**Lab 01 – Week 2 (Single-row Functions)**

This week’s lab continues using the SELECT command and learning the interfaces for both SQL Developer and introduces the use of single-line functions.

Submission

***Your submission will be a single text-based SQL file with appropriate header and commenting. Please ensure your file runs when the entire file is executed in SQL Developer.***

Create a new Worksheet in SQL Developer. Save the file as L01\_ID#\_LASTNAME.sql

Your submission needs to be commented and include the question, the solutions.

Do not comment the solutions (SQL Statements).

**Tasks**

-- **Q1:** Write a query to display the tomorrow’s date in the following format:  
 January 10th of year 2019  
the result will depend on the day when you RUN/EXECUTE this query. Label the column “Tomorrow”.  
  
***Advanced Option*:** Define an SQL variable called “tomorrow”, assign it a value of tomorrow’s date and use it in an SQL statement. Here the question is asking you to use a Substitution variable. Instead of using the constant values in your queries, you can use variables to store and reuse the values.

**See the following example:**

**select \***   
**from employees  
where employee\_id = 107;**

**You can also have the following code:**

**define emp\_id number = 107;  
select \*  
from employees  
where employee\_id = &emp\_id;  
After you use the variable, you can undefined the variable:**

**undefine emp\_id;**

**Define a variable of type datetime:**

**define toay datetime = sysdate; -- Assigning current date to the today variable.**

**Or**

**define tomorrow = sysdate + 1;**

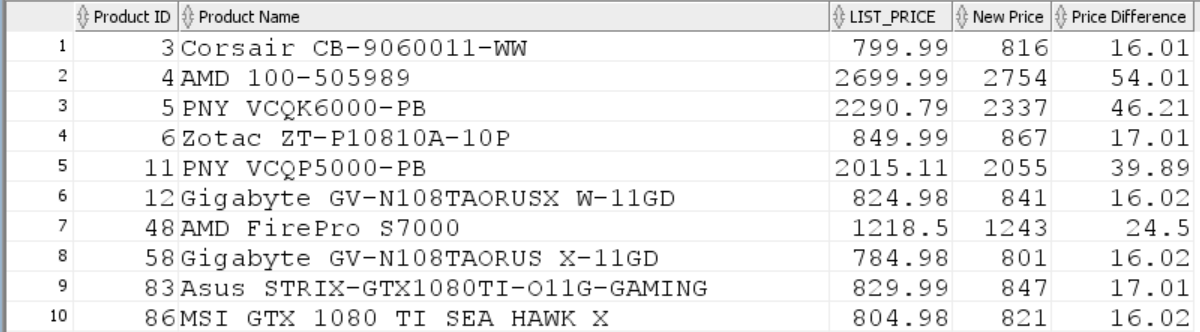
-- **Q2:** For each product in category 2, 3, and 5, show product ID, product name, list price, and the new list price increased by 2%. Display a new list price as a whole number.

In your result, add a calculated column to show the difference of old and new list prices.

Sort the result according to category ID first and then based on product ID.

You output has to match the following result. This result is partially displayed as it has 158 rows.

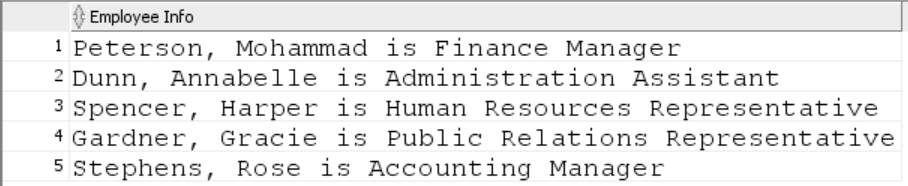
See the result for the first 10 rows.

****

-- **Q3:** For employees whose manager ID is 2, write a query that displays the employee’s Full Name and Job Title in the following format:

Summer, Payne is Public Accountant.

Sort the result based on employee ID.

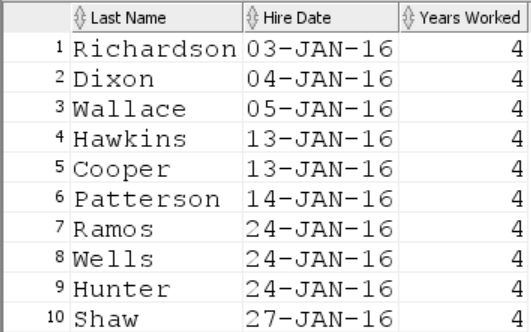


-- **Q4:** For each employee hired before October 2016, display the employee’s last name, hire date and calculate the number of YEARS between TODAY and the date the employee was hired.

* Label the column Years worked.
* Order your results by the number of years employed. Round the number of years employed up to the closest whole number.

The output result includes 89 rows. See the partial result (The first 10 rows).

**If you get the result in a different order, sort the result first based on the hire date column and then based on the number of years worked.**



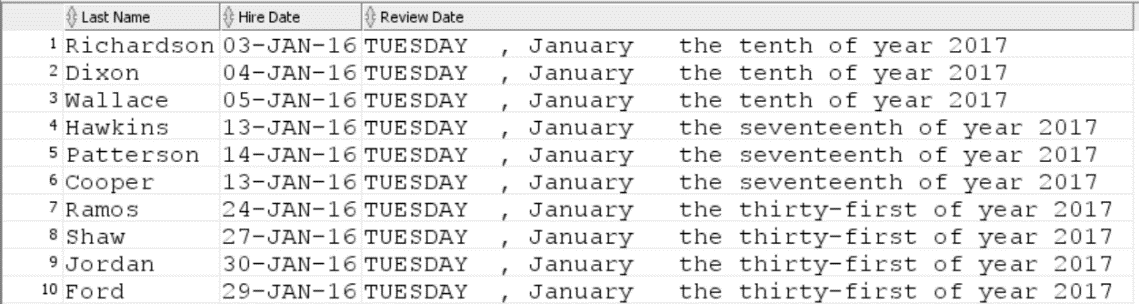
-- **Q5:** Display each employee’s last name, hire date, and the review date, which is the first Tuesday after a year of service, but only for those hired after January 1, 2016.

* Label the column REVIEW DAY.
* Format the dates to appear in the format like:  
   TUESDAY, August the Thirty-First of year 2016

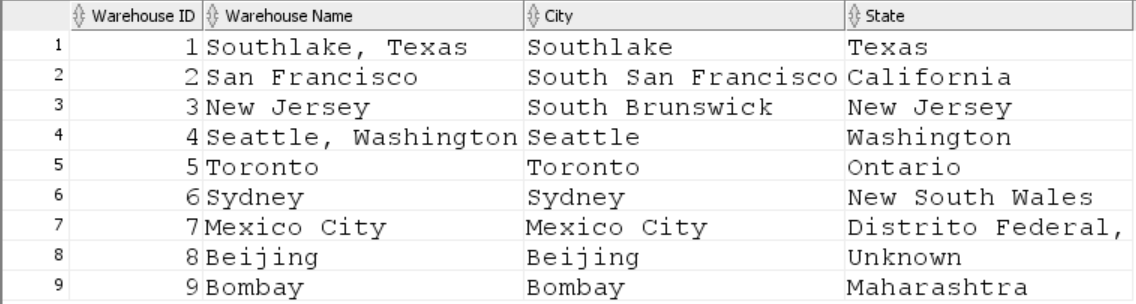
You can use ***ddspth*** to have the above format for the day.

* Sort by review date

The Query returns 107 rows. See the first 10 rows of the output result.



-- **Q6:** For all warehouses, display warehouse id, warehouse name, city, and state. For warehouses with the null value for the state column, display “unknown”. Sort the result based on the warehouse ID.



Example Submission

-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
-- Name: Your Name  
-- ID: #########  
-- Date: The current date  
-- Purpose: Lab 1 DBS311  
-- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
-- Question 1 – write a brief note about what the question is asking  
-- Q1 SOLUTION --  
  
SELECT \* FROM TABLE;

-- Question 2 –   
-- Q2 Solution –

SELECT \* FROM TABLE;