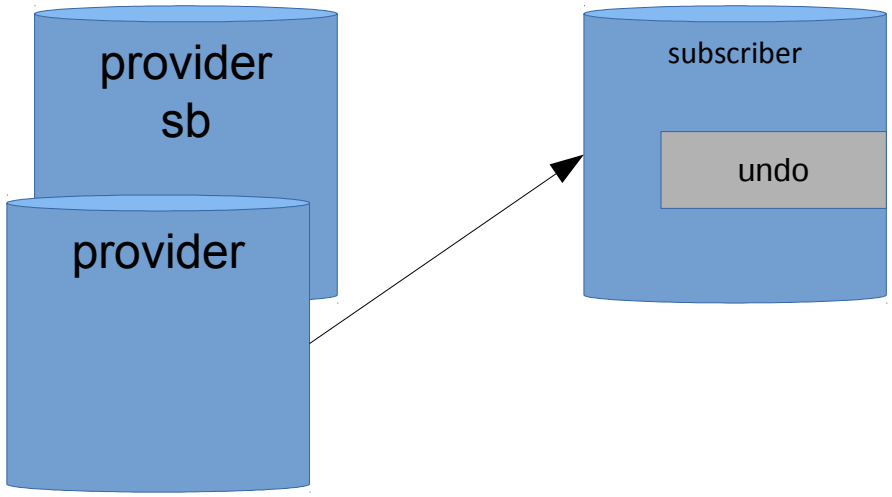
MatView + Londiste



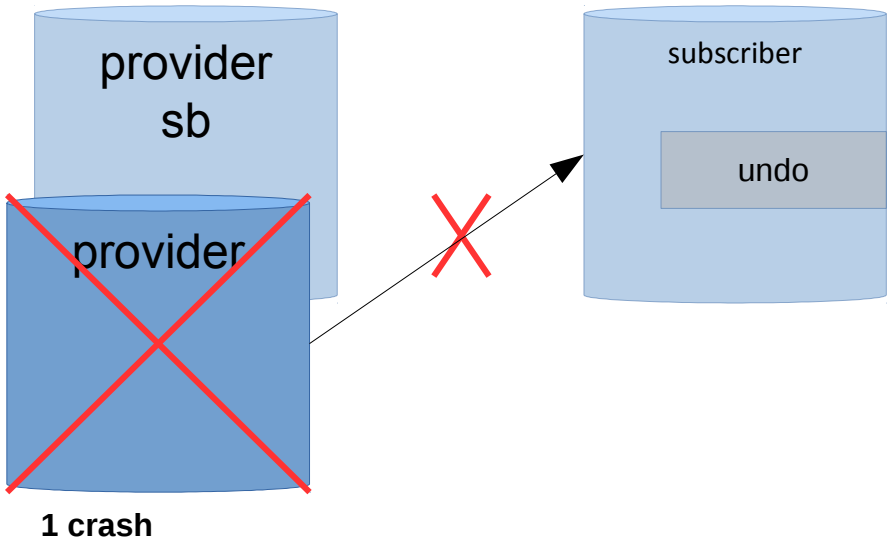
Производное от данных

- Deffered trigger
- Собрать все сигналы (GUC for session)
- На приемнике конвеер обработки и передача по цепочке

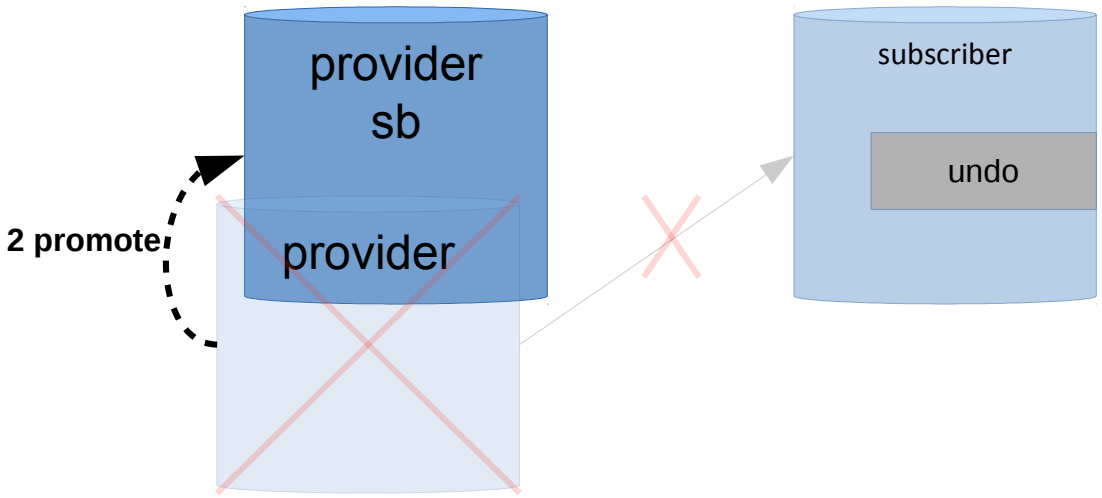
Падение provider



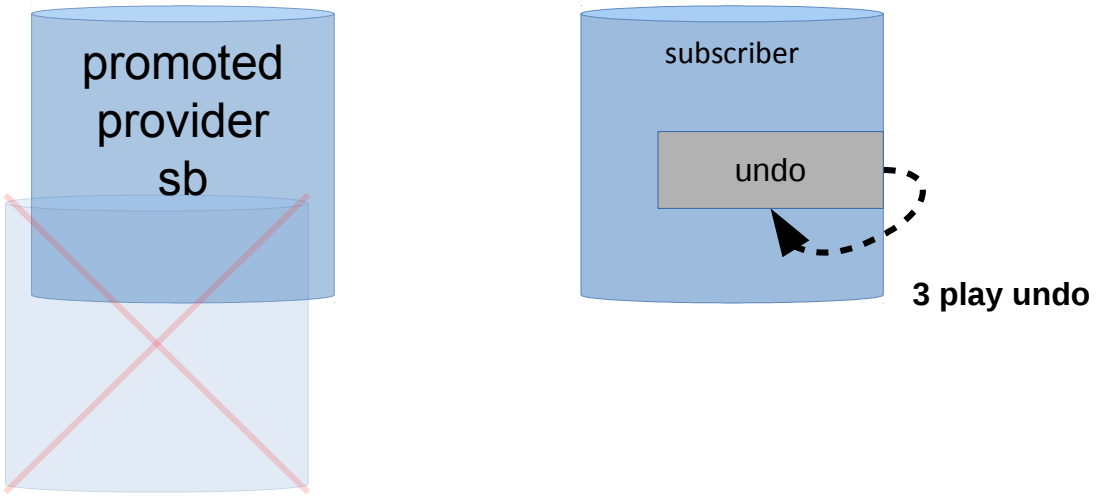
Падение provider



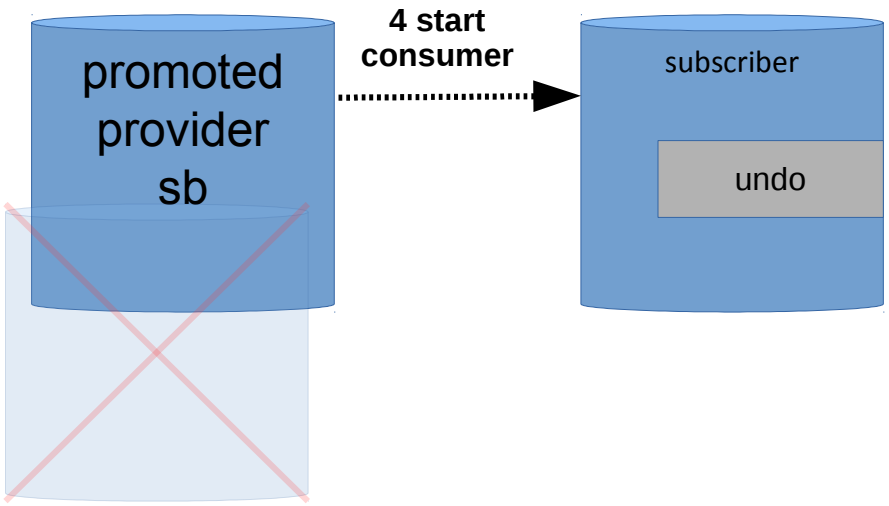
Падение provider



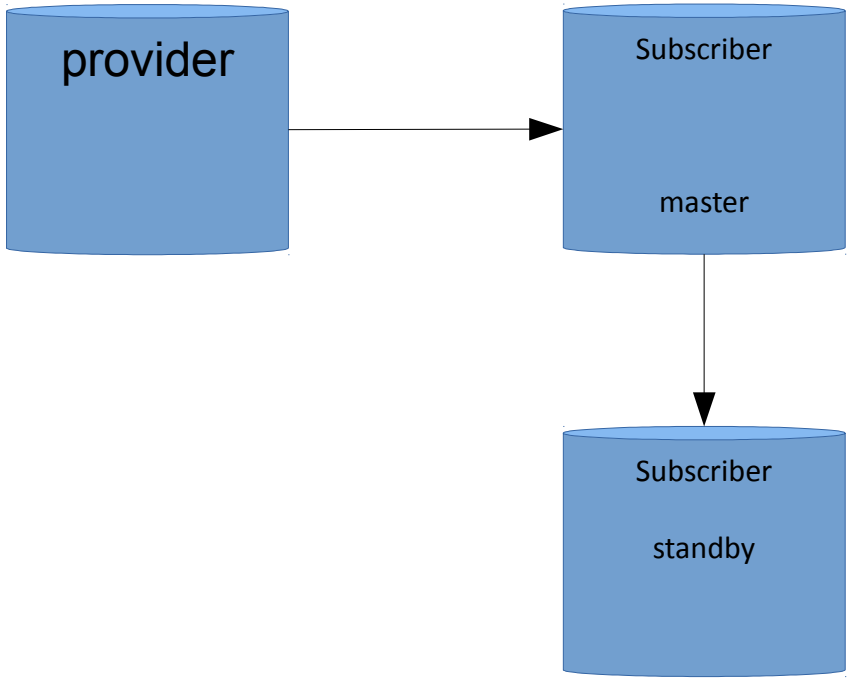
Падение provider



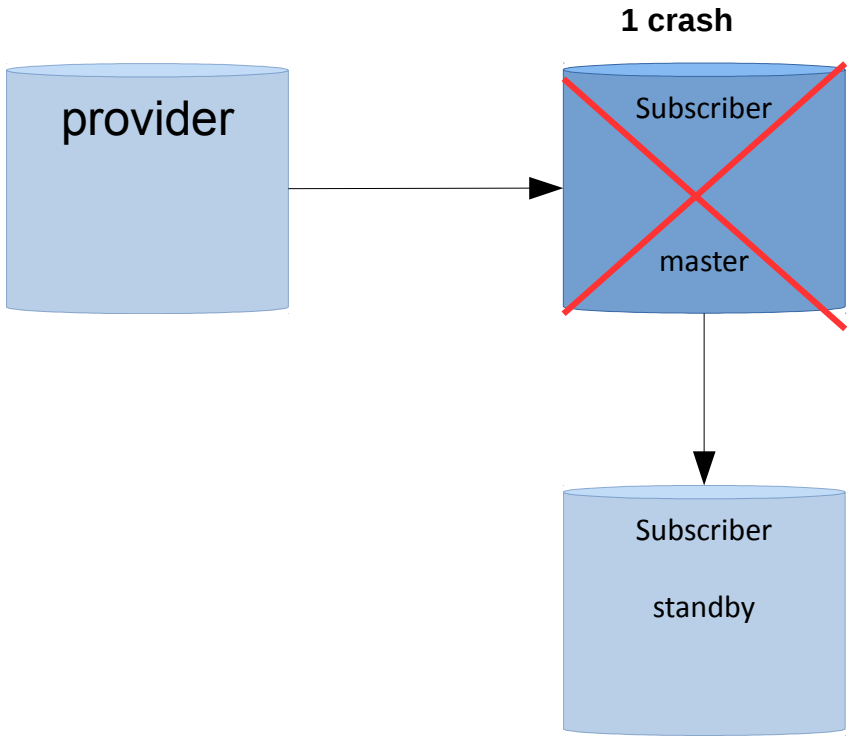
Падение provider



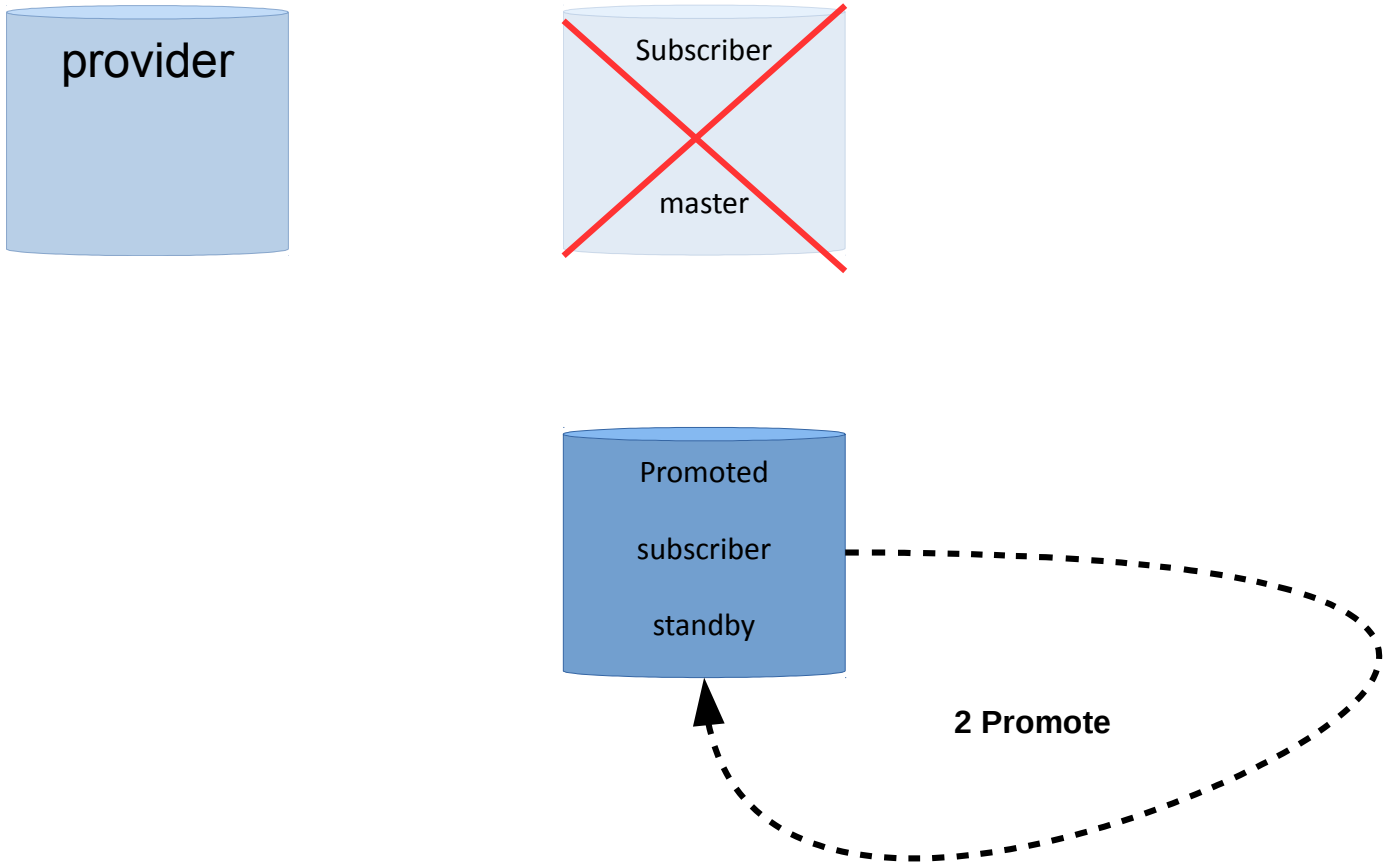
Падение subscriber



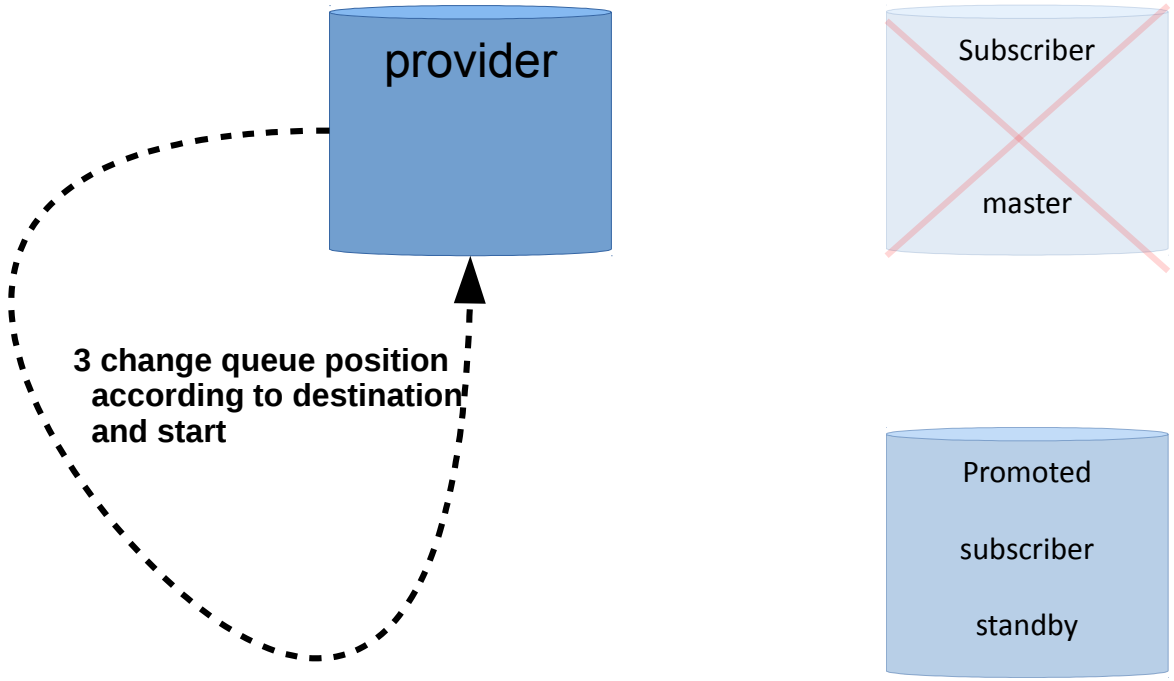
Падение subscriber



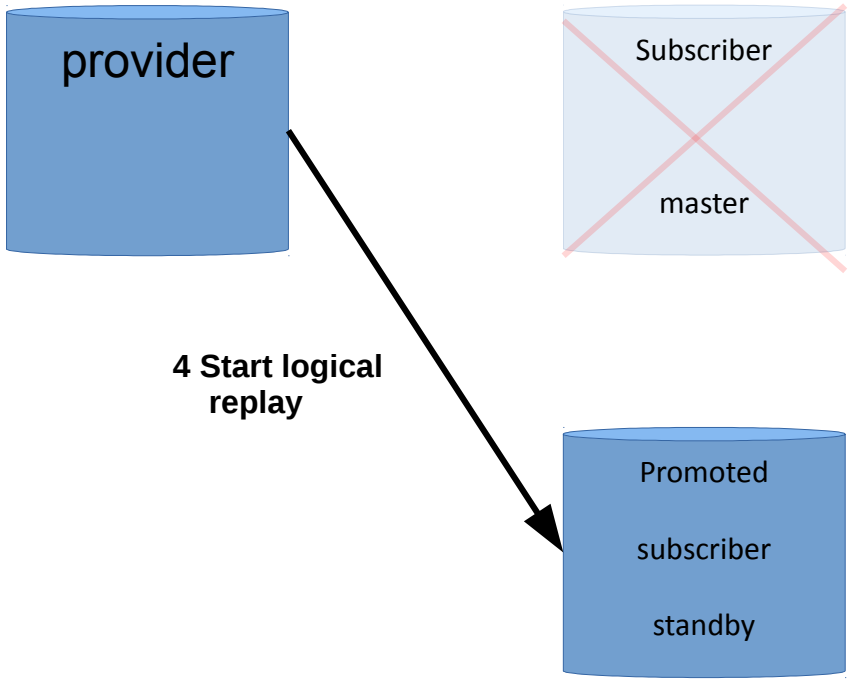
Падение subscriber



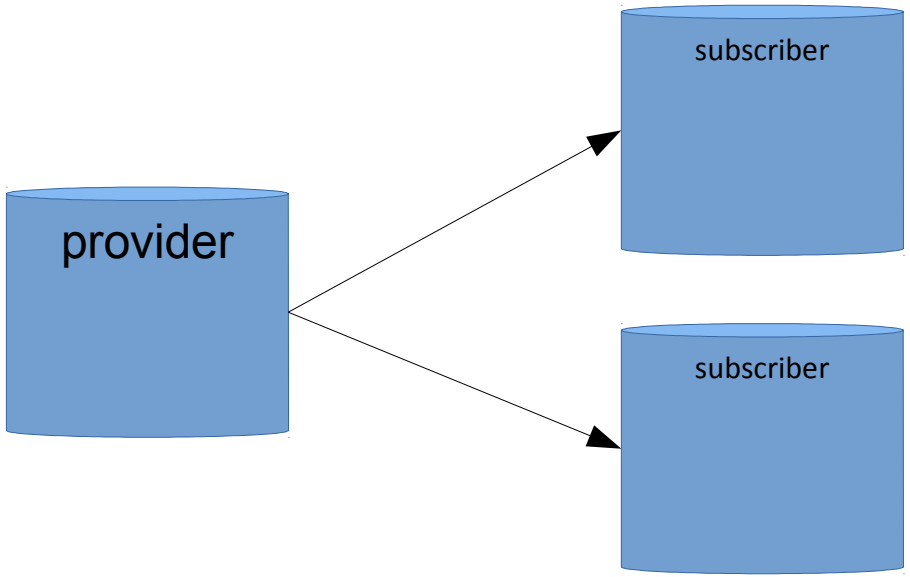
Падение subscriber



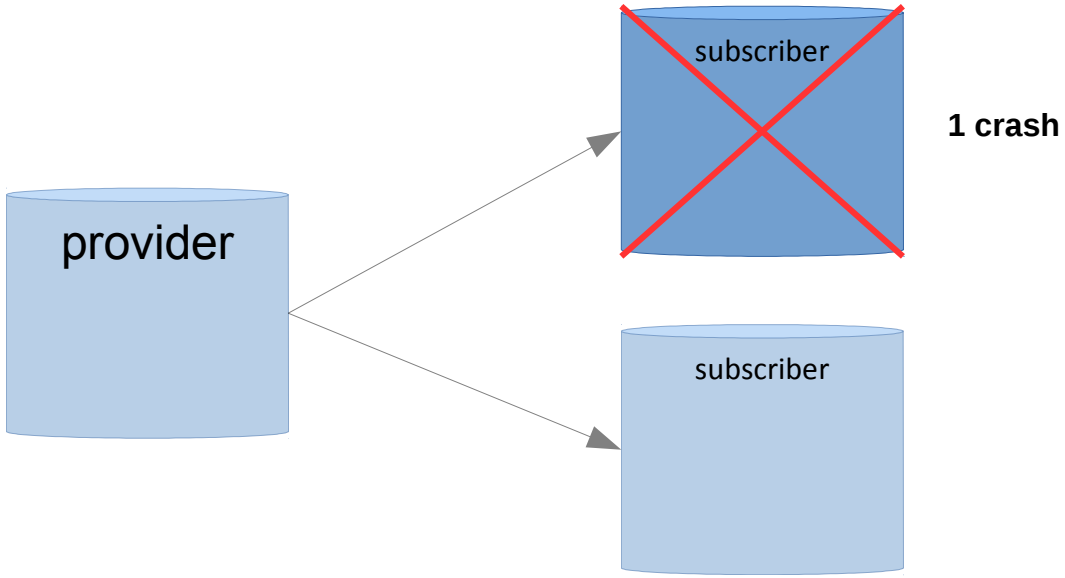
Падение subscriber



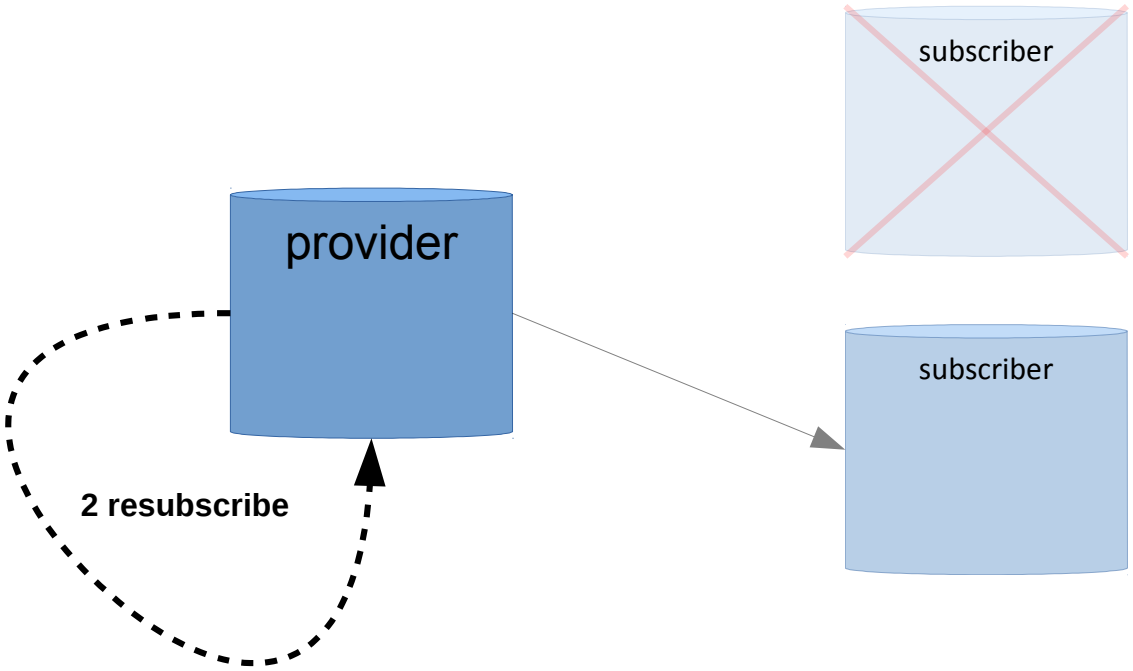
Создание копии подписчика



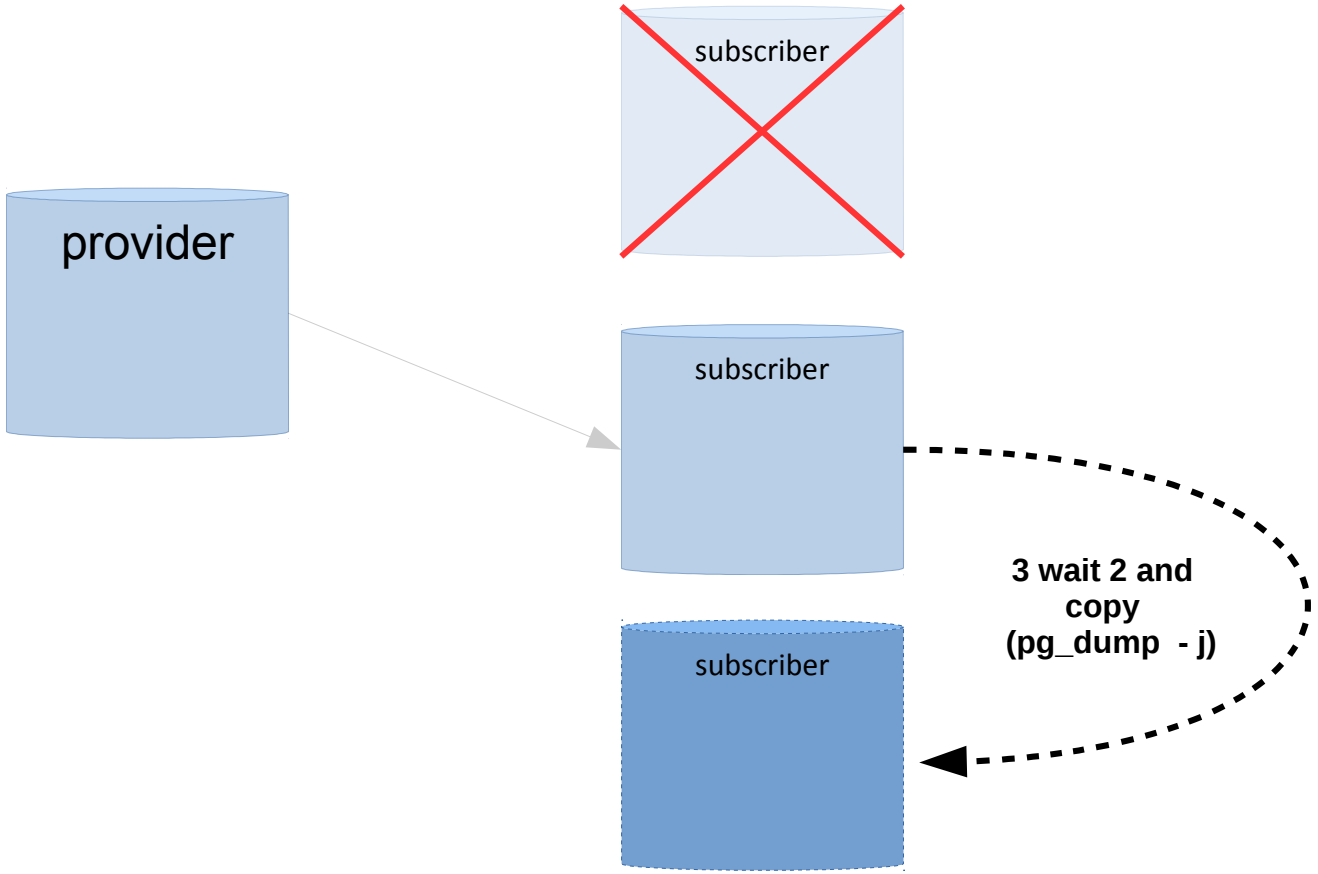
Создание копии подписчика



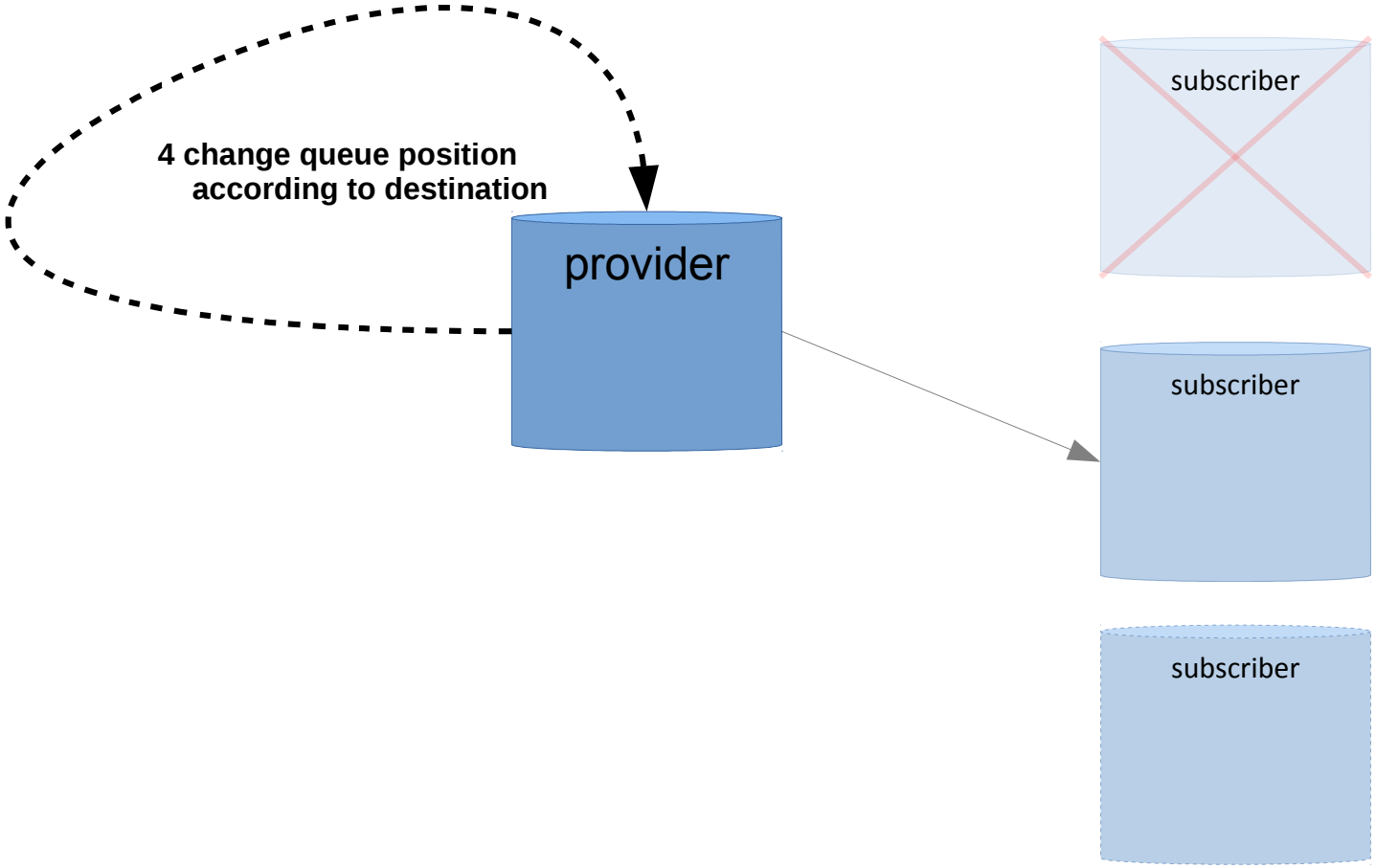
Создание копии подписчика



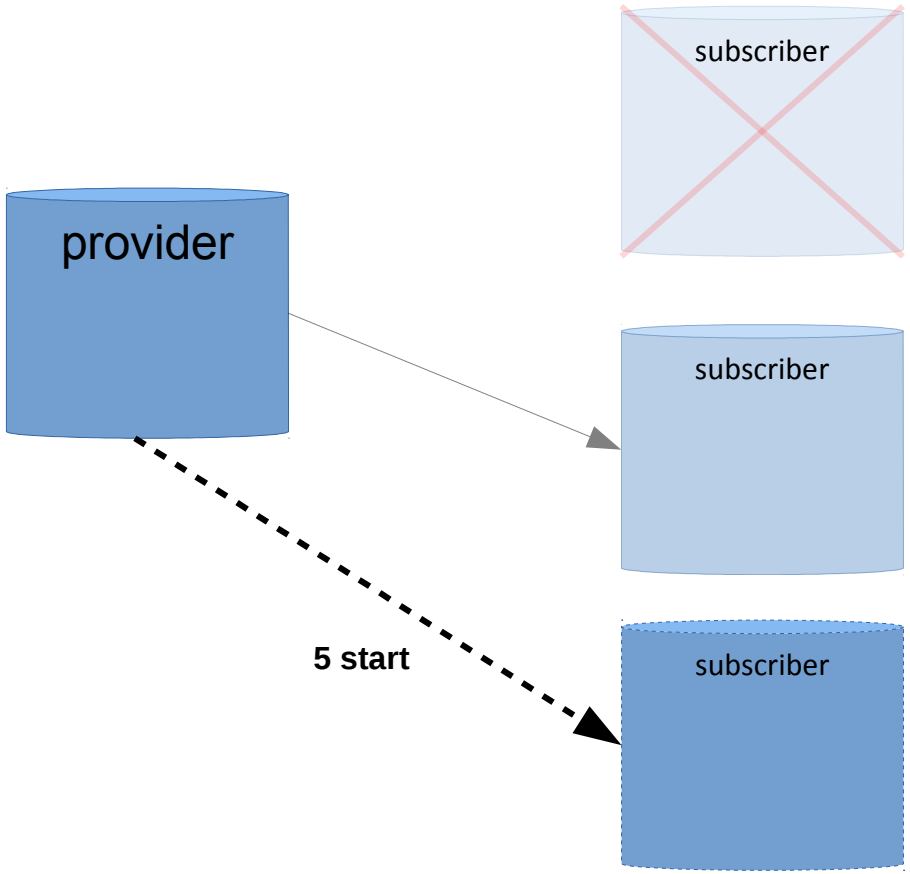
Создание копии подписчика



Создание копии подписчика



Создание копии подписчика



План демонстрации

Londiste (9.6) и Logical (10):

- инициализация базы master, standby, repca1 (items, cats, INSERT)
- создание логической репликации
- создать вторую реплику через pg_dump
- авария и promote standby
- UNDO
- ok!

План демонстрации

https://github.com/avito-tech/dba-docs/blob/master/2017-highload/plan.text

Проблемы logical

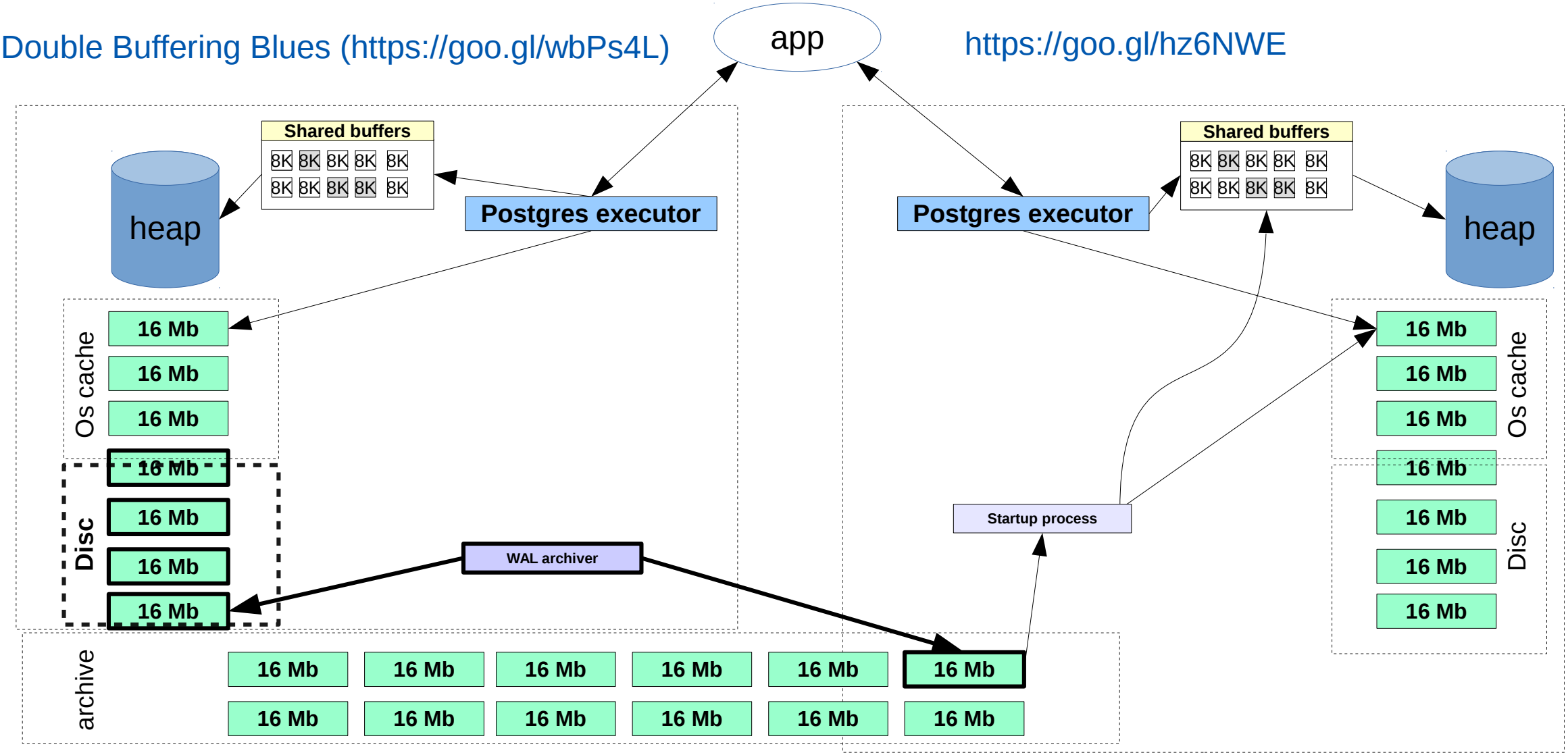
- текущая сессия: LSN, replication origin
- lock replication
- для pg_replication_origin_advance нужно commit stop
- pg_replication_origin_advance не сдвигает слот на src (REDO)
- replication origin не транзакционен (shared memory + WAL реплики)
- pg_dump (replication origin, pg_subscription_rel, и т.п.)
- logical slots на standby (дырка после promote)

	SkyTools	PostgreSQL10
Repca	OK	OK
Sphinx	OK	OK
Event tracking	OK	OK (overhead на proxy таблицу)
Rt index	OK	OK(sql protocol)
Unlogged table	OK	-
Undo	OK	OK (хочется простой реализации из коробки)
Redo	OK	OK (хочется чтобы pg_replication_origin_advance сдвигал слот на src)
Upgrade	OK	OK

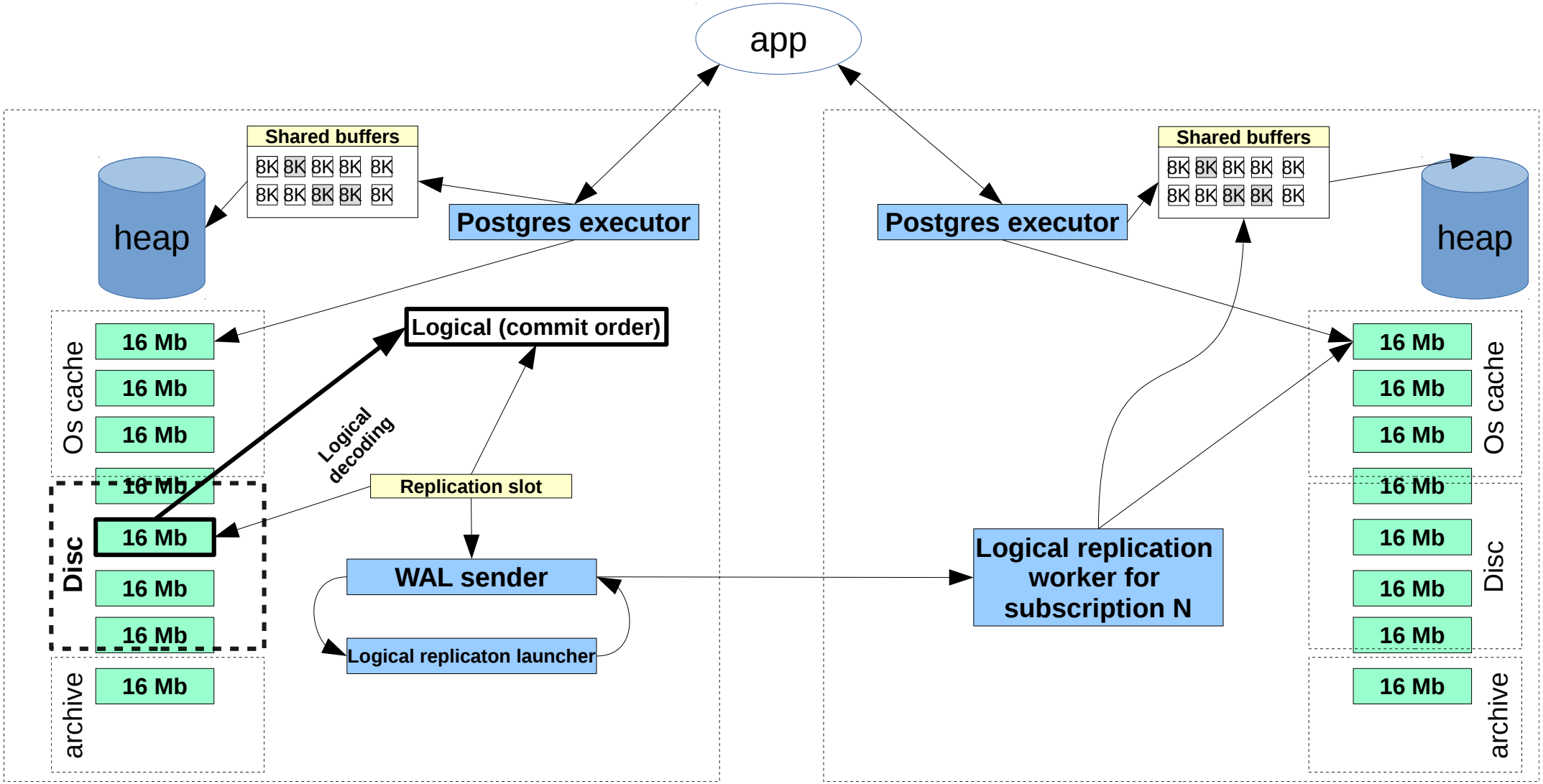
WALs out of page cache (archive)

Double Buffering Blues (https://goo.gl/wbPs4L)

https://goo.gl/hz6NWE



WALs out of page cache (logical)



kevteev@avito.ru

https://habrahabr.ru/company/avito/

https://github.com/avito-tech/dba-docs