Microsoft

Oracle to Azure PostgreSQL Migration Workarounds

Prepared by

Paula Berenguel
pabereng@microsoft.com

DM Jumpstart Engineering Team

Disclaimer

This document is provided “as-is”. Information and views expressed in this document, including URL and other Internet Web site references, may change without notice.

Some examples depicted herein are provided for illustration only and are fictitious. No real association or connection is intended or should be inferred.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

Contents

Introduction

This document purpose is to provide Architects, Consultants, DBAs and related roles with a guide for quick fixing / working around issues while migrating workloads from Oracle to Azure Database for PostgreSQL.

List of Workarounds

Find below the list of workarounds available in this document:

Oracle	PostgreSQL
Database Link	Foreign Data Wrapper
External Table	Foreign Table
Synonym	View / Set search_path
Global Temporary Table	Unlogged Table / Temp Table
Virtual column	View / Function / Trigger
Connect by	With Recursive
Reverse Index	Functional Index
Index Organized Table (IOT table)	Cluster the table according to an Index

1. Database Link

Oracle syntax:

```
CREATE PUBLIC DATABASE LINK remote_service USING 'remote_db';

SELECT * FROM employees@remote_service;
```

Azure PostgreSQL syntax:

```
CREATE SERVER remote_service FOREIGN DATA WRAPPER oracle_fdw OPTIONS (dbserver 'remote_db');

CREATE USER MAPPING FOR current_user SERVER remote_service OPTIONS (user 'scott', password
'tiger');

CREATE FOREIGN TABLE employees_fdw (<columns_list>) SERVER remote_service OPTIONS(schema 'HR',
table 'EMPLOYEES');
```

2. External Tables

Oracle Syntax:

```
 CREATE OR REPLACE DIRECTORY ext_dir AS '/data/ext/';
 CREATE TABLE ext_table (empno VARCHAR2(4), firstname VARCHAR2(20),  lastname VARCHAR2(20),
  age VARCHAR2(2)) ORGANIZATION EXTERNAL (DEFAULT DIRECTORY ext_dir ACCESS PARAMETERS (…
LOCATION ('file_ext.csv')));
cat /data/ext/file_ext.csv
1234,ALBERT,GRANT,21
1235,ALFRED,BLUEOS,
26 1236,BERNY,JOLYSE,34
```

Azure PostgreSQL syntax:

```
CREATE FOREIGN TABLE ext_table
(empno VARCHAR(4),
 firstname VARCHAR(20),
 lastname VARCHAR(20),
  age VARCHAR(2) )
SERVER ext_dir OPTIONS (filename '/data/ext/file_ext.csv', format 'csv', delimiter ',');
```

3. Synonyms

A synonym is an alias name for objects. They are used to make access to an object from another schema or a remote database simpler. Synonyms are not supported in PostgreSQL.

Oracle Syntax:

```
CREATE PUBLIC SYNONYM emp_table FOR hr.employees [@ dblink];
```

Azure PostgreSQL syntax:

There are two options for migrating synonyms: use search_path or use views:

--search path – session level – no permanent effect, it needs to be set for every connection

```
SET search_path TO other_schema;
```

--search path – role or database level – it takes permanent effect

```
--@postgresql

alter database <database_name> set search_path = "other_schema";

--@database_name

alter role <role_name> set search_path = "other_schema";
```

--view:

```
CREATE VIEW public.emp_table AS SELECT * FROM hr.employees;
ALTER VIEW public.emp_table OWNER TO hr;
GRANT ALL ON public.emp_table TO PUBLIC;
```

Note that with @DBLINK clause, the creation of a foreign table HR.EMPLOYEES using a foreign server is required (Ora2Pg will warn you to see DBLINK and FDW export type).

4. Global Temporary Tables

There are two options for migrating global temporary tables to Azure Database for PostgreSQL: use Unlogged tables or Temp Tables.

Option 1: Unlogged Table:

Oracle Syntax:

```
CREATE GLOBAL TEMPORARY TABLE MY_CONTRACT_MONTH
(ID NUMBER(10),
 CMONTH DATE
)
ON COMMIT DELETE ROWS;
```

Azure PostgreSQL Syntax:

```
CREATE UNLOGGED TABLE MY_CONTRACT_MONTH
(
  ID number,
  CMONTH timestamp,
  pid bigint default pg_backend_pid()
);
ALTER TABLE MY_CONTRACT_MONTH ENABLE ROW LEVEL SECURITY;
ALTER TABLE MY_CONTRACT_MONTH FORCE ROW LEVEL SECURITY;
CREATE POLICY cm_pid ON MY_CONTRACT_MONTH TO <role_name> USING (pid = (select
pg_backend_pid()));
```

The Role is also the User. Please read PostgreSQL Users and Roles implementations if questions arise.

With Unlogged table, row level security must be implemented, to prevent sessions from querying other sessions data.

There is also a need of implementing a job that eliminates the data on the unlogged table for the inactive sessions.

```
DELETE FROM smdr.contract_months cm WHERE not exists (select 1 from pg_stat_activity psa where psa.pid = cm.pid);
```

Option 2 – Temp table

Oracle Syntax:

```
CREATE GLOBAL TEMPORARY TABLE MY_CONTRACT_MONTH
(ID NUMBER(10),
 CMONTH DATE
)
ON COMMIT DELETE ROWS;
```

Oracle stores the definitions of temporary tables permanently like the definitions of regular tables.

Azure PostgreSQL Syntax:

```
CREATE GLOBAL TEMPORARY TABLE MY_CONTRACT_MONTH
(ID NUMERIC,
 CMONTH TIMESTAMP
)
ON COMMIT DELETE ROWS;
```

CREATE TEMPORARY TABLE statement creates a temporary table that is automatically dropped at the end of a session, or the current transaction (ON COMMIT DROP option).

During the conversion, you need to extract CREATE TEMPORARY TABLE statement from application code, stored procedures, triggers etc. and execute them once to create the temporary table definition.

- Oracle does not support ON COMMIT DROP, so if this option is required, you need to explicitly execute DROP TABLE statement after each COMMIT
- ON COMMIT PRESERVE ROWS is the default in PostgreSQL, while ON COMMIT DELETE ROWS is the default in Oracle

5. Virtual Column

One way of implementing the equivalent of a virtual column in PostgreSQL is by creating a view, as following:

Oracle Syntax

```
CREATE TABLE VIRT_COL_TABLE (
  id          NUMBER,
  first_name  VARCHAR2(10),
  last_name   VARCHAR2(10),
```

```
  salary      NUMBER(9,2),
  comm1       NUMBER(3),
  comm2       NUMBER(3),
  salary1     AS (ROUND(salary*(1+comm1/100),2)),
  salary2     NUMBER GENERATED ALWAYS AS (ROUND(salary*(1+comm2/100),2)) VIRTUAL,
);
```

Azure Database for PostgreSQL Syntax

```
CREATE TABLE virt_col_table (
        id bigint NOT NULL,
        first_name varchar(10),
        last_name varchar(10),
        salary double precision,
        comm1 smallint,
        comm2 smallint,
        salary1 bigint,
        salary2 bigint
) ;
```

Ora2pg implements it by creating an additional export file named VIRTUAL_COLUMNS_(...).sql contains the trigger definition to apply the default values of the original virtual columns:

```
DROP TRIGGER IF EXISTS virt_col_VIRT_COL_TABLE_trigger ON VIRT_COL_TABLE CASCADE;

CREATE OR REPLACE FUNCTION fct_virt_col_VIRT_COL_TABLE_trigger() RETURNS trigger AS $BODY$
BEGIN
        NEW.SALARY2 = ROUND(NEW.SALARY*(1+NEW.COMM2/100),2);
        NEW.SALARY1 = ROUND(NEW.SALARY*(1+NEW.COMM1/100),2);

RETURN NEW;
end
$BODY$
 LANGUAGE 'plpgsql' SECURITY DEFINER;

CREATE TRIGGER virt_col_VIRT_COL_TABLE_trigger
        BEFORE INSERT OR UPDATE ON VIRT_COL_TABLE FOR EACH ROW
        EXECUTE PROCEDURE fct_virt_col_VIRT_COL_TABLE_trigger();
```

6. Connect By – Hierarchical query

Hierarchical queries in Oracle using the CONNECT BY clause are converted in PostgreSQL to queries using WITH RECURSIVE clause

Oracle Syntax

```
CREATE TABLE taxonomy    (
    key NUMBER(11) NOT NULL CONSTRAINT taxPkey PRIMARY KEY,
    value VARCHAR2(255),
    taxHier NUMBER(11)    );

ALTER TABLE     taxonomy ADD CONSTRAINT     taxTaxFkey  FOREIGN KEY    (taxHier) REFERENCES
tax(key);

SELECT
     value
```

```
FROM
    taxonomy
CONNECT BY
    PRIOR key = taxHier
START WITH
    key = 0;
```

Azure Database for PostgreSQL Syntax

```
WITH RECURSIVE cte AS (
   SELECT key, value, 1 AS level
   FROM   taxonomy
   WHERE  key = 0

   UNION  ALL
   SELECT t.key, t.value, c.level + 1
   FROM   cte      c
   JOIN   taxonomy t ON t.taxHier = c.key
   )
SELECT value
FROM   cte
ORDER  BY level;
```

7. Reverse Index

This workaround is valid when the reverse index is applied against a TEXT column.

Oracle Syntax

```
CREATE TABLE REV_TEMP (

       Id NUMBER(10) NOT NULL PRIMARY KEY,

       Description VARCHAR2(512) NOT NULL

) ;

CREATE INDEX REV_TEMP_N1 ON REV_TEMP(Description) REVERSE;
```

Azure Database for PostgreSQL Syntax

```
CREATE TABLE REV_TEMP (

       Id NUMERIC(10) NOT NULL PRIMARY KEY,

       Description VARCHAR(512) NOT NULL

) ;

CREATE INDEX REV_TEMP_N1 ON REV_TEMP(REVERSE(Description));
```

8. Index Organized Table

The Oracle database uses heap tables by default. Index-organized tables can be created using the ORGANIZATION INDEX clause:

Oracle Syntax

```
CREATE TABLE IOT_TEMP (

        Id NUMBER(10) NOT NULL PRIMARY KEY,

        Description VARCHAR2(512) NOT NULL

) ORGANIZATION INDEX;
```

The Oracle database always uses the primary key as the clustering key.

Azure Database for PostgreSQL Syntax

PostgreSQL only uses heap tables. however, use the CLUSTER clause to align the contents of the heap table with an index.

```
CREATE TABLE IOT_TEMP (

        Id NUMERIC(10) NOT NULL PRIMARY KEY,

        Description VARCHAR(512) NOT NULL

) ;

CREATE INDEX IOT_TEMP_N1 ON IOT_TEMP(ID);
```

Known Unsupported features:

- Type inheritance and type with member method are not supported
- Global indexes over partitions are not supported
- Compound triggers are not supported

Feedback and suggestions

If you have feedback or suggestions for improving this data migration asset, please contact the Data Migration Jumpstart Team (askdmjfordmtools@microsoft.com). Thanks for your support!

Note: For additional information about migrating various source databases to Azure, see the Azure Database Migration Guide.