Geographical Distance Calculation Using Oracle® Database

ZIPCodeWorld™ & PostalCodeWorld™ Database

This paper describes steps to implement distance calculation between postal codes. You may want to modify some names and parameters. Most appropriate for modification places are highlighted for your convenience. Script has been tested against version 9 and 10.

1) Create Oracle user (optional in case if you want to set up Oracle objects using existing user). Execute as system (any other DBA) user.

```
CREATE USER POST_CODE
IDENTIFIED BY POST_CODE
DEFAULT TABLESPACE USERS
TEMPORARY TABLESPACE "TEMP"
ACCOUNT UNLOCK;
GRANT "CONNECT" TO "POST_CODE";
GRANT "RESOURCE" TO "POST_CODE";
```

2) Create directories (make sure that Operation system Oracle user have read access on data dir and full access on log and bad dir). Execute as system (any other DBA) user.

```
CREATE OR REPLACE DIRECTORY post_code_data_dir AS 'C: \PostCode\data';
CREATE OR REPLACE DIRECTORY post_code_log_dir AS 'C:\ PostCode\log';
CREATE OR REPLACE DIRECTORY post_code_bad_dir AS 'C: \PostCode\bad';

GRANT READ ON DIRECTORY post_code_data_dir TO post_code;
GRANT WRITE ON DIRECTORY post_code_log_dir TO post_code;
GRANT WRITE ON DIRECTORY post_code_bad_dir TO post_code;
```

3) Create external table (Execute as **post_code user)**. We use PostalcodeWorld™ Premium Edition for example.

```
CREATE TABLE L_POST_CODE_EXT
(
      POST_CODE         VARCHAR2(7 BYTE),
      CITY              VARCHAR2(45 BYTE),
      PROVINCE_NAME     VARCHAR2(45 BYTE),
      PROVINCE          VARCHAR2(2 BYTE),
      AREA_CODE         VARCHAR2(3 BYTE),
      CITYTYPE          VARCHAR2(1 BYTE),
      TIME_ZONE         NUMBER,
      DAY_LIGHT_SAVING  VARCHAR2(1 BYTE),
      LATITUDE          NUMBER(12,6),
      LONGITUDE         NUMBER(12,6)
)
```

```
ORGANIZATION EXTERNAL
(     TYPE ORACLE_LOADER
      DEFAULT DIRECTORY POST_CODE_DATA_DIR
      ACCESS PARAMETERS
      (     records delimited BY newline
            badfile post_code_bad_dir:'post_code_temp%a_%p.bad'
            LOGFILE post_code_log_dir:'post_code_temp%a_%p.log'
            skip 1
            fields terminated BY ',' OPTIONALLY ENCLOSED BY '"'
            missing field VALUES are NULL
            (
                  POST_CODE,
                  CITY,
                  PROVINCE_NAME,
                  PROVINCE,
                  AREA_CODE,
                  CITYTYPE,
                  TIME_ZONE,
                  DAY_LIGHT_SAVING,
                  LATITUDE,
                  LONGITUDE
            )
      )
      LOCATION (POST_CODE_DATA_DIR:'POSTALCODEWORLD-CA-PREMIUM.CSV')
)
REJECT LIMIT UNLIMITED
NOLOGGING
NOCACHE
NOPARALLEL;
```

3) Create regular table (Execute as **post_code** user)

```
CREATE TABLE L_POST_CODE
(
      POST_CODE         VARCHAR2(6 BYTE),
      CITY              VARCHAR2(30 BYTE),
      PROVINCE          VARCHAR2(2 BYTE),
      LATITUDE          NUMBER(18,15),
      LONGITUDE         NUMBER(18,15),
      AREA_CODE         VARCHAR2(3 BYTE),
      CITYTYPE          VARCHAR2(1 BYTE),
      TIME_ZONE         NUMBER(3),
      DAY_LIGHT_SAVING  VARCHAR2(1 BYTE)
);

alter table L_POST_CODE
add constraint
pk_L_POST_CODE primary key (POST_CODE)
using index
tablespace INDX;
```

4) Create oracle package (Execute as **post_code** user)

```
CREATE OR REPLACE PACKAGE load_post_code IS
PROCEDURE update_post_code;
FUNCTION get_pc_distance(p_post_code1  IN VARCHAR2, p_post_code2  IN VARCHAR2) RETURN
NUMBER;
FUNCTION get_geo_distance(lat1 IN NUMBER, lon1 IN NUMBER,lat2 IN NUMBER, lon2 IN NUMBER)
RETURN NUMBER;
END ;
/


CREATE OR REPLACE PACKAGE BODY Load_Post_Code AS
p_title_package_name VARCHAR2(100) := 'load_post_code';
PROCEDURE update_post_code
IS
 p_title_procedure_name VARCHAR2(100) := 'update_post_code';
 CURSOR cur_det IS
 SELECT *
 FROM L_POST_CODE_EXT;
 v_count NUMBER(10) :=0;
 v_upd NUMBER(10) :=0;
 v_ins NUMBER(10) :=0;
 v_radians NUMBER :=57.29577951;
BEGIN
   dbms_output.ENABLE(1000000);
   dbms_output.put_line('-------------------------------------------');
   dbms_output.put_line('Update l_post_code table');
   dbms_output.put_line('-------------------------------------------');
  FOR rec_det IN cur_det LOOP
     v_count:=v_count+1;
     UPDATE L_POST_CODE SET
          CITY=UPPER(rec_det.CITY),
          PROVINCE=UPPER(rec_det.PROVINCE),
          AREA_CODE=rec_det.AREA_CODE,
          CITYTYPE=rec_det.CITYTYPE,
          TIME_ZONE=rec_det.TIME_ZONE,
          LATITUDE=rec_det.LATITUDE/57.29577951,
          LONGITUDE=rec_det.LONGITUDE/57.29577951
     WHERE post_code=REPLACE(rec_det.post_code,' ','');
     IF  SQL%ROWCOUNT=0 THEN
       v_ins:=v_ins+1;
       INSERT INTO
L_POST_CODE(POST_CODE,CITY,PROVINCE,LATITUDE,LONGITUDE,AREA_CODE,CITYTYPE,TIME_ZONE)
       VALUES  (REPLACE(rec_det.post_code,'
',''),UPPER(rec_det.CITY),UPPER(rec_det.PROVINCE),rec_det.LATITUDE/57.29577951,rec_det.LO
NGITUDE/57.29577951, rec_det.AREA_CODE,rec_det.CITYTYPE, rec_det.TIME_ZONE);
     ELSE
       v_upd:=v_upd+1;
     END IF;

  END LOOP;
  COMMIT;
    dbms_output.put_line('Total:'||v_count);
    dbms_output.put_line('Inserted:'||v_ins);
    dbms_output.put_line('Updated:'||v_upd);
  EXCEPTION WHEN OTHERS THEN
    dbms_output.put_line('Error:'||p_title_package_name||'.'||p_title_procedure_name||
                         ':'||SQLERRM);

END;


/*
/*::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::*/
/*:: :*/
/*:: This routine calculates the distance between two points (given the :*/
/*:: latitude/longitude of those points).
/*:: Passed to function: :*/
```

```
/*:: lat1, lon1 = Latitude and Longitude of point 1 (in radians) :*/
/*:: lat2, lon2 = Latitude and Longitude of point 2 (in radians) :*/
/*:: v_r earth radius  :*/
/*:: result returned in kilometers  :*/
/*:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::*/
FUNCTION get_geo_distance(lat1 IN NUMBER, lon1 IN NUMBER,lat2 IN NUMBER, lon2 IN NUMBER)
RETURN NUMBER
IS
  v_r NUMBER :=6378.7;
BEGIN
  IF lat1=lat2 AND lon1=lon2 THEN
    RETURN 0;
  END IF;
  RETURN v_r * ACOS (
                    SIN(lat1) * SIN(lat2) +
                    COS(lat1) * COS(lat2) * COS(lon2- lon1)
                 );
EXCEPTION WHEN OTHERS THEN
  RETURN NULL;
END;

FUNCTION get_pc_distance(p_post_code1  IN VARCHAR2, p_post_code2  IN VARCHAR2) RETURN
NUMBER
IS
  v_LATITUDE1 NUMBER;
  v_LONGITUDE1 NUMBER;
  v_LATITUDE2 NUMBER;
  v_LONGITUDE2  NUMBER;
  a NUMBER;
BEGIN
  IF p_post_code1=p_post_code2 THEN
    RETURN 0;
  END IF;
  BEGIN
    SELECT LATITUDE,LONGITUDE
    INTO  v_LATITUDE1,v_LONGITUDE1
    FROM L_POST_CODE
    WHERE post_code=p_post_code1;
  EXCEPTION WHEN OTHERS THEN
    RETURN NULL;
  END;
  BEGIN
    SELECT LATITUDE,LONGITUDE
    INTO  v_LATITUDE2,v_LONGITUDE2
    FROM L_POST_CODE
    WHERE post_code=p_post_code2;
  EXCEPTION WHEN OTHERS THEN
    RETURN NULL;
  END;
  RETURN get_geo_distance(v_LATITUDE1,v_LONGITUDE1,v_LATITUDE2,v_LONGITUDE2);
EXCEPTION WHEN OTHERS THEN

  RETURN NULL;
END;
END;
/
```

5) Load data from file to permanent table

```
SQL> set serverout on
SQL> execute load_post_code.update_post_code

-------------------------------------------
Update l_post_code table
-------------------------------------------
Total:80896
Inserted:80896
```

```
Updated:0
```

6) Calculate distance between postal codes

```
SQL> SELECT load_post_code.get_pc_distance('A0A1B0','A1A1B4') FROM
dual;

LOAD_POST_CODE.GET_PC_DISTANCE('A0A1B0','A1A1B4')
-------------------------------------------------
                                       41.4562521
```

Using get_pc_distance formula you can do all sorts of actions including finding distance between customers and retail outlet, closest retail outlet to customer and ultimately increase marketing campaigns performance