Oracle to PostgreSQL Migration: a hard way ?

PgConf.RU 2015
Moscow, Feb. 7

< gilles@darold.net >

About me

- Author : Gilles Darold
  - Works at Dalibo (http://www.dalibo.com/) as PostgreSQL consultant

- Author and maintainer of
  - Ora2Pg (http://ora2pg.darold.net)
  - PgBadger (http://dalibo.github.io/pgbadger/)
  - PgCluu (http://pgcluu.darold.net)
  - PgFormatter (http://sqlformat.darold.net)
  - … and more (http://www.darold.net)

About Ora2Pg

- Ora2Pg, first release on May 2001 (last version: 15.1)
  - 14 years of development !
  - Near 10,000 lines of Perl code
  - What users say about Ora2Pg?
    - « Terrific program! »
    - « You save my life! »
    - « Invaluable! »
- Where are we now ?
  - Hundred of Oracle database migration
  - Industrial deployment of Ora2Pg
    - When one database is migrated others follow
    - Some others can not because of editor's locks
  - Ask PostgreSQL support to software editors !

2015 – What Ora2Pg can do ?

- Automatic Oracle database discovery
- Automatic creation of migration projects
- Oracle database migration cost assessment
- Automatic database schema export
- Full and automatic data export
- Automatic conversion of PL/SQL to PLPGSQL
- Oracle Spatial to PostGis export

Automatic discovery

- Set the Oracle connection DSN
  - ora2pg -u system -w manager -t SHOW_VERSION --source « dbi:Oracle:host=localhost;sid=testdb »
- Set the configuration file /etc/ora2pg/ora2pg.conf
  - ORACLE_DSN   dbi:Oracle:host=localhost;sid=testdb
  - ORACLE_USER   system
  - ORACLE_PWD   manager
- Look for schema to export and set it into configuration file:
  - ora2pg -c /etc/ora2pg/ora2pg.conf -t SHOW_SCHEMA
  - SCHEMA   HR
- Lookup database tables and columns:
  - ora2pg -c /etc/ora2pg/ora2pg.conf -t SHOW_TABLE
  - ora2pg -c /etc/ora2pg/ora2pg.conf -t SHOW_COLUMN

Create a migration project

```
ora2pg --init_project my_db_mig  --project_base /full/path/to/project
```

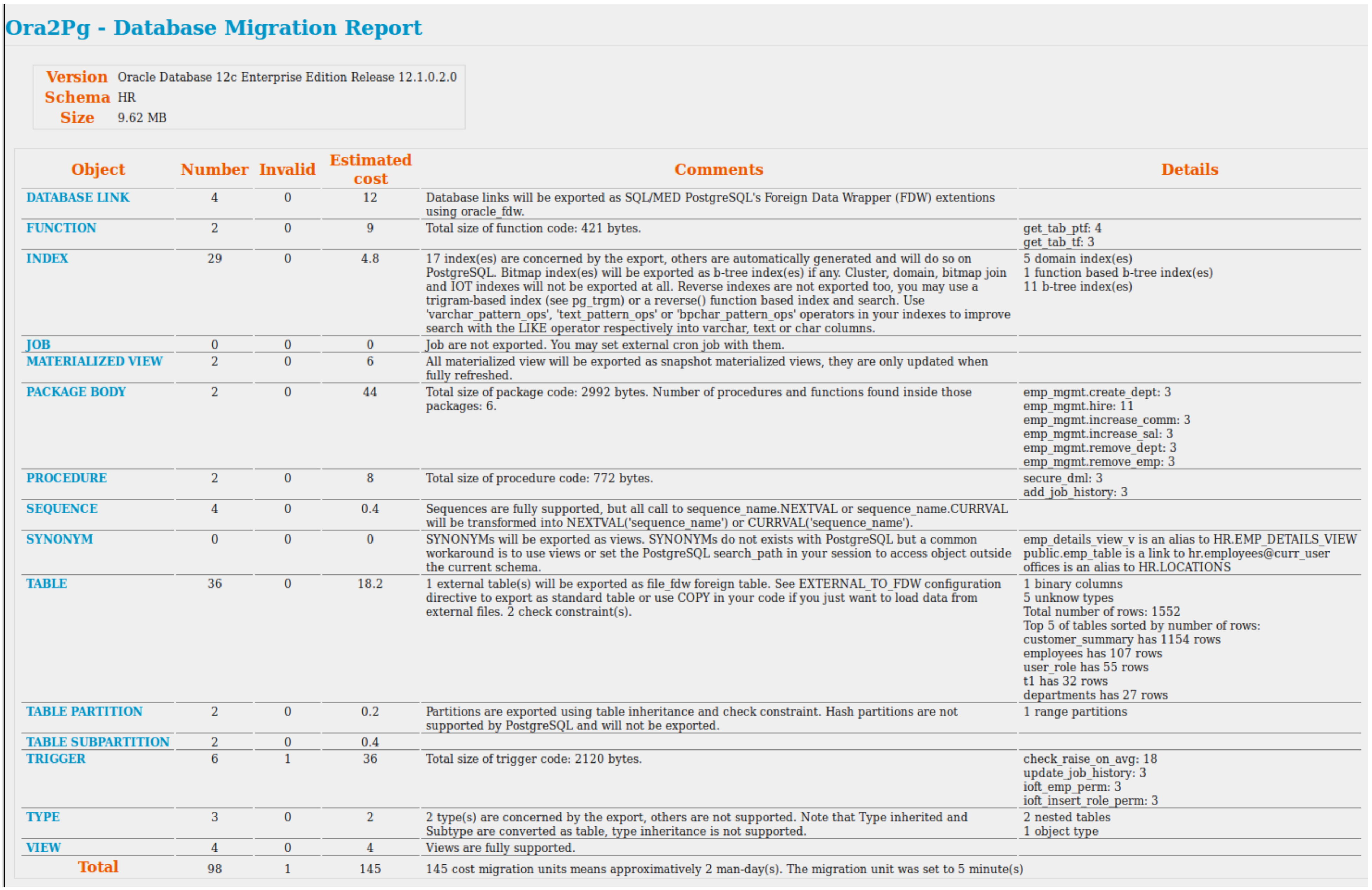
```
/full/path/to/project/my_db_mig/
        ├── config/
        │   └── ora2pg.conf
        ├── data/
        ├── export_schema.sh
        ├── reports/
        ├── schema/
        │   ├── dblinks/  functions/  grants/  mviews/  packages/
        │   ├── partitions/  procedures/  sequences/  synonyms/
        │   └── tables/  tablespaces/ directories/  triggers/  types/  views/
        └── sources/
            ├── functions/  mviews/  packages/  partitions/
            └── procedures/  triggers/  types/  views/
```

Migration assessment

- What database might be migrated first ?
  - Don't choose the Oracle Application database, you will fail !
  - Choose the smallest with few PL/SQL to learn Ora2Pg usage
  - Then choose the most representative, you need to forge your experience

- But how much human-days this work will cost me?
  - Buy an expensive audit
  - Use Ora2Pg migration assessment report

```
ora2pg -c /etc/ora2pg.conf -t SHOW_REPORT --estimate_cost
--dump_as_html > report.html
```



Ora2Pg - Database Migration Report

Version	Oracle Database 12c Enterprise Edition Release 12.1.0.2.0
Schema	HR
Size	9.62 MB

Object	Number	Invalid	Estimated cost	Comments	Details
DATABASE LINK	4	0	12	Database links will be exported as SQL/MED PostgreSQL's Foreign Data Wrapper (FDW) extentions using oracle_fdw.	
FUNCTION	2	0	9	Total size of function code: 421 bytes.	get_tab_ptf: 4 get_tab_tf: 3
INDEX	29	0	4.8	17 index(es) are concerned by the export, others are automatically generated and will do so on PostgreSQL. Bitmap index(es) will be exported as b-tree index(es) if any. Cluster, domain, bitmap join and IOT indexes will not be exported at all. Reverse indexes are not exported too, you may use a trigram-based index (see pg_trgm) or a reverse() function based index and search. Use 'varchar_pattern_ops', 'text_pattern_ops' or 'bpchar_pattern_ops' operators in your indexes to improve search with the LIKE operator respectively into varchar, text or char columns.	5 domain index(es) 1 function based b-tree index(es) 11 b-tree index(es)
JOB	0	0	0	Job are not exported. You may set external cron job with them.	
MATERIALIZED VIEW	2	0	6	All materialized view will be exported as snapshot materialized views, they are only updated when fully refreshed.	
PACKAGE BODY	2	0	44	Total size of package code: 2992 bytes. Number of procedures and functions found inside those packages: 6.	emp_mgmt.create_dept: 3 emp_mgmt.hire: 11 emp_mgmt.increase_comm: 3 emp_mgmt.increase_sal: 3 emp_mgmt.remove_dept: 3 emp_mgmt.remove_emp: 3
PROCEDURE	2	0	8	Total size of procedure code: 772 bytes.	secure_dml: 3 add_job_history: 3
SEQUENCE	4	0	0.4	Sequences are fully supported, but all call to sequence_name.NEXTVAL or sequence_name.CURRVAL will be transformed into NEXTVAL('sequence_name') or CURRVAL('sequence_name').	
SYNONYM	0	0	0	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 session to access object outside the current schema.	emp_details_view_v is an alias to HR.EMP_DETAILS_VIEW public.emp_table is a link to hr.employees@curr_user offices is an alias to HR.LOCATIONS
TABLE	36	0	18.2	1 external table(s) will be exported as file_fdw foreign table. See EXTERNAL_TO_FDW configuration directive to export as standard table or use COPY in your code if you just want to load data from external files. 2 check constraint(s).	1 binary columns 5 unknow types Total number of rows: 1552 Top 5 of tables sorted by number of rows: customer_summary has 1154 rows employees has 107 rows user_role has 55 rows t1 has 32 rows departments has 27 rows
TABLE PARTITION	2	0	0.2	Partitions are exported using table inheritance and check constraint. Hash partitions are not supported by PostgreSQL and will not be exported.	1 range partitions
TABLE SUBPARTITION	2	0	0.4		
TRIGGER	6	1	36	Total size of trigger code: 2120 bytes.	check_raise_on_avg: 18 update_job_history: 3 ioft_emp_perm: 3 ioft_insert_role_perm: 3
TYPE	3	0	2	2 type(s) are concerned by the export, others are not supported. Note that Type inherited and Subtype are converted as table, type inheritance is not supported.	2 nested tables 1 object type
VIEW	4	0	4	Views are fully supported.	
Total	98	1	145	145 cost migration units means approximatively 2 man-day(s). The migration unit was set to 5 minute(s)	

Schema migration

- Almost everything is exported :
  - table, constraint, index, sequence, trigger, view, tablespace, grant, type, partition
  - procedure, function, package, synonym, database link, materialized view, ...
- but some are not exported and need adaptation :
  - IOT / Cluster indexes can be replaced by « CLUSTER table_name USING index_name ».
  - Bitmap indexes are internally build by PostgreSQL when needed.
  - Reverse indexes can be replaced by a trigram-based index (see pg_trgm) or a reverse() function based index and search.
  - Type inheritance and type with member method are not supported
  - Global indexes over partitions are not supported
  - Global Temporary Table does not exists
  - Virtual Columns does not exists, use view instead
  - Compound triggers are not supported

DATA migration

- Can you migrate Big data ?
  - Tera bytes of data and billions of rows in tables takes hours
  - Purge or archive unused or rarely used data
  - Import live data first, open to production then import remaining data
- The Oracle and PostgreSQL database must be responsive
  - Parallel table export (-P ncores)
  - Multiple process to fill PostgreSQL tables (-j ncores)
  - Multiprocess to extract data from Oracle (-J ncores)
  - Both ? (-J ncores x -j ncores)
- Simple table (only columns with numbers) : +1 millions rows / second
- Complex table (lot of CLOB and/or BLOB) : 100 rows / second
- Always use COPY data export mode, INSERT is too slow

What's new

- Version 15.0 Ora2Pg has cool new features:
  - Autonomous transaction
  - Database Link
  - External table
  - BFILE
  - DIRECTORY
  - SYNONYM
  - More Spatial support

Autonomous transactions

- Autonomous transactions are not natively supported by PostgreSQL.
- Ora2Pg use a wrapper function to call the function through DBLINK
  - The original function is renamed with suffix '_atx'
  - The wrapper function take the name of the original function
- Waiting for **pg_background**
  - run commands in a background worker, and get the results.
  - Work in progress by Robert Haas - EnterpriseDB

Autonomous transaction

```
CREATE OR REPLACE FUNCTION log_action (msg text)  RETURNS VOID AS
$body$
DECLARE
        -- Change this to reflect the dblink connection string
        v_conn_str  text := 'port=5432 dbname=testdb host=localhost user=pguser
password=pgpass';
        v_query     text;
BEGIN
        v_query := 'SELECT true FROM log_action_atx ( ' || quote_literal(msg) || ' )';
        PERFORM * FROM dblink(v_conn_str, v_query) AS p (ret boolean);
END;
$body$
LANGUAGE plpgsql STRICT SECURITY DEFINER;
```

DATABASE LINK

- Access objects on a remote database
  - CREATE PUBLIC DATABASE LINK remote_service USING 'remote_db';
  - SELECT * FROM employees@remote_service;
- Ora2Pg will export it as Foreign Data Wrapper using **oracle_fdw**
  - 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');
- Remote tables need to be created as FDW tables:
  - ora2pg -c ora2pg.conf -t FDW -a EMPLOYEES
  - CREATE FOREIGN TABLE employees_fdw (…) SERVER remote_service OPTIONS(schema 'HR', table 'EMPLOYEES');

EXTERNAL TABLES

- Oracle EXTERNAL TABLE does not exists internally into PostgreSQL
  - CREATE OR REPLACE DIRECTORY ext_dir AS '/data/ext/';
  - CREATE TABLE ext_table (id NUMBER, …) 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
```

- Ora2Pg will export them as remote tables using extension **file_fdw** :

```
CREATE FOREIGN TABLE ext_tab (
        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 ',');
```

BFILE

- The BFILE data type stores unstructured binary data in flat files outside the database.
- A BFILE column stores a file locator that points to an external file containing the data: (DIRECTORY, FILENAME)
- By default Ora2Pg will transform it as bytea by loading file content :
  - CREATE TABLE bfile_test (id bigint, bfilecol bytea);

    ```
    COPY bfile_test (id,bfilecol) FROM STDIN;
    1
    1234,ALBERT,GRANT,21\\0121235,ALFRED,BLUEOS,26\\0121236,BERNY,JOLYSE,34\\012
    \.
    ```
- DATA_TYPE = BFILE:TEXT, only the path is exported : '/data/ext/file_ext.csv'
- DATA_TYPE = BFILE:EFILE, will use the **external_file** extension
  - https://github.com/darold/external_file

DIRECTORY

- DIRECTORY can be exported to be used with the **external_file** extension.

(https://github.com/darold/external_file)

```
INSERT INTO external_file.directories (directory_name, directory_path)
VALUES ('EXT_DIR', '/data/ext/');

INSERT INTO external_file.directory_roles (directory_name, directory_role,
directory_read, directory_write) VALUES ('EXT_DIR', 'hr', true, false);


INSERT INTO external_file.directories (directory_name, directory_path)
VALUES ('SCOTT_DIR', '/usr/home/scott/');

INSERT INTO external_file.directory_roles(directory_name, directory_role,
directory_read, directory_write) VALUES ('SCOTT_DIR', 'hr', true, true);
```

SYNONYM

- A synonym is an alias name for objects. They are used to grant access to an object from another schema or a remote database.
  - CREATE SYNONYM synonym_name FOR object_name [@ dblink];

- SYNONYMs doesn't exists in PostgreSQL
  - SET search_path TO other_schema,...
  - Ora2Pg will export them as VIEWS :

```
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;
```

With DBLINK, you have to create a foreign table HR.EMPLOYEES using a foreign server (Ora2Pg will warn you to see DBLINK and FDW export type).

ROWNUM

- Oracle : SELECT * FROM table WHERE ROWNUM <= 10
- PostgreSQL : SELECT * FROM table LIMIT 10
- Take care to the result, Oracle's sort ORDER BY is done after ROWNUM !!! To have the same behavior than LIMIT
  - SELECT * FROM (SELECT * FROM A ORDER BY id) WHERE ROWNUM <= 10;
- Ora2Pg replace automatically ending ROWNUM with LIMIT :
  - ROWNUM = N rewritten as LIMIT 1 OFFSET N
  - ROWNUM < or <= N rewritten as LIMIT N
  - ROWNUM > or >= N rewritten as LIMIT ALL OFFSET N
- ROWNUM to enumerate rows, not covered by Ora2Pg
  - Need to be rewritten as window function

Empty string vs NULL

- A zero length string is NULL in Oracle:
  - '' = NULL
- PostgreSQL and SQL standard:
  - '' <> NULL

- Constraint violation on Oracle but not in PostgreSQL

```
CREATE TABLE tempt (
        id NUMBER NOT NULL,
        descr VARCHAR2(255) NOT NULL
) ;
INSERT INTO temp_table (id, descr) VALUES (2, '');
ORA-01400: cannot insert NULL into ("HR"."TEMPT"."DESCR")
```

Empty string vs NULL

- By default Ora2Pg replace all conditions with a test on NULL by a call to the coalesce() function.
  - (field1 IS NULL) is replaced by (coalesce(field1::text, '') = '')
  - (field2 IS NOT NULL) is replaced by (field2 IS NOT NULL AND field2::text <> '')

- Default is replacement to be sure that your application will have the same behavior.
- You can not insert an empty string into a numeric so the replacement is no necessary.
- Set NULL_EQUAL_EMPTY to 0 to disable this automatic replacement.

PL/SQL to PLPGSL

- All triggers, functions, procedures and packages are exported and converted to PLPGSQL by Ora2Pg.
  - This will really save your life !
- But some parts are not :
  - Global variables in packages, use dedicated tables instead
  - Anonymous/initialization block in package, use an init function with this code
  - Function created inside an other one, drop the code into a normal function
- Oracle specific code always need to be rewritten :
  - External modules (DBMS, UTL, ...)
  - CONNECT BY (use CTE « WITH RECURSIVE »)
  - OUTER JOIN (+)
  - DECODE (Ora2Pg can only transform simple forms)

Oracle DBMS modules

- Some are implemented in orafce library

  (https://github.com/orafce/orafce)
  - DBMS_OUTPUT
  - UTL_FILE
  - DBMS_PIPE
  - DBMS_ALERT
- Some advanced functionalities are implemented in external PostgreSQL tools, contribs or extensions:
  - Oracle Advanced Queuing => see PGQ from Skytools
  - Oracle Jobs scheduler => see pgAgent / JobScheduler
- Others can easily be rewritten in extended language like Perl.
  - You used to send email from your Oracle database using UTL_SMTP ?

Example UTIL_SMTP

```
CREATE OR REPLACE FUNCTION send_email(name,inet, text, text, text) RETURNS integer AS
$body$
        use Net::SMTP;
        my ($Db, $Ip, $sendTo, $Subject, $Message) = @_;
        my $smtp = Net::SMTP->new("mailhost", Timeout => 60);
        $smtp->mail("$Db\@$Ip");
        $smtp->recipient($sendTo);
        $smtp->data();
        $smtp->datasend("To: $sendTo\n");
        $smtp->datasend("Subject: $Subject\n");
        $smtp->datasend("Content-Type: text/plain;\n\n");
        $smtp->datasend("$Message\n");
        $smtp->dataend();
        $smtp->quit();
        return 1;
$body$ language 'plperlu';

SELECT send_email(current_database(), inet_server_addr(), 'dba@dom.com', 'test pg_utl_smtp', 'This is a test');
```

Oracle OUTER JOIN (+)

- LEFT OUTER JOIN
  - SELECT * FROM a, b WHERE a.id = b.id (+)
  - SELECT * FROM a LEFT OUTER JOIN b ON (id)
- RIGHT OUTER JOIN
  - SELECT * FROM a, b, c WHERE a.id = b.id (+) AND a.id (+) = c.id
  - SELECT * FROM a LEFT OUTER JOIN b ON (a. id = b.id) RIGHT OUTER JOIN c ON (a.id = c.id)
- FULL OUTER JOIN
  - SELECT * FROM a, b WHERE a.id = b.id (+) UNION ALL SELECT * FROM a, b WHERE a.id (+) = b.id AND a.id = NULL
  - SELECT * FROM a FULL OUTER JOIN b ON (a.id = b.id)

Conversion of (+) to ANSI Joins

- Your PL/SQL code if filled of queries like that?
- Your developers still use (+) notation?
- How can you automatically convert this code to ANSI-compliant joins syntax?
  - Ora2Pg is not able to convert this code, at least not now.
- Please help!!!
  - First stop to produce code with (+) notation it is recommended by Oracle itself since Oracle 9i.

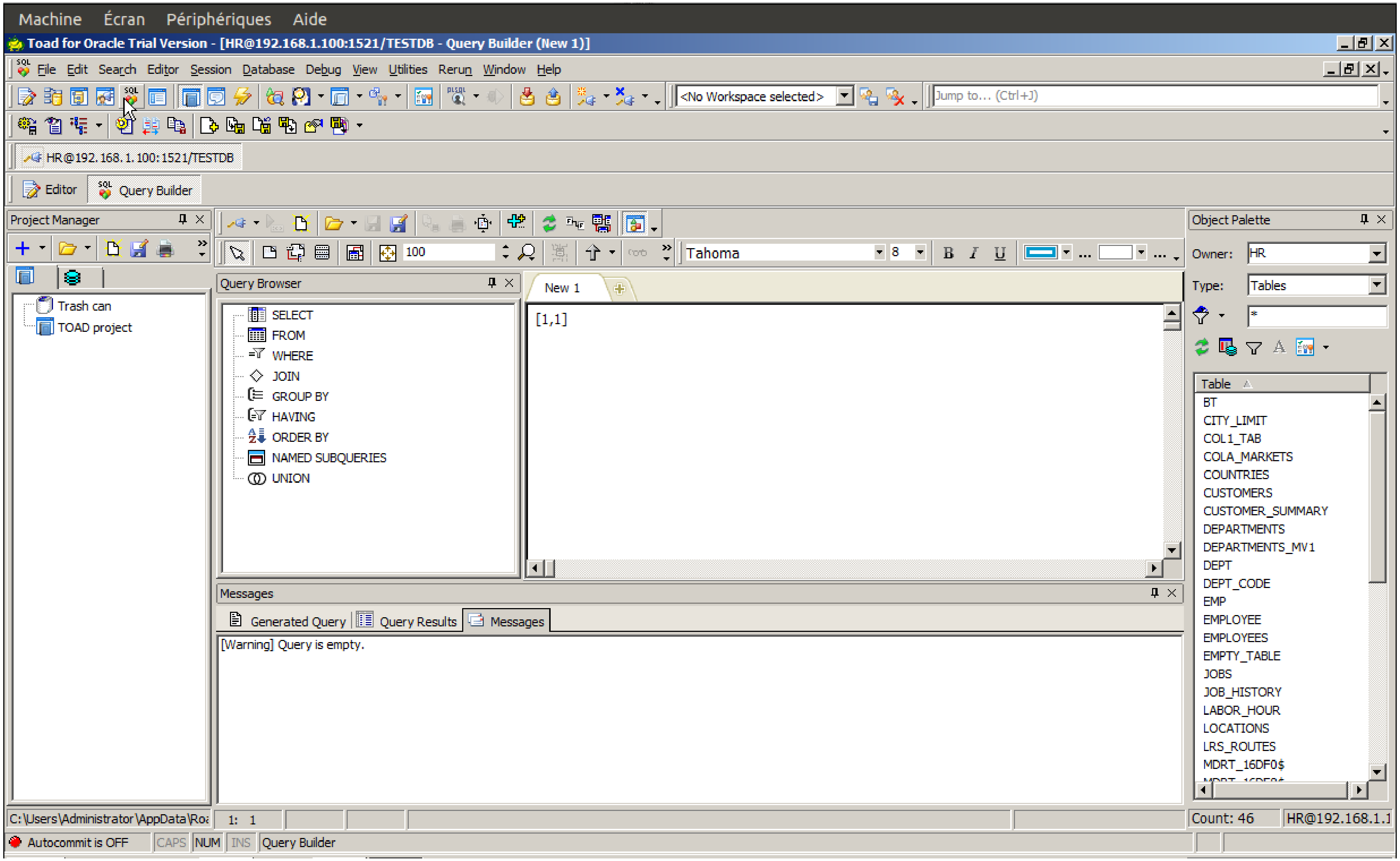
Automatic conversion of (+)

- I can't migrate without automation, it will takes months!

Ok, keep calm, Toad is your friend !

Does Oracle SQL Developer too ?

Open the TOAD Query Builder

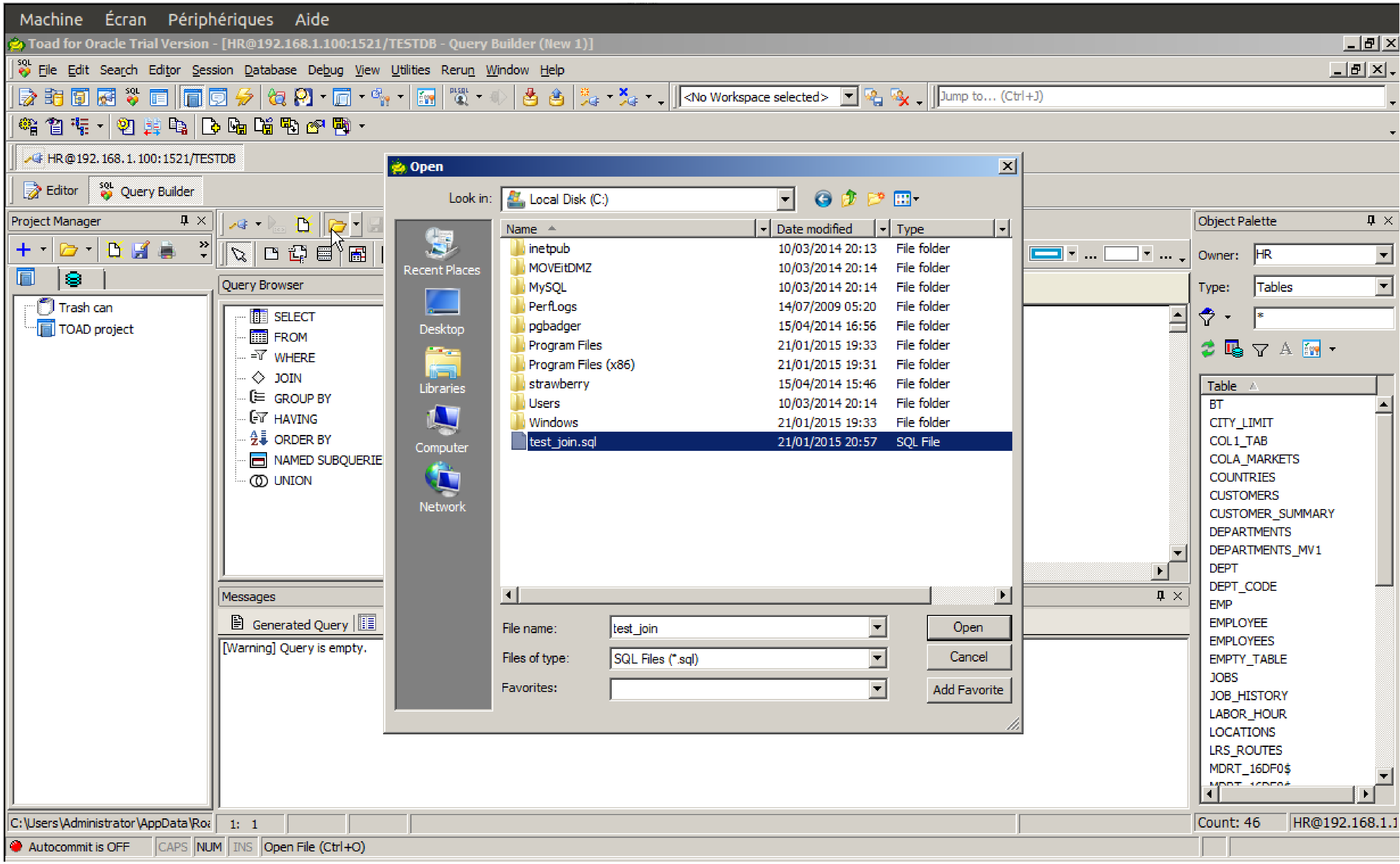


[1,1]

[Warning] Query is empty.

BT
CITY_LIMIT
COL1_TAB
COLA_MARKETS
COUNTRIES
CUSTOMERS
CUSTOMER_SUMMARY
DEPARTMENTS
DEPARTMENTS_MV1
DEPT
DEPT_CODE
EMP
EMPLOYEE
EMPLOYEES
EMPTY_TABLE
JOBS
JOB_HISTORY
LABOR_HOUR
LOCATIONS
LRS_ROUTES
MDRT_16DF0$

then load your SQL code


Machine Écran Périphériques Aide

Toad for Oracle Trial Version - [HR@192.168.1.100:1521/TESTDB - Query Builder (New 1)]

File Edit Search Editor Session Database Debug View Utilities Rerun Window Help

<No Workspace selected> Jump to... (Ctrl+J)

HR@192.168.1.100:1521/TESTDB

Editor | Query Builder

Project Manager: Trash can, TOAD project

Query Browser: SELECT, FROM, WHERE, JOIN, GROUP BY, HAVING, ORDER BY, NAMED SUBQUERIE, UNION

Messages — Generated Query

[Warning] Query is empty.

Open

Look in: Local Disk (C:)

Recent Places, Desktop, Libraries, Computer, Network

Name	Date modified	Type
inetpub	10/03/2014 20:13	File folder
MOVEitDMZ	10/03/2014 20:14	File folder
MySQL	10/03/2014 20:14	File folder
PerfLogs	14/07/2009 05:20	File folder
pgbadger	15/04/2014 16:56	File folder
Program Files	21/01/2015 19:33	File folder
Program Files (x86)	21/01/2015 19:31	File folder
strawberry	15/04/2014 15:46	File folder
Users	10/03/2014 20:14	File folder
Windows	21/01/2015 19:33	File folder
test_join.sql	21/01/2015 20:57	SQL File

File name: test_join

Files of type: SQL Files (*.sql)

Favorites:

Open | Cancel | Add Favorite

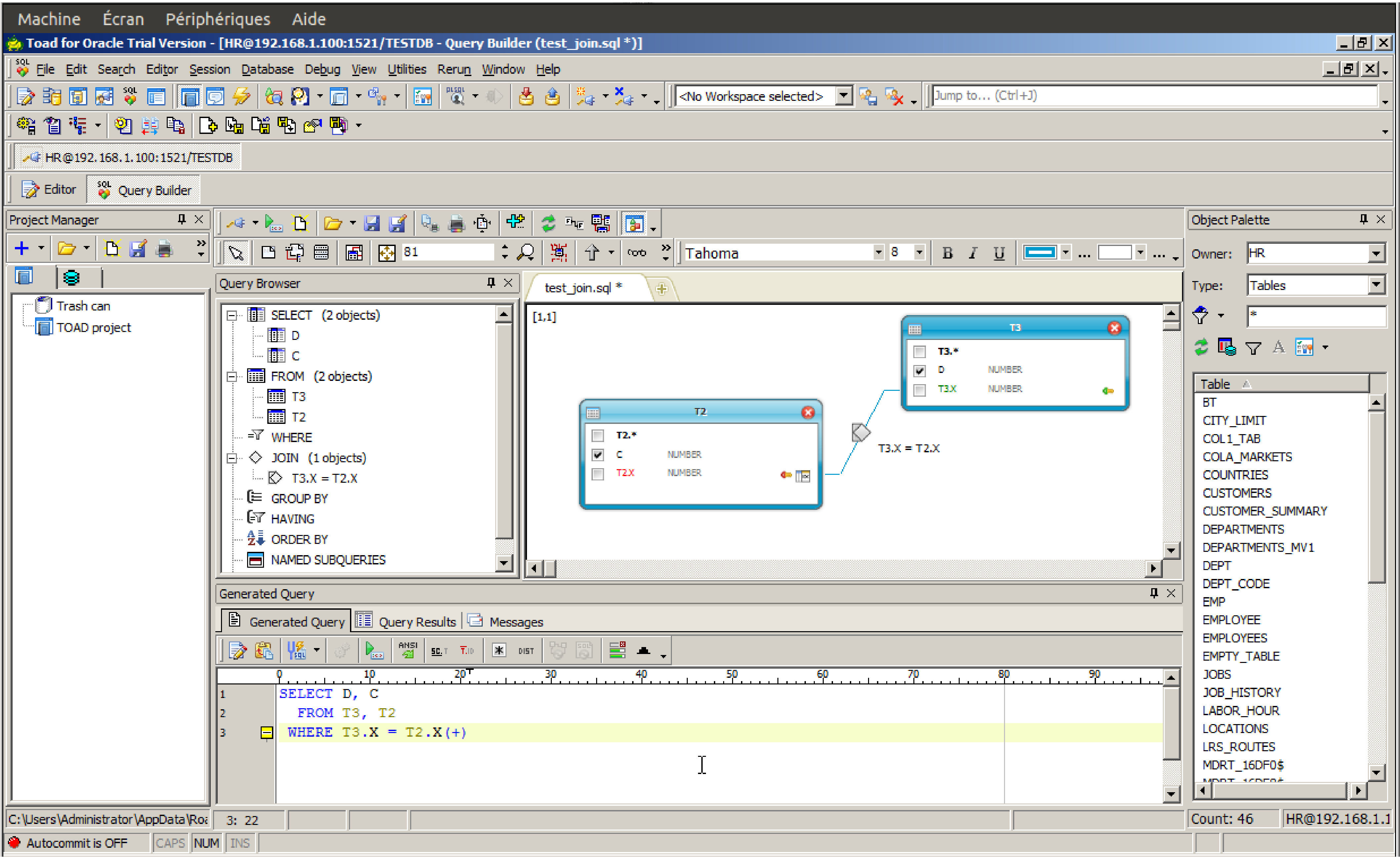
Object Palette — Owner: HR — Type: Tables — *

Table
BT
CITY_LIMIT
COL1_TAB
COLA_MARKETS
COUNTRIES
CUSTOMERS
CUSTOMER_SUMMARY
DEPARTMENTS
DEPARTMENTS_MV1
DEPT
DEPT_CODE
EMP
EMPLOYEE
EMPLOYEES
EMPTY_TABLE
JOBS
JOB_HISTORY
LABOR_HOUR
LOCATIONS
LRS_ROUTES
MDRT_16DF0$

C:\Users\Administrator\AppData\Roa 1: 1 Count: 46 HR@192.168.1.1

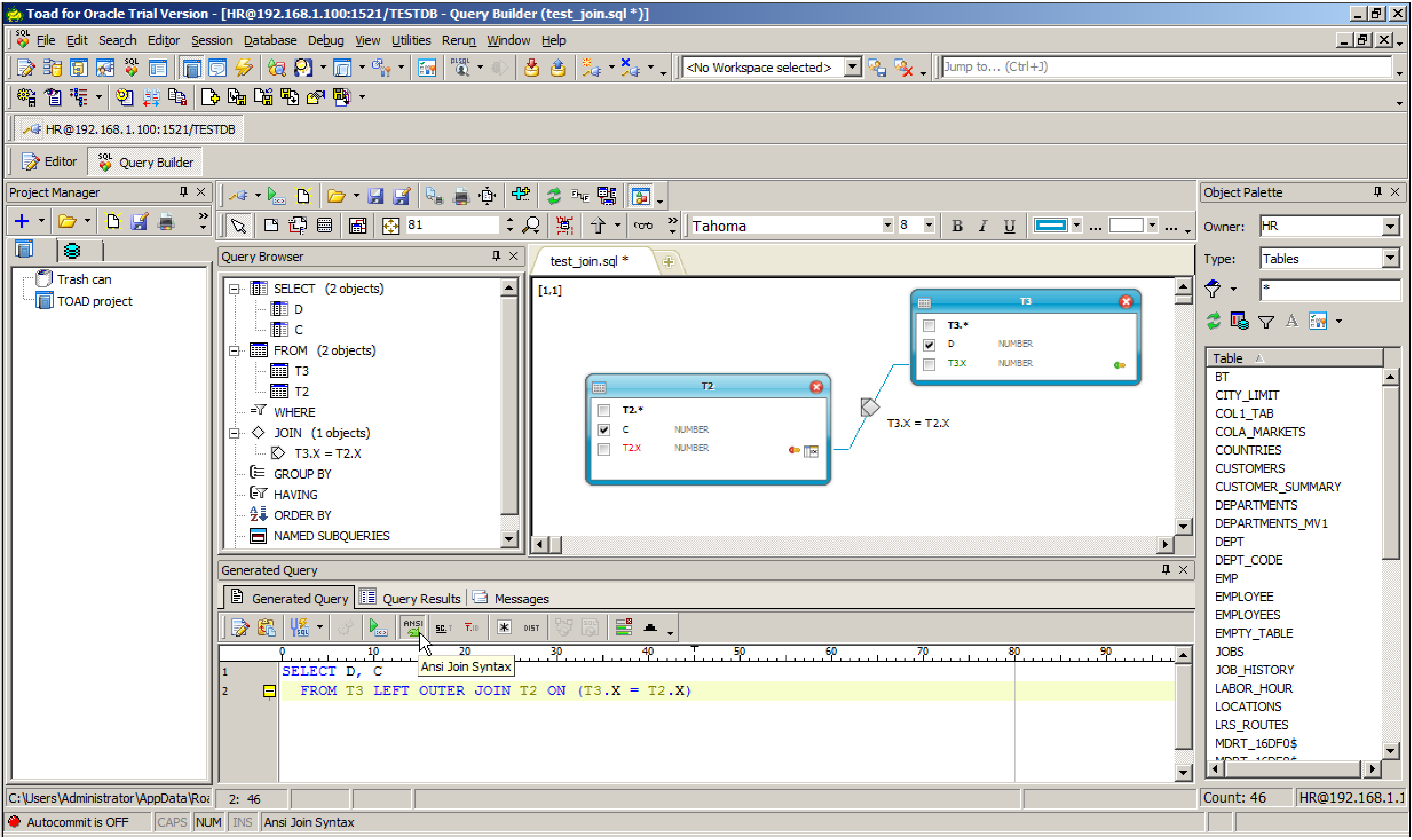
Autocommit is OFF CAPS NUM INS Open File (Ctrl+O)

Oracle outer join syntax



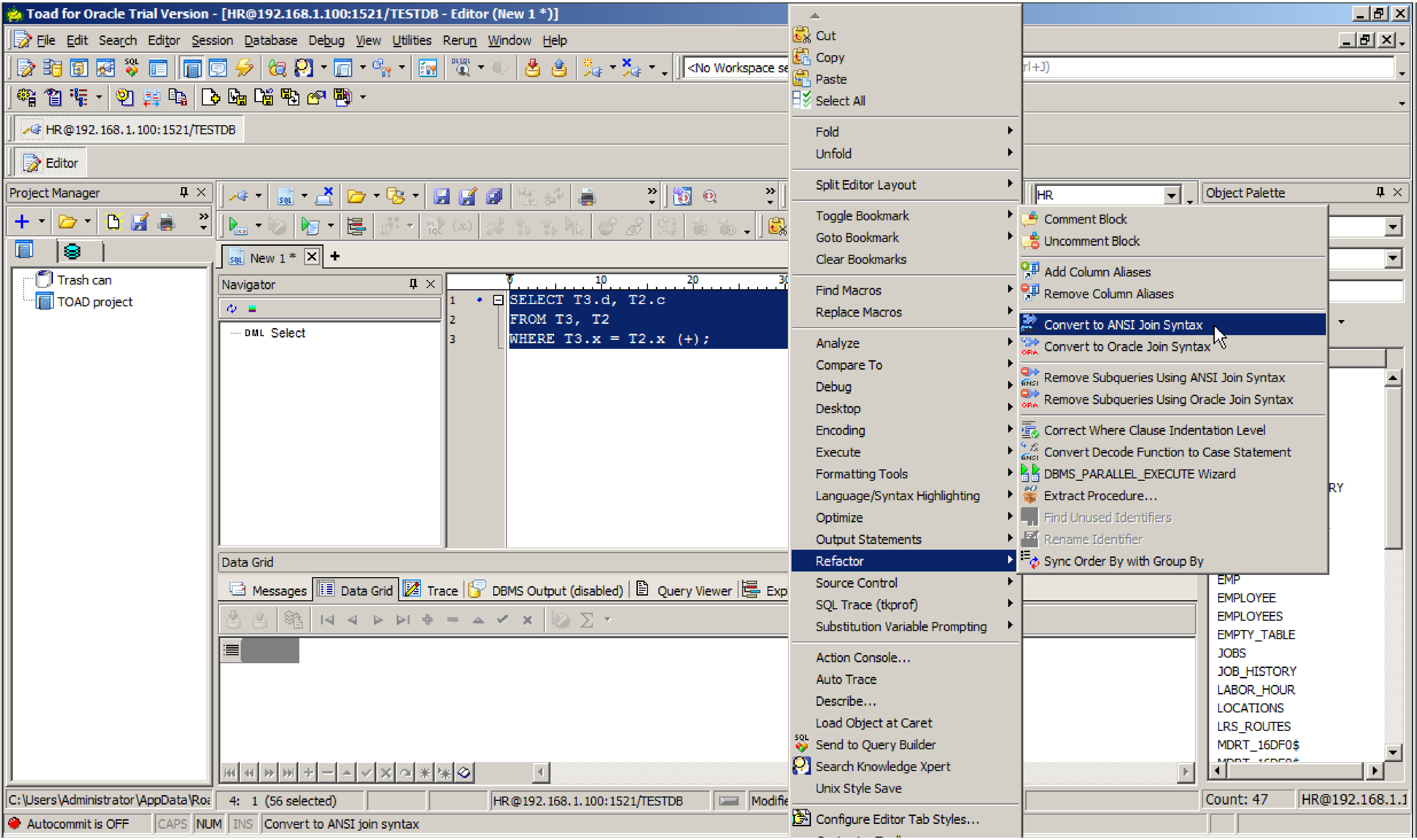
```
SELECT D, C
  FROM T3, T2
 WHERE T3.X = T2.X(+)
```

and the ANSI-compliant Join



```
SELECT D, C
  FROM T3 LEFT OUTER JOIN T2 ON (T3.X = T2.X)
```

Refactor → Convert to ANSI Join Syntax

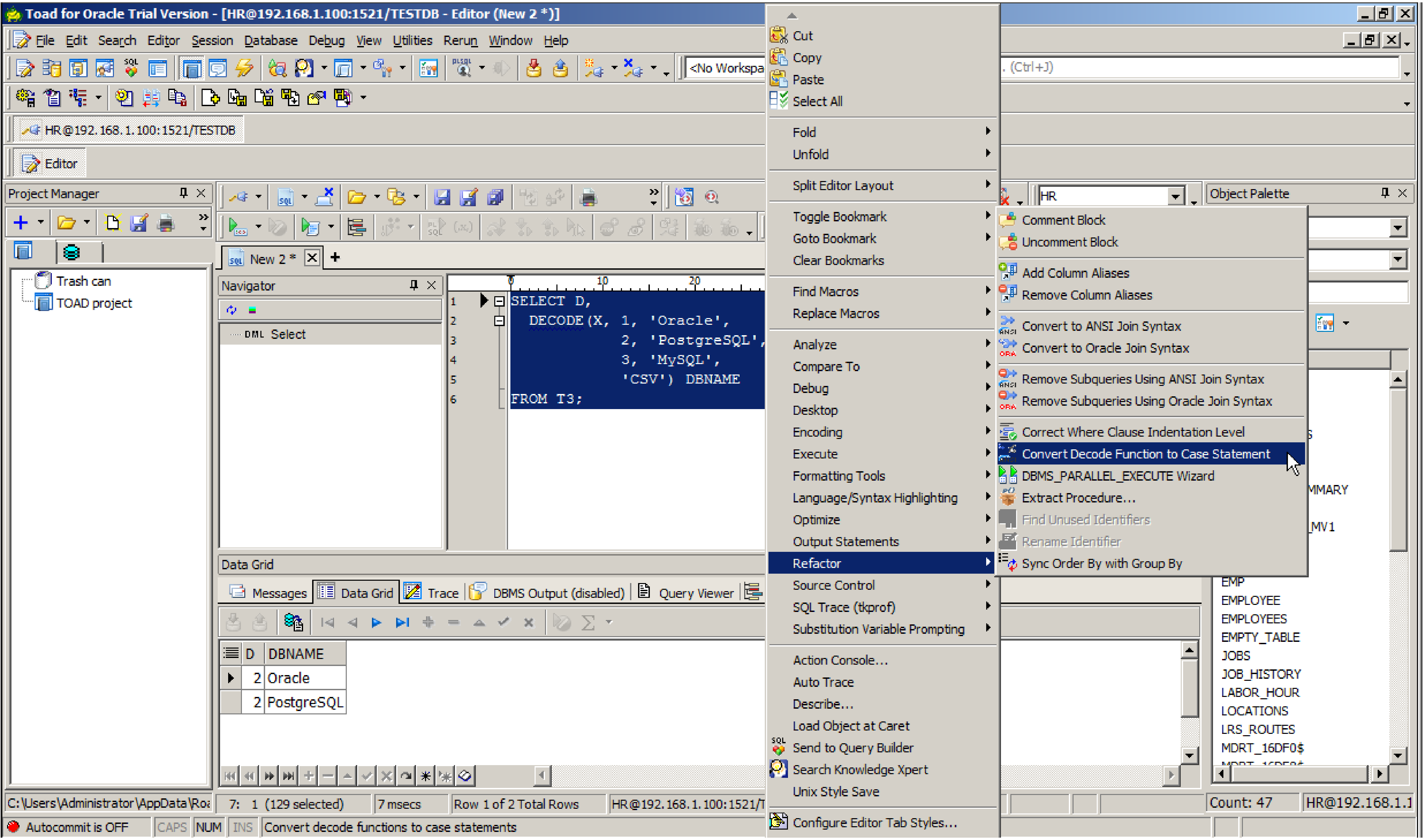


```
SELECT T3.d, T2.c
FROM T3, T2
WHERE T3.x = T2.x (+);
```

DECODE

- This is an Oracle specific function :
  - DECODE (expression, search, result [, search, result]... [, default])
  - CASE WHEN expr = search THEN result ... ELSE default END
- You have tons of functions and queries using it!
  - Use SQL standard CASE clause or why not the Orafce decode() function
- My developers still use it!
  - Oracle recommend the use of CASE since 9i
- Please help!!!
  - Ora2Pg can only replace simple form of the function up to 10 parameters
  - But remember your friend, TOAD !

Refactor → Convert Decode to Case



Toad for Oracle Trial Version - [HR@192.168.1.100:1521/TESTDB - Editor (New 2 *)]

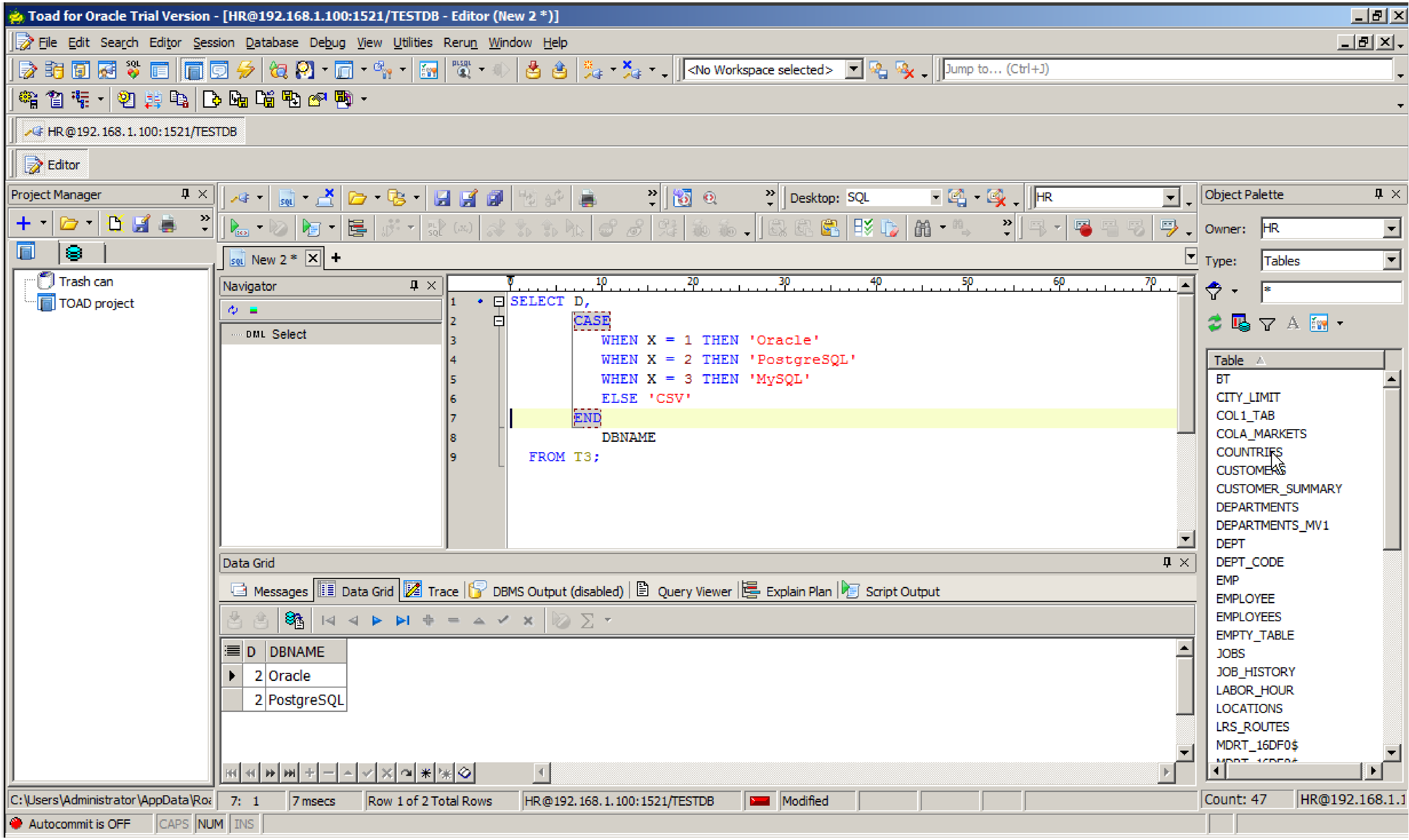
```
SELECT D,
  DECODE(X, 1, 'Oracle',
            2, 'PostgreSQL',
            3, 'MySQL',
            'CSV') DBNAME
FROM T3;
```

D	DBNAME
2	Oracle
2	PostgreSQL

7: 1 (129 selected) | 7 msecs | Row 1 of 2 Total Rows

Autocommit is OFF | Convert decode functions to case statements

Decode converted to Case



Toad for Oracle Trial Version - [HR@192.168.1.100:1521/TESTDB - Editor (New 2 *)]
File Edit Search Editor Session Database Debug View Utilities Rerun Window Help
Project Manager
Trash can
TOAD project
Navigator
Select
Desktop: SQL
HR
Object Palette
Owner: HR
Type: Tables

```
SELECT D,
       CASE
         WHEN X = 1 THEN 'Oracle'
         WHEN X = 2 THEN 'PostgreSQL'
         WHEN X = 3 THEN 'MySQL'
         ELSE 'CSV'
       END
         DBNAME
  FROM T3;
```

D	DBNAME
2	Oracle
2	PostgreSQL

```
7: 1   7 msecs   Row 1 of 2 Total Rows   HR@192.168.1.100:1521/TESTDB   Modified
Autocommit is OFF
Count: 47
```

Oracle Spatial/Locator type

```
CREATE TABLE cola_markets (
        mkt_id NUMBER PRIMARY KEY,
        name VARCHAR2(32),
        shape **SDO_GEOMETRY**
    );
```

Type SDO_GEOMETRY:

```
        SDO_GEOMETRY(
          2001,  – Indicates the type of the geometry, here a point
          NULL, -- Identify a coordinate system (SRID: spatial reference system)
          NULL, -- SDO_POINT attributes X, Y, and Z, all of type NUMBER
          SDO_ELEM_INFO_ARRAY(1,1,1), --  Element informations array
          SDO_ORDINATE_ARRAY(10, 5) -- Coordinates Array
         )
```

PostGis Spatial type

- Corresponding type in PostGis : GEOMETRY

```
CREATE TABLE cola_markets (
        mkt_id bigint PRIMARY KEY,
        name varchar(32),
        shape geometry(GEOMETRY)
);
```

- Type GEOMETRY :
  - WKT (Well-Know Text)
    - Ex: 'LINESTRING(0 0, 1 1, 2 1, 2 2)'
  - WKB (Well-Know Binary)
    - Ex : 0101000020040000000000000000000000000000...

Geometry Constraints

- With PostGis you can enforce the type of spatial object that must be used :

```
CREATE TABLE stores (
    id  integer,
    gps_position geometry(POINT),
    sale_area geometry(POLYGONZ)
);
```

- 3D objects are signified with suffix Z and 4D using ZM :
  - GEOMETRY / GEOMETRYZ / GEOMETRYZM
  - POINT / POINTZ / POINTZM
  - POLYGON / POLYGONZ / POLYGONZM

Default geometry

- You can mixed several geometry types (points / lines / polygons…) in the same column.
  - shape geometry(**GEOMETRY**)
  - shape geometry(**GEOMETRY**, 4326)
- This correspond to the generic use of the GEOMETRY type.
- This is the default type used by Ora2Pg.

SRID

- SRID : Spatial reference system
- Oracle "legacy" vs standard "EPSG"
  - CONVERT_SRID   1
- Conversion function : **map_oracle_srid_to_epsg()**
  - Returns often NULL
  - DEFAULT_SRID   4326
- To enforce the use of a particular SRID :
  - CONVERT_SRID   27572

Detecting geometry constraint

- Ora2Pg is able to detect the geometry type of a column by
  - Looking at the constrained type in parameters of spatial indexes
    - Ex : CREATE INDEX ... PARAMETERS ('**sdo_indx_dims=2, layer_gtype=line**');
  - Or using a sequential scan to search distinct geometry types
    - AUTODETECT_SPATIAL_TYPE 1
    - When only one geometry type is found, it is applied as constraint
- Sequential scan is only used when there's no constraint type defined.
- it need to be limited or the whole table will be scanned
  - SELECT DISTINCT c.SDO_GTYPE FROM MYTABLE c WHERE ROWNUM < ?;
    - AUTODETECT_SPATIAL_TYPE = 1 then ROWNUM=50000 by default
    - AUTODETECT_SPATIAL_TYPE > 1, ROWNUM=AUTODETECT_SPATIAL_TYPE

Inserting geometry : Oracle

A simple rectangle inserted into Oracle :

```
INSERT INTO cola_markets VALUES (
   302, 'Rectangle',
   SDO_GEOMETRY(
      2003,  -- 2D polygon
      8307,
      NULL,
      SDO_ELEM_INFO_ARRAY(1,1003,3), -- a rectangle
      SDO_ORDINATE_ARRAY(1,1, 5,7) -- 2 points define the rectangle
   )
);
```

```
INSERT INTO cola_markets VALUES (302, 'Rectangle', GeomFromText('POLYGON
((1.0 1.0, 5.0 1.0, 5.0 7.0, 1.0 7.0, 1.0 1.0))'));
```

Inserting geometry : PostGis

Same rectangle inserted into PostgreSQL using WKT :

```
INSERT INTO cola_markets (mkt_id,name,shape) VALUES (
        302,
        'rectangle',
        'POLYGON ((1.0 1.0, 5.0 1.0, 5.0 7.0, 1.0 7.0, 1.0 1.0))'
);
```

And WKB:

```
INSERT INTO cola_markets VALUES (302,'rectangle',
'01ea030000003000000000000000000000000000000000000000000000000
00000000000000000000f03f000000000000000000000000000000000000
000000000f03f000000000000f03f0000000000000040');
```

Spatial data export

- Ora2Pg first lookup for SRID by querying the ALL_SDO_GEOM_METADATA table.
- Then export data as EWKT, using COPY mode:

```
COPY cola_markets (mkt_id,name,shape) FROM STDIN;
301	polygon SRID=4326;POLYGON ((5.0 1.0, 8.0 1.0, 8.0 6.0, 5.0 7.0, 5.0 1.0))
\.
```

- Or when using INSERT mode:

```
INSERT INTO cola_markets (mkt_id,name,shape) VALUES (301,E'polygon',ST_GeomFromText('POLYGON ((5.0 1.0, 8.0 1.0, 8.0 6.0, 5.0 7.0, 5.0 1.0))',4326));
```

Spatial Indexes

Oracle spatial indexes

```
CREATE INDEX cola_spatial_idx
   ON cola_markets(shape)
   INDEXTYPE IS MDSYS.SPATIAL_INDEX;
```

PostgreSQL spatial index

```
CREATE INDEX cola_spatial_idx
    ON cola_markets USING gist(shape);
```

Supported Geometries

- 2D and 3D geometry are exported
- SDO_POINT
- UNKNOWN_GEOMETRY
- POINT
- POLYGON
- COLLECTION
- MULTIPOINT
- MULTILINE or MULTICURVE
- MULTIPOLYGON
- Unsupported: CIRCLE, RASTER

Spatial Function

Ora2Pg replace all call to SDO_* functions into PostGis ST_* functions in converted PL/SQL code

- SDO_GEOM.RELATE => ST_Relate
- SDO_GEOM.VALIDATE_GEOMETRY_WITH_CONTEXT => ST_IsValidReason
- SDO_GEOM.WITHIN_DISTANCE => ST_DWithin
- SDO_DISTANCE => ST_Distance
- SDO_BUFFER => ST_Buffer
- SDO_CENTROID => ST_Centroid
- SDO_UTIL.GETVERTICES => ST_DumpPoints
- SDO_TRANSLATE => ST_Translate
- SDO_SIMPLIFY => ST_Simplify
- SDO_AREA => ST_Area
- SDO_CONVEXHULL => ST_ConvexHull
- SDO_DIFFERENCE => ST_Difference
- SDO_INTERSECTION => ST_Intersection
- SDO_LENGTH => ST_Length
- SDO_POINTONSURFACE => ST_PointOnSurface
- SDO_UNION => ST_Union
- SDO_XOR => ST_SymDifference

The hidden part of the magic

- Aka, the todo list:
  - Use regexp only => need a real PL/SQL parser/lexer
    - Ora2Pg replace sometime SELECT by PERFORM wrongly
    - Replacement of complex form of code
  - Hash and multicolumn partitioning
  - Add a mechanism to handle global variables in packages
  - Allow user custom function to modify data on the fly
  - Allow incremental data migration
  - Embedded SQL code formatter
  - Parallelized creation of indexes and constraint
  - ...

Tools equivalence 1/3

- SQLPLUS: PSQL but much more
- TOAD / Oracle SQL Developper: TORA (http://torasql.com/) or pgAdmin
- EXPLAIN PLAN: EXPLAIN ANALYZE
- ANALYZE TABLE: ANALYZE
- Cold backup: both are file system backup
- Hot backup: REDOLOGS = ARCHIVELOGS
- Logical Export: exp = pg_dump
- Logical Import: imp = pg_restore or psql
- SQL Loader: pgLoader (http://pgloader.io/)
- RMAN: Barman (http://www.pgbarman.org/) or Pitrery (https://dalibo.github.io/pitrery/)
- AUDIT TRAIL: pgAudit (https://github.com/2ndQuadrant/pgaudit)

Tools equivalence 2/3

- Pooling / Dispatcher:
  - PgBouncer (http://pgfoundry.org/projects/pgbouncer)
  - PgPool (http://www.pgpool.net/)
- Active Data Guard:
  - PostgreSQL master / slave replication
  - Slony (http://slony.info/)
- Replication master / master:
  - PostgreSQL-XC (http://sourceforge.net/projects/postgres-xc/)
  - Bucardo (https://bucardo.org/)
- Logical replication:
  - PostgreSQL 9.5 / 10 ?
  - Slony
- Official binary packages for all these projects can be found at http://yum.postgresql.org or http://apt.postgresql.org

Tools equivalence 3/3

- RAC Horizontal scaling: PostgreSQL-XC – PostgreSQL-XL - plProxy, pg_shard
- Oracle => Postgres Plus Advanced Server
  - Same as PostgreSQL but with proprietary code and database feature compatibility for Oracle.
  - Compatible with applications written for Oracle.
  - No need to rewrite PL/SQL into PLPGSQL
  - Applications written for Oracle run on Postgres Plus Advanced Server without modification.
  - http://www.enterprisedb.com/
- This is not an exhaustive list of the existing tools, there's much more interesting projects.

Monitoring / Audit tools

- **PgBadger**: A fast PostgreSQL log analyzer
  - http://dalibo.github.io/pgbadger/
- **PgCluu**: PostgreSQL and system performances monitoring and auditing tool
  - http://pgcluu.darold.net/
- **Powa**: PostgreSQL Workload Analyzer. Gathers performance stats and provides real-time charts and graphs to help monitor and tune your PostgreSQL servers. Similar to Oracle AWR.
  - http://dalibo.github.io/powa/
- **PgObserver**: monitor performance metrics of different PostgreSQL clusters.
  - http://zalando.github.io/PGObserver/
- **OPM**: Open PostgreSQL Monitoring. Gather stats, display dashboards and send warnings when something goes wrong. Tend to be similar to Oracle Grid Control.
  - http://opm.io/
- **check_postgres**: script for monitoring various attributes of your database. It is designed to work with Nagios, MRTG, or in standalone scripts.
  - https://bucardo.org/wiki/Check_postgres
- **Pgwatch**: monitor PostgreSQL databases and provides a fast and efficient overview of what is really going on.
  - http://www.cybertec.at/en/products/pgwatch-cybertec-enterprise-postgresql-monitor/
- More tools at https://wiki.postgresql.org/wiki/Monitoring

What else ?

- Other OSS tool that can help to migrate
  - Pentaho Kettle
    - http://community.pentaho.com/projects/data-integration/
  - JTS Topology Suite for spatial data import
    - http://www.vividsolutions.com/jts/JTSHome.htm
  - oracle_fdw, with Oracle spatial support since 1.1.0
    - http://pgxn.org/dist/oracle_fdw/
  - Orafce, Oracle's compatibility functions and packages
    - http://pgxn.org/dist/orafce/

- Don't forget to migrate your SQL Server database too :-)
    - https://github.com/dalibo/sqlserver2pgsql

You are not alone !

Community support on Ora2Pg :

- Any PostgreSQL's forum can help
- Github for feature requests
- Github issues and bugs reports
  - https://github.com/darold/ora2pg
- Feedback / suggestion to < gilles@darold.net >

Buy professional help to migrate and commercial support :

- Any PostgreSQL company near from you listed in http://www.postgresql.org/support/professional_support/
- Support the community !

Acknowledgments

- DGFiP (French Public Finance Government) for the migration cost assessment sponsoring.
  - http://www.impots.gouv.fr/
- BRGM (French Geological and Mining Survey) for the Oracle Spatial to PostGis sponsoring.
  - http://www.brgm.eu/
- Very specials thanks to Dominique Legendre who help me a lot on Spatial understanding and testing Ora2Pg features. He is also the author of the external_file extension.
- Oslandia for Spatial to PostGis specification and for they works on oracle_fdw.
  - http://www.oslandia.com/index-en.html
- Dalibo who give me time to develop Ora2Pg and opportunities to work on Oracle to PostgreSQL migrations.
  - http://www.dalibo.com/
- And all great contributors to Ora2Pg!

Thanks for your attention

Question ?