From Oracle to PostgreSQL (with ora2pg)

Markus Flechtner

@markusdba

www.markusdba.net
Trivadis – Our key figures.

- 16 Trivadis locations with more than 650 employees.
- Sales of CHF 115 million (EUR 106 million).
- Over 250 Service Level Agreements.
- More than 4000 training participants.
- Research and development budget: CHF 5.0 million.
- More than 1900 projects each year with over 800 customers.
- Financially independent and sustainably profitable.
Markus Flechtner

- Principal Consultant, Trivadis, Düsseldorf
- Oracle since 1990: Development, Support, DBA
- Focus: RAC, HA, Upgrade + Migration
- Teacher: RAC, New Features, Multitenant, PostgreSQL for Oracle DBAs
- Co-Author of the book "Der Oracle DBA" (Hanser, 2016)

@markusdba  www.markusdba.net | .de
Agenda

• Things to consider beforehand: Incompatibilities and different behaviour

• The tool ora2pg

• Database Assessment

• Database Migration

• Summary & Recommendations
Things to consider beforehand
General (1)

- PostgreSQL architecture is more like the Multitenant Architecture of Oracle
- PostgreSQL relies more on OS features and existing OS functionalities, e.g.
  - Less caching in the database, but more using OS caching
  - No internal archiver process but using OS commands for archiving (more flexible)
- Both ..
  - .. are ACID compliant
  - .. support MVCC
  - .. support ANSI SQL
- "Enhanced" Oracle compatibility for PostgreSQL
  - Enterprise DB Advanced Server
  - Extension orafce - https://github.com/orafce/orafce
General (2)

- Don't get confused by terms
  - Some terms have a different meaning in both RDBMS

<table>
<thead>
<tr>
<th>Diagram Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>User/Roles</td>
</tr>
<tr>
<td>Databases</td>
</tr>
<tr>
<td>Tables spaces</td>
</tr>
<tr>
<td>Schemas</td>
</tr>
<tr>
<td>Relations</td>
</tr>
<tr>
<td>File</td>
</tr>
</tbody>
</table>

- Users are roles with login privilege
- Tablespaces are directories, defined on cluster level
- Basically users and schemas are independent
General (3) - Housekeeping

- A database (schema) migration project is a good moment to do some housekeeping in your existing Oracle database.

- Check for
  - Invalid objects
  - Obsolete schemas
  - Obsolete tables
  - Obsolete PL/SQL code
Different behaviour

• When migrating databases and applications from Oracle to PostgreSQL, (small) differences in the behaviour of both RDBMS are hard to discover

• Test your applications carefully

• Let's see some examples of different behaviour ..
Handling NULLs

- Handling NULL (character) values is different in Oracle and PostgreSQL
- In Oracle an empty string is equal to NULL

```
SQL> select 'TEST'||null from dual;
TES
----
TEST
```

- The PostgreSQL way (empty string is different to NULL) is ANSI-SQL compatible

```
postgres=# select 'TEST'||null;
?column?
--------

(1 row)
```
Transaction Handling: Rollback

• In PostgreSQL DDL can be rolled back
• With Oracle, an error in the middle of a transaction will rollback just the last statement
• With PostgreSQL, an error in the middle of a transaction will rollback all the previous statements
  • Developers relying on the rollback behaviour must be careful
  • Using savepoints might do the job

• Example:

```sql
db=# BEGIN;
BEGIN
db=# INSERT INTO t VALUES (1);
INSERT 0 1
db=# SELECT 0/0;
ERROR: division by zero
```

```sql
db=# select * from t;
ERROR: current transaction is aborted, commands ignored until end of transaction block
```

```sql
db=# commit;
ROLLBACK
```
Constraint Behaviour

- Oracle enforces constraints per statement
- PostgreSQL enforces constraints per row
- Workaround: set constraint to "DEFERRABLE"

```sql
SQL> create table t (c number primary key);
Table created.
SQL> insert into t values (1);
1 row created.
SQL> insert into t values (2);
1 row created.
SQL> update t set c=c+1;
2 rows updated.
SQL> commit;
Commit complete.
```

```sql
postgres=# create table t (c numeric primary key);
CREATE TABLE
postgres=# insert into t values (1), (2);
INSERT 0 2

postgres=# update t set c=c+1;
ERROR:  duplicate key value violates unique constraint "t_pkey"
DETAIL:  Key (c)=(2) already exists.
```
Programming: PL/SQL vs. PL/pgSQL

• Both procedural languages are similar
• But code has to be converted and tested carefully
• Example:

```
CREATE FUNCTION get_bal(acc_no IN NUMBER)
RETURN NUMBER
IS
  acc_bal NUMBER(11,2);
BEGIN
  SELECT balance
  INTO acc_bal
  FROM accounts
  WHERE account_id = acc_no;
  RETURN(acc_bal);
END;
```

```
CREATE OR REPLACE FUNCTION get_bal(acc_no IN INT)
RETURN INT
IS
  acc_bal DECIMAL(11,2);
BEGIN
  SELECT balance
  INTO acc_bal
  FROM accounts
  WHERE account_id = acc_no;
  RETURN(acc_bal);
END;
$$ LANGUAGE plpgsql;
```

• When the query returns more than one row, Oracle will raise an error, but PostgreSQL returns the first rows
## Data Types (1)

<table>
<thead>
<tr>
<th>Oracle</th>
<th>PostgreSQL</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>VARCHAR2(n)</td>
<td>VARCHAR2(n)</td>
<td>in PostgreSQL n = number of characters, in Oracle it depends on NLS_LENGTH_SEMANTICS</td>
</tr>
<tr>
<td>NUMBER(n,m)</td>
<td>NUMERIC(n,m)</td>
<td></td>
</tr>
<tr>
<td>NUMBER(4)</td>
<td>SMALLINT</td>
<td>NUMERIC could be used for all variants</td>
</tr>
<tr>
<td>NUMBER(9)</td>
<td>INT</td>
<td></td>
</tr>
<tr>
<td>NUMBER(18)</td>
<td>BIGINT</td>
<td></td>
</tr>
<tr>
<td>NUMBER(n)</td>
<td>NUMERIC(n)</td>
<td>n&gt;18</td>
</tr>
</tbody>
</table>
## Data Types (2)

<table>
<thead>
<tr>
<th>Oracle</th>
<th>PostgreSQL</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATE</td>
<td>TIMESTAMP(0)</td>
<td>DATE in PostgreSQL does not contain time information</td>
</tr>
<tr>
<td>TIMESTAMP WITH LOCAL TIMEZONE</td>
<td>TIMESTAMPTZ</td>
<td></td>
</tr>
<tr>
<td>CLOB</td>
<td>TEXT</td>
<td>(1 GB limit in PostgreSQL)</td>
</tr>
<tr>
<td>BLOB</td>
<td>BYTEA</td>
<td>1 GB limit for BYTEA</td>
</tr>
<tr>
<td></td>
<td>LargeObject</td>
<td></td>
</tr>
</tbody>
</table>
Other differences

• Sequences

```sql
SQL> SELECT test_seq.NEXTVAL FROM dual;
```

```sql
postgres=# SELECT NEXTVAL('test_seq');
```

• Outer join
  • ANSI-SQL only, no "+" syntax

• DECODE function
  • ANSI-SQL: CASE

• Most of these differences can be handled by migration tools
Oracle things which are not available ..

- Synonyms
  - Use views instead
  - Remember the search order in "search path"
- Database Links
  - Use the extension "Foreign Data Wrapper to Oracle" instead
  - Available on [https://github.com/laurenz/oracle_fdw](https://github.com/laurenz/oracle_fdw)
- Temporary Tables
- Bitmap indexes
- Table DUAL
- SYSDATE
- ROWNUM, ROWID
The tool ora2pg
Ora2pg - Overview

- Open Source
- Free software (GNU GPL)
- Developed by Gilles Darold
- Available on http://ora2pg.darold.net
- Current version: 20 (January 2019)
- Written in Perl
- Requirements:
  - Oracle Client
  - Perl 5.10 or higher
  - DBI Perl module > 1.614
  - DBD::Oracle Perl
Installation

- Download current version from http://ora2pg.darold.net
- Install as root
  - Oracle Client must be installed beforehand
  - ORACLE_HOME must be set
  - LD_LIBRARY_PATH must be set to $ORACLE_HOME/lib

```bash
yum -y install cpan
cpan YAML
cpan Test::NoWarnings
cpan DBI
cpan DBD::Oracle

tar xjf ora2pg-20.0.tar.bz2
cd ora2pg-20.0/
perl Makefile.PL
make && make install
```
Features

• Migrates databases resp. schemas from Oracle to PostgreSQL
• Oracle database migration cost assessment
• Automatic database schema export
• Full and automatic data export and import
• Automatic conversion of PL/SQL to PL/pgSQL
• Oracle Spatial to PostGIS export
Migration project

• An "ora2pg migration project" is a directory structure which will be used by ora2pg
• Scripts for exporting and importing schema definitions and data are provided, too

```
postgres@pg4ora:~/ ora2pg --project_base /home/postgres/pg4ora \ 
--init_project hr_migration
postgres@pg4ora:~/ tree -d ~/mig_example/hr_migration/
/home/postgres/mig_example/hr_migration/
|-- config
|-- data
|-- reports
|-- schema
| |-- dblink
[..]
| |-- tables
| |-- sources
| | |-- functions
[..]
```
Configuration file

- Ora2pg requires a configuration file (usually named "ora2pg.conf")

```
ORACLE_DSN
dbi:Oracle:host=pg4ora.trivadistraing.com;SID=XEPDB1.trivadistraing.com

ORACLE_USER    system
ORACLE_PWD     manager

SCHEMA    HR
PG_VERSION  11
EXPORT_SCHEMA  1
CREATE_SCHEMA  1

OUTPUT hr_mig.sql
DROP_FKEY  1
TYPE TABLE PACKAGE COPY VIEW SEQUENCE TRIGGER FUNCTION PROCEDURE
```
Database Assessment
Database Assessment

• Every database migration starts with an assessment of the existing Oracle database
• Check for ..
  • Objects which cannot be migrated (with the tool)
  • Objects which are using Oracle specific syntax (e.g. "+" for outer joins or "DECODE"-function)
  • Objects which are converted from an Oracle object type to another PostgreSQL object type (e.g. synonyms -> views)
  • Objects with need additional attention ("manual work")
  • Objects which are using Oracle specific packages (DBMS_*, UTL_*)
• Estimation of the duration/cost of the migration
Database Assessment with ora2pg

- Estimate the costs and the additional time for modifications (e.g. PL/SQL-code)
  - Default time per "cost evaluation unit": 5 minutes
- Output formats:
  - Txt (default)
  - Html
  - Csv
- Example:

  ```
  postgres@pg4ora:~$ ora2pg \
  -c /home/postgres/p4ora/hr_migration/config/ora2pg.conf -t SHOW_REPORT \
  --estimate_cost --cost_unit_value 10 --dump_as_html >hr_mig_assessment.html
  ```
### Assessment Report – Example (1)

#### Ora2Pg - Database Migration Report

<table>
<thead>
<tr>
<th>Object</th>
<th>Number</th>
<th>Invalid</th>
<th>Estimated cost</th>
<th>Comments</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATABASE LINK</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GLOBAL TEMPORARY TABLE</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INDEX</td>
<td>19</td>
<td>0</td>
<td>5</td>
<td>11 index(es) are concerned by the export, which are automatically generated and will be so on PostgreSQL. Bitmap index will be exported as b-tree index(es) and hash index(es) will be exported as b-tree index(es). If any. Domain index are exported as b-tree but commented. Each index name not exists in PDB. Cluster, bitmap joins and IOT indexers will not be exported. Use the reverse function to reindex and search.</td>
<td></td>
</tr>
<tr>
<td>JOB</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Job are not exported. You may set external user job with them.</td>
<td></td>
</tr>
<tr>
<td>PROCEDURE</td>
<td>2</td>
<td>0</td>
<td>8</td>
<td>4248 bytes on</td>
<td></td>
</tr>
<tr>
<td>SEQUENCE</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>4244 bytes on</td>
<td></td>
</tr>
<tr>
<td>SYNONYM</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>SYNONYMS will be exported as views. SYNONYMS do not exists with PostgreSQL but a common workaround is to use views or set the PostgreSQL search path in your sessions to access object outside the current schema.</td>
<td></td>
</tr>
<tr>
<td>TABLE</td>
<td>7</td>
<td>0</td>
<td>1.2</td>
<td>2 check constraint.</td>
<td></td>
</tr>
<tr>
<td>TRIGGER</td>
<td>2</td>
<td>0</td>
<td>5</td>
<td>Total size of trigger code: 123 bytes.</td>
<td></td>
</tr>
<tr>
<td>VIEW</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>Views are not supported but can use specific functions.</td>
<td></td>
</tr>
</tbody>
</table>

**Total: 34**

19.20 cost migration units means approximately 1 man-day(s). The migration unit was set to 10 minute(s).
## Ora2Pg - Database Migration Report

<table>
<thead>
<tr>
<th>Object</th>
<th>Number</th>
<th>Invalid</th>
<th>Estimated cost</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATABASE LINK</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Database links will be exported as SQL/ADM. PostgreSQL’s Foreign Data Wrapper (FDW) extensions using oracle_fdw.</td>
</tr>
<tr>
<td>FUNCTION</td>
<td>4</td>
<td>0</td>
<td>27.8</td>
<td>Total size of function code: 1928 bytes. plate_counter: 5, f_pda_dirstring: 3</td>
</tr>
<tr>
<td>GLOBAL TEMPORARY TABLE</td>
<td>11</td>
<td>0</td>
<td>11.3</td>
<td>Migration units means approximately 23 man-day(s). Total 731. 1879.00 cost.</td>
</tr>
<tr>
<td>INDEX</td>
<td>112</td>
<td>0</td>
<td>11.3</td>
<td>Migration units means approximately 23 man-day(s). Total 731. 1879.00 cost.</td>
</tr>
</tbody>
</table>

### Migration level: C-5

- Migration levels:
  - A - Migration that might be run automatically
  - B - Migration with code rewrite and a human-days cost up to 5 days
  - C - Migration with code rewrite and a human-days cost above 5 days

- Technical levels:
  - 1 = trivial: no stored functions and no triggers
  - 2 = easy: no stored functions but with triggers, no manual rewriting
  - 3 = simple: stored functions and/or triggers, no manual rewriting
  - 4 = manual: no stored functions but with triggers or views with code rewriting
  - 5 = difficult: stored functions and/or triggers with code rewriting
Database Migration
Database Migration

- When using a "migration project" ora2pg provides two scripts for database (schema) migration

```
postgres@pg4ora:~/mig_example/hr_migration/ ls -l 
```

```
total 20
drwxrwxr-x. 2 postgres postgres  25 Sep 18 17:59 config
drwxrwxr-x. 2 postgres postgres  22 Sep 18 17:57 data
-rwx------. 1 postgres postgres 2010 Sep 18 16:56 export_schema.sh
-rwx------. 1 postgres postgres 16061 Sep 18 16:56 import_all.sh
drwxrwxr-x. 2 postgres postgres  62 Sep 18 16:58 reports
drwxrwxr-x. 17 postgres postgres 245 Sep 18 16:56 schema
drwxrwxr-x. 10 postgres postgres  131 Sep 18 16:56 sources
```

- Without a migration project you have to run the commands for extracting and importing data yourself
Step 1: Export Definitions

postgres@pg4ora:~/mig_example/hr_migration/ ./export_schema.sh
[====================================> 7/7 tables (100.0%) end of scanning.
[====================================> 10/10 objects types (100.0%) end of objects
auditing.
Running: ora2pg -p -t TABLE -o table.sql -b ./schema/tables -c
./config/ora2pg.conf
[====================================> 7/7 tables (100.0%) end of scanning.
[====================================> 7/7 tables (100.0%) end of table export.
Fixing function calls in output files...
Running: ora2pg -p -t PACKAGE -o package.sql -b ./schema/packages -c
./config/ora2pg.conf
[====================================> 0/0 packages (100.0%) end of output.
Fixing function calls in output files...
[...
To extract data use the following command:
ora2pg -t COPY -o data.sql -b ./data -c ./config/ora2pg.conf
Step 2: Extract Data

postgres@pg4ora:~/mig_example/hr_migration/  ora2pg -t COPY -o data.sql -b ./data -c ./config/ora2pg.conf

[========================>]  7/7 tables (100.0%) end of scanning.
[--------------------------]  25/25 rows (100.0%) Table COUNTRIES (25 recs/sec)
[=>]                        ]  25/215 total rows (11.6%) - (0 sec., avg: 25 recs/sec).
[--------------------------]  27/27 rows (100.0%) Table DEPARTMENTS (27 recs/sec)
[<==>]                     ]  52/215 total rows (24.2%) - (0 sec., avg: 52 recs/sec).
[--------------------------]  107/107 rows (100.0%) Table EMPLOYEES (107 recs/sec)
[==========================] 159/215 total rows (74.0%) - (0 sec., avg: 159 recs/sec).
[--------------------------]  19/19 rows (100.0%) Table JOBS (19 recs/sec)
[--------------------------]  178/215 total rows (82.8%) - (0 sec., avg: 178 recs/sec).
[--------------------------]  10/10 rows (100.0%) Table JOB_HISTORY (10 recs/sec)
[--------------------------]  188/215 total rows (87.4%) - (0 sec., avg: 188 recs/sec).
[--------------------------]  23/23 rows (100.0%) Table LOCATIONS (23 recs/sec)
[--------------------------]  211/215 total rows (98.1%) - (0 sec., avg: 211 recs/sec).
[--------------------------]  4/4 rows (100.0%) Table REGIONS (4 recs/sec)
[--------------------------]  215/215 total rows (100.0%) - (0 sec., avg: 215 recs/sec).
[--------------------------]  215/215 rows (100.0%) on total estimated data (1 sec.,
avg: 215 recs/sec)

Fixing function calls in output files...
Step 3: modify scripts

• It may be necessary to modify the scripts which were generated by ora2pg

• Examples:
  • Change tablespace definition
    Ora2pg output points to the directory of the Oracle tablespace
  • Remove grants
Step 4: run schema creation + import

postgres@pg4ora:~/mig_example/hr_migration/ ./import_all.sh -d hr_db -o hr -y
Database owner hr already exists, skipping creation.
Running: dropdb hr_db
Running: createdb -E UTF8 --owner hr hr_db
Running: psql --single-transaction -U hr -d hr_db -f ./schema/tables/table.sql
SET
SET
CREATE SCHEMA
ALTER SCHEMA
SET
CREATE TABLE
COMMENT
COMMENT
[...]
Step 5: Verify the Results

postgres@pg4ora:~/mig_example/hr_migration/ psql -d hr_db -U hr
psql (11.5)
Type "help" for help.

hr_db=> \d
List of relations

<table>
<thead>
<tr>
<th>Schema</th>
<th>Name</th>
<th>Type</th>
<th>Owner</th>
</tr>
</thead>
<tbody>
<tr>
<td>hr</td>
<td>countries</td>
<td>table</td>
<td>hr</td>
</tr>
<tr>
<td>hr</td>
<td>departments</td>
<td>table</td>
<td>hr</td>
</tr>
<tr>
<td>hr</td>
<td>departments_seq</td>
<td>sequence</td>
<td>hr</td>
</tr>
<tr>
<td>hr</td>
<td>emp_details_view</td>
<td>view</td>
<td>hr</td>
</tr>
<tr>
<td>hr</td>
<td>employees</td>
<td>table</td>
<td>hr</td>
</tr>
<tr>
<td>hr</td>
<td>employees_seq</td>
<td>sequence</td>
<td>hr</td>
</tr>
<tr>
<td>hr</td>
<td>job_history</td>
<td>table</td>
<td>hr</td>
</tr>
<tr>
<td>hr</td>
<td>jobs</td>
<td>table</td>
<td>hr</td>
</tr>
<tr>
<td>hr</td>
<td>locations</td>
<td>table</td>
<td>hr</td>
</tr>
<tr>
<td>hr</td>
<td>locations_seq</td>
<td>sequence</td>
<td>hr</td>
</tr>
<tr>
<td>hr</td>
<td>regions</td>
<td>table</td>
<td>hr</td>
</tr>
</tbody>
</table>

(11 rows)
Summary & Recommendations
Summary & Recommendations (1)

• It's possible to migrate databases from Oracle to PostgreSQL
• But think about the things around (support, administration, knowledge)
• Migrating the data model
  • Do it beforehand, choose data types carefully
• Migrating the data
  • Easy, but perhaps time consuming (depending on the size of the database)
• Migrating Application Code
  • The more PL/SQL code you have in your database, the more difficult a migration will be
• Do not forget the clients
• Look for features which are not available in PostgreSQL
  • A lot of them can be replaced by PostgreSQL functionality (e.g. Database links by FDWs)
Summary & Recommendations (2)

- Start with the "low hanging fruits", e.g. commercial applications which support PostgreSQL and Oracle or "simple databases" resp. "simple applications"

- It's not only DBA work
  - Involve application developers, system administrators, users (for testing ..)

- It's an iterative process
  - Don't expect a successful migration in the first attempt

- **Test, test, test ... your applications after the migration**

- Instead of migrating existing applications and databases to PostgreSQL it can be easier to start new applications on top of PostgreSQL and make experiences with the new database platform with new applications first
  - Enter the PostgreSQL world without the burden of an Oracle history
  - Oracle Compatibility is less important
Summary & Recommendations (3)

- Ora2pg is a very helpful Open Source for migrating Oracle (and MySQL) databases to PostgreSQL

- Can migrate "almost" everything

- Very flexible

- Can do simple conversions of PL/SQL-Code
Questions & Answers

Markus Flechtner
markus.flechtner@trivadis.com
Phone +49 211 5866 64725

@markusdba  www.markusdba.de
• Trivadis Training "**PostgreSQL for Oracle DBAs**"

<table>
<thead>
<tr>
<th>Date Range</th>
<th>City</th>
</tr>
</thead>
<tbody>
<tr>
<td>7. – 10. Oktober 2019</td>
<td>München</td>
</tr>
<tr>
<td>2. – 5. Dezember 2019</td>
<td>Düsseldorf</td>
</tr>
<tr>
<td>2. – 5. Dezember 2019</td>
<td>Wien</td>
</tr>
<tr>
<td>2. – 5. März 2020</td>
<td>Stuttgart</td>
</tr>
</tbody>
</table>

• More information: [https://www.trivadis-training.com/de/training/postgresql-fuer-oracle-dbas-o-pg4ora](https://www.trivadis-training.com/de/training/postgresql-fuer-oracle-dbas-o-pg4ora)
Making a **WORLD** possible in which **intelligent IT** facilitates **LIFE and WORK as a matter of COURSE.**