**Average electricity usage:**

usage(zipCode number, typeOfUsage varchar)

*Algorithm:* The fucntion computes the average usage for gas and electricity in a specific zip-code region.

The records will be grouped for each zip-code to analyze the average consumption of electricity and gas for the top 20% records. Top 20% records will be obtained by using the sort function in sql.

*Input:*1) zipCode

2) typeOfUsage (gas/electricity)

*Output:* Display the top 20% usage for the specified type of usage.

SET SERVEROUTPUT ON;

CREATE OR REPLACE PROCEDURE usage (zipCode IN varchar, monthOfService IN int, yearOfService IN int) IS

zip\_code varchar (30); --Zip code.

avg\_eu number; --Average electricity usage

avg\_eu\_b20 number; --Average electricity usage of the bottom 20 percent.

avg\_gu number; --Average gas usage

avg\_gu\_b20 number; --Average gas usage of the bottom 20 percent.

BEGIN

SELECT avg(electricity\_usage) AS eu\_bottom\_20percent, zip , total\_avg INTO avg\_eu\_b20, zip\_code, avg\_eu

            FROM (

                  SELECT h.\*, a.\*, m.\*, avg(electricity\_usage) OVER (PARTITION BY zip) AS total\_avg,

--We take everything from the 3 joined tables for reference.

--We calculate the average electricity\_usage by zip.

                  row\_number() OVER (PARTITION BY zip ORDER BY electricity\_usage) AS row\_num,

--We use the Oracle row\_number() function to assign the electricity\_usage column a row number (after sorting it).

--These row numbers are assigned after partitioning the zip and then sorting by electricity\_usage (from smallest to largest).

                  greatest(1,round((COUNT(DISTINCT h.hid) OVER (PARTITION BY zip)) \* 0.2)) AS bottom\_20\_cnt

--We identify the total number of distinct houses by zip and multiply it with .2 to get the bottom 20%.

--The round function is used to ensure we have 20% (in case we have an odd number of rows).

--The greatest function is used to ensure that the bottom\_20\_cnt is greater than 0.

--This way, if we have less than 5 houses in a zip, bottom 20% of houses will equal total average for that zip.

                  FROM house\_address h

                       INNER JOIN account a ON (h.hid = a.hid)

                       INNER JOIN monthly\_bill m ON (a.aid = m.aid)

                  WHERE zip = zipCode --Input Parameter limits results to a specified zip code.

                         AND extract (year FROM bill\_date) = yearOfService --Input Parameter limits the results to a specific year.

                         AND extract (month FROM bill\_date) = monthOfService --Input Parameter limits the results to a specific month.

                   ) t

            WHERE t.row\_num <= bottom\_20\_cnt --This WHERE clause limits the AVG(electricity\_usage) to bottom 20%.

            GROUP BY zip, total\_avg

            ORDER BY zip;

dbms\_output.put\_line('The average electricity usage for zipcode ' || zip\_code || ' is ' || avg\_eu ||

                     ' and the average usage of the bottom 20% is ' || avg\_eu\_b20 || '.');

--The same process is repeated to calculate the averages for gas usage.

SELECT avg(gas\_usage) AS gu\_bottom\_20percent, zip , total\_avg INTO avg\_gu\_b20, zip\_code, avg\_gu

            FROM (

                  SELECT h.\*, a.\*, m.\*, avg(gas\_usage) OVER (PARTITION BY zip) AS total\_avg ,

                  row\_number() OVER (PARTITION BY zip ORDER BY gas\_usage) AS row\_num,

                  greatest(1,round((COUNT(DISTINCT h.hid) OVER (PARTITION BY zip)) \* 0.2)) AS bottom\_20\_cnt

                  FROM house\_address h

                       INNER JOIN account a ON (h.hid = a.hid)

                       INNER JOIN monthly\_bill m ON (a.aid = m.aid)

                  WHERE zip = zipCode --Input Parameter limits results to specified zip code.

                         AND extract (year FROM bill\_date) = yearOfService

                         AND extract (month FROM bill\_date) = monthOfService

                   ) t

            WHERE t.row\_num <= bottom\_20\_cnt

            GROUP BY zip, total\_avg

            ORDER BY zip;

dbms\_output.put\_line('The average gas usage for zipcode ' || zip\_code || ' is ' || avg\_gu ||

                     ' and the average usage of the bottom 20% is ' || avg\_gu\_b20 || '.');

EXCEPTION

   WHEN no\_data\_found THEN

DBMS\_OUTPUT.PUT\_LINE('No usage data found.');

END;

EXEC usage (21225, 10, 2016);