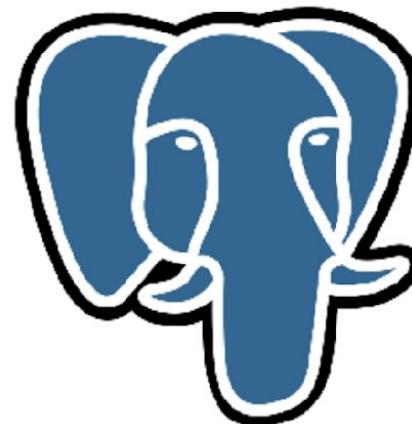


PostgreSQL 11 ile Gelen Yenilikler



PGDay Türkiye

22 Eylül 2018

Günce Kaya

guncenkaya.blogspot.com

PostgreSQL 11 Release Notes

<https://www.postgresql.org/docs/11/static/release-11.html>



İçerik

1. Partitioning

- Hash Partition
- Partition Key Güncellemesi
- Partition Tablolarında Index
- Default Partition
- Partition Tablolarında Unique Key ve Foreign Key
- PartitionWise Query Planning

2. Include Indexes

3. Arttırılmış Paralellik

4. Default Roles

5. WAL Size

6. Backup and Streaming Replication

- Logical Replication'da TRUNCATE
- Base Backup - Unlogged ve Temporary Tablolar
- Base Backup ile Replication Slot Oluşturulması

7. Alter Table Add Column

8. SHA-2 Fonksiyonları

9. Stored Procedure

10. Group Access on Data Folder

11. Insert On Conflict Do Nothing

12. Pg_stat statement İvileştirmesi

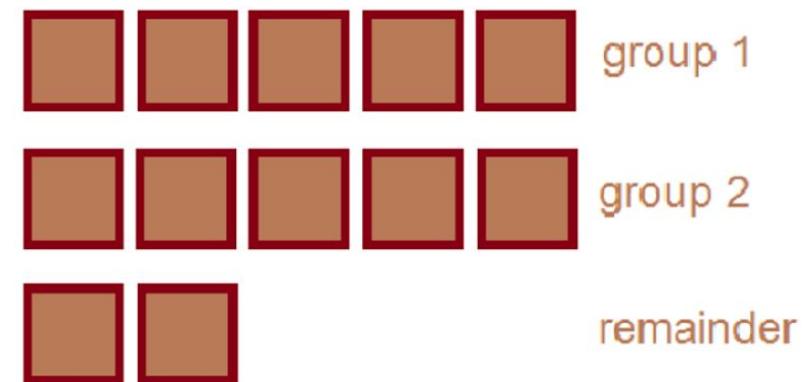


PARTITION



HASH PARTITION

- Hash değere göre verileri partition tablolarına yerleştirir.
- MODULUS, REMAINDER



HASH PARTITION

```
CREATE TABLE <partitioned_table>
  (col1 data_type,
  ...,
  ...)
PARTITION BY HASH(<hash_key>);
```

```
CREATE TABLE <partition_table>
PARTITION OF <partitioned_table>
FOR VALUES WITH (MODULUS X, REMAINDER Y);
```

hash_key = sütun adı

HASH PARTITION

CREATE TABLE ->

```
[pg11] # create table part_t(date_ date)
partition by hash (date_);
CREATE TABLE
```

CREATE
PARTITION ->
TABLES

```
[pg11] # create table part_t1
partition of part_t for values with (modulus 3, remainder 0);
CREATE TABLE
[pg11] # create table part_t2
partition of part_t for values with (modulus 3, remainder 1);
CREATE TABLE
[pg11] # create table part_t3
partition of part_t for values with (modulus 3, remainder 2);
CREATE TABLE
```

INSERT 10000 ->



```
[pg11] # insert into part_t
      select i from
      generate_Series('20180101',
                      '20190202',
                      '1 hour'::interval
      ) as i;
INSERT 0 9529
```

HASH PARTITION

```
[pa11] # select count(*) from part_t1;  
count  
-----  
 2904  
1 row)
```

```
[pg11] # select count(*) from part_t2;  
count  
-----  
 3144  
1 row)
```

```
[pg11] # select count(*) from part_t3;  
count  
-----  
 3481  
1 row)
```

PARTITION KEY GÜNCELLENMESİ

- PG10'da sadece partition içi veri güncellenmesi
- PG11'de partition tabloları arasında verilerin güncellenmesi



PARTITION TABLOLARINDA INDEX



- PG10'da manuel index yapıliyordu
- PG11'de partitioned table -> partition table
 - CREATE INDEX
 - DROP INDEX
- Partition tablolarına ait index oluşturulabilir.
- Aynı isimde index varsa!
- Lock seviyesi -> **ShareLock!**

PARTITION TABLOLARINDA INDEX



- PG10'da manuel index yapıliyordu
- PG11'de partitioned table -> partition table
 - CREATE INDEX
 - DROP INDEX
- Partition tablolarına ait index oluşturulabilir.
- Aynı isimde index varsa!
- Lock seviyesi -> **ShareLock!**
- Partition tablolarında otomatik oluşturulmuş indexler o tablo için manuel silinmez.

DEFAULT PARTITION

- Partition kriterlerinin dışında kalanlar
- Diğer partitionlardan önce yaratıldıysa DETACH!
 - Partition tablolarını oluştur.
 - ATTACH Default Partition.
- HASH partitionlarla kullanılmaz.



PARTITION TABLOLARDADA UNIQUE KEY VE FOREIGN KEY

- PG10'da Partition tablosu üzerinde UNIQUE KEY oluşturulamazdı.
- PG11 ile Partitionlar arası UNIQUE KEY
 - Mevcut/olşturulacak child tablolarda otomatik oluşturulur.
- Foreign Key desteği geldi!

ENABLE_PARTITIONWISE_AGGREGATE

- Default off
- Query Planner'i her bir partition tablosunun AGGREGATE yapması için değiştirebiliriz.
- Partition tablolarında GRUPLAMA ve AGGREGATION işlemlerinin ayrı ayrı yapılır.
- Gruplama partition key'e göre yapılır!
- Daha performanslı
- Daha fazla IO ve Memory

ENABLE_PARTITIONWISE_AGGREGATE

```
SET ENABLE_PARTITIONWISE_AGGREGATE=OFF
```

```
[pg11] # explain select p_status, count(*) from products group by p_status;
          QUERY PLAN
-----
HashAggregate  (cost=255.00..257.00 rows=200 width=16)
  Group Key: products_p1.p_status
    -> Append  (cost=0.00..205.00 rows=10000 width=8)
        -> Seq Scan on products_p1  (cost=0.00..53.41 rows=3441 width=8)
        -> Seq Scan on products_p2  (cost=0.00..50.35 rows=3235 width=7)
        -> Seq Scan on products_p3  (cost=0.00..51.24 rows=3324 width=8)
(6 rows)
```

```
SET ENABLE_PARTITIONWISE_AGGREGATE=ON
```

```
[pg11] # explain select p_status, count(*) from products group by p_status;
          QUERY PLAN
-----
Append  (cost=70.62..205.05 rows=3 width=16)
  -> HashAggregate  (cost=70.62..70.63 rows=1 width=16)
      Group Key: products_p1.p_status
        -> Seq Scan on products_p1  (cost=0.00..53.41 rows=3441 width=8)
  -> HashAggregate  (cost=66.53..66.54 rows=1 width=16)
      Group Key: products_p2.p_status
        -> Seq Scan on products_p2  (cost=0.00..50.35 rows=3235 width=7)
  -> HashAggregate  (cost=67.86..67.87 rows=1 width=16)
      Group Key: products_p3.p_status
        -> Seq Scan on products_p3  (cost=0.00..51.24 rows=3324 width=8)
(1w rows)
```



ENABLE_PARTITIONWISE_JOIN

```
SET ENABLE_PARTITIONWISE_JOIN_AGGREGATE=ON
```

- Default off
- Partitioned tablolar arasındaki JOIN, partition keyleri eşleşen partition tablolarının JOIN yapılmasıyla gerçekleşir!
- JOIN koşulunda tüm partition key'lerinin olması gereklidir!
 - Partition key'lerin veri tipi aynı olmalıdır.
- Daha fazla IO ve Memory



INCLUDE INDEXES

- Non-key alanlarıyla birlikte index-only scan
- Sadece Index-only scan
- Sadece B-tree Index
- Index-only scan non-key sütun değerini index tablosuna uğramadan döndürür.
- Sorgunun tüm sütunları için INDEX oluşturmak
 - Gereksiz Disk kullanımı
 - Silinen/güncellenen satırlar için daha çok VACUUM!

INCLUDE INDEXES

- Non-key alanlarıyla birlikte index-only scan
- Sadece Index-only scan
- Sadece B-tree Index
- Index-only scan non-key sütun değerini index tablosuna uğramadan döndürür.
- ~~Sorgunun tüm sütunları için INDEX oluşturmak~~
 - Gereksiz Disk kullanımı
 - Silinen/güncellenen satırlar için daha çok VACUUM!



INCLUDE INDEXES

```
CREATE UNIQUE INDEX id_idx ON products_2  
USING btree (id) INCLUDE (p_status, date);
```

- p_status ve date sütunlarını içeren sorgular index-only kullanmaya devam edecekler.
- ROUND gibi fonksiyonlar INCLUDE ile kullanılamaz.

```
=# CREATE UNIQUE INDEX new_unique_idx_2  
ON new_example(a, b) INCLUDE (round(c));  
ERROR: 0A000: expressions are not supported in  
included columns
```

ARTTIRILMIŞ PARALELLİK



GÜNDÜZ
BİLİŞİM
DANIŞMANLIĞI

ARTTIRILMIŞ PARALELLİK

- CREATE TABLE AS SELECT
- CREATE MATERIALIZED VIEW

```
[pg11] # EXPLAIN CREATE MATERIALIZED VIEW mv1
AS SELECT COUNT(*) FROM products;
                                         QUERY PLAN
-----
Finalize Aggregate  (cost=255070.26..255070.27 rows=1 width=8)
 ->  Gather  (cost=255070.05..255070.26 rows=2 width=8)
     Workers Planned: 2
       ->  Partial Aggregate  (cost=254070.05..254070.06 rows=1 width=8)
           ->  Parallel Append  (cost=0.00..233226.32 rows=8337489 width=0)
               ->  Parallel Seq Scan on products_p2  (cost=0.00..63928.14 rows=2782714 width=0)
               ->  Parallel Seq Scan on products_p1  (cost=0.00..63809.85 rows=2777585 width=0)
               ->  Parallel Seq Scan on products_p3  (cost=0.00..63800.90 rows=2777190 width=0)
(8 rows)
```



ARTTIRILMIŞ PARALELLİK

- SELECT INTO

```
[pg11] # EXPLAIN
SELECT max(id) INTO val FROM products;
                                         QUERY PLAN
-----
Finalize Aggregate  (cost=255070.26..255070.27 rows=1 width=4)
->  Gather  (cost=255070.05..255070.26 rows=2 width=4)
    Workers Planned: 2
    ->  Partial Aggregate  (cost=254070.05..254070.06 rows=1 width=4)
        ->  Parallel Append  (cost=0.00..233226.32 rows=8337489 width=4)
            ->  Parallel Seq Scan on products_p2  (cost=0.00..63928.14 rows=2782714 width=4)
            ->  Parallel Seq Scan on products_p1  (cost=0.00..63809.85 rows=2777585 width=4)
            ->  Parallel Seq Scan on products_p3  (cost=0.00..63800.90 rows=2777190 width=4)
(8 rows)
```

- CREATE INDEX
 - `max_parallel_maintenance_workers` en az 2 olmalı
- UNION ifadesi kullanılan scriptler



DEFAULT ROLES

- pg_read_server_files
- pg_write_server_files
- pg_execute_server_program

DEFAULT ROLES

- pg_read_server_files
- pg_write_server_files
- pg_execute_server_program

Superuser olmayan kullanıcıların sunucu üzerinde işlem yapmasını sağlar.



```
GRANT pg_execute_server_program TO admin_user;
```

DEFAULT ROLES

```
[pg11] # \du gunce
          List of roles
  Role name | Attributes |           Member of
-----+-----+-----+
  gunce     |             | { }
```

```
[pg11] # select current_user;
      current_user
-----+
  gunce
(1 row)
```

```
[pg11] # copy(select 1 ) to '/tmp/copytest.csv';
ERROR: 42501: must be superuser or a member of the
pg_write_server_files role to COPY to a file
```

```
[pg11] # GRANT pg_write_server_files to gunce ;
GRANT ROLE
```

```
[pg11] # copy(select 1 ) to '/home/guncek/copytest.csv';
ERROR: 42501: could not open file "/home/guncek/copytest.csv"
       for writing: Permission denied
HINT: COPY TO instructs the PostgreSQL server process to write
      a file. You may want a client-side facility such as psql's \copy.
LOCATION: BeginCopyTo, copy.c:1866
[pg11] # copy(select 1 ) to '/var/lib/pgsql/copytest.csv';
COPY 1
```



DEFAULT ROLES

```
[pg11] # \du postgres
List of roles
-[ RECORD 1 ]-----
Role name | postgres
Attributes | Superuser, Create role, Create DB, Replication, Bypass RLS
Member of | {}

[pg11] # select current_user;
 current_user
-----
 postgres
(1 row)

[pg11] # copy(select 1 ) to '/home/guncek/copytest.csv';
ERROR:  42501: could not open file
        "/home/guncek/copytest.csv" for writing: Permission denied
HINT:  COPY TO instructs the PostgreSQL server process to write
      a file. You may want a client-side facility such as psql's \copy.
LOCATION:  BeginCopyTo, copy.c:1866
```



WAL-SIZE

- Default WAL size 16MB
- 2'nin 0 ile 10'uncu kuvvetleri arasında 2'nin herhangi bir kuvveti olabilir.

```
initdb -D /pg_data --wal-segsize=32
```

```
SHOW wal_segment_size;
```

```
/usr/pgsql-11/bin/pg_controldata | grep "Bytes per WAL segment"
```

- Transaction sayısı yüksek veritabanlarında daha çok WAL dosyası oluşturulur.
- Disk üzerinde daha fazla write işlemi

WAL-SIZE

- Sunucu Üzerinde daha az WAL trafiği
 - archive_command çağrıılma sıklığının azaltılması
- Çok büyük veritabanları için avantaj sağlarken, küçük veritabanları için dezavantaj sağlayabilir.

WAL-SIZE

```
[postgres] # show wal_segment_size ;  
wal_segment_size  
-----  
64MB  
(1 row)
```



```
[postgres@asus-guncek ~]$ /usr/pgsql-11/bin/pg_ctl  
-D /var/lib/pgsql/11-3/data/ stop
```

```
[postgres@asus-guncek ~]$ /usr/pgsql-11/bin/pg_resetwal  
-D /var/lib/pgsql/11-3/data/  
--wal-segsize=32  
Write-ahead log reset
```

```
[postgres@asus-guncek ~]$ /usr/pgsql-11/bin/pg_ctl  
-D /var/lib/pgsql/11-3/data/ start
```

```
[postgres] # show wal_segment_size ;  
wal_segment_size  
-----  
32MB  
(1 row)
```



WAL-SIZE

archive_command -> cp, scp, rsync

16MB'lik 100 adet dosya

```
[postgres@asus-guncek ~]$ mkdir test_local
[postgres@asus-guncek ~]$ for i in {1..100}; do
> dd if=/dev/zero of=/var/lib/pgsql/test_local/${i} bs=16M count=1
> done
1+0 records in
1+0 records out
16777216 bytes (17 MB, 16 MiB) copied, 0.0151023 s, 1.1 GB/s
1+0 records in
1+0 records out
```

64MB'lik 25 adet dosya

```
[postgres@asus-guncek test_local]$ for i in {1..25};
> do dd if=/dev/zero of=/var/lib/pgsql/test_local_64mb/${i} bs=64M count=1
> done
1+0 records in
1+0 records out
67108864 bytes (67 MB, 64 MiB) copied, 0.0514324 s, 1.3 GB/s
1+0 records in
1+0 records out
67108864 bytes (67 MB, 64 MiB) copied, 0.0445698 s, 1.5 GB/s
```



WAL-SIZE

16MB'lik dosyaların SCP ile aktarımı -> 24sn

```
[postgres@asus-guncuk test_local]$ time for i in `ls /var/lib/pgsql/test_local`; do scp /var/lib/pgsql/test_local/${i} postgres@192.168.122.37:/tmp/test16mb/; done
1                                         100%   16MB 492.7MB/s  00:00
10                                        100%   16MB 367.7MB/s  00:00
100                                       100%   16MB 437.7MB/s  00:00
11                                         100%   16MB 205.7MB/s  00:00
95                                         100%   16MB 435.0MB/s  00:00
96                                         100%   16MB 398.8MB/s  00:00
97                                         100%   16MB 489.5MB/s  00:00
98                                         100%   16MB 417.7MB/s  00:00
99                                         100%   16MB 308.4MB/s  00:00
real    0m24.243s
user    0m2.328s
sys     0m2.838s
[postgres@asus-guncuk test_local]$
```



WAL-SIZE

64MB'lik dosyaların SCP ile aktarımı -> 8sn

```
[postgres@asus-guncuk test_local_64mb]$ time for i in `ls /var/lib/pgsql/test_local_64mb`; do scp /var/lib/pgsql/test_local_64mb/${i} postgres@192.168.122.37:/tmp/test64mb/; done
```

| | | | | |
|------|----------|------|-----------|-------|
| 8 | 100% | 64MB | 453.7MB/s | 00:00 |
| 9 | 100% | 64MB | 191.1MB/s | 00:00 |
| real | 0m8.944s | | | |
| user | 0m1.554s | | | |
| sys | 0m2.511s | | | |

WAL-SIZE

16MB'lik dosyaların RSYNC ile aktarımı -> 3dk24sn

```
[postgres@asus-guncuk ~]$ time for i in `ls /var/lib/pgsql/test_local`; do rsync -a /var/lib/pgsql/test_local/${i} postgres@192.168.122.37:/tmp/test16mb/; done  
real    3m24.708s  
user    0m3.949s  
sys     0m3.103s
```

64MB'lik dosyaların RSYNC ile aktarımı -> 14sn

```
[postgres@asus-guncuk test_local_64mb]$ time for i in `ls /var/lib/pgsql/test_local_64mb`; do rsync -a /var/lib/pgsql/test_local_64mb/${i} postgres@192.168.122.37:/tmp/test64mb/; done  
real    0m14.826s  
user    0m4.850s  
sys     0m2.601s  
[postgres@asus-guncuk test_local_64mb]$
```

BASE BACKUP AND STREAMING REPLICATION

LOGICAL REPLICATION'da

TRUNCATE

- PG11'de Logical Replication ile TRUNCATE ifadesini kullanabiliyoruz.
- PG10'da TRUNCATE yerine DELETE kullanmak zorundaydık.

LOGICAL REPLICATION'da TRUNCATE

- PG11'de Logical Replication ile TRUNCATE ifadesini kullanabiliyoruz.
- PG10'da TRUNCATE yerine DELETE kullanmak zorundaydık.
 - Daha fazla VACUUM
 - Daha fazla IO



BASEBACKUP VE UNLOGGED - TEMPORARY TABLOLAR



BASEBACKUP İLE UNLOGGED VE TEMPORARY TABLOLAR

- UNLOGGED tablolar veri tutar WAL dosyalarına yazılmaz.
 - Standby'a replike olmazlar!
 - Performans amaçlı kullanılır.
 - Restore işleminde sadece tablo TRUNCATE edilir.
- Temporary tablolar ve ayrıca pg_internal.init dosyası da çıkarıldı!

ALTER TABLE ADD COLUMN

- Bir tabloya nasıl sütun eklenir?

ALTER TABLE ADD COLUMN

- Bir tabloya nasıl sütun eklenir?
 - Tablo baştan yazılır

ALTER TABLE ADD COLUMN

- Bir tabloya nasıl sütun eklenir?
 - Tablo baştan yazılır
 - Tablodaki INDEXler de baştan yazılır

ALTER TABLE ADD COLUMN

- Bir tabloya nasıl sütun eklenir?
 - Tablo **baştan** yazılır
 - Tablodaki **INDEX**ler de baştan yazılır
 - Sonuç: Oldukça maliyetli...

ALTER TABLE ADD COLUMN

- pg_attribute tablosunda iki yeni sütun
 - attmissingval
 - atthasmissing
- Ciddi oranda azalan lock süresi
- Index'lerin yeniden diske yazılması önlenir



ALTER TABLE ADD COLUMN

PG10

```
[test] # select version();
          version
-----
 PostgreSQL 10.5 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 7.3.1 20180712 (Red Hat 7.3.1-6), 64-bit
(1 row)

Time: 0.428 ms
[test] # create table add_col as select i, random() as rand from generate_series(1,10000000) as i;
SELECT 10000000
Time: 16016.593 ms (00:16.017)
[test] # alter table add_col add column g character varying default 'not-null';
ALTER TABLE
Time: 33977.474 ms (00:33.977)
```

PG11

```
[test] # select version();
          version
-----
 PostgreSQL 11beta3 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 7.3.1 20180712 (Red Hat 7.3.1-6), 64-bit
(1 row)

Time: 0.394 ms
[test] # create table add_col as select i, random() as rand from generate_series(1,10000000) as i;
SELECT 10000000
Time: 14163.042 ms (00:14.163)
[test] # alter table add_col add column g character varying default 'not-null';
ALTER TABLE
Time: 7.534 ms
```

ALTER TABLE ADD COLUMN

PG10 daki tablonun boyutu

```
[test] # SELECT
... #     relname as "Table",
... #     pg_size.pretty(pg_total_relation_size(relid)) As "Size",
... #     pg_size.pretty(pg_total_relation_size(relid) - pg_relation_size(relid)) as "External Size"
... #     FROM pg_catalog.pg_statio_user_tables where relname='add_col' ORDER BY pg_total_relation_size(relid) DESC;
   Table |    Size    | External Size
-----+-----+-----
 add_col | 574 MB | 8192 bytes
(1 row)
```

PG11 deki tablonun boyutu

```
[test] # SELECT
... #     relname as "Table",
... #     pg_size.pretty(pg_total_relation_size(relid)) As "Size",
... #     pg_size.pretty(pg_total_relation_size(relid) - pg_relation_size(relid)) as "External Size"
... #     FROM pg_catalog.pg_statio_user_tables where relname='add_col' ORDER BY pg_total_relation_size(relid) DESC;
   Table |    Size    | External Size
-----+-----+-----
 add_col | 422 MB | 8192 bytes
(1 row)
```

ALTER TABLE ADD COLUMN

add_col tablosunun "g" sütunun pg_attribute tablosundaki görüntüsü

```
[test] # select attname, attrhasmissing, attmissingval from pg_attribute where attrelid='add_col'::regclass and attname='g';
attname | attrhasmissing | attmissingval
-----+-----+-----+
g      | t             | {not-null}
(1 row)
```

SHA-2 FONKSIYONLARI

sha224(), sha256(), sha384(), sha512()

```
[postgres] # SELECT LENGTH(sha224('test')),sha224('a');
length | sha224
-----+-----
 28 | \xabd37534c7d9a2efb9465de931cd7055ffdb8879563ae98078d6d6d5
(1 row)

[postgres] # SELECT LENGTH(sha256('test')),sha256('a');
length | sha256
-----+-----
 32 | \xca978112ca1bbdcfac231b39a23dc4da786eff8147c4e72b9807785afee48bb
(1 row)

[postgres] # SELECT LENGTH(sha384('test')),sha384('a');
length | sha384
-----+-----
 48 | \x54a59b9f22b0b80880d8427e548b7c23abd873486e1f035dce9cd697e85175033caa88e6d57bc35efae0b5afd3145f31
(1 row)

[postgres] # SELECT LENGTH(sha512('test')),sha512('a');
-[ RECORD 1 ]-----
length | 64
sha512 | \x1f40fc92da241694750979ee6cf582f2d5d7d28e18335de05abc54d0560e0f5302860c652bf08d560252aa5e74210546f369

[postgres] #
```

STORED PROCEDURE



GÜNDÜZ
BİLİŞİM
DANIŞMANLIĞI

STORED PROCEDURE

- CREATE PROCEDURE
- CALL <PROCEDURENAME>

STORED PROCEDURE

```
[pg11] # CREATE OR REPLACE PROCEDURE public.test_procedure(a integer, INOUT b integer)
... # LANGUAGE plpgsql
... # AS $procedure$ BEGIN
... # select a*b into b;
... # END;
... # $procedure$
... #
... # ;
CREATE PROCEDURE
[pg11] # CALL test_procedure (2,3);
    b
---
    6
(1 row)
```

STORED PROCEDURE

- Transactional Control
 - Cursor Loop
 - COMMIT/ROLLBACK birlikte

GROUP ACCESS on DATA FOLDER



GROUP ACCESS on DATA FOLDER

```
-g veya --allow-group-access
```

- PG11(default PG10 ile aynı)
 - Dizin yetkisi 640
 - Dosya yetkisi 750
- PG10
 - Dizin yetkisi 600
 - Dosya yetkisi 700

GROUP ACCESS on DATA FOLDER

```
/usr/pgsql-11/bin/initdb -g -D /var/lib/pgsql/11-2/data
```

- Amaç
 - Cluster sahibi olmayan kullanıcıların backup processlerini gerçekleştirmesi
- Write yetkisine sahip olmayan kullanıcılar backup alabilir

BASEBACKUP İLE REPLICATION SLOT OLUŞTURULMASI

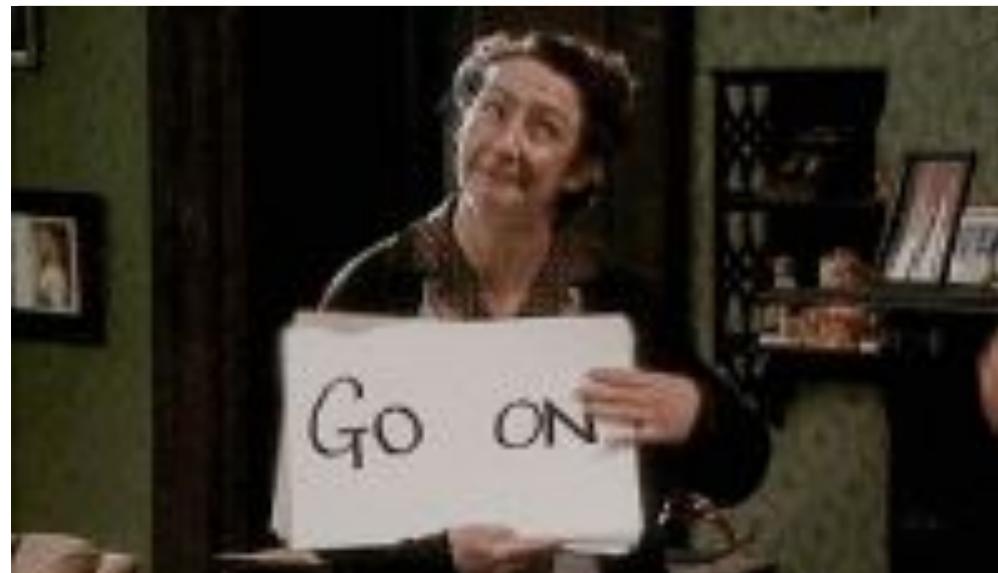
- --create-slot
- replication slot adı (--slot)
- WAL streaming method (--wal-method=stream)

```
[postgres@asus-guncuk ~]$ /usr/pgsql-11/bin/pg_basebackup  
      -D /var/lib/pgsql/11-3/data/ -p5411 -h 192.168.122.100  
      --create-slot --slot='test'  
  
[postgres] # SELECT slot_name, slot_type, restart_lsn  
              FROM pg_replication_slots;  
 slot_name | slot_type | restart_lsn  
-----+-----+-----  
 test      | physical | 3/6E000000  
(1 row)
```

```
pg_basebackup: could not send replication command  
"CREATE_REPLICATION_SLOT "test" PHYSICAL RESERVE_WAL":  
ERROR: replication slot "test" already exists
```



DiĞER GELİŞTİRMELER



INSERT ON CONFLICT DO NOTHING

PG10

```
[pg10] # insert into students(id,name) values(11,'name_1') on
conflict do nothing;
ERROR:  0A000: ON CONFLICT clause is not supported with partit
ioned tables
LOCATION:  transformInsertStmt, analyze.c:855
```

PG11

```
[pg11] # insert into students(id,name) values(11,'name_1') on conflict do nothing;
INSERT 0 1
```

PG_STAT_STATEMENTS

- Queryid alanı -> 32bit'ten 64bit olarak güncellendi

..Ve Diğerleri

- Partition tablolarında TRIGGER desteği
- Partition Foreign tablolarında postgres_fdw için push down aggregate özelliği geldi.
- SP-GIST için yeni TEXT prefix operator -> ^@
 - LIKE 'kelime%' ile aynı
 - Daha etkili

..Ve Diğerleri

- VACUUM veya ANALYZE ifadelerinin birden fazla tablo için çalıştırılması
- pg_verify_checksums
 - Veritabanı offline durumundayken checksum yapar
- Diğer fonksiyonlar;
 - websearch_to_tsquery()
 - json(b)_to_tsvector()

Teşekkürler

Günce Kaya



guncekaya.blogspot.com

twitter: @ggunesi