Úvod do logické replikace

Prague PostgreSQL Developer Day 2024

2024-06-04 Aleš Zelený, Tomáš Vondra

Osnova

- Co je logická replikace
- Příklady využití
- Zjednodušený příklad
- Trocha teorie
- Základní vlastnosti
- Omezení logické replikace
- Konfigurace a řízení logické replikace – příklad
- Monitoring logické replikace
- Obvyklé problémy a jejich řešení

Co je logická replikace

- Replikace v PostgreSQL
  - binární (streaming/log-shipping)
    - standby/slave/follower instance(s)
  - logická
- Write Ahead Log - WAL
  - crash recovery
  - online zálohy, PITR
  - binární replikace (streaming/log-shipping standby)
  - logická replikace
- Log Sequence Number - LSN
  - unsigned 64-bit integer
  - trvale rostoucí ukazatel na pozici záznamu v WAL logu

Fyzická replikace

Přenáší změny ukládané v transakčním logu na cílový server, kde **probíhá RECOVERY** - aplikují se změny bloků (stránek) v **datových souborech**.

Přenáší se tedy *všechny DB PostgreSQL instance*.

Image by Gerd Altmann from Pixabay

Logická replikace

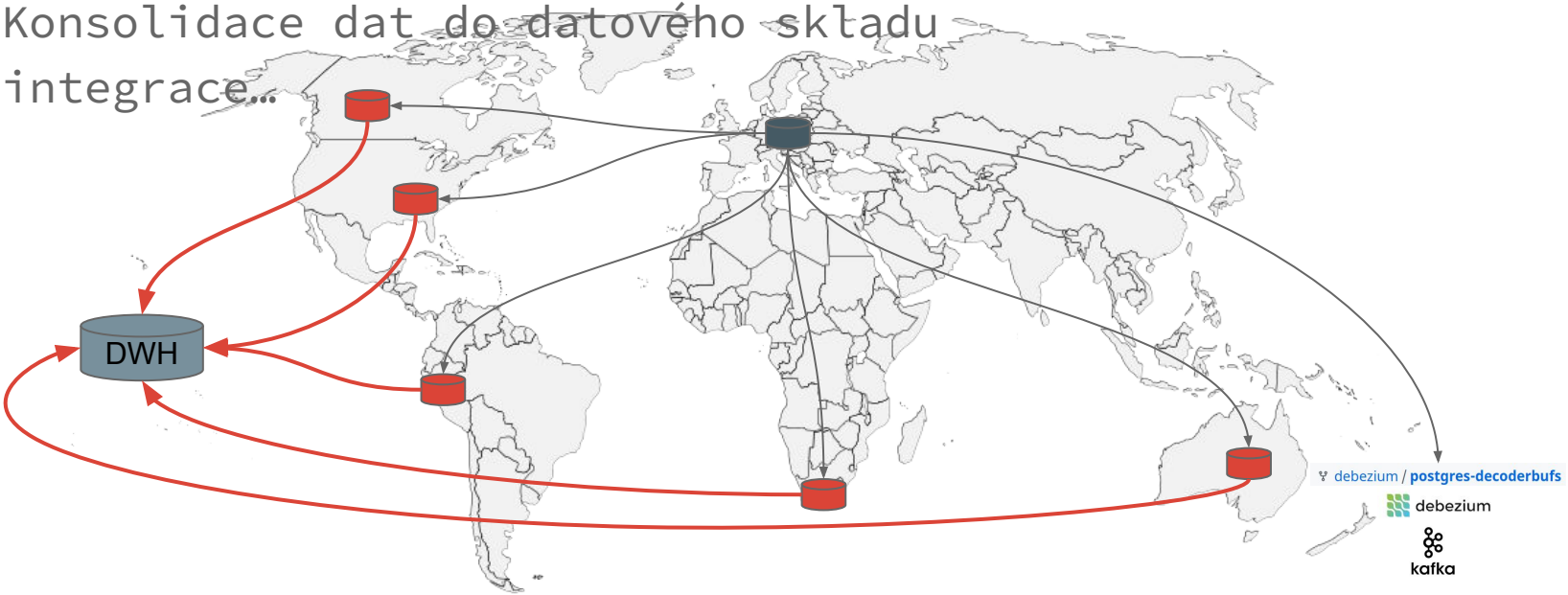
Zjišťuje změny v datech na úrovni **řádek** v **tabulkách** a tyto změny aplikuje v tabulce cílové databázi formou SQL příkazu.

Zúčastněné databáze jsou dostupné i pro *zápis*.

Image by Nile from Pixabay

Příklady využití

- Distribuce vybraných tabulek
  - konfigurace, ceníky, produktový katalog…
- Konsolidace dat do datového skladu
- integrace…



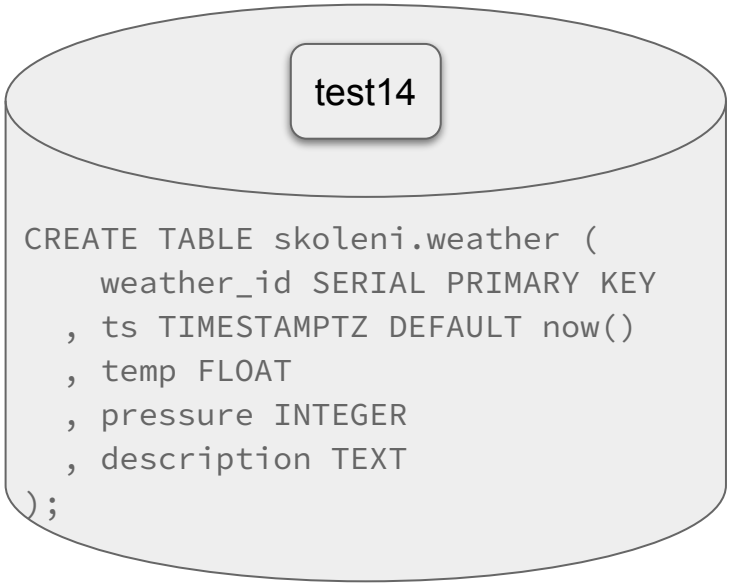
PŘÍKLADY VYUŽITÍ...

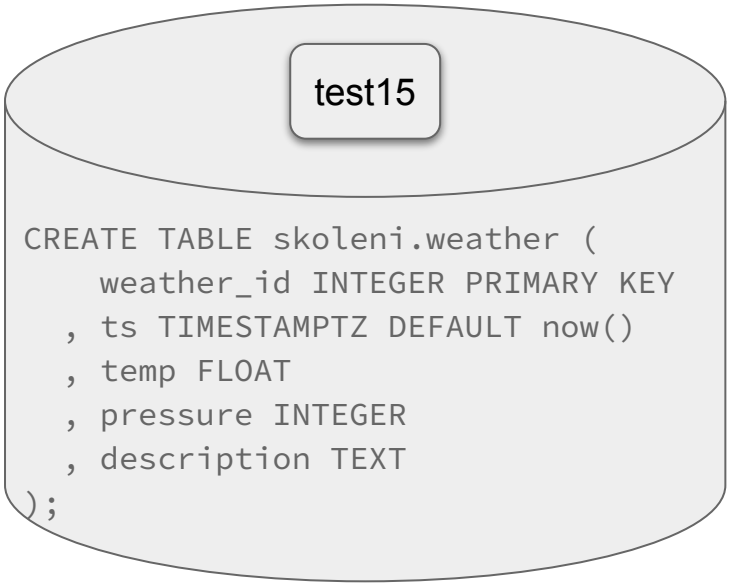
- změna platformy (OS, endian)
- přenos dat mezi různými verzemi PostgreSQL
- upgrade s minimálním časem výpadku - omezení, příklady
- distribuce dat
- konsolidace dat
- integrace s dalšími produkty (debezium & kafka/pub_sub…)
- lze využít triggery na straně příjemce dat

MINIMALISTICKÁ UKÁZKA

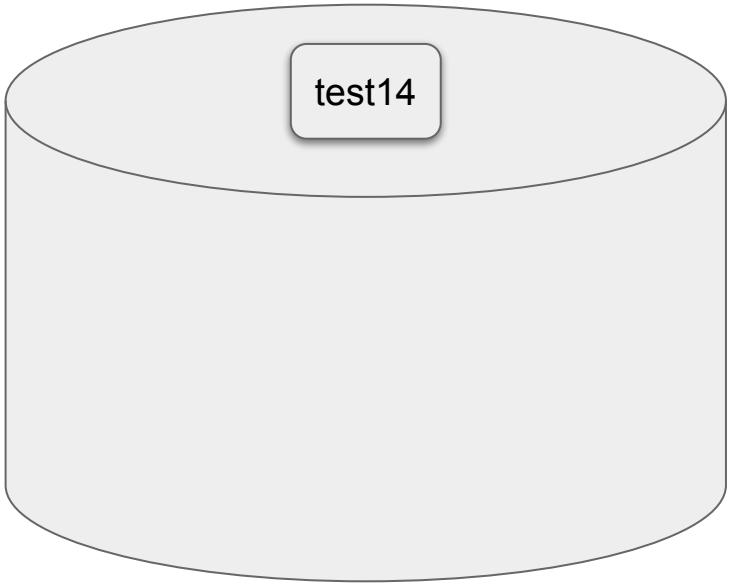
- Příprava
  - dvě PostgreSQL instance (ukázky: porty 5432, 5433)
  - v každé instancí DB, každá se může jmenovat jinak
  - tabulka shodného plně kvalifikovaného jména (schema.tabulka)
  - subscriber tabulka může mít nadmnožinu sloupců zdrojové
    - PostgreSQL15 umožňuje i výběr replikovaných sloupců
- Publikace
- Subscription

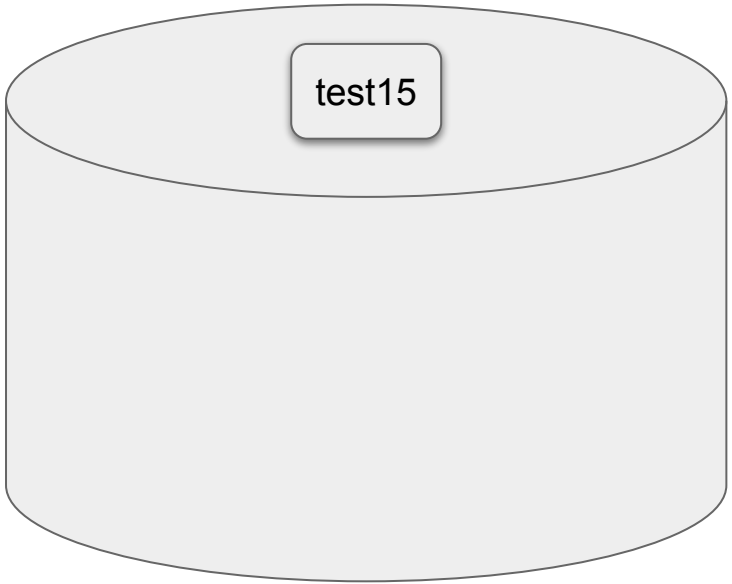
Databáze: Publisher a Subscriber





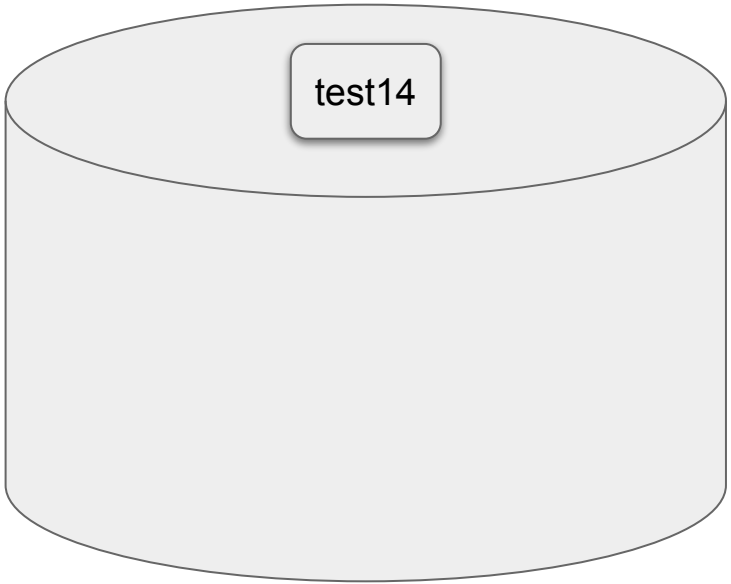
Publication

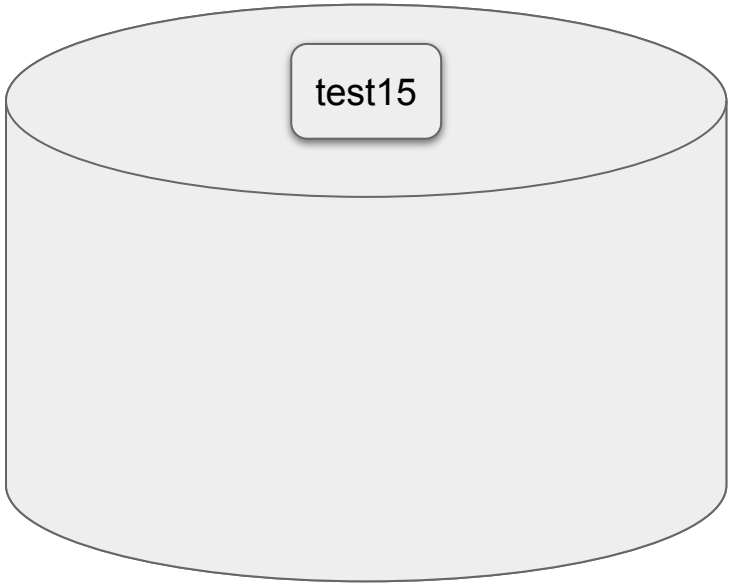





```
CREATE PUBLICATION pub_skoleni
  FOR TABLE skoleni.weather
  WITH ( publish = 'insert, update, delete, truncate' );
```

Subscription






```
CREATE SUBSCRIPTION sub_skoleni
  CONNECTION 'port=5432 dbname=test14'
  PUBLICATION pub_skoleni;
```

Data jsou synchronizována, replikace přenáší změny

```
test15=# table skoleni.weather limit 3;
 weather_id |           ts           | temp | pressure |              description
------------+------------------------+------+----------+-----------------------------------
          1 | 1970-01-01 20:31:01+01 |  -10 |      810 | Thursday  01, 01 1970
          2 | 1970-01-02 12:56:55+01 |  -13 |      998 | Prev: Thursday  01, 01 1970 20:31:01
          3 | 1970-01-02 14:26:32+01 |  -10 |     1014 | Prev: Friday    02, 01 1970 12:56:55
(3 rows)
```

```
test14=# DELETE FROM skoleni.weather WHERE weather_id = 1;
DELETE 1
test14=# UPDATE skoleni.weather SET pressure = 500 WHERE weather_id = 2;
UPDATE 1
```

```
test15=# SELECT * FROM skoleni.weather WHERE weather_id < 4 ORDER BY weather_id;
 weather_id |           ts           | temp | pressure |              description
------------+------------------------+------+----------+-----------------------------------
          2 | 1970-01-02 12:56:55+01 |  -13 |      500 | Prev: Thursday  01, 01 1970 20:31:01
          3 | 1970-01-02 14:26:32+01 |  -10 |     1014 | Prev: Friday    02, 01 1970 12:56:55
(2 rows)
```

DISABLE (SUSPEND) A ENABLE SUBSCRIPTION

```
test15=# ALTER SUBSCRIPTION sub_skoleni DISABLE;
```

```
test15=# SELECT * FROM skoleni.weather WHERE weather_id < 4 ORDER BY weather_id;
 weather_id |           ts           | temp | pressure |              description
------------+------------------------+------+----------+---------------------------------------
          2 | 1970-01-02 12:56:55+01 |  -13 |      500 | Prev: Thursday  01, 01 1970 20:31:01
          3 | 1970-01-02 14:26:32+01 |  -10 |     1014 | Prev: Friday    02, 01 1970 12:56:55
```

```
test14=# UPDATE skoleni.weather SET temp= 0, pressure = 1000 WHERE weather_id = 3;
UPDATE 1
```

```
test15=# ALTER SUBSCRIPTION sub_skoleni ENABLE;
```

```
test15=# SELECT * FROM skoleni.weather WHERE weather_id < 4 ORDER BY weather_id;
 weather_id |           ts           | temp | pressure |              description
------------+------------------------+------+----------+---------------------------------------
          2 | 1970-01-02 12:56:55+01 |  -13 |      500 | Prev: Thursday  01, 01 1970 20:31:01
          3 | 1970-01-02 14:26:32+01 |    0 |     1000 | Prev: Friday    02, 01 1970 12:56:55
```

Změny schéma (struktury)

1. přidání sloupce
   a. subscriber
   b. publisher
2. odebrání sloupce
   a. publisher
   b. subscriber

Subscriber tabulka může mít nadmnožinu sloupců publisher tabulky.

Odebrání tabulky z publikace

```
-- Publisher DB:

ALTER PUBLICATION pub_skoleni DROP TABLE skoleni.weather;

-- Změny nejsou od dokončení příkazu přenášeny.


-- Subscriber DB:

ALTER SUBSCRIPTION sub_skoleni REFRESH PUBLICATION;

-- Subscriber aktualizuje seznam tabulek, pro které očekává -- data (smaže z svého seznamu tabulku skoleni.weather)
```

Přidání tabulky do publikace

```
test14=# \dRp+ pub_skoleni
                         Publication pub_skoleni
  Owner   | All tables | Inserts | Updates | Deletes | Truncates | Via root
----------+------------+---------+---------+---------+-----------+----------
 postgres | f          | t       | t       | t       | t         | f
(1 row)


test14=# ALTER PUBLICATION pub_skoleni ADD TABLE skoleni.weather;
ALTER PUBLICATION


test14=# \dRp+ pub_skoleni
                         Publication pub_skoleni
  Owner   | All tables | Inserts | Updates | Deletes | Truncates | Via root
----------+------------+---------+---------+---------+-----------+----------
 postgres | f          | t       | t       | t       | t         | f
Tables:
    "skoleni.weather"

test14=# truncate table skoleni.weather;
```

Přidání tabulky do publikace... co na to subscriber?

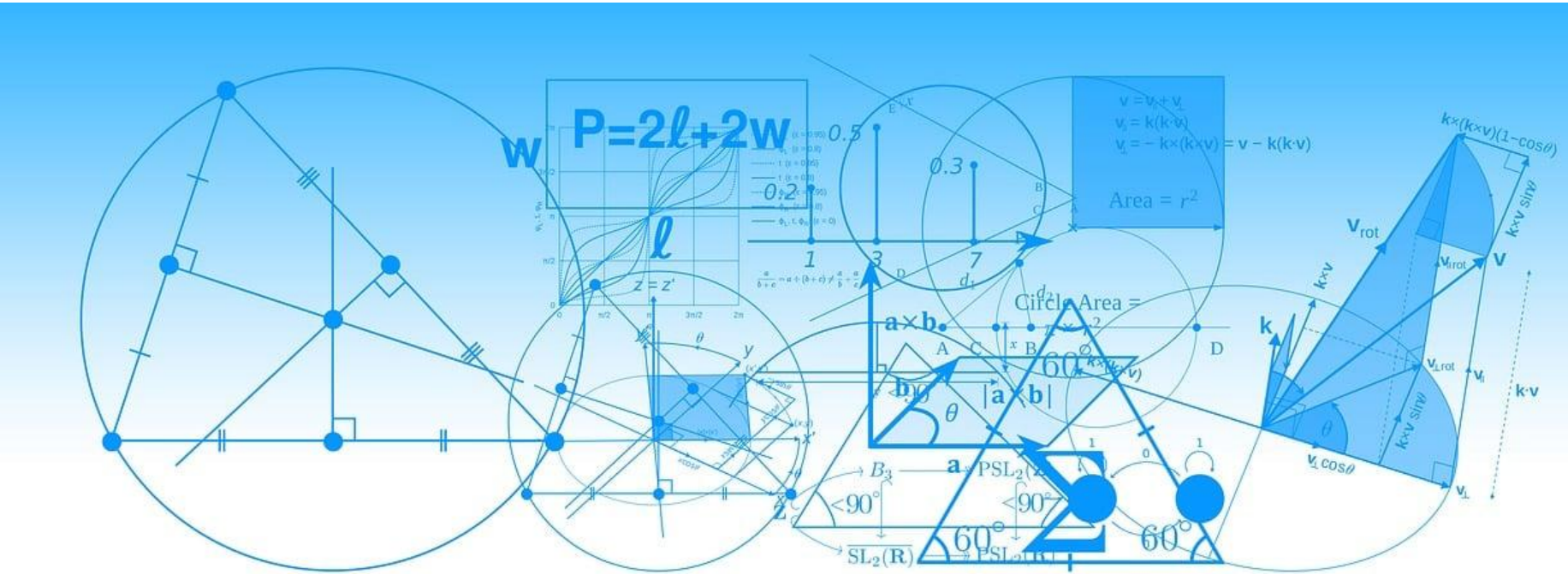
```
test14=# truncate table skoleni.weather;
```

```
test15=# SELECT * FROM skoleni.weather WHERE weather_id < 4 ORDER BY weather_id;
 weather_id |           ts           | temp | pressure |             description
------------+------------------------+------+----------+-------------------------------------
          2 | 1970-01-02 12:56:55+01 |  -13 |      500 | Prev: Thursday  01, 01 1970 20:31:01
          3 | 1970-01-02 14:26:32+01 |    0 |     1000 | Prev: Friday    02, 01 1970 12:56:55

test15=# ALTER SUBSCRIPTION sub_skoleni REFRESH PUBLICATION;

test15=# SELECT * FROM skoleni.weather WHERE weather_id < 4 ORDER BY weather_id;
 weather_id |           ts           | temp | pressure |             description
------------+------------------------+------+----------+-------------------------------------
          2 | 1970-01-02 12:56:55+01 |  -13 |      500 | Prev: Thursday  01, 01 1970 20:31:01
          3 | 1970-01-02 14:26:32+01 |    0 |     1000 | Prev: Friday    02, 01 1970 12:56:55
```

```
test14=# INSERT INTO skoleni.weather (weather_id, ts, temp, pressure) VALUES (1, now(), -2, 1024);
INSERT 0 1
```

```
test15=# SELECT * FROM skoleni.weather WHERE weather_id < 4 ORDER BY weather_id;
 weather_id |              ts               | temp | pressure |             description
------------+-------------------------------+------+----------+-------------------------------------
          1 | 2023-01-25 00:03:01.692848+01 |   -2 |     1024 |
          2 | 1970-01-02 12:56:55+01        |  -13 |      500 | Prev: Thursday  01, 01 1970 20:31:01
          3 | 1970-01-02 14:26:32+01        |    0 |     1000 | Prev: Friday    02, 01 1970 12:56:55
```

Trocha teorie



Fotka od Gerd Altmann z Pixabay

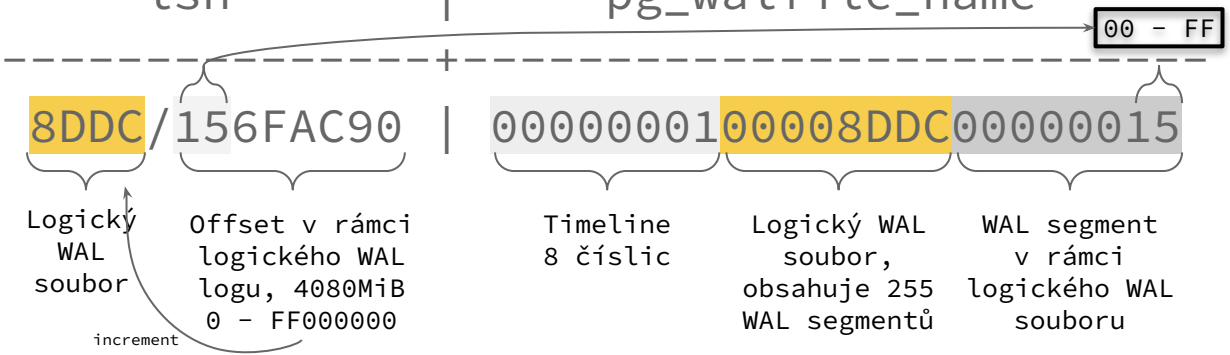
API vs. implementace

- infrastruktura + API
  - generická funkcionalita
  - sada callbacků (různé události)
- konkrétní implementace
  - built-in replikace
  - pglogical, BDR
  - debezium
  - …
  - různé koncepty, další funkce, architektury

Write Ahead Log, Log Sequence Number

- WAL – “logický log” (18.446.744.073.709.551.615 ~ 15.9 [Ei] exbi)
  - rozdělen na segmenty – soubory v pg_wal adresáři (16MiB – výchozí)
- LSN – Log Sequence number – ukazatel pozice v WAL

```
SELECT cwl.lsn, pg_catalog.pg_walfile_name(cwl.lsn)
FROM pg_catalog.pg_current_wal_lsn() AS cwl(lsn);
     lsn      |     pg_walfile_name
--------------+--------------------------
 8DDC/156FAC90 | 000000010000 8DDC 00000015
```



WAL - CO OBSAHUJE? PG_WALDUMP

```
rmgr: Heap        len (rec/tot):    119/   119, tx: 3937076997, lsn: 1C2C8/C800D480, prev 1C2C8/C800D448, desc: INSERT ...
rmgr: Btree       len (rec/tot):     64/    64, tx: 3937076997, lsn: 1C2C8/C800D4F8, prev 1C2C8/C800D480, desc: INSERT_LEAF ...
rmgr: Btree       len (rec/tot):     80/    80, tx: 3937076997, lsn: 1C2C8/C800D608, prev 1C2C8/C800D5C0, desc: INSERT_LEAF ...
rmgr: Heap        len (rec/tot):     98/    98, tx: 3937076998, lsn: 1C2C8/C800DFD8, prev 1C2C8/C800D690, desc: UPDATE ...
rmgr: Btree       len (rec/tot):     53/  1733, tx: 3937076998, lsn: 1C2C8/C800E058, prev 1C2C8/C800DFD8, desc: INSERT_LEAF ...
rmgr: Btree       len (rec/tot):     64/    64, tx: 3937076998, lsn: 1C2C8/C800E720, prev 1C2C8/C800E058, desc: INSERT_LEAF ...
rmgr: Heap        len (rec/tot):     54/    54, tx: 3937076998, lsn: 1C2C8/C800E760, prev 1C2C8/C800E720, desc: LOCK ...
rmgr: Heap        len (rec/tot):     54/    54, tx: 3937076998, lsn: 1C2C8/C800E798, prev 1C2C8/C800E760, desc: LOCK ...
rmgr: Heap        len (rec/tot):    490/   490, tx: 3937076998, lsn: 1C2C8/C800E7D0, prev 1C2C8/C800E798, desc: UPDATE ...
rmgr: Btree       len (rec/tot):     66/    66, tx: 3937076998, lsn: 1C2C8/C800E9C0, prev 1C2C8/C800E7D0, desc: INSERT_POST ...
rmgr: Heap        len (rec/tot):    119/   119, tx: 3937076997, lsn: 1C2C8/C800EA08, prev 1C2C8/C800E9C0, desc: INSERT ...
rmgr: Btree       len (rec/tot):     64/    64, tx: 3937076997, lsn: 1C2C8/C800EA80, prev 1C2C8/C800EA08, desc: INSERT_LEAF ...
rmgr: Btree       len (rec/tot):     66/    66, tx: 3937076998, lsn: 1C2C8/C800EC28, prev 1C2C8/C800EBD8, desc: INSERT_POST ...
rmgr: Btree       len (rec/tot):     64/    64, tx: 3937076998, lsn: 1C2C8/C800EC70, prev 1C2C8/C800EC28, desc: INSERT_LEAF …
```

```
..., desc: INSERT off 80 f_lags 0x08, blkref #0: rel 1663/16474/132812 blk 2609754
```

```
rel - tablespace/db/relfilenode
blk - 8kB stránka
```

TEST_DECODING

TEST_DECODING

- extenze (plugin) v contrib balíku
- minimální funkčnost, ilustruje infrastrukturu
- ideální pro demonstraci základních konceptů

TEST_DECODING - VYTVOŘENÍ SLOTU

```
SELECT * FROM pg_catalog.pg_create_logical_replication_slot('my_slot', 'test_decoding');
 slot_name |    lsn
-----------+------------
 my_slot   | 1/A26B3F58
(1 row)
```

jméno slotu

decoding plugin

log sequence number

```
SELECT slot_name, plugin, slot_type, database, confirmed_flush_lsn FROM pg_replication_slots;
 slot_name |    plugin     | slot_type | database | confirmed_flush_lsn
-----------+---------------+-----------+----------+---------------------
 my_slot   | test_decoding | logical   | zelenya  | 1/A26B3F58
(1 row)
```

PG_LOGICAL_SLOT_GET_CHANGES() - ODBĚR DAT Z REPL. SLOTU

```
CREATE TABLE public.foo (foo_id SERIAL PRIMARY KEY, a NUMERIC, b TEXT);


SELECT slot_name, plugin, slot_type, database, confirmed_flush_lsn FROM pg_replication_slots;
 slot_name |    plugin     | slot_type | database | confirmed_flush_lsn
-----------+---------------+-----------+----------+---------------------
 my_slot   | test_decoding | logical   | zelenya  | 1/A26B3F58


SELECT * FROM pg_catalog.pg_logical_slot_get_changes('my_slot', NULL, NULL);
    lsn     | xid |    data
------------+-----+------------
 1/A26B3F88 | 990 | BEGIN 990
 1/A26BDA58 | 990 | COMMIT 990


SELECT slot_name, plugin, slot_type, database, confirmed_flush_lsn FROM pg_replication_slots;
 slot_name |    plugin     | slot_type | database | confirmed_flush_lsn
-----------+---------------+-----------+----------+---------------------
 my_slot   | test_decoding | logical   | zelenya  | 1/A26BDA58
```

PG_LOGICAL_SLOT_GET_CHANGES() - POSUN LSN SLOTU

```
INSERT INTO public.foo (a, b) VALUES (sqrt(2)/2, 'test data');


SELECT slot_name, plugin, slot_type, database, confirmed_flush_lsn FROM pg_replication_slots;
 slot_name |    plugin     | slot_type | database | confirmed_flush_lsn
-----------+---------------+-----------+----------+---------------------
 my_slot   | test_decoding | logical   | zelenya  | 1/A26BDA58


SELECT * FROM pg_catalog.pg_logical_slot_get_changes('my_slot', NULL, NULL);
    lsn     | xid |                                                  data
------------+-----+-------------------------------------------------------------------------------------------
 1/A26BDA58 | 991 | BEGIN 991
 1/A26BDAC0 | 991 | table public.foo: INSERT: foo_id[integer]:1 a[numeric]:0.707106781186548 b[text]:'test data'
 1/A26BDBE0 | 991 | COMMIT 991


SELECT slot_name, plugin, slot_type, database, confirmed_flush_lsn FROM pg_replication_slots;
 slot_name |    plugin     | slot_type | database | confirmed_flush_lsn
-----------+---------------+-----------+----------+---------------------
 my_slot   | test_decoding | logical   | zelenya  | 1/A26BDBE0
```

PG_LOGICAL_SLOT_PEEK_CHANGES() - "NÁHLED", LSN ZŮSTÁVÁ

```
INSERT INTO public.foo (a, b) VALUES (exp(1), 'test data');


SELECT slot_name, plugin, slot_type, database, confirmed_flush_lsn FROM pg_replication_slots;
 slot_name |    plugin     | slot_type | database | confirmed_flush_lsn
-----------+---------------+-----------+----------+---------------------
 my_slot   | test_decoding | logical   | zelenya  | 1/A26BDBE0


SELECT * FROM pg_catalog.pg_logical_slot_peek_changes('my_slot', NULL, NULL);
    lsn     | xid |                                                 data
------------+-----+-------------------------------------------------------------------------------------------------
 1/A26BDBE0 | 992 | BEGIN 992
 1/A26BDBE0 | 992 | table public.foo: INSERT: foo_id[integer]:2 a[numeric]:2.71828182845905 b[text]:'test data'
 1/A26BDCA8 | 992 | COMMIT 992


SELECT slot_name, plugin, slot_type, database, confirmed_flush_lsn FROM pg_replication_slots;
 slot_name |    plugin     | slot_type | database | confirmed_flush_lsn
-----------+---------------+-----------+----------+---------------------
 my_slot   | test_decoding | logical   | zelenya  | 1/A26BDBE0
```

PG_LOGICAL_SLOT_[GET|PEEK]_CHANGES()

```
SELECT slot_name, plugin, slot_type, database, confirmed_flush_lsn FROM pg_replication_slots;
 slot_name |    plugin     | slot_type | database | confirmed_flush_lsn
-----------+---------------+-----------+----------+---------------------
 my_slot   | test_decoding | logical   | zelenya  | 1/A26BDBE0
```

```
SELECT * FROM pg_catalog.pg_logical_slot_get_changes('my_slot', NULL, NULL);
    lsn     | xid |                                                 data
------------+-----+-------------------------------------------------------------------------------------------------
 1/A26BDBE0 | 992 | BEGIN 992
 1/A26BDBE0 | 992 | table public.foo: INSERT: foo_id[integer]:2 a[numeric]:2.71828182845905 b[text]:'test data'
 1/A26BDCA8 | 992 | COMMIT 992
```

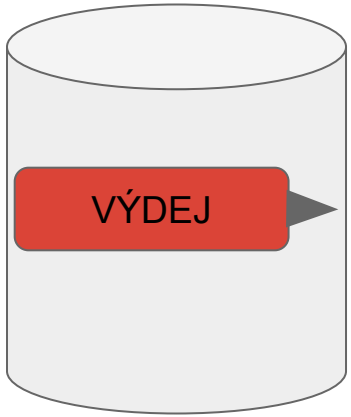
```
SELECT slot_name, plugin, slot_type, database, confirmed_flush_lsn FROM pg_replication_slots;
 slot_name |    plugin     | slot_type | database | confirmed_flush_lsn
-----------+---------------+-----------+----------+---------------------
 my_slot   | test_decoding | logical   | zelenya  | 1/A26BDCA8
```

Update může přenášet i další replikované sloupce

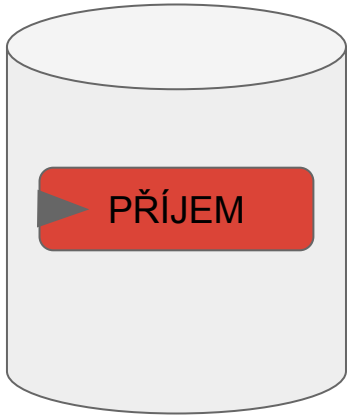
```
UPDATE public.foo SET b = 'Leonhard Euler' WHERE foo_id = 2;

SELECT * FROM pg_catalog.pg_logical_slot_get_changes('my_slot', NULL, NULL);
    lsn     | xid |                                                  data
------------+-----+-----------------------------------------------------------------------------------------------------
 1/A26BDCA8 | 993 | BEGIN 993
 1/A26BDCA8 | 993 | table public.foo: UPDATE: foo_id[integer]:2 a[numeric]:2.71828182845905 b[text]:'Leonhard Euler'
 1/A26BDD40 | 993 | COMMIT 993

SELECT slot_name, plugin, slot_type, database, confirmed_flush_lsn FROM pg_replication_slots;
 slot_name |    plugin     | slot_type | database | confirmed_flush_lsn
-----------+---------------+-----------+----------+---------------------
 my_slot   | test_decoding | logical   | zelenya  | 1/A26BDD40


SELECT pg_catalog.pg_drop_replication_slot('my_slot');
 pg_drop_replication_slot
--------------------------
```

ZABUDOVANÁ LOGICKÁ REPLIKACE

Publisher & Subscriber

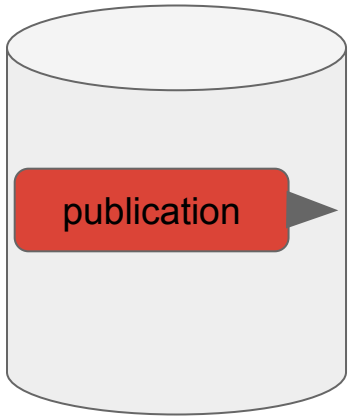


Publisher



Subscriber

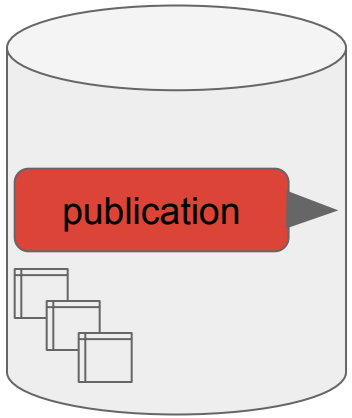
Publication & Subscription



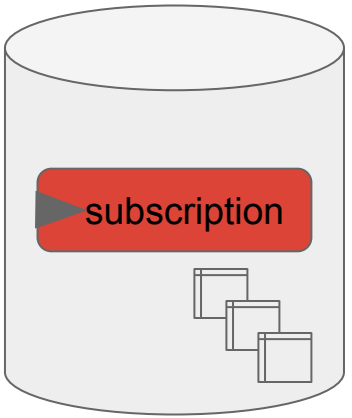
Publisher

Subscriber

Publication & Subscription - tables

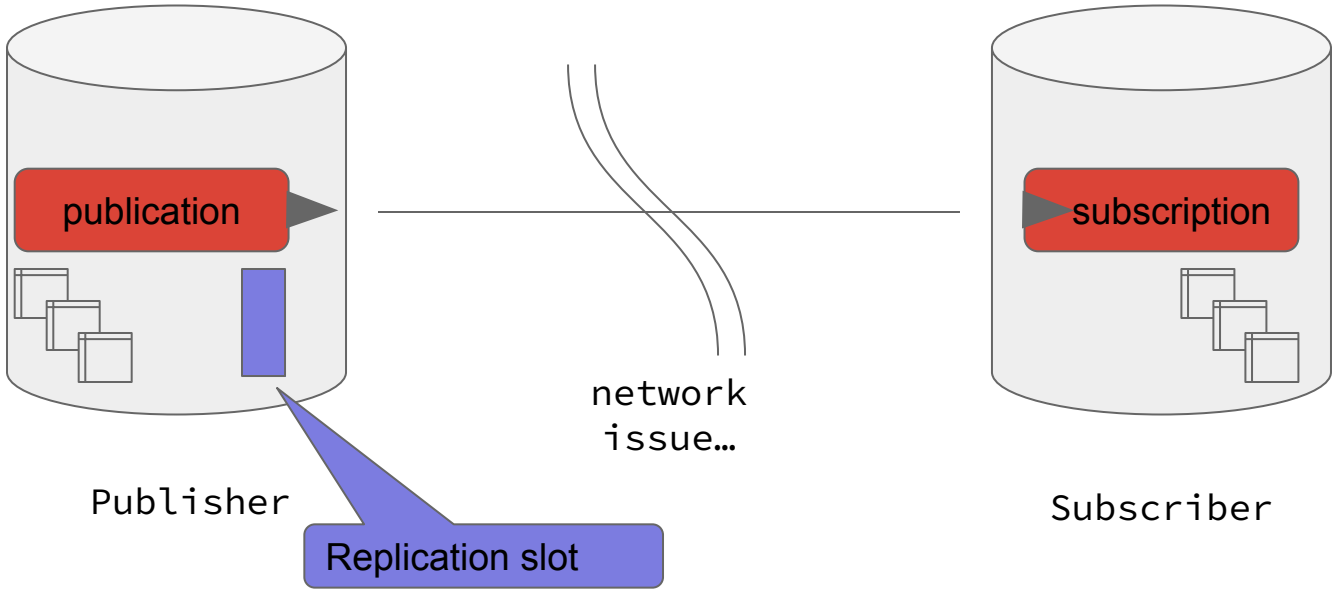


Publisher

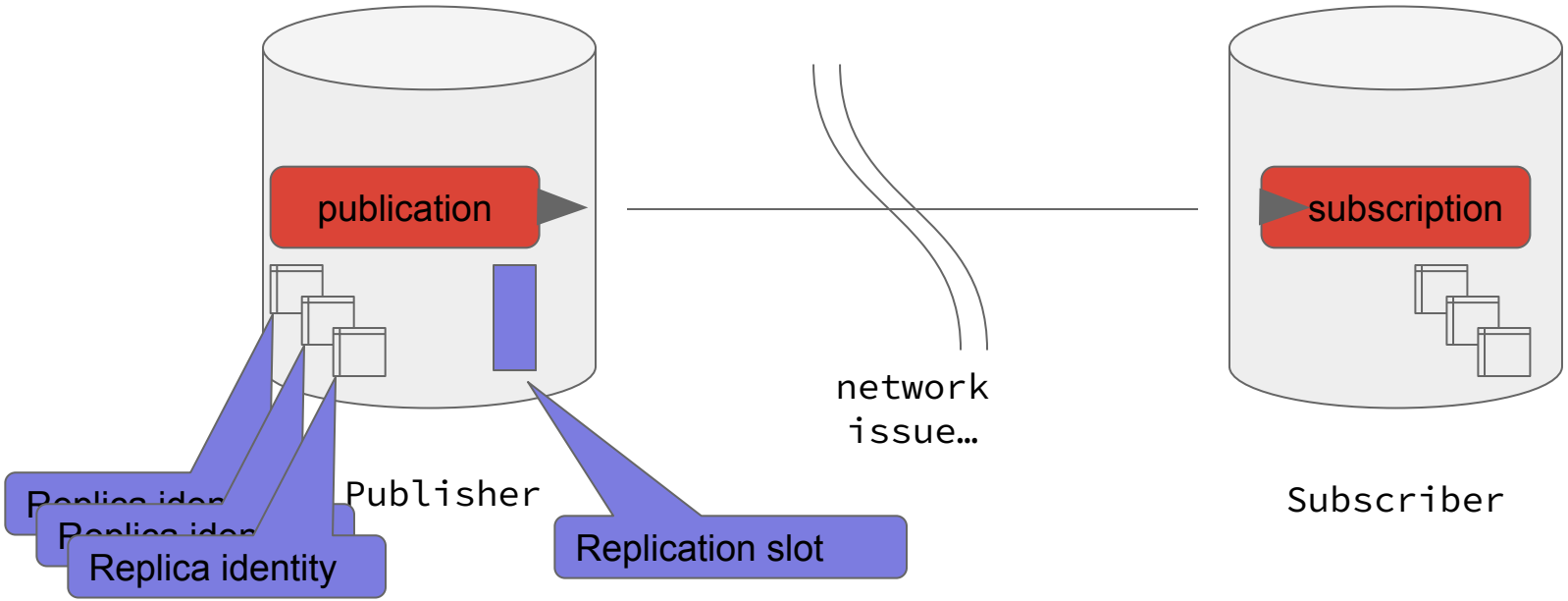


Subscriber

Replication slot



Replica identity



Vlastnosti logické replikace

- implementace Change Data Capture technologie
- WAL záznamy -> SQL příkazy pro změnu dat
- dodržuje pořadí transakcí
- data celého řádku (pg15 umí vybrané sloupce)
- úvodní kopie dat (výchozí volba)
- jednosměrná (produkty jako BDR umí multimaster)
- logical decoding on standby instance (pg 16)
- pouze pro DML
- nepřenáší
  - DDL
  - sekvence

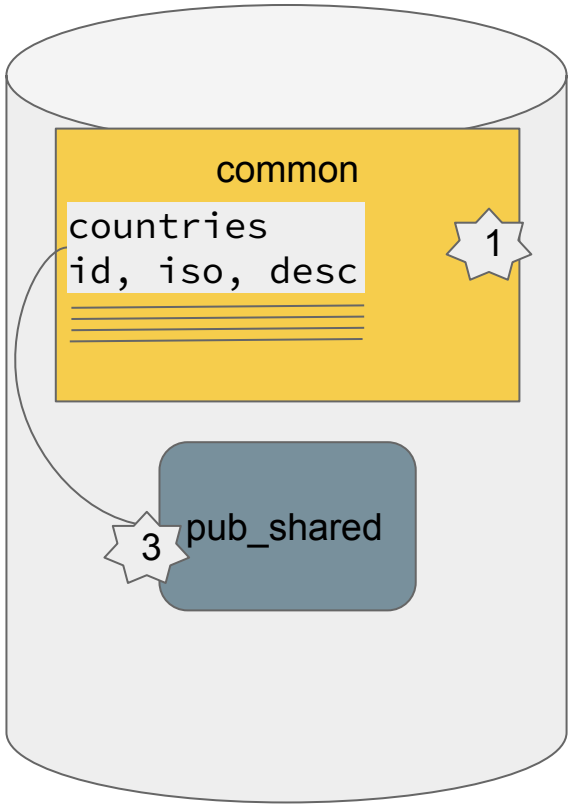
Limity/omezení logické replikace

- shoda plně kvalifikovaných jmen
- partitioning (pg 14+ via root partition)
- LOB (large objects) není podporován
- Truncate (pg11+) funguje
  - pozor na referenční integritu - cascade cizí klíče na tabulkách v publikaci

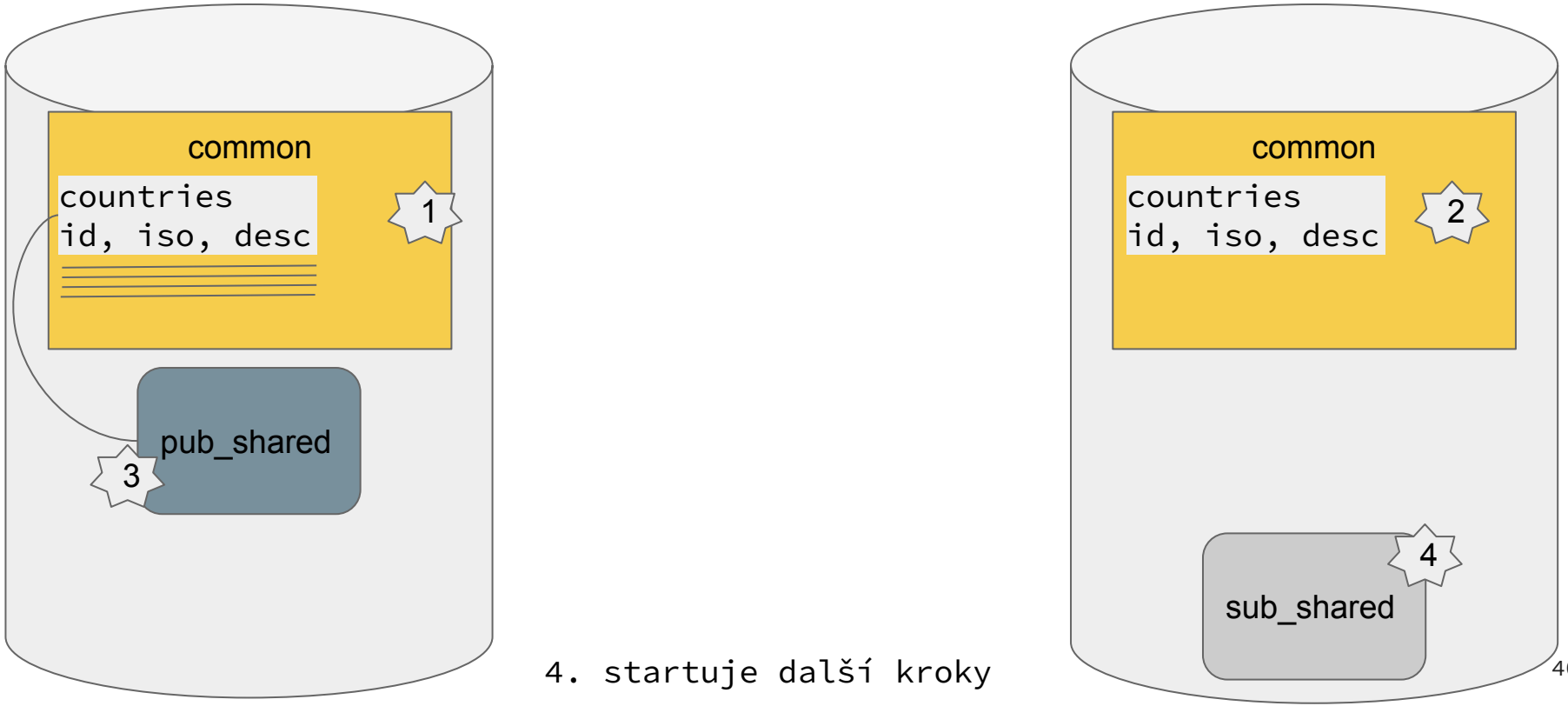
TABULKU SHODNÉHO PLNĚ KVALIFIKOVANÉHO JMÉNA


```
common.countries
```

Publication

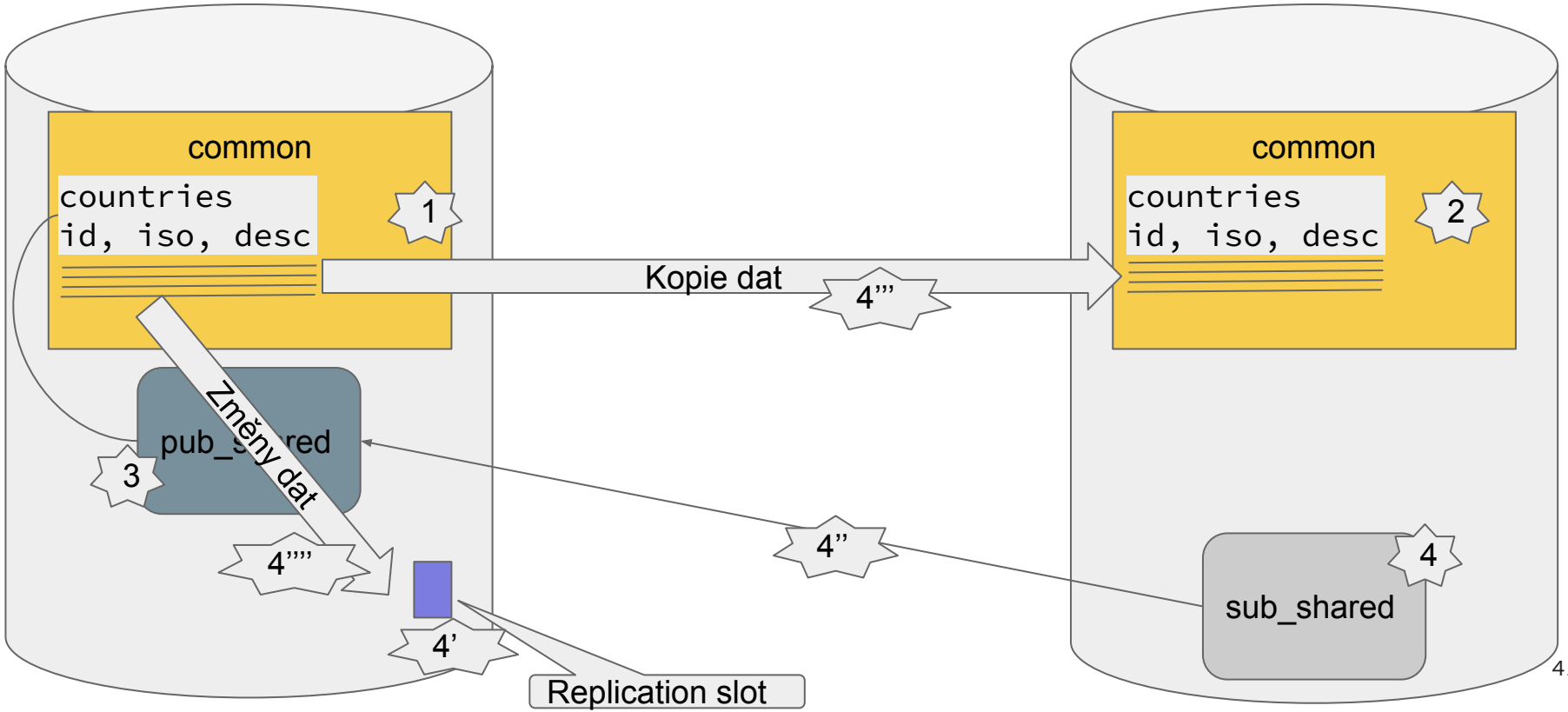


Subscription

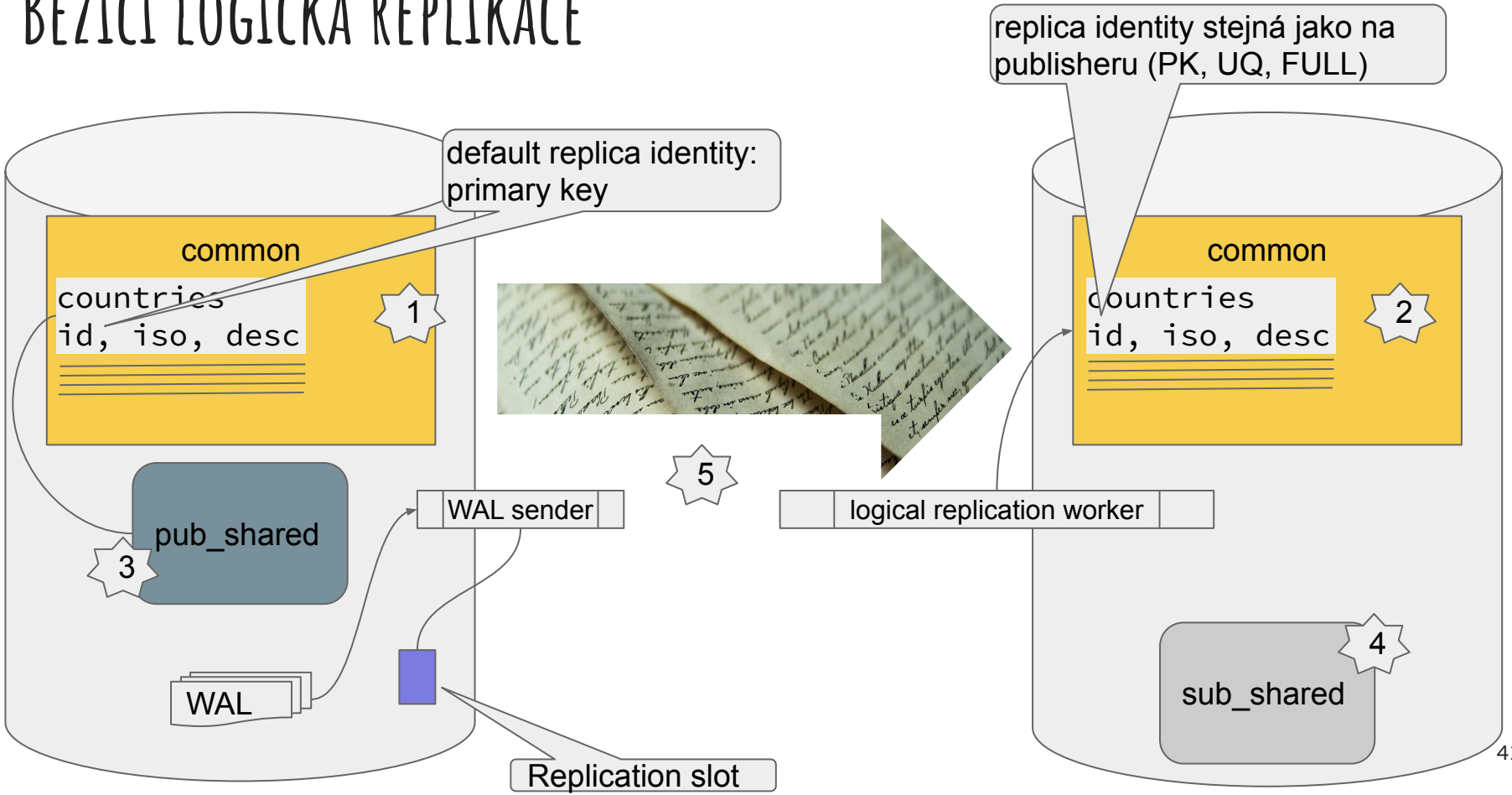


4. startuje další kroky

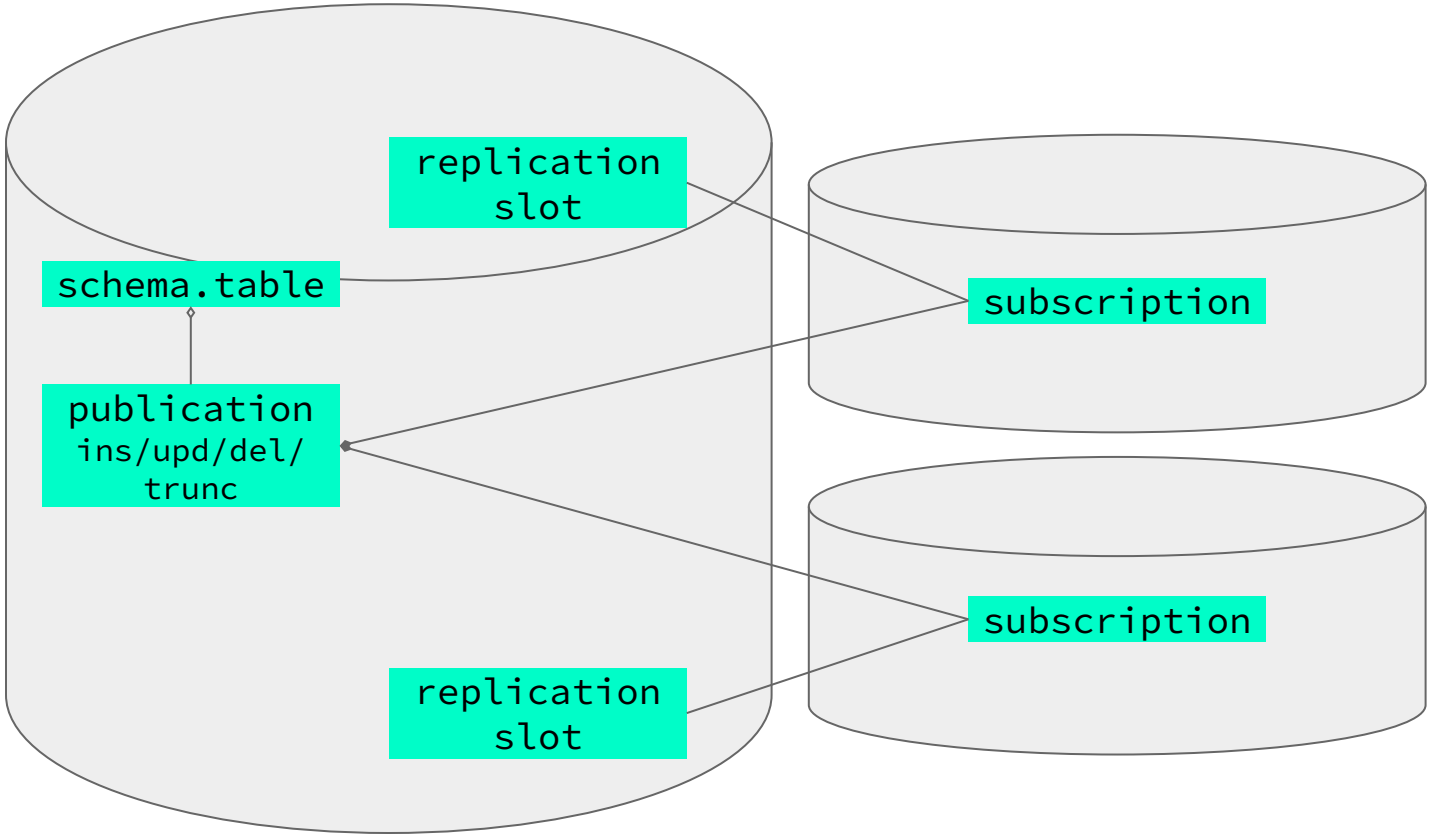
Inicializace vyvolaná založením subscription



Běžící logická replikace



Entity logické replikace



Ukázky

Přehled instancí pro ukázky

```
Ver Cluster  Port Status Owner    Data directory                  Log file
14  pg14main 5432 online postgres /var/lib/postgresql/14/pg14main /var/log/postgresql/postgresql-14-pg14main.log
15  pg15main 5433 online postgres /var/lib/postgresql/15/pg15main /var/log/postgresql/postgresql-15-pg15main.log
15  pg15test 5434 online postgres /var/lib/postgresql/15/pg15test /var/log/postgresql/postgresql-15-pg15test.log

$ psql -X -p 5432 -c "select current_setting('cluster_name'), current_setting('server_version');"

 current_setting |         current_setting
-----------------+---------------------------------
 14/pg14main     | 14.6 (Ubuntu 14.6-1.pgdg22.04+1)

 current_setting |         current_setting
-----------------+---------------------------------
 15/pg15main     | 15.1 (Ubuntu 15.1-1.pgdg22.04+1)

 current_setting |         current_setting
-----------------+---------------------------------
 15/pg15test     | 15.1 (Ubuntu 15.1-1.pgdg22.04+1)
```

Co potřebujeme?

- publisher: zdrojovou DB (instanci)
  - wal_level = logical
  - max_replication_slots (počet **subscriptions** které se připojí, standby, pg_basebackup, max_sync_workers_per_subscription)
  - max_wal_senders >= max_replication_slots (+ pg_basebackup, pg_receivewal…, standby…)
- subscriber: cílovou DB (instanci)
  - max_replication_slots (*)
  - max_logical_replication_workers (*)
  - max_worker_processes (počet subscr. + 1, + extenze…)

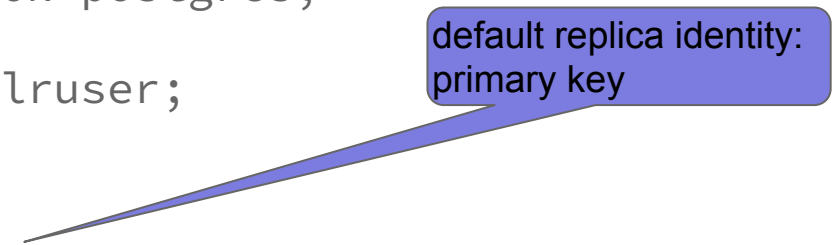
(*) nejméně podle počtu subscriptions, + max_sync_workers_per_subscription

PUBLISHER - DATABÁZE

```
-- Založení uživatelské role pro logickou replikaci.
CREATE ROLE lruser WITH LOGIN REPLICATION PASSWORD 'heslo';

-- Založení DB pro ukázky.
CREATE DATABASE test14;

-- Nastavení oprávnění pro připojení k databázi.
REVOKE CONNECT ON DATABASE test14 FROM PUBLIC;
GRANT CONNECT ON DATABASE test14 TO lruser;
```

PUBLISHER - TABULKA

```
CREATE SCHEMA skoleni AUTHORIZATION postgres;

GRANT USAGE ON SCHEMA skoleni TO lruser;

CREATE TABLE skoleni.weather (
    weather_id SERIAL PRIMARY KEY
  , ts TIMESTAMPTZ DEFAULT now()
  , temp FLOAT
  , pressure INTEGER
  , description TEXT
);

GRANT SELECT ON skoleni.weather TO lruser;
```



PUBLISHER - TESTOVACÍ DATA

```
pg_restore -v -a -d test14 skoleni_weather.dump
```

```
SELECT
    s.schemaname
  , s.sequenceowner
  , s.data_type
  , s.last_value
FROM pg_catalog.pg_sequences s
WHERE true
  AND s.schemaname = 'skoleni'
  AND s.sequencename = 'weather_weather_id_seq';
```

```
 schemaname | sequenceowner | data_type | last_value
------------+---------------+-----------+------------
 skoleni    | zelenya       | integer   |      38812
```

PUBLISHER - TESTOVACÍ DATA - NÁHLED

```
SELECT count(*), min(w.weather_id), max(w.weather_id), sum(w.weather_id) FROM skoleni.weather AS w;
```

```
 count | min |  max  |    sum
-------+-----+-------+-----------
 38812 |   1 | 38812 | 753205078
```

```
SELECT w.* FROM skoleni.weather w ORDER BY w.weather_id ASC LIMIT 3;
```

```
 weather_id |           ts           | temp | pressure |             description
------------+------------------------+------+----------+-------------------------------------
          1 | 1970-01-01 20:31:01+01 |  -10 |      810 | Thursday  01, 01 1970
          2 | 1970-01-02 12:56:55+01 |  -13 |      998 | Prev: Thursday  01, 01 1970 20:31:01
          3 | 1970-01-02 14:26:32+01 |  -10 |     1014 | Prev: Friday    02, 01 1970 12:56:55
```

```
SELECT q.* FROM ( SELECT w.* FROM skoleni.weather w ORDER BY w.weather_id DESC LIMIT 3 ) AS Q
ORDER BY weather_id ASC;
```

```
 weather_id |           ts           | temp | pressure |             description
------------+------------------------+------+----------+-------------------------------------
      38810 | 2023-01-31 06:15:04+01 |   -5 |      996 | Prev: Monday    30, 01 2023 22:38:02
      38811 | 2023-01-31 07:12:20+01 |   -8 |      942 | Prev: Tuesday   31, 01 2023 06:15:04
      38812 | 2023-02-01 06:04:05+01 |   -7 |      857 | Prev: Tuesday   31, 01 2023 07:12:20
```

PUBLISHER - PUBLICATION

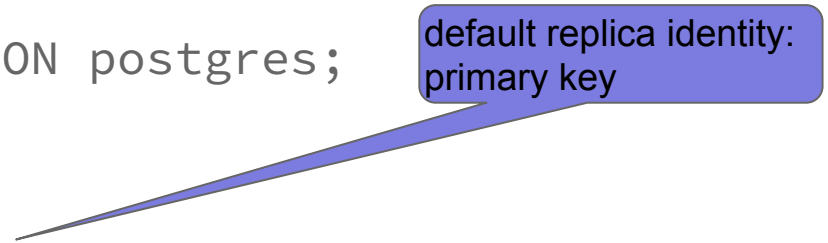
```
CREATE PUBLICATION pub_skoleni FOR TABLE skoleni.weather
  WITH ( publish = 'insert, update, delete, truncate' ); -- defaults
```

Chybné nastavené wal_level: je zobrazeno varování:

```
psql:02_ukazka_5432_pub.sql:81: WARNING:  wal_level is insufficient to publish logical changes
HINT:  Set wal_level to logical before creating subscriptions.
CREATE PUBLICATION
```

Subscriber - databáze, schema, tabulka

```
CREATE DATABASE test15;

CREATE SCHEMA skoleni AUTHORIZATION postgres;

CREATE TABLE skoleni.weather (
    weather_id SERIAL PRIMARY KEY
  , temp FLOAT
  , pressure INTEGER
  , description TEXT
  , measured TIMESTAMPTZ DEFAULT now()
);
```



Subscriber - SUBSCRIPTION

```
CREATE SUBSCRIPTION sub_skoleni
  CONNECTION 'host=127.0.0.1 port=5432 user=lruser password=heslo dbname=test14'
  PUBLICATION pub_skoleni;
```

```
psql:03_ukazka_5433_sub.sql:65: NOTICE:  created replication slot "sub_skoleni" on publisher
CREATE SUBSCRIPTION
```

- Connection - LibPQ podporované metody
  - pg_service
  - pgpass

Subscriber - testovací data - náhled

```
CREATE SUBSCRIPTION sub_skoleni
  CONNECTION 'host=127.0.0.1 port=5432 user=lruser password=heslo dbname=test14'
  PUBLICATION pub_skoleni;
```

```
SELECT count(*), min(w.weather_id), max(w.weather_id), sum(w.weather_id)
FROM skoleni.weather AS w;
 count | min | max | sum
-------+-----+-----+-----
     0 |     |     |
```

```
SELECT s.schemaname, s.sequenceowner, s.data_type, s.last_value
FROM pg_catalog.pg_sequences s
WHERE true AND s.schemaname = 'skoleni' AND s.sequencename = 'weather_weather_id_seq';
 schemaname | sequenceowner | data_type | last_value
------------+---------------+-----------+------------
 skoleni    | zelenya       | integer   |
```

Subscriber - initial synchronization

```
SELECT d.datname, s.subname, sr.srrelid::regclass, sr.srsubstate FROM pg_catalog.pg_subscription AS s
INNER JOIN pg_subscription_rel AS sr ON s.oid = sr.srsubid
INNER JOIN pg_catalog.pg_database d  ON s.subdbid = d.oid;
 datname |   subname   |     srrelid     | srsubstate
---------+-------------+-----------------+------------
 test15  | sub_skoleni | skoleni.weather | i
```

Q1

```
SELECT count(*), min(w.weather_id), max(w.weather_id), sum(w.weather_id) FROM skoleni.weather AS w;
 count | min | max | sum
-------+-----+-----+-----
     0 |     |     |
```

Q2

```
 datname |   subname   |     srrelid     | srsubstate
---------+-------------+-----------------+------------
 test15  | sub_skoleni | skoleni.weather | d
```

Q1

```
 count | min | max | sum
-------+-----+-----+-----
     0 |     |     |
```

Q2

Subscriber - initial synchronization - hotovo

```
SELECT d.datname, s.subname, sr.srrelid::regclass, sr.srsubstate
FROM pg_catalog.pg_subscription AS s
INNER JOIN pg_subscription_rel AS sr
  ON s.oid = sr.srsubid
INNER JOIN pg_catalog.pg_database d
  ON s.subdbid = d.oid;
 datname |   subname   |     srrelid     | srsubstate
---------+-------------+-----------------+------------
 test15  | sub_skoleni | skoleni.weather | r
```

```
SELECT
    count(*)
  , min(w.weather_id)
  , max(w.weather_id)
  , sum(w.weather_id)
FROM skoleni.weather AS w;
 count | min |  max  |    sum
-------+-----+-------+-----------
 38812 |   1 | 38812 | 753205078
```

Data jsou již synchronizovány

Subscriber - sequence

```
SELECT s.schemaname, s.sequenceowner, s.data_type, s.last_value FROM pg_catalog.pg_sequences s
WHERE s.schemaname = 'skoleni' AND s.sequencename = 'weather_weather_id_seq';
 schemaname | sequenceowner | data_type | last_value
------------+---------------+-----------+------------
 skoleni    | zelenya       | integer   |
```

Logická replikace sequence nepřenáší

```
SELECT w.* FROM skoleni.weather w ORDER BY w.weather_id ASC LIMIT 3;
 weather_id |           ts           | temp | pressure |             description
------------+------------------------+------+----------+-------------------------------------
          1 | 1970-01-01 20:31:01+01 |  -10 |      810 | Thursday  01, 01 1970
          2 | 1970-01-02 12:56:55+01 |  -13 |      998 | Prev: Thursday  01, 01 1970 20:31:01
          3 | 1970-01-02 14:26:32+01 |  -10 |     1014 | Prev: Friday    02, 01 1970 12:56:55
```

```
SELECT q.* FROM ( SELECT w.* FROM skoleni.weather w ORDER BY w.weather_id DESC LIMIT 3 ) AS Q ORDER BY
weather_id ASC;
 weather_id |           ts           | temp | pressure |             description
------------+------------------------+------+----------+-------------------------------------
      38810 | 2023-01-31 06:15:04+01 |   -5 |      996 | Prev: Monday    30, 01 2023 22:38:02
      38811 | 2023-01-31 07:12:20+01 |   -8 |      942 | Prev: Tuesday   31, 01 2023 06:15:04
      38812 | 2023-02-01 06:04:05+01 |   -7 |      857 | Prev: Tuesday   31, 01 2023 07:12:20
```

Monitoring

Monitoring

Postgres poskytuje funkce a pohledy v systémovém katalogu vhodné pro monitoring:

- pg_catalog.pg_replication_slots
- pg_catalog.pg_stat_replication
- pg_catalog.pg_stat_replication_slots
- pg_catalog.pg_subscription
- pg_catalog.pg_subscription_rel
- pg_catalog.pg_stat_subscription

Stav na publisheru

```
SELECT
    prs.slot_name
  , prs.slot_type
  , prs.plugin
  , prs.database
  , prs.temporary
  , prs.active
  , psr.pid
  , psr.client_addr
  , psr.usename
  , psr.application_name
  , psr.state
  , pg_xact_commit_timestamp(prs.catalog_xmin) AS cat_xmin_tm
  , pg_size_pretty(pg_wal_lsn_diff(pg_current_wal_lsn(), prs.restart_lsn)) AS slot_size
  , pg_current_wal_lsn() AS current_lsn
  , prs.restart_lsn
  , psr.sent_lsn
  , pg_size_pretty(pg_wal_lsn_diff(pg_current_wal_lsn(), psr.sent_lsn)) AS sent_diff
  , write_lsn, pg_size_pretty(pg_wal_lsn_diff(pg_current_wal_lsn(), psr.write_lsn)) AS write_diff
  , replay_lsn, pg_size_pretty(pg_wal_lsn_diff(pg_current_wal_lsn(), psr.replay_lsn)) AS replay_diff
  , psr.write_lag
  , psr.flush_lag
  , psr.replay_lag
FROM pg_catalog.pg_replication_slots prs
FULL JOIN pg_stat_replication psr ON prs.active_pid = psr.pid
ORDER BY prs.slot_name, psr.pid;
```

```
┌[ RECORD 1 ]──────┬──────────────┐
│ slot_name        │ sub_skoleni  │
│ slot_type        │ logical      │
│ plugin           │ pgoutput     │
│ database         │ test14       │
│ temporary        │ f            │
│ active           │ t            │
│ pid              │ 13740        │
│ client_addr      │ 127.0.0.1    │
│ usename          │ lruser       │
│ application_name │ sub_skoleni  │
│ state            │ streaming    │
│ cat_xmin_tm      │ «¤»          │
│ slot_size        │ 10 MB        │
│ current_lsn      │ 1/7E0000D8   │
│ restart_lsn      │ 1/7D588308   │
│ sent_lsn         │ 1/7E0000D8   │
│ sent_diff        │ 0 bytes      │
│ write_lsn        │ 1/7E0000D8   │
│ write_diff       │ 0 bytes      │
│ replay_lsn       │ 1/7E0000D8   │
│ replay_diff      │ 0 bytes      │
│ write_lag        │ «¤»          │
│ flush_lag        │ «¤»          │
│ replay_lag       │ «¤»          │
└──────────────────┴──────────────┘
```

Problémy

Časté obtíže a jejich řešení

- “nerozjelo se to”
  - oprávnění pro uživatele použitého v subscription connection
    - login, replication
    - usage na schema a select na tabulky (copy_data=true) je default
- “zastavilo se to”
  - Parametery: wal_sender_timeout, wal_receiver_timeout (default 1 min)
  - constraint na straně subscription
    - logy
- publisher nebo subscriber jsou trvale nedostupné
  - například rušení jedné lokality

"ONO SE TO ZASTAVILO" - ANALÝZA

Publisher:

```
-[ RECORD 1 ]-----+-----------------------------
slot_name         | sub_skoleni
slot_type         | logical
plugin            | pgoutput
database          | test14
temporary         | f
active            | f
pid               | «¤»
client_addr       | «¤»
usename           | «¤»
application_name  | «¤»
state             | «¤»
cat_xmin_tm       | 2023-01-28 01:17:43.040343+01
slot_size         | 16 kB
current_lsn       | 1/7E004030
restart_lsn       | 1/7E0001C0
sent_lsn          | «¤»
sent_diff         | «¤»
write_lsn         | «¤»
write_diff        | «¤»
replay_lsn        | «¤»
replay_diff       | «¤»
write_lag         | «¤»
flush_lag         | «¤»
replay_lag        | «¤»
```

Subscriber:

```
2023-01-28 01:32:41.260 CET [14960] LOG:  logical replication apply worker for
subscription "sub_skoleni" has started


2023-01-28 01:32:41.280 CET [14960] ERROR:  duplicate key value violates unique
constraint "weather_pkey"


2023-01-28 01:32:41.280 CET [14960] DETAIL:  Key (weather_id)=(0) already
exists.


2023-01-28 01:32:41.282 CET [13718] LOG:  background worker "logical replication
worker" (PID 14960) exited with exit code 1
```

"ONO SE TO ZASTAVILO" - ANALÝZA - POSTGRESQL 15

Publisher:

```
-[ RECORD 1 ]-----+-----------------------------
slot_name         | sub_skoleni
slot_type         | logical
plugin            | pgoutput
database          | test14
temporary         | f
active            | f
pid               | «¤»
client_addr       | «¤»
usename           | «¤»
application_name  | «¤»
state             | «¤»
cat_xmin_tm       | 2023-01-28 01:17:43.040343+01
slot_size         | 16 kB
current_lsn       | 1/7E004030
restart_lsn       | 1/7E0001C0
sent_lsn          | «¤»
sent_diff         | «¤»
write_lsn         | «¤»
write_diff        | «¤»
replay_lsn        | «¤»
replay_diff       | «¤»
write_lag         | «¤»
flush_lag         | «¤»
replay_lag        | «¤»
```

Subscriber:

```
2023-01-28 01:23:15.623 CET [14355] LOG:  logical replication apply worker for
subscription "sub_skoleni" has started


2023-01-28 01:23:15.652 CET [14355] ERROR:  duplicate key value violates unique
constraint "weather_pkey"


2023-01-28 01:23:15.652 CET [14355] DETAIL:  Key (weather_id)=(0) already
exists.


2023-01-28 01:23:15.652 CET [14355] CONTEXT:  processing remote data for
replication origin "pg_16730" during message type "INSERT" for replication
target relation "skoleni.weather" in transaction 1638, finished at 1/7E003FB0


2023-01-28 01:23:15.656 CET [1916] LOG:  background worker "logical replication
worker" (PID 14355) exited with exit code 1
```

Integritní omezení na subscriberu platí... FK, constraint...

```
Subscriber:
[local]:5433 zelenya@test15:17677
=# ALTER TABLE skoleni.weather ADD CHECK (weather_id > 0);
ALTER TABLE


Publisher:
[local]:5432 zelenya@test14:17788
=# INSERT INTO skoleni.weather AS w (weather_id, ts, temp, pressure, description)
VALUES (0, '1970-01-01 00:00:00+00', 0, 1000, 'initial');
INSERT 0 1


Log sbscriberu:
2023-01-28 02:05:26.180 CET [17810] LOG:  logical replication apply worker for subscription "sub_skoleni" has started
2023-01-28 02:05:26.208 CET [17810] ERROR:  new row for relation "weather" violates check constraint "weather_weather_id_check"
2023-01-28 02:05:26.208 CET [17810] DETAIL:  Failing row contains (0, 1970-01-01 01:00:00+01, 0, 1000, initial).
2023-01-28 02:05:26.208 CET [17810] CONTEXT:  processing remote data for replication origin "pg_16811" during message type
"INSERT" for replication target relation "skoleni.weather" in transaction 1803, finished at 1/7F226EA8
2023-01-28 02:05:26.212 CET [14827] LOG:  background worker "logical replication worker" (PID 17810) exited with exit code 1
```

SUBSCRIBER - ZRUŠENÍ SUBSCRIPTION

- funguje konektivita subsciber -> publisher
  - ALTER SUBSCRIPTION sub_skoleni DISABLE;
  - DROP SUBSCRIPTION sub_skoleni;
    - automaticky odstraní replikační slot na straně publisheru
- **NE**funguje konektivita subsciber -> publisher
  - ALTER SUBSCRIPTION sub_skoleni DISABLE;
  - ALTER SUBSCRIPTION sub_skoleni SET (slot_name = NONE);
  - DROP SUBSCRIPTION sub_skoleni;
- pokud publisher existuje (jen nemá spojení)
  - je třeba zrušit replikační slot, jinak by WAL logy trvale jen rostly
  - SELECT pg_catalog.pg_drop_replication_slot('sub_skoleni');

PG_LOGICAL_SLOT_...

investigace problémů / přeskočení transakcí

- pg_logical_slot_peek_changes
- pg_logical_slot_get_changes

Konec, pauza na oběd

Fotka od SecularEthos z Pixabay

PG_WALDUMP

```
postgres@alesnbbs:~/14/pg14main_arch$ /usr/lib/postgresql/14/bin/pg_waldump -p /var/lib/postgresql/14/pg14main_arch/ 000000010000000000000062 000000010000000000000063
rmgr: Standby       len (rec/tot):      50/       50, tx:             0, lsn: 0/62000028, prev 0/6102B888, desc: RUNNING_XACTS nextXid 1194 latestCompletedXid 1193 oldestRunningXid 1194
rmgr: Standby       len (rec/tot):      42/       42, tx:   1194, lsn: 0/62000060, prev 0/62000028, desc: LOCK xid 1194 db 13797 rel 16905
rmgr: Standby       len (rec/tot):      42/       42, tx:   1194, lsn: 0/62000090, prev 0/62000060, desc: LOCK xid 1194 db 13797 rel 16909
rmgr: Standby       len (rec/tot):      42/       42, tx:   1194, lsn: 0/620000C0, prev 0/62000090, desc: LOCK xid 1194 db 13797 rel 16904
rmgr: Standby       len (rec/tot):      42/       42, tx:   1194, lsn: 0/620000F0, prev 0/620000C0, desc: LOCK xid 1194 db 13797 rel 16911
rmgr: Standby       len (rec/tot):      42/       42, tx:   1194, lsn: 0/62000120, prev 0/620000F0, desc: LOCK xid 1194 db 13797 rel 16910
rmgr: Heap2         len (rec/tot):      56/       56, tx:   1194, lsn: 0/62000150, prev 0/62000120, desc: PRUNE latestRemovedXid 0 nredirected 0 ndead 1, blkref #0: rel 1663/13797/1249
blk 16
rmgr: Heap2         len (rec/tot):      60/       60, tx:   1194, lsn: 0/62000188, prev 0/62000150, desc: NEW_CID rel 1663/13797/2610; tid 0/41; cmin: 4294967295, cmax: 0, combo:
4294967295
rmgr: Heap          len (rec/tot):      54/       54, tx:   1194, lsn: 0/620001C8, prev 0/62000188, desc: DELETE off 41 flags 0x00 KEYS_UPDATED , blkref
…
rmgr: XLOG          len (rec/tot):      114/   114, tx:             0, lsn: 0/63000098, prev 0/63000060, desc: CHECKPOINT_ONLINE redo 0/63000060; tli 1; prev tli 1; fpw true; xid
0:1195; oid 25096; multi 1; offset 0; oldest xid 726 in DB 1; oldest multi 1 in DB 1; oldest/newest commit timestamp xid: 0/0; oldest running xid 1195; online
rmgr: Standby       len (rec/tot):      50/       50, tx:             0, lsn: 0/63000110, prev 0/63000098, desc: RUNNING_XACTS nextXid 1195 latestCompletedXid 1194 oldestRunningXid 1195
rmgr: Storage       len (rec/tot):      42/       42, tx:             0, lsn: 0/63000148, prev 0/63000110, desc: CREATE base/16889/16913
rmgr: Heap2         len (rec/tot):      60/       60, tx:   1195, lsn: 0/63000178, prev 0/63000148, desc: NEW_CID rel 1663/16889/1259; tid 0/9; cmin: 0, cmax: 4294967295, combo:
4294967295
rmgr: Heap          len (rec/tot):      54/  2154, tx:      1195, lsn: 0/630001B8, prev 0/63000178, desc: INSERT off 9 flags 0x00, blkref #0: rel 1663/16889/1259 blk 0 FPW
rmgr: Btree         len (rec/tot):      53/  2493, tx:      1195, lsn: 0/63000A28, prev 0/630001B8, desc: INSERT_LEAF off 120, blkref #0: rel 1663/16889/2662 blk 2 FPW
rmgr: Btree         len (rec/tot):      53/  5685, tx:      1195, lsn: 0/630013E8, prev 0/63000A28, desc: INSERT_LEAF off 29, blkref #0: rel 1663/16889/2663 blk 1 FPW
rmgr: Btree         len (rec/tot):      53/  3793, tx:      1195, lsn: 0/63002A38, prev 0/630013E8, desc: INSERT_LEAF off 185, blkref #0: rel 1663/16889/3455 blk 4 FPW
rmgr: Heap2         len (rec/tot):      60/       60, tx:   1195, lsn: 0/63003910, prev 0/63002A38, desc: NEW_CID rel 1663/16889/1249; tid 54/39; cmin: 0, cmax: 4294967295, combo:
4294967295
rmgr: Heap2         len (rec/tot):      60/       60, tx:   1195, lsn: 0/63003950, prev 0/63003910, desc: NEW_CID rel 1663/16889/1249; tid 54/40; cmin: 0, cmax: 4294967295, combo:
4294967295
rmgr: Heap2         len (rec/tot):      60/       60, tx:   1195, lsn: 0/63003990, prev 0/63003950, desc: NEW_CID rel 1663/16889/1249; tid 54/41; cmin: 0, cmax: 4294967295, combo:
4294967295
rmgr: Heap2         len (rec/tot):      61/  6153, tx:      1195, lsn: 0/630039D0, prev 0/63003990, desc: MULTI_INSERT 3 tuples flags 0x02, blkref #0: rel 1663/16889/1249 blk 54 FPW
rmgr: Btree         len (rec/tot):      53/  5041, tx:      1195, lsn: 0/630051F8, prev 0/630039D0, desc: INSERT_LEAF off 141, blkref #0: rel 1663/16889/2658 blk 14 FPW
rmgr: Btree         len (rec/tot):      53/  1073, tx:      1195, lsn: 0/630065C8, prev 0/630051F8, desc: INSERT_LEAF off 49, blkref #0: rel 1663/16889/2659 blk 10 FPW
rmgr: Btree         len (rec/tot):      72/       72, tx:   1195, lsn: 0/63006A00, prev 0/630065C8, desc: INSERT_LEAF off 142, blkref #0: rel 1663/16889/2658 blk 14
rmgr: Btree         len (rec/tot):      64/       64, tx:   1195, lsn: 0/63006A48, prev 0/63006A00, desc: INSERT_LEAF off 50, blkref #0: rel 1663/16889/2659 blk 10
rmgr: Btree         len (rec/tot):      72/       72, tx:   1195, lsn: 0/63006A88, prev 0/63006A48, desc: INSERT_LEAF off 141, blkref #0: rel 1663/16889/2658 blk 14
rmgr: Btree         len (rec/tot):      64/       64, tx:   1195, lsn: 0/63006AD0, prev 0/63006A88, desc: INSERT_LEAF off 51, blkref #0: rel 1663/16889/2659 blk 10
```

PG_WALFILE_NAME_OFFSET()

```
pg_waldump -r Heap -p /var/lib/postgresql/14/pg14main_arch/ 000000020000000100000051
rmgr: Heap       len (rec/tot):  89/  89, tx:    1432, lsn: 1/51000058, prev 1/51000028, desc:
INSERT off 110 flags 0x08, blkref #0: rel 1663/16889/16914 blk 1
```

```
test14=# select * from pg_walfile_name_offset('1/51000058');
        file_name         | file_offset
--------------------------+-------------
 000000020000000100000051 |          88
```

```
postgres@alesnbbs:~/14$ xxd --seek 88 --len 89
/var/lib/postgresql/14/pg14main_arch/000000020000000100000051
00000058: 5900 0000 9805 0000 2800 0051 0100 0000  Y.......(..Q....
00000068: 000a 0000 72b7 bb6e 0020 2800 7f06 0000  ....r..n. (.....
00000078: f941 0000 1242 0000 0100 0000 ff03 0200  .A...B..........
00000088: 0208 1800 e600 0000 3d32 3032 332d 3031  ........=2023-01
00000098: 2d32 3320 3233 3a35 303a 3033 2e35 3234  -23 23:50:03.524
000000a8: 3138 322b 3031 6e00 08                182+01n..
```

Na co nezapomenout

- *Zakladni prikazy*
  - *disable = suspend (data ze slotu nezmizi)*
  - *jak dropnout subscr. kdyz se nemuze pripojit na publisher pro dropnuti slotu*
- *refresh subscr - kdy je potreba*
- *poradi pridavani/odebirani sloupcu z repl. tabulek*
- constrainity na subscriberu se uplatňují
- *práva pro roli nastavenou v subscription*
  - *replication*
  - *select na zdrojové tabulky (jen pokud copy_data = true)*

Obvyklé problémy a jejich řešení

- *UPDATE přenáší může celý řádek, nikoliv jen změněné sloupce* (Pg15 - všechny vybrané sloupce)
- V rámci jedné instance nejde založit v jedné DB subscription s automatickým vytvořením replikačního slotu v druhé z databází
- *práva pro roli nastavenou v subscription*
  - *replication*
  - *select na zdrojové tabulky (jen pokud copy_data = true)*
- *zapomenute sloty - WAL roste ad limitum*