Blue/Green Deployments Pre-Check Runbook

Revision 1.2, August 2024

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents current AWS product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

Contents

Introduction

Runbook audience and overview

This runbook is designed for internal use by Solutions Architects, Specialist Sales, and other roles who work directly with customers to troubleshoot issues related to Aurora PostgreSQL and RDS Postgres. It describes the recommended steps and best practices for evaluating the readiness of customer RDS/Aurora PostgreSQL databases for Blue/Green Deployments, and should be used when

1. A customer is wanting to automate the verification of a large number of database clusters' readiness for usage with Blue/Green Deployments
2. Diagnosing a cluster on which creation of Blue/Green Deployments already failed.

The runbook is specifically written for Blue/Green Deployments as it relates to the PostgreSQL engine, and will focus on issues that can prevent the usage of Blue/Green Deployments with RDS Postgres and Aurora PostgreSQL

By following this runbook, you should be able to educate the customer about key best practices, and assist the customer in identifying current limitations in Blue/Green Deployments for RDS/Aurora PostgreSQL. This document will be updated in response to field feedback and lessons learned. We encourage you to share your feedback, which will allow us to evolve and improve the runbook.

Why do I need a Blue/Green Deployments readiness check?

While Blue/Green Deployments contains guardrails that prevent customers from both inadvertently breaking replication once established or creating an unhealthy Blue/Green Deployments cluster, two high level issues can prevent users from enabling this feature. These issues currently include: 1). Lack of a primary key or identifier on any table within any database within the cluster to used and 2). Existing logical replication slots which are used to consume/replicate data into/out of the cluster to be used by Blue/Green Deployments.

While Blue/Green Deployments already checks for these conditions prior to creating a Blue/Green Deployment, there is currently no feature capability to estimate readiness of a fleet of clusters to be used with Blue/Green Deployments (and propose solutions for identified problems preventing Blue/Green Deployment's usage).

Why does Blue/Green Deployments have these limitations with PostgreSQL?

Blue/Green Deployments for RDS/Aurora PostgreSQL is a feature built primarily on PostgreSQL core Logical Replication. Due to the current implementation of Blue/Green Deployments, this managed feature requires many of the same preconditions which are already required when creating replication between two PostgreSQL clusters using Logical Replication. The main requirement stemming from Logical Replication is the requirement for each table to either 1). have a Primary Key or 2). the REPLICA IDENTITY setting of the table altered to FULL using an ALTER TABLE command.

Another limitation to be aware of when enabling Blue/Green Deployments for RDS/Aurora PostgreSQL is the current requirement to drop all logical replication slots before initiating a Blue/Green Deployment. This means that Blue/Green Deployments required a pre-condition which will cause upstream and downstream sources of data from the proposed Blue/Green cluster to lose their logical replication slot (and the LSN at which Blue/Green Deployments was started/ended. If your customer is currently using PostgreSQL logical replication as a primary component of their data model, Blue/Green Deployments is likely not a good fit for the customer's use-case.

Queries used to Determine Blue/Green Readiness:

These scripts can be used to estimate readiness of customers to use Blue/Green Deployments with the current feature requirements can be observed below. Note, all of these queries are per-database. Additioannly, all of these queries are contained in a ready-to-use script which can be used to check Blue/Green Readiness across an entire RDS/Aurora PostgreSQL Cluster:

```
WITH no_primary_key_or_replica_identity AS (
    SELECT
        n.nspname AS schema_name,
        c.relname AS table_name,
        'Missing primary key or replica identity' AS reason
    FROM
        pg_class c
    JOIN
        pg_namespace n ON n.oid = c.relnamespace
    LEFT JOIN
        pg_index i ON i.indrelid = c.oid AND i.indisprimary
    WHERE
        c.relkind = 'r'  -- Only ordinary tables
        AND n.nspname NOT IN ('pg_catalog', 'information_schema')
        AND (i.indisprimary IS NULL OR c.relreplident = 'd')  --
Missing primary key or default replica identity
),
pg_largeobject_tables AS (
    SELECT
        'pg_catalog' AS schema_name,
        'pg_largeobject' AS table_name,
        'Contains pg_largeobject' AS reason
),
foreign_tables AS (
    SELECT
        table_schema AS schema_name,
        table_name,
        'Foreign table' AS reason
    FROM
        information_schema.tables
    WHERE
        table_type = 'FOREIGN'
        AND table_schema NOT IN ('pg_catalog', 'information_schema')
)
SELECT * FROM no_primary_key_or_replica_identity
UNION ALL
SELECT * FROM pg_largeobject_tables
UNION ALL
SELECT * FROM foreign_tables
ORDER BY schema_name, table_name;
```

Usage: *Run this query in each database within the cluster that we desire to use with Blue/Green Deployments*

Query description: *Lists any tables incompatible with Blue/Green Deployments and list the reason why*

Sample output:

```
 schema_name |   table_name    |                  reason
-------------+-----------------+---------------------------------
 hr          | departments     | Missing primary key or replica identity
 hr          | employees       | Missing primary key or replica identity
 hr          | positions       | Missing primary key or replica identity
 hr          | salaries        | Missing primary key or replica identity
 pg_catalog  | pg_largeobject  | Contains pg_largeobject
(5 rows)
```

```
SELECT slot_name, slot_type, database, active FROM
pg_replication_slots WHERE slot_type = 'logical';
```

Usage: *Run this query in each database within the cluster that we desire to use with Blue/Green Deployments*

Query description: *Lists any existing PostgreSQL replication slots which must be removed before attempting to utilize Blue/Green Deployments. The output from this query applies at the cluster level.*

Sample Output:

```
  slot_name   | slot_type | database | active
--------------+-----------+----------+--------
 hr_db1_slot | logical   | hr_db1   | f
 hr_db2_slot | logical   | hr_db2   | f
 hr_db3_slot | logical   | hr_db3   | f
(3 rows)
```

```
SELECT * FROM pg_largeobject;
```

Usage: *Run this query in each database within the cluster that we desire to use with Blue/Green Deployments*

Query description: *Lists any pg_largeobjects which exist in a given database*

Sample Output:

```
 loid | pageno |           data
------+--------+---------------
-----------------
 17294 |   0    | this is a test
data file. \015
(1 row)
```

```
SELECT foreign_table_schema, foreign_table_name,
foreign_server_name FROM
information_schema.foreign_tables;
```

Usage: *Run this query in each database within the cluster that we desire to use with Blue/Green Deployments*

Query description: *Lists any existing PostgreSQL foreign tables. These tables will not be replicated using Blue/Green Deployments, but will not break replication.*

Sample Output:

```
foreign_table_schema | foreign_table_name | foreign_server_name
---------------------+--------------------+---------------------
 oracle_fdw_hr_schema |      employees     |   onprem_ora_1
(1 row)
```

Interpreting and Acting on the Script Output

Tables missing PRIMARY KEYS or not using REPLICA IDENTITY = FULL

In order to utilize Blue/Green Deployments, all tables in all databases within the cluster to be used with Blue/Green deployments needs to have a PRIMARY KEY enabled on the table, or the table must have the REPLICA IDENTITY setting = Full. The following SQL example can be used to add a primary key to a table missing it. Alternatively, a SET command can also be used to change the REPLICA IDENTITY setting for the affected table:

Address Existing Logical Replication Slots

All logical replication slots must be removed from the candidate cluster before it can be used with Blue/Green Deployments. The impact of this action must be understood, since removing logical replication slots from active publications/subscriptions has the potential to cause data loss (especially if done in error). If logical replication slots must be preserved, Blue/Green Deployments is not currently an option for these workloads.

Best Practices

Working with a large number of databases/tables

As previously mentioned, Blue/Green deployments is based on PostgreSQL logical replication. Because this feature uses the same publisher/subscriber model, we must be mindful that replication occurs at the database level (not the cluster level. This implies that as the number of databases in a given PostgreSQL cluster increases, the number of publishers and subscribers managed by Blue/Green Deployments must also increase. For each database replicated, there will be a corresponding increase in CPU and memory consumption on the Blue cluster, which impacts performance on that (production) cluster. Tuning several parameters based on the number of databases in the cluster to be replicated can be helpful, such as `max_replication_slots, max_wal_sender, max_logical_replication_worker`, and `max_worker_process`. More information on tuning these settings can be found in the following blog: New – Fully managed Blue/Green Deployment in Amazon Aurora PostgreSQL and Amazon RDS for PostgreSQL

Large Objects

Large database objects (pg_largeobject) are not currently supported by PostgreSQL logical replication. If your database cluster contains large objects, they will not be logically replicated into the Green environment (resulting in data loss). Migrating all large database objects into S3 or using another storage mechanism such as `bytea` can be workarounds to this current limitation.

Foreign Tables

Foreign tables are not currently supported by Blue/Green Deployments and will not be replicated into the green environment. These items must be manually configured in the Green environment.

Resources

Public knowledge resources

- Blue/Green Readiness code repository: Sample Blue/Green Deployment Precheck Resources for RDS/Aurora PostgreSQL
- Documentation: Overview of Amazon RDS Blue/Green Deployments for Aurora
- Blog : New – Fully managed Blue/Green Deployment in Amazon Aurora PostgreSQL and Amazon RDS for PostgreSQL

Training/Hands-On Workshops

- Training: Aurora PostgreSQL Immersion Day – Blue/Green Deployment
- Training: RDS PostgreSQL Immersion Day – Blue/Green Deployments
- Training: Troubleshoot Amazon Aurora PostgreSQL Performance Workshop
- Training: PostgreSQL Fundamentals

Getting help troubleshooting PostgreSQL workloads

- Use Premium Support engagements (Support Cases) for service troubleshooting and break & fix assistance.
- Internal wiki supported by Specialist Solutions Architects for PostgreSQL
- Use SpecReqs for architectural guidance, and assistance with troubleshooting customer PostgreSQL performance concerns. Note that all SpecReqs must be created as “Sales Support”, and must be linked to an SFDC opportunity with an estimated non-zero opportunity amount. Requests that don’t meet those requirements may not be fulfilled.

Contributors

The following individuals and organizations contributed to this document:

- Peter Celentano - Senor DB Specialist Solutions Architect, Amazon Web Services

Document history

Change	Description	Date
Initial publication	Initial publication	August 2024