Kubernetes StackGres/PostgreSQL

Állapot ellenőrzés

Kubernetes állapot ellenőrzés

A következő utasítások futtatása előtt be kell állítani a névteret.

```
export NAMESPACE=<névtér>
```

POD-ok ellenőrzése

```
$ kubectl get pods -n $NAMESPACE -o wide
NAME           READY   STATUS    RESTARTS     AGE   IP           NODE
NOMINATED NODE   READINESS GATES
pg-cluster-0   5/5     Running   6 (5d ago)   37d   10.244.0.5   kube01
<none>           <none>
pg-cluster-1   5/5     Running   7 (5d ago)   37d   10.244.2.5   kube03
<none>           <none>
pg-cluster-2   5/5     Running   8 (5d ago)   37d   10.244.1.8   kube02
<none>           <none>
```

Amennyiben hiba van (bármelyik POD állapota nem `Running`, akkor az alábbi ellenőrzések szükségesek a hibás POD nevét megadva (a példában ez a `pg-cluster-0` nevű pod).

```
$ kubectl describe pod pg-cluster-0 -n $NAMESPACE
Name:             pg-cluster-0
Namespace:        lapodb-postgres
Priority:         0
Service Account:  pg-cluster-patroni
Node:             kube01/192.168.110.161
Start Time:       Sun, 03 May 2026 17:31:59 +0200
Labels:           app=StackGresCluster
                  apps.kubernetes.io/pod-index=0
                  controller-revision-hash=pg-cluster-7f856d76dc
                  nofailover=false
                  noloadbalance=false
                  role=primary
                  stackgres.io/cluster=true
                  stackgres.io/cluster-name=pg-cluster
                  stackgres.io/cluster-scope=pg-cluster
                  stackgres.io/cluster-
uid=27cdba8f-3477-4740-848d-6f82c45c405a
                  stackgres.io/disruptible=true
                  statefulset.kubernetes.io/pod-name=pg-cluster-0
```

```
Annotations:      stackgres.io/cluster-controller-version: 1.18.6
                  stackgres.io/operatorVersion: 1.18.6
                  stackgres.io/patroni-version: 4.1.0
                  stackgres.io/pgbouncer-version: 1.25.1
                  stackgres.io/postgresql-version: 18.3
                  stackgres.io/prometheus-postgres-exporter-version: 0.19.0
                  status:
{"conn_url":"postgres://10.244.0.5:5432/postgres","api_url":"http://10.244.0
.5:8009/patroni","state":"running","role":"primary","version":...
Status:           Running
IP:               10.244.0.5
IPs:
  IP:           10.244.0.5
Controlled By:  StatefulSet/pg-cluster
...
Node-Selectors:              node-role.kubernetes.io/postgres=true
Tolerations:                 node.kubernetes.io/not-ready:NoExecute
op=Exists for 300s
                             node.kubernetes.io/unreachable:NoExecute
op=Exists for 300s
Events:                      <none>
```

POD-ban található konténerek nevének meghatározása.

```
$ kubectl get pod pg-cluster-0 -n $NAMESPACE -o
jsonpath='{.spec.containers[*].name}{"\n"}'
patroni pgbouncer prometheus-postgres-exporter postgres-util cluster-
controller
```

A PostgreSQL példány a `patroni` konténerben fut.

```
$ kubectl logs pg-cluster-0 -c patroni -n $NAMESPACE
...
2026-06-10 08:26:26 UTC [596361]:
db=postgres,user=postgres,app=[unknown],client=127.0.0.1 LOG:  connection
authorized: user=postgres database=postgres SSL enabled (protocol=TLSv1.3,
cipher=TLS_AES_256_GCM_SHA384, bits=256)
2026-06-10 08:26:26 UTC [596361]: db=postgres,user=postgres,app=PostgreSQL
JDBC Driver,client=127.0.0.1 LOG:  disconnection: session time: 0:00:00.026
user=postgres database=postgres host=127.0.0.1 port=48302
2026-06-10 08:26:33,319 INFO: no action. I am (pg-cluster-0), the leader
with the lock
```

StackGres objektumok ellenőrzése

```
$ kubectl get sgcluster -n $NAMESPACE -o wide
NAME         VERSION   FLAVOR   INSTANCES   PROFILE              DISK   AGE
REPLICATION   BACKUP OBJECT STORAGE   BACKUP SCHEDULE
pg-cluster   18.3              3           pg-cluster-default   8Gi    37d
```

```
async

$ kubectl describe sgcluster pg-cluster -n $NAMESPACE
$ kubectl describe sgcluster pg-cluster -n $NAMESPACE
Name:         pg-cluster
Namespace:    lapodb-postgres
Labels:       <none>
Annotations:  stackgres.io/operatorVersion: 1.18.6
API Version:  stackgres.io/v1
Kind:         SGCluster
Metadata:
  Creation Timestamp:  2026-05-03T15:25:56Z
  Generation:          17
  Resource Version:    8510436
  UID:                 27cdba8f-3477-4740-848d-6f82c45c405a
Spec:
  Configurations:
    Observability:
      Disable Metrics:      false
      Prometheus Autobind:  false
    Sg Pooling Config:      pg-cluster-default
    Sg Postgres Config:     pg-config
  Instances:                3
  Managed Sql:
    Continue On SGScript Error:  false
    Scripts:
      Id:         0
      Sg Script:  pg-cluster-default
  Pods:
    Disable Connection Pooling:  false
    Disable Envoy:               true
    Disable Metrics Exporter:    false
    Disable Postgres Util:       false
    Management Policy:           OrderedReady
    Persistent Volume:
      Size:           8Gi
      Storage Class:  local-postgres
    Scheduling:
      Node Selector:
        node-role.kubernetes.io/postgres:  true
      Pod Anti Affinity:
        Required During Scheduling Ignored During Execution:
          Label Selector:
            Match Labels:
              stackgres.io/cluster:  pg-cluster
          Topology Key:              kubernetes.io/hostname
    Update Strategy:
      Method:  InPlace
      Type:    OnlyDbOps
  Postgres:
    Flavor:  vanilla
```

```
    Ssl:
      Enabled:  true
    Version:    18
  Postgres Services:
    Primary:
      Enabled:  true
      Type:     ClusterIP
    Replicas:
      Enabled:  true
      Type:     ClusterIP
  Profile:      production
  Replication:
    Mode:               async
    Role:               ha-read
  Sg Instance Profile:  pg-cluster-default
Status:
  Arch:  x86_64
  Binding:
    Name:         pg-cluster-binding
  Build Version:  6.49
  Conditions:
    Last Transition Time:  2026-05-03T15:35:52.134132Z
    Reason:                ClusterInitialScriptApplied
    Status:                True
    Type:                  InitialScriptsApplied
    Last Transition Time:  2026-05-03T15:35:52.071432Z
    Reason:                ClusterBootstrapped
    Status:                True
    Type:                  Bootstrapped
    Last Transition Time:  2026-05-03T15:31:59.887576Z
    Reason:                FalsePendingUpgrade
    Status:                False
    Type:                  PendingUpgrade
    Last Transition Time:  2026-05-03T15:31:59.887531Z
    Reason:                FalsePendingRestart
    Status:                False
    Type:                  PendingRestart
    Last Transition Time:  2026-05-03T15:31:59.313942Z
    Reason:                FalseFailed
    Status:                False
    Type:                  Failed
  Extensions:
  Instances:       3
  Label Selector:  app=StackGresCluster,stackgres.io/cluster-name=pg-
cluster,stackgres.io/cluster-scope=pg-
cluster,stackgres.io/cluster=true,stackgres.io/cluster-
uid=27cdba8f-3477-4740-848d-6f82c45c405a
  Managed Sql:
    Scripts:
      Completed At:  2026-05-03T15:35:47.613868Z
      Id:            0
```

```
      Scripts:
        Id:        0
        Version:   0
        Id:        4
        Version:   0
      Started At:  2026-05-03T15:35:44.096419Z
      Updated At:  2026-05-03T15:35:47.193412Z
  Os:              linux
  Pod Statuses:
    Installed Postgres Extensions:
    Name:               pg-cluster-0
    Node Name:          kube01
    Pending Restart:    false
    Primary:            true
    Replication Group:  0
    Installed Postgres Extensions:
    Name:               pg-cluster-1
    Node Name:          kube03
    Pending Restart:    false
    Primary:            false
    Replication Group:  0
    Installed Postgres Extensions:
    Name:               pg-cluster-2
    Node Name:          kube02
    Pending Restart:    false
    Primary:            false
    Replication Group:  0
  Postgres Version:     18.3
Events:                 <none>

$ kubectl get events -n $NAMESPACE --sort-by=.lastTimestamp
LAST SEEN   TYPE      REASON               OBJECT
MESSAGE
1s          Warning   BackupConfigFailed   sgbackup/bk2026-05-03-17-52-04
SGBackup reconciliation cycle failed: SGCluster pg-cluster has no backup
configured
```

StatefulSet ellenőrzése

```
$ kubectl get statefulset -n $NAMESPACE -o wide
NAME         READY   AGE   CONTAINERS
IMAGES
pg-cluster   3/3     37d   patroni,pgbouncer,prometheus-postgres-
exporter,postgres-util,cluster-controller   quay.io/ongres/patroni:v4.1.0-
pg18.3-build-6.49,quay.io/ongres/pgbouncer:v1.25.1-
build-6.49,quay.io/ongres/prometheus-postgres-exporter:v0.19.0-
build-6.49,quay.io/ongres/postgres-util:v18.3-
build-6.49,quay.io/stackgres/cluster-controller:1.18.6
```

PVC-k ellenőrzése

```
$ kubectl get pvc -n $NAMESPACE -o wide
NAME                          STATUS   VOLUME            CAPACITY   ACCESS
MODES   STORAGECLASS     VOLUMEATTRIBUTESCLASS   AGE   VOLUMEMODE
pg-cluster-data-pg-cluster-0   Bound    local-pg-kube01   8Gi        RWO
local-postgres   <unset>                 37d   Filesystem
pg-cluster-data-pg-cluster-1   Bound    local-pg-kube03   8Gi        RWO
local-postgres   <unset>                 37d   Filesystem
pg-cluster-data-pg-cluster-2   Bound    local-pg-kube02   8Gi        RWO
local-postgres   <unset>                 37d   Filesystem
```

Patroni állapot ellenőrzés

Normál állapot

```
$ kubectl exec -n $NAMESPACE -it pg-cluster-0 -c patroni -- patronictl list
+ Cluster: pg-cluster (7635689612010042939) ------+----+-------------+-----
+------------+-----+
| Member       | Host       | Role    | State     | TL | Receive LSN | Lag |
Replay LSN | Lag |
+--------------+------------+---------+-----------+----+-------------+-----
+------------+-----+
| pg-cluster-0 | 10.244.0.5 | Leader  | running   |  2 |             |     |
|     |
| pg-cluster-1 | 10.244.2.5 | Replica | streaming |  2 |   0/40002B0 |   0 |
0/40002B0 |   0 |
| pg-cluster-2 | 10.244.1.8 | Replica | streaming |  2 |   0/40002B0 |   0 |
0/40002B0 |   0 |
+--------------+------------+---------+-----------+----+-------------+-----
+------------+-----+
```

Egy replika node kiesése

```
$ kubectl exec -n $NAMESPACE -it pg-cluster-0 -c patroni -- patronictl list
+ Cluster: pg-cluster (7635689612010042939) ------+----+-------------+-----
+------------+-----+
| Member       | Host       | Role    | State     | TL | Receive LSN | Lag |
Replay LSN | Lag |
+--------------+------------+---------+-----------+----+-------------+-----
+------------+-----+
| pg-cluster-0 | 10.244.0.5 | Leader  | running   |  2 |             |     |
|     |
| pg-cluster-1 | 10.244.2.5 | Replica | streaming |  2 |   0/40002B0 |   0 |
0/40002B0 |   0 |
| pg-cluster-2 |            | Replica |           |    |     unknown |     |
unknown |     |
+--------------+------------+---------+-----------+----+-------------+-----
```

```
+------------+-----+
```

Leader node kiesése

```
# kubectl get nodes
NAME     STATUS     ROLES                     AGE   VERSION
kube01   NotReady   control-plane,postgres   38d   v1.36.0
kube02   Ready      control-plane,postgres   38d   v1.36.0
kube03   Ready      control-plane,postgres   38d   v1.36.0

# kubectl get pods -n $NAMESPACE
NAME           READY   STATUS        RESTARTS       AGE
pg-cluster-0   5/5     Terminating   6 (5d2h ago)   37d
pg-cluster-1   5/5     Running       7 (5d2h ago)   37d
pg-cluster-2   5/5     Running       1 (15m ago)    16m

# kubectl exec -n $NAMESPACE pg-cluster-1 -c patroni -- patronictl list
2026-06-10 09:43:17,883 - WARNING - Retrying (Retry(total=0, connect=None,
read=None, redirect=0, status=None)) after connection broken by
'NewConnectionError("HTTPSConnection(host='10.96.0.1', port=443): Failed to
establish a new connection: [Errno 113] No route to host")':
/api/v1/namespaces/lapodb-
postgres/endpoints?labelSelector=app%3DStackGresCluster%2Cstackgres.io%2Fclu
ster%3Dtrue%2Cstackgres.io%2Fcluster-scope%3Dpg-cluster
+ Cluster: pg-cluster (7635689612010042939) ------+----+-------------+-----
+------------+-----+
| Member       | Host       | Role    | State     | TL | Receive LSN | Lag |
Replay LSN | Lag |
+--------------+------------+---------+-----------+----+-------------+-----
+------------+-----+
| pg-cluster-0 | 10.244.0.5 | Replica | running   |  2 |     unknown |     |
0/40002B0 |   0 |
| pg-cluster-1 | 10.244.2.5 | Leader  | running   |  3 |             |     |
|     |
| pg-cluster-2 | 10.244.1.9 | Replica | streaming |  3 |   0/40004C0 |   0 |
0/40004C0 |   0 |
+--------------+------------+---------+-----------+----+-------------+-----
+------------+-----+

# kubectl exec -n $NAMESPACE pg-cluster-2 -c patroni -- patronictl list
+ Cluster: pg-cluster (7635689612010042939) ------+----+-------------+-----
+------------+-----+
| Member       | Host       | Role    | State     | TL | Receive LSN | Lag |
Replay LSN | Lag |
+--------------+------------+---------+-----------+----+-------------+-----
+------------+-----+
| pg-cluster-0 | 10.244.0.5 | Replica | running   |  2 |     unknown |     |
0/40002B0 |   0 |
| pg-cluster-1 | 10.244.2.5 | Leader  | running   |  3 |             |     |
|     |
```

```
| pg-cluster-2 | 10.244.1.9 | Replica | streaming |  3 |   0/40004C0 |   0 |
0/40004C0 |   0 |
+--------------+------------+---------+-----------+----+-------------+-----
+------------+-----+
```

Patroni automatikus failover után (10-60 másodperc) elérhető a szolgáltatás.

```
$ kubectl run psql-test -n $NAMESPACE --rm -it --image=postgres:17 -- bash
All commands and output from this session will be recorded in container
logs, including credentials and sensitive information passed through the
command prompt.
If you don't see a command prompt, try pressing enter.

root@psql-test:/# pg_isready -h pg-cluster-primary -p 5432
pg-cluster-primary:5432 - accepting connections

root@psql-test:/# exit
exit
Session ended, resume using 'kubectl attach psql-test -c psql-test -n
lapodb-postgres -i -t' command
pod "psql-test" deleted from lapodb-postgres namespace

$ kubectl exec -n $NAMESPACE -it pg-cluster-1 -c patroni -- patronictl list
+ Cluster: pg-cluster (7635689612010042939) -------+----+-------------+-----
+------------+-----+
| Member       | Host        | Role    | State     | TL | Receive LSN | Lag
| Replay LSN | Lag |
+--------------+-------------+---------+-----------+----+-------------+-----
+------------+-----+
| pg-cluster-0 | 10.244.0.9  | Replica | streaming |  4 |   0/40014F0 |   0
|  0/40014F0 |   0 |
| pg-cluster-1 | 10.244.2.16 | Replica | streaming |  4 |   0/40014F0 |   0
|  0/40014F0 |   0 |
| pg-cluster-2 | 10.244.1.9  | Leader  | running   |  4 |             |
|            |     |
+--------------+-------------+---------+-----------+----+-------------+-----
+------------+-----+
```

PostgreSQL állapot ellenőrzés

Upstream állapot ellenőrzés

```
$ kubectl exec -n $NAMESPACE pg-cluster-1 -c patroni -- patronictl list
+ Cluster: pg-cluster (7635689612010042939) -------+----+-------------+-----
+------------+-----+
| Member       | Host        | Role    | State     | TL | Receive LSN | Lag
| Replay LSN | Lag |
+--------------+-------------+---------+-----------+----+-------------+-----
+------------+-----+
```

```
| pg-cluster-0 | 10.244.0.9  | Replica | streaming |  4 |   0/40014F0 |   0
|  0/40014F0 |   0 |
| pg-cluster-1 | 10.244.2.16 | Replica | streaming |  4 |   0/40014F0 |   0
|  0/40014F0 |   0 |
| pg-cluster-2 | 10.244.1.9  | Leader  | running   |  4 |             |
|            |     |
+--------------+-------------+---------+-----------+----+-------------+-----
+------------+-----+
```

A leaderen kell ellenőrizni a replikáció állapotát

```
$ kubectl exec -n $NAMESPACE -it pg-cluster-2 -c patroni -- psql -U postgres
-c "SELECT application_name, client_addr, state, sync_state FROM
pg_stat_replication"
Pager usage is off.
 application_name | client_addr |   state   | sync_state
------------------+-------------+-----------+------------
 pg-cluster-0     | 10.244.0.9  | streaming | async
 pg-cluster-1     | 10.244.2.16 | streaming | async
(2 rows)
```

Részletesebb megjelenítés

```
$ kubectl exec -n $NAMESPACE -it pg-cluster-2 -c patroni -- psql -U postgres
-x -c "SELECT * FROM pg_stat_replication"
Pager usage is off.
-[ RECORD 1 ]----+------------------------------
pid              | 9267
usesysid         | 16384
usename          | replicator
application_name | pg-cluster-0
client_addr      | 10.244.0.9
client_hostname  |
client_port      | 41370
backend_start    | 2026-06-10 11:05:15.664685+00
backend_xmin     |
state            | streaming
sent_lsn         | 0/40014F0
write_lsn        | 0/40014F0
flush_lsn        | 0/40014F0
replay_lsn       | 0/40014F0
write_lag        |
flush_lag        |
replay_lag       |
sync_priority    | 0
sync_state       | async
reply_time       | 2026-06-10 11:42:47.125+00
-[ RECORD 2 ]----+------------------------------
pid              | 9962
```

```
usesysid         | 16384
usename          | replicator
application_name | pg-cluster-1
client_addr      | 10.244.2.16
client_hostname  |
client_port      | 47432
backend_start    | 2026-06-10 11:13:44.348486+00
backend_xmin     |
state            | streaming
sent_lsn         | 0/40014F0
write_lsn        | 0/40014F0
flush_lsn        | 0/40014F0
replay_lsn       | 0/40014F0
write_lag        |
flush_lag        |
replay_lag       |
sync_priority    | 0
sync_state       | async
reply_time       | 2026-06-10 11:42:47.158991+00
```

- `pid` WAL sender PostgreSQL folyamat PID-je a leaderen
- `usesysid` Kapcsolódó PostgreSQL felhasználó azonosítója
- `usename` Replikációs kapcsolatot használó felhasználó (replicator)
- `application_name` Replika neve. Patroni használatakor a pod neve
- `client_addr` Replika IP címe
- `client_hostname` DNS név, amennyiben feloldható
- `client_port` Replika TCP forrásportja
- `backend_start` Replikációs kapcsolat létrejöttének ideje
- `backend_xmin` Hot standby VACUUM védelemhez használt xmin.
- `state` Replikáció állapota (`startup` kapcsolat indul, `catchup` utoléri a leadert, `streaming` folyamatos replikáció, `backup` base backup készül)
- `sent_lsn` A leader eddig a pontig küldte ki a WAL rekordokat
- `write_lsn` A replika eddig a pontig írta lemezre
- `flush_lsn` A replika eddig a pontig törölte a WAL-t
- `replay_lsn` A replika eddig a pontig alkalmazta a WAL rekordokat
- `write_lag` Lemezre írás késleltetése
- `flush_lag` Törlés késleltetése
- `replay_lag` WAL rekordok alkalmazásának késleltetése
- `sync_priority` Szinkron replikációnál a szinkronizáció elsőbbségét adja meg
- `sync_state` Replikáció típusa (`async` aszinkron replikáció, `sync` szinkron replikáció)
- `reply_time` Replika utoljára ekkor válaszolt a leadernek

Replika állapot ellenőrzés

Az ellenőrzés elvégezhető valamelyik replikán. A `t` azt jelenti, hogy a replikáció fut. Amennyiben `f` az eredmény, akkor a replikáció nem fut (ilyen eset lehet, ha a lekérdezést a leader-en futtatjuk).

```
$ kubectl exec -n $NAMESPACE -it pg-cluster-0 -c patroni -- psql -U postgres
-x -c "SELECT pg_is_in_recovery()"
Pager usage is off.
```

```
-[ RECORD 1 ]-----+--
pg_is_in_recovery | t

$ kubectl exec -n $NAMESPACE -it pg-cluster-1 -c patroni -- psql -U postgres
-x -c "SELECT pg_is_in_recovery()"
Pager usage is off.
-[ RECORD 1 ]-----+--
pg_is_in_recovery | t

$ kubectl exec -n $NAMESPACE -it pg-cluster-2 -c patroni -- psql -U postgres
-x -c "SELECT pg_is_in_recovery()"
Pager usage is off.
-[ RECORD 1 ]-----+--
pg_is_in_recovery | f
```

Monitorozás

Monitor szkriptek

PostgreSQL replikáció állapot

A PostgreSQL állapotának ellenőrzését végző szkriptek első sorában meg kell adni a használni kívánt névteret.

POD-ok ellenőrzése

01-check-pods.sh

```
#!/usr/bin/env bash

NAMESPACE="lapodb-postgres"

NOT_READY=$(kubectl get pods -n "$NAMESPACE" --no-headers | awk '
{
  split($2, ready, "/")
  if (ready[1] != ready[2] || $3 != "Running") {
    print
  }
}')

if [ -n "$NOT_READY" ]; then
  echo "CRITICAL - Van olyan PostgreSQL pod, amiben legalább egy
konténer nem megfelelő állapotban van"
  echo "$NOT_READY"
  exit 2
fi
```

```
echo "OK - Minden pod és konténer megfelelő állapotban van"

exit 0
```

Futtatás eredménye

```
$ ./01-check-pods.sh
OK - Minden pod és konténer megfelelő állapotban van
```

StackGres/PostgreSQL replikáció

02-check-stackgres-postgresql-status.sh

```
#!/usr/bin/env bash

NAMESPACE="lapodb-postgres"
CLUSTER_PREFIX="pg-cluster"
EXPECTED_PODS=3

PODS=$(kubectl get pods -n "$NAMESPACE" --no-headers 2>/dev/null | awk
-v p="$CLUSTER_PREFIX" '$1 ~ "^"p"-[0-9]+$" {print $1}')

if [ -z "$PODS" ] ; then
  echo "CRITICAL - Nincs PostgreSQL pod a namespace-ben: $NAMESPACE"
  exit 2
fi

FIRST_POD=$(echo "$PODS" | head -n 1)

PATRONI_STATUS="$(kubectl exec -n "$NAMESPACE" "$FIRST_POD" -c patroni
-- patronictl list 2>/dev/null)"

if [ -z "$PATRONI_STATUS" ] ; then
  echo "CRITICAL - Nem sikerült lekérdezni a Patroni állapotot"
  exit 2
fi

READY_COUNT=0
POD_COUNT=0
LEADER_COUNT=0
REPLICA_COUNT=0
DETAILS=""

for POD in $PODS ; do
    POD_COUNT=$((POD_COUNT + 1))

    NODE=$(kubectl get pod "$POD" -n "$NAMESPACE" -o
```

```
jsonpath='{.spec.nodeName}' 2>/dev/null)
    PHASE=$(kubectl get pod "$POD" -n "$NAMESPACE" -o
jsonpath='{.status.phase}' 2>/dev/null)
    READY="$(kubectl get pod "$POD" -n "$NAMESPACE" -o jsonpath='{range
.status.containerStatuses[*]}{.ready}{" "}{end}' 2>/dev/null | awk
'{ready=0; total=0; for (i=1;i<=NF;i++) {total++; if ($i=="true")
ready++} print ready "/" total}')"
    RESTARTS="$(kubectl get pod "$POD" -n "$NAMESPACE" -o
jsonpath='{range .status.containerStatuses[*]}{.restartCount}{"
"}{end}' 2>/dev/null | awk '{sum=0; for (i=1;i<=NF;i++) sum+=$i; print
sum}')"
    ROLE="$(echo "$PATRONI_STATUS" | awk -F'|' -v pod="$POD" '$2 ~ pod
{gsub(/^[ \t]+|[ \t]+$/, "", $4) ; print tolower($4)}')"

    [ -z "$ROLE" ] && ROLE="unknown"

    if [ "$PHASE" = "Running" ] && echo "$READY" | awk -F/ '{exit
!($1==$2 && $2>0)}' ; then
        READY_COUNT=$((READY_COUNT + 1))
    fi

    case "$ROLE" in
        leader)
            LEADER_COUNT=$((LEADER_COUNT + 1))
        ;;
        replica)
            REPLICA_COUNT=$((REPLICA_COUNT + 1))
        ;;
    esac

    DETAILS="${DETAILS}${POD} role=${ROLE} node=${NODE} ready=${READY}
phase=${PHASE} restarts=${RESTARTS}; "
done

UNIQUE_NODES=$(for POD in $PODS ; do kubectl get pod "$POD" -n
"$NAMESPACE" -o jsonpath='{.spec.nodeName}{"\n"}' ; done | sort -u | wc
-l)

if [ "$POD_COUNT" -ne "$EXPECTED_PODS" ] ; then
    echo "CRITICAL - Pod szám eltérés: ${POD_COUNT}/${EXPECTED_PODS}.
${DETAILS}"
    exit 2
fi

if [ "$READY_COUNT" -ne "$EXPECTED_PODS" ] ; then
    echo "CRITICAL - Nem minden pod ready:
${READY_COUNT}/${EXPECTED_PODS}. ${DETAILS}"
    exit 2
fi

if [ "$LEADER_COUNT" -ne 1 ] ; then
```

```
    echo "CRITICAL - Leader darabszám hibás: ${LEADER_COUNT}.
${DETAILS}"
    exit 2
fi

if [ "$REPLICA_COUNT" -lt 2 ] ; then
    echo "WARNING - Kevesebb mint 2 replica látszik: ${REPLICA_COUNT}.
${DETAILS}"
    exit 1
fi

if [ "$UNIQUE_NODES" -lt "$EXPECTED_PODS" ] ; then
    echo "WARNING - Nem minden PostgreSQL pod külön node-on fut:
${UNIQUE_NODES}/${EXPECTED_PODS}. ${DETAILS}"
    exit 1
fi

echo "OK - StackGres PostgreSQL rendben.
Pods=${READY_COUNT}/${EXPECTED_PODS}, Leader=${LEADER_COUNT},
Replicas=${REPLICA_COUNT}, Nodes=${UNIQUE_NODES}/${EXPECTED_PODS}.
${DETAILS}"

exit 0
```

Futtatás eredménye

```
$ ./02-check-stackgres-postgresql-status.sh
OK - StackGres PostgreSQL rendben. Pods=3/3, Leader=1, Replicas=2,
Nodes=3/3. pg-cluster-0 role=leader node=kube01 ready=5/5 phase=Running
restarts=0; pg-cluster-1 role=replica node=kube03 ready=5/5 phase=Running
restarts=13; pg-cluster-2 role=replica node=kube02 ready=5/5 phase=Running
restarts=0;
```

Riport szkriptek

StackGres riport

00-report-stackgres-postgresql-status.sh

```
#!/usr/bin/env bash

NAMESPACE="lapodb-postgres"

echo "=================================================="
echo "StackGres PostgreSQL klaszter állapot"
echo "Dátum: $(date)"
echo "=================================================="
```

```
echo -e "\n=== Kubernetes node-ok ===\n"
kubectl get nodes -o wide


echo -e "\n=== PostgreSQL pod-ok ===\n"
kubectl get pods -n "$NAMESPACE" -o wide


echo -e "\n=== Patroni klaszter állapot ===\n"
LEADER_POD=$(kubectl get pods -n "$NAMESPACE" -o name | grep pg-cluster
| head -1 | sed 's#pod/##')
kubectl exec -n "$NAMESPACE" "$LEADER_POD" -c patroni -- patronictl
list 2>/dev/null || "ERROR: patronictl nem található"


echo -e "\n=== PostgreSQL pod-ok elhelyezkedése ===\n"
printf "%-20s %-10s %-15s %-8s %-8s\n" "POD" "ROLE" "NODE" "READY"
"RESTARTS"
for POD in $(kubectl get pods -n "$NAMESPACE" -o name | grep pg-cluster
| sed 's#pod/##') ; do
    NODE=$(kubectl get pod "$POD" -n "$NAMESPACE" -o
jsonpath='{.spec.nodeName}')
    READY=$(kubectl get pod "$POD" -n "$NAMESPACE" -o
jsonpath='{.status.containerStatuses[*].ready}' | awk
'{c=0;for(i=1;i<=NF;i++) if($i=="true") c++; print c"/"NF}')
    RESTARTS=$(kubectl get pod "$POD" -n "$NAMESPACE" -o
jsonpath='{.status.containerStatuses[*].restartCount}' | awk
'{s=0;for(i=1;i<=NF;i++) s+=$i; print s}')
    ROLE=$(kubectl exec -n "$NAMESPACE" "$LEADER_POD" -c patroni --
patronictl list 2>/dev/null | awk -F'|' -v pod="$POD" '$2 ~ pod {
gsub(/^[ \t]+|[ \t]+$/, "", $4) ; print tolower($4) }')
    printf "%-20s %-10s %-15s %-8s %-8s\n" "$POD" "${ROLE:-unknown}"
"$NODE" "$READY" "$RESTARTS"
done


echo -e "\n=== Leader(ek) ===\n"
LEADERS=$(kubectl exec -n "$NAMESPACE" "$LEADER_POD" -c patroni --
patronictl list 2>/dev/null | awk -F'|' 'tolower($4) ~ /leader/ {
gsub(/^[ \t]+|[ \t]+$/, "", $2) ; print $2 }')
echo "$LEADERS"

echo -e "\n======================================================="
```

Futtatás eredménye

```
$ ./00-report-stackgres-postgresql-status.sh
=======================================================
StackGres PostgreSQL klaszter állapot
```

```
Dátum: 2026. jún. 10., szerda, 15:10:21 CEST
=================================================

=== Kubernetes node-ok ===

NAME     STATUS   ROLES                    AGE   VERSION   INTERNAL-IP
EXTERNAL-IP   OS-IMAGE                         KERNEL-VERSION
CONTAINER-RUNTIME
kube01   Ready    control-plane,postgres   38d   v1.36.0   192.168.110.161
<none>        AlmaLinux 10.1 (Heliotrope Lion)
6.12.0-124.52.3.el10_1.x86_64 (amd64)   containerd://2.2.3
kube02   Ready    control-plane,postgres   38d   v1.36.0   192.168.110.162
<none>        AlmaLinux 10.1 (Heliotrope Lion)
6.12.0-124.52.3.el10_1.x86_64 (amd64)   containerd://2.2.3
kube03   Ready    control-plane,postgres   38d   v1.36.0   192.168.110.163
<none>        AlmaLinux 10.1 (Heliotrope Lion)
6.12.0-124.52.3.el10_1.x86_64 (amd64)   containerd://2.2.3

=== PostgreSQL pod-ok ===

NAME           READY   STATUS    RESTARTS        AGE    IP            NODE
NOMINATED NODE   READINESS GATES
pg-cluster-0   5/5     Running   0               128m   10.244.0.9    kube01
<none>           <none>
pg-cluster-1   5/5     Running   13 (118m ago)   37d    10.244.2.16   kube03
<none>           <none>
pg-cluster-2   5/5     Running   0               39m    10.244.1.15   kube02
<none>           <none>

=== Patroni klaszter állapot ===

+ Cluster: pg-cluster (7635689612010042939) -------+----+-------------+-----
+------------+-----+
| Member       | Host        | Role    | State     | TL | Receive LSN | Lag
| Replay LSN | Lag |
+--------------+-------------+---------+-----------+----+-------------+-----
+------------+-----+
| pg-cluster-0 | 10.244.0.9  | Leader  | running   |  6 |             |
|            |     |
| pg-cluster-1 | 10.244.2.16 | Replica | streaming |  6 |   0/60002B0 |   0
|  0/60002B0 |   0 |
| pg-cluster-2 | 10.244.1.15 | Replica | streaming |  6 |   0/60002B0 |   0
|  0/60002B0 |   0 |
+--------------+-------------+---------+-----------+----+-------------+-----
+------------+-----+

=== PostgreSQL pod-ok elhelyezkedése ===

POD                  ROLE       NODE            READY    RESTARTS
pg-cluster-0         leader     kube01          5/5      0
pg-cluster-1         replica    kube03          5/5      13
```

```
pg-cluster-2         replica    kube02          5/5      0

=== Leader(ek) ===

pg-cluster-0

=====================================================
```

Tesztelés

Kiinduló állapot

Előkészítő műveletek

Először be kell állítani a tesztelés során használt környezeti változókat

```
export NAMESPACE=<névtér>
export CLUSTER=<klaszter>
```

Kubernetes objektumok lekérdezése

```
$ kubectl get pods -n $NAMESPACE -o wide
NAME           READY   STATUS    RESTARTS       AGE    IP            NODE
NOMINATED NODE   READINESS GATES
pg-cluster-0   5/5     Running   0              75m    10.244.0.9    kube01
<none>           <none>
pg-cluster-1   5/5     Running   13 (65m ago)   37d    10.244.2.16   kube03
<none>           <none>
pg-cluster-2   5/5     Running   1 (169m ago)   170m   10.244.1.9    kube02
<none>           <none>

$ kubectl get sgcluster -n $NAMESPACE
NAME         VERSION   FLAVOR   INSTANCES   PROFILE              DISK   AGE
pg-cluster   18.3              3           pg-cluster-default   8Gi    37d
```

Patroni állapot lekérdezése

```
$ kubectl exec -n $NAMESPACE -it pg-cluster-0 -c patroni -- patronictl list
+ Cluster: pg-cluster (7635689612010042939) -------+----+-------------+-----
+------------+-----+
| Member       | Host        | Role    | State     | TL | Receive LSN | Lag
| Replay LSN | Lag |
+--------------+-------------+---------+-----------+----+-------------+-----
+------------+-----+
| pg-cluster-0 | 10.244.0.9  | Replica | streaming |  5 |   0/50002B0 |   0
```

```
|  0/50002B0 |   0 |
| pg-cluster-1 | 10.244.2.16 | Replica | streaming |  5 |   0/50002B0 |   0
|  0/50002B0 |   0 |
| pg-cluster-2 | 10.244.1.9  | Leader  | running   |  5 |             |
|            |     |
+--------------+-------------+---------+-----------+----+-------------+-----
+------------+-----+
```

Ebben a példában a leader a `pg-cluster-2`.

Leader POD törlés

```
$ kubectl delete pod -n $NAMESPACE pg-cluster-2
pod "pg-cluster-2" deleted from lapodb-postgres namespace
```

POD ellenőrzése

```
$ watch -n 2 "kubectl get pods -n $NAMESPACE -o wide"

Every 2,0s: kubectl get pods -n lapodb-postgres -o wide
note02: 2026. jún. 10., szerda, 14:24:24 CEST
in 0,252s (0)
NAME           READY   STATUS    RESTARTS   AGE     IP            NODE
NOMINATED NODE   READINESS GATES
pg-cluster-0   5/5     Running   0              82m     10.244.0.9    kube01
<none>           <none>
pg-cluster-1   5/5     Running   13 (72m ago)   37d     10.244.2.16   kube03
<none>           <none>
pg-cluster-2   5/5     Running   0              2m10s   10.244.1.14   kube02
<none>           <none>
```

Ellenőrzés

Új leader POD került kiválasztásra. Mivel a leader szerepkör átkerült egy másik POD-ra, az adatbázis kapcsolat megszakadt a kliens és a szerver között. Ezt a szakadást kezelni kell az alkalmazásban.

```
$ kubectl exec -n $NAMESPACE -it pg-cluster-0 -c patroni -- patronictl list
+ Cluster: pg-cluster (7635689612010042939) -------+----+-------------+-----
+------------+-----+
| Member       | Host        | Role    | State     | TL | Receive LSN | Lag
| Replay LSN | Lag |
+--------------+-------------+---------+-----------+----+-------------+-----
+------------+-----+
| pg-cluster-0 | 10.244.0.9  | Leader  | running   |  6 |             |
|            |     |
| pg-cluster-1 | 10.244.2.16 | Replica | streaming |  6 |   0/60002B0 |   0
|  0/60002B0 |   0 |
```

```
| pg-cluster-2 | 10.244.1.14 | Replica | streaming |  6 |   0/60002B0 |   0
|  0/60002B0 |   0 |
+--------------+-------------+---------+-----------+----+-------------+-----
+------------+-----+
```

Replika POD törlés

Replika pod törlése esetén a kliens és szerver közötti kapcsolat nem szakad meg.

```
$ kubectl delete pod -n $NAMESPACE pg-cluster-2
pod "pg-cluster-2" deleted from lapodb-postgres namespace

$ kubectl get pods -n $NAMESPACE -o wide
NAME           READY   STATUS     RESTARTS       AGE   IP            NODE
NOMINATED NODE   READINESS GATES
pg-cluster-0   5/5     Running    0              89m   10.244.0.9    kube01
<none>           <none>
pg-cluster-1   5/5     Running    13 (79m ago)   37d   10.244.2.16   kube03
<none>           <none>
pg-cluster-2   0/5     Init:0/1   0              9s    10.244.1.15   kube02
<none>           <none>

$ kubectl get pods -n $NAMESPACE -o wide
NAME           READY   STATUS            RESTARTS       AGE   IP
NODE     NOMINATED NODE   READINESS GATES
pg-cluster-0   5/5     Running           0              89m   10.244.0.9
kube01   <none>           <none>
pg-cluster-1   5/5     Running           13 (79m ago)   37d   10.244.2.16
kube03   <none>           <none>
pg-cluster-2   0/5     PodInitializing   0              12s   10.244.1.15
kube02   <none>           <none>

$ kubectl get pods -n $NAMESPACE -o wide
NAME           READY   STATUS    RESTARTS       AGE   IP            NODE
NOMINATED NODE   READINESS GATES
pg-cluster-0   5/5     Running   0              90m   10.244.0.9    kube01
<none>           <none>
pg-cluster-1   5/5     Running   13 (79m ago)   37d   10.244.2.16   kube03
<none>           <none>
pg-cluster-2   4/5     Running   0              34s   10.244.1.15   kube02
<none>           <none>

$ kubectl exec -n $NAMESPACE -it pg-cluster-0 -c patroni -- patronictl list
+ Cluster: pg-cluster (7635689612010042939) -------+----+-------------+-----
+------------+-----+
| Member       | Host        | Role    | State     | TL | Receive LSN | Lag
| Replay LSN | Lag |
+--------------+-------------+---------+-----------+----+-------------+-----
+------------+-----+
| pg-cluster-0 | 10.244.0.9  | Leader  | running   |  6 |             |
```

```
|            |     |
| pg-cluster-1 | 10.244.2.16 | Replica | streaming |  6 |   0/60002B0 |   0
|  0/60002B0 |   0 |
| pg-cluster-2 | 10.244.1.15 | Replica | streaming |  6 |   0/60002B0 |   0
|  0/60002B0 |   0 |
+--------------+-------------+---------+-----------+----+-------------+-----
+------------+-----+
```

Ellenőrzések

- Leader: `pg-cluster-0`
- Replikák: `pg-cluster-1`, `pg-cluster-2`

Leader oldali ellenőrzés

```
$ kubectl exec -n $NAMESPACE -it pg-cluster-0 -c postgres-util -- psql -U
postgres -c "SELECT application_name, client_addr, state, sync_state,
sent_lsn, replay_lsn FROM pg_stat_replication"
 application_name | client_addr |   state   | sync_state | sent_lsn  |
replay_lsn
------------------+-------------+-----------+------------+-----------+------
------
 pg-cluster-1     | 10.244.2.16 | streaming | async      | 0/60002B0 |
0/60002B0
 pg-cluster-2     | 10.244.1.15 | streaming | async      | 0/60002B0 |
0/60002B0
```

Replika oldali ellenőrzés

```
$ kubectl exec -n $NAMESPACE -it pg-cluster-1 -c postgres-util -- psql -U
postgres -c "SELECT pg_is_in_recovery(), pg_last_wal_receive_lsn(),
pg_last_wal_replay_lsn()"
 pg_is_in_recovery | pg_last_wal_receive_lsn | pg_last_wal_replay_lsn
-------------------+-------------------------+-----------------------
 t                 | 0/60002B0               | 0/60002B0
(1 row)

$ kubectl exec -n $NAMESPACE -it pg-cluster-2 -c postgres-util -- psql -U
postgres -c "SELECT pg_is_in_recovery(), pg_last_wal_receive_lsn(),
pg_last_wal_replay_lsn()"
 pg_is_in_recovery | pg_last_wal_receive_lsn | pg_last_wal_replay_lsn
-------------------+-------------------------+-----------------------
 t                 | 0/6000000               | 0/60002B0
(1 row)
```

Mentés, helyreállítás

Előkészítő műveletek

Felhasználó és adatbázis létrehozás

A mentés teszteléséhez szükséges felhasználó és adatbázisok létrehozása.

```
$ kubectl exec -n $NAMESPACE -it pg-cluster-0 -c postgres-util -- bash
bash-4.4$ createuser eesztuser
bash-4.4$ createdb -O eesztuser eesztdb
bash-4.4$ createdb -O eesztuser eesztdb_restore
bash-4.4$ exit
exit
```

Teszt tábla létrehozása, adatok feltöltése

```
$ kubectl exec -n $NAMESPACE -it pg-cluster-0 -c postgres-util -- psql -U
eesztuser -d eesztdb
psql (18.3 (OnGres 18.3-build-6.49))
Type "help" for help.

eesztdb=> CREATE TABLE backup_test (
eesztdb(>     id           BIGSERIAL PRIMARY KEY,
eesztdb(>     created_at   TIMESTAMPTZ NOT NULL DEFAULT now(),
eesztdb(>     username     VARCHAR(50) NOT NULL,
eesztdb(>     email        VARCHAR(100),
eesztdb(>     amount       NUMERIC(12,2),
eesztdb(>     active       BOOLEAN DEFAULT true,
eesztdb(>     note         TEXT
eesztdb(> );
CREATE TABLE
eesztdb=> INSERT INTO backup_test
eesztdb-> (username,email,amount,active,note)
eesztdb-> SELECT
eesztdb->     'user' || g,
eesztdb->     'user' || g || '@example.com',
eesztdb->     round((random()*10000)::numeric,2),
eesztdb->     random() > 0.2,
eesztdb->     md5(random()::text)
eesztdb-> FROM generate_series(1,10000) g;
INSERT 0 10000
eesztdb=> SELECT count(*) FROM backup_test;
 count
-------
 10000
(1 row)
```

```
eesztdb=> \q
```

Mentő szkript

A szkript kiolvassa a használt névteret, és az elsődleges POD-ot.

03-stackgres-postgresql-backup.sh

```
#!/usr/bin/env bash

DEBUG=1
TIMESTAMP=$(date +%Y%m%d_%H%M%S)

PGDATABASE=eesztdb
PGUSER=eesztuser

NAMESPACE=$(kubectl get sgcluster --no-headers -A | awk '{ print $1 }')
PRIMARY=$(kubectl get pod -n $NAMESPACE -l
app=StackGresCluster,role=primary -o
jsonpath='{.items[0].metadata.name}')

if [ $DEBUG -gt 0 ] ; then
    echo "NAMESPACE: $NAMESPACE"
    echo "PRIMARY: $PRIMARY"
fi

if [ -z "$PRIMARY" ] ; then
    echo "Nem található primary pod"
    exit 1
fi

kubectl exec -n "$NAMESPACE" "$PRIMARY" -c postgres-util -- pg_dump -U
"$PGUSER" -d "$PGDATABASE" --no-owner --no-privileges >
"$PGDATABASE"_"$TIMESTAMP".sql
```

Futtatás eredménye

```
$ ./03-stackgres-postgresql-backup.sh
NAMESPACE: lapodb-postgres
PRIMARY: pg-cluster-0

$ ls -1
eesztdb_20260610_162054.sql
```

Helyreállító szkript

A helyreállítás az `eesztdb_restore` adatbázist használja.

03-stackgres-postgresql-restore.sh

```
#!/usr/bin/env bash

DEBUG=1

if [ $# -ne 1 ] ; then
    echo "Használat: $0 backupfile.sql"
    exit 1
fi

BACKUP_FILE="$1"

if [ ! -f "$BACKUP_FILE" ] ; then
    echo "Nem található a mentési fájl: $BACKUP_FILE"
    exit 1
fi

PGDATABASE=eesztdb_restore
PGUSER=eesztuser

NAMESPACE=$(kubectl get sgcluster --no-headers -A | awk '{ print $1 }')
PRIMARY=$(kubectl get pod -n "$NAMESPACE" -l
app=StackGresCluster,role=primary -o
jsonpath='{.items[0].metadata.name}')

if [ $DEBUG -gt 0 ] ; then
    echo "NAMESPACE: $NAMESPACE"
    echo "PRIMARY: $PRIMARY"
    echo "BACKUP_FILE: $BACKUP_FILE"
fi

if [ -z "$PRIMARY" ] ; then
    echo "Nem található primary pod"
    exit 1
fi

kubectl exec -i -n "$NAMESPACE" "$PRIMARY" -c postgres-util -- psql -U
"$PGUSER" -d "$PGDATABASE" < "$BACKUP_FILE"
```

Futtatás eredménye

```
$ ./03-stackgres-postgresql-restore.sh eesztdb_20260610_162054.sql
NAMESPACE: lapodb-postgres
PRIMARY: pg-cluster-0
BACKUP_FILE: eesztdb_20260610_162054.sql
SET
SET
SET
SET
```

```
SET
SET
 set_config
------------

(1 row)

SET
SET
SET
SET
SET
SET
CREATE TABLE
CREATE SEQUENCE
ALTER SEQUENCE
ALTER TABLE
COPY 10000
 setval
--------
  10000
(1 row)

ALTER TABLE
```

Ellenőrzés

```
$ kubectl exec -n $NAMESPACE -it pg-cluster-0 -c postgres-util -- psql -U
eesztuser -d eesztdb -c "SELECT count(*) FROM backup_test"
 count
-------
 10000
(1 row)

$ kubectl exec -n $NAMESPACE -it pg-cluster-0 -c postgres-util -- psql -U
eesztuser -d eesztdb_restore -c "SELECT count(*) FROM backup_test"
 count
-------
 10000
(1 row)
```

From:
https://wiki.r-l.hu/ - **Reverse-Logic wiki**

Permanent link:
https://wiki.r-l.hu/doku.php?id=kubernetes:stackgres:monitoring

Last update: **2026/06/10 14:38**