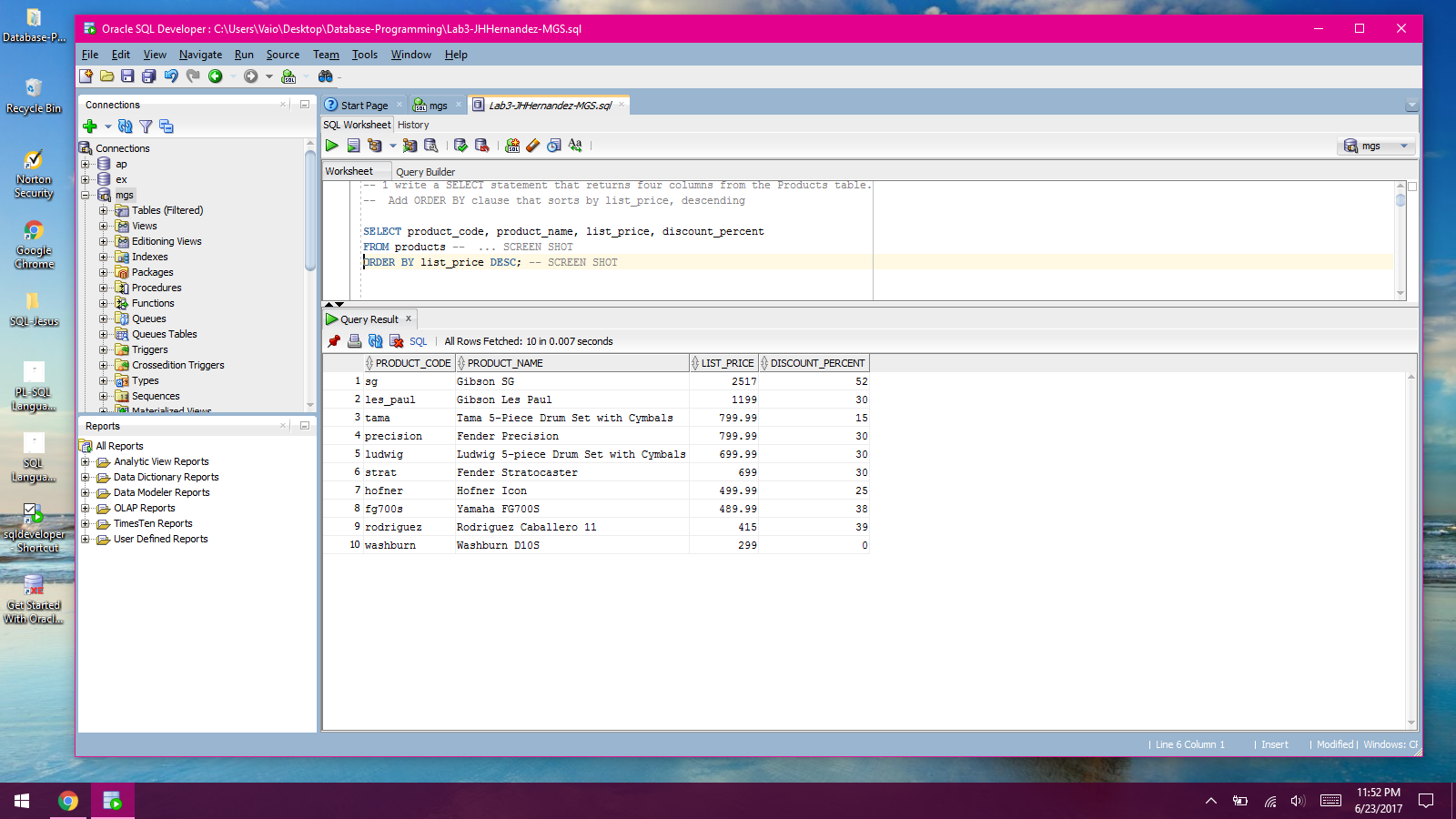
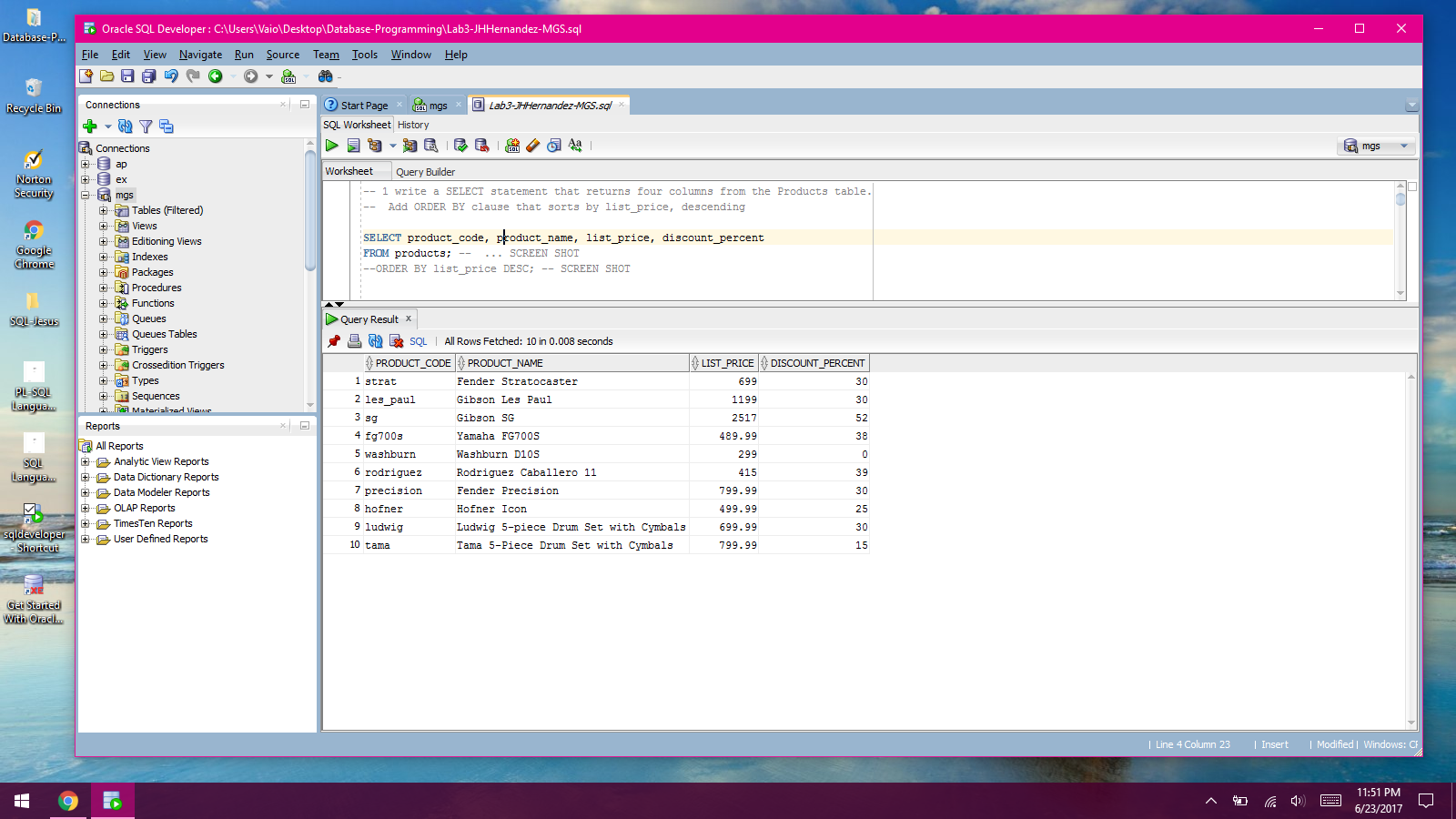
-- 1 write a SELECT statement that returns four columns from the Products table.

-- Add ORDER BY clause that sorts by list\_price, descending

SELECT product\_code, product\_name, list\_price, discount\_percent

FROM products -- SCREEN SHOT

ORDER BY list\_price DESC; -- SCREEN SHOT



-- 2 Write a SELECT statement that returns one column from

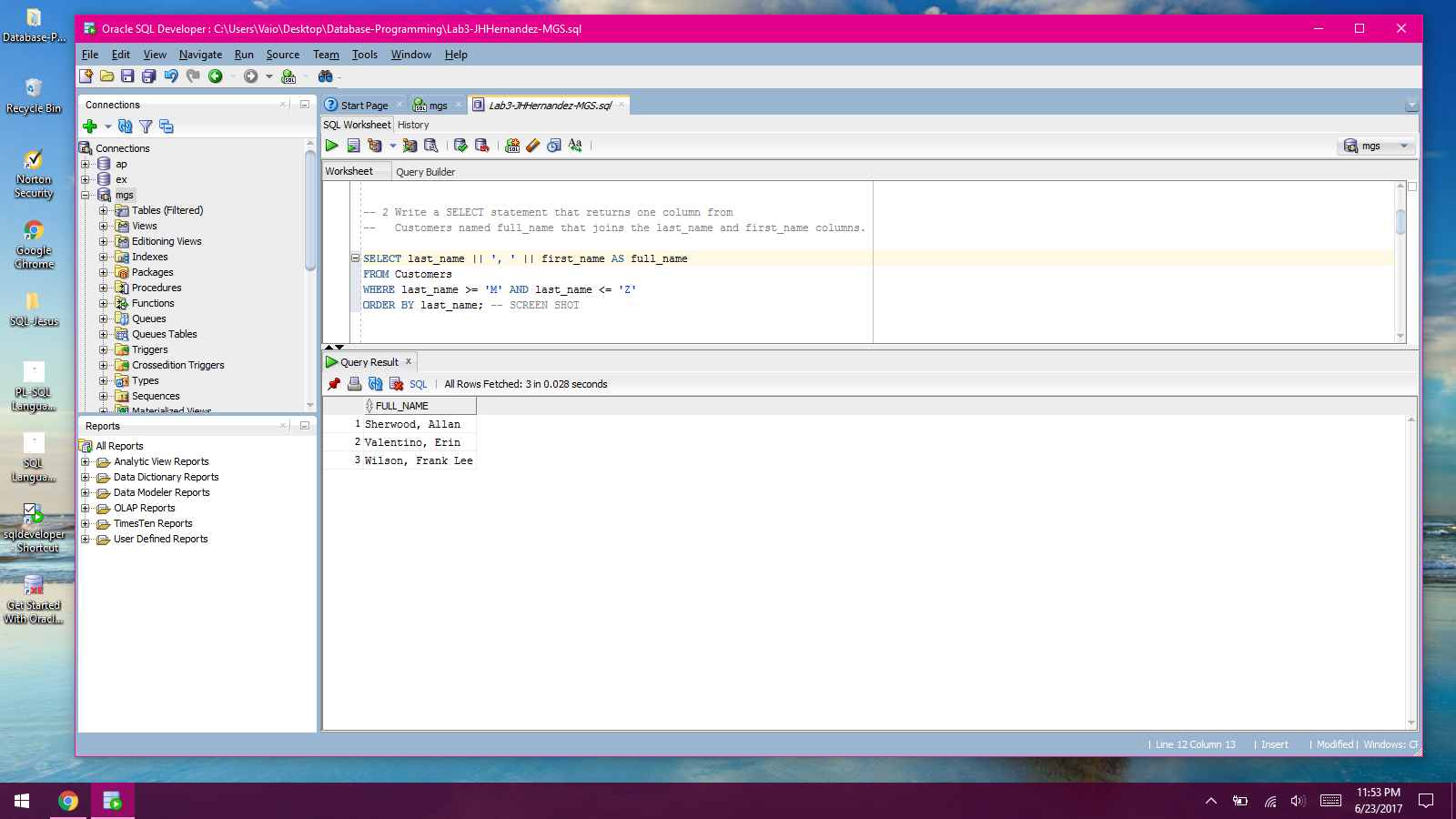
-- Customers named full\_name that joins the last\_name and first\_name columns.

SELECT last\_name || ', ' || first\_name AS full\_name

FROM Customers

WHERE last\_name >= 'M' AND last\_name <= 'Z'

ORDER BY last\_name; -- SCREEN SHOT



--3 Write a SELECT statment that returns columns from Products table:

-- Return only the rows with a list price that's greater than 500

-- and less than 2000.

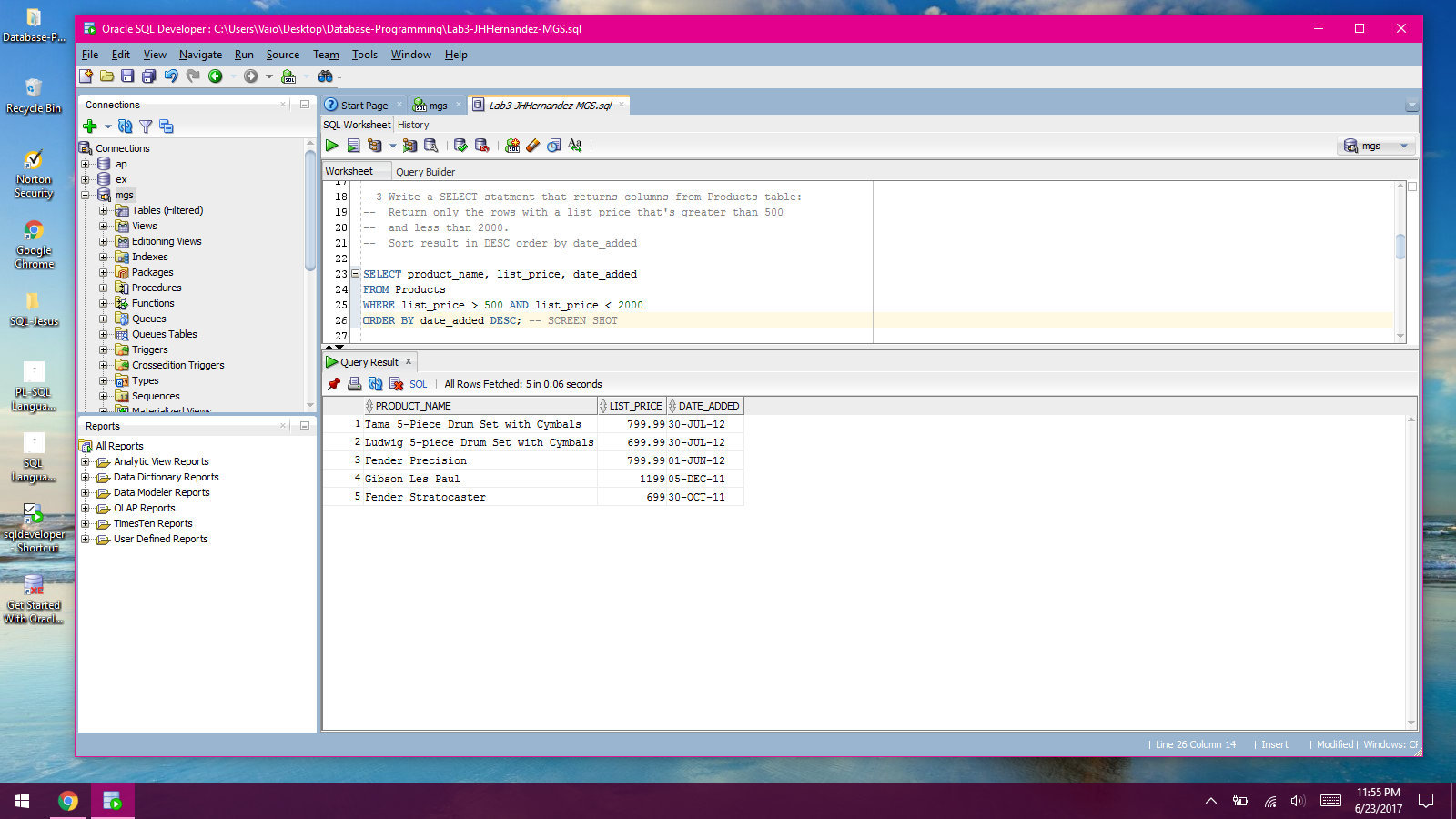
-- Sort result in DESC order by date\_added

SELECT product\_name, list\_price, date\_added

FROM Products

WHERE list\_price > 500 AND list\_price < 2000

ORDER BY date\_added DESC; -- SCREEN SHOT



-- 4 Write a SELECT statement that returns column names from Products table:

-- Use the ROWNUM pseudo column so the result set contains only the first 5 rows.

-- Sort the result set by discount price in descending sequence.

SELECT product\_name, list\_price, discount\_percent,

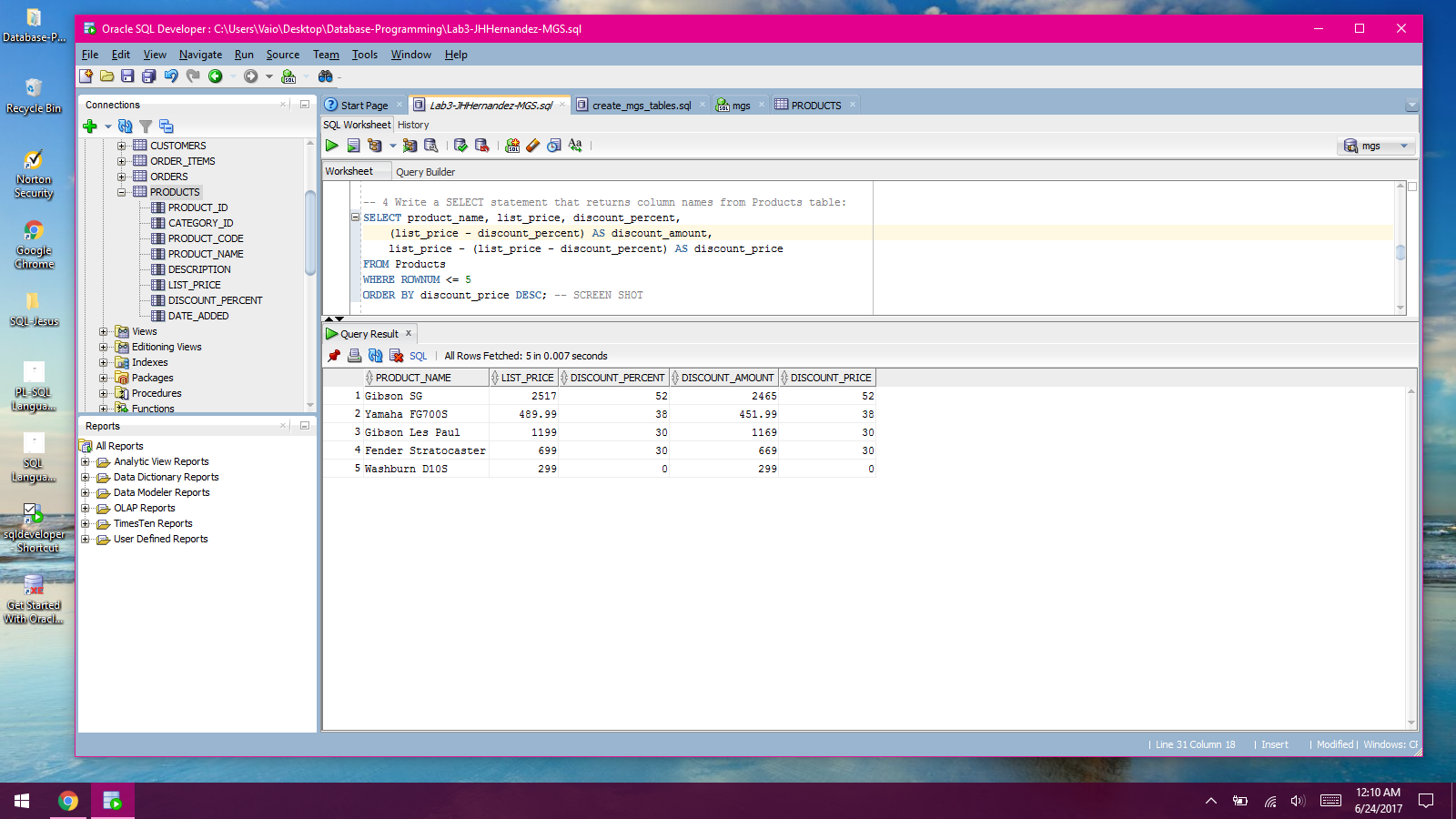
(list\_price - discount\_percent) AS discount\_amount,

list\_price - (list\_price - discount\_percent) AS discount\_price

FROM Products

WHERE ROWNUM <= 5

ORDER BY discount\_price DESC; -- SCREEN SHOT



-- 5 Write a SELECT statement that returns columns from Order\_Items table:

-- Only return rows where the item\_total is greater than 500.

-- Sort the result set by item total in descending sequence.

SELECT item\_id, item\_price, discount\_amount, quantity,

(item\_price \* quantity) AS price\_total,

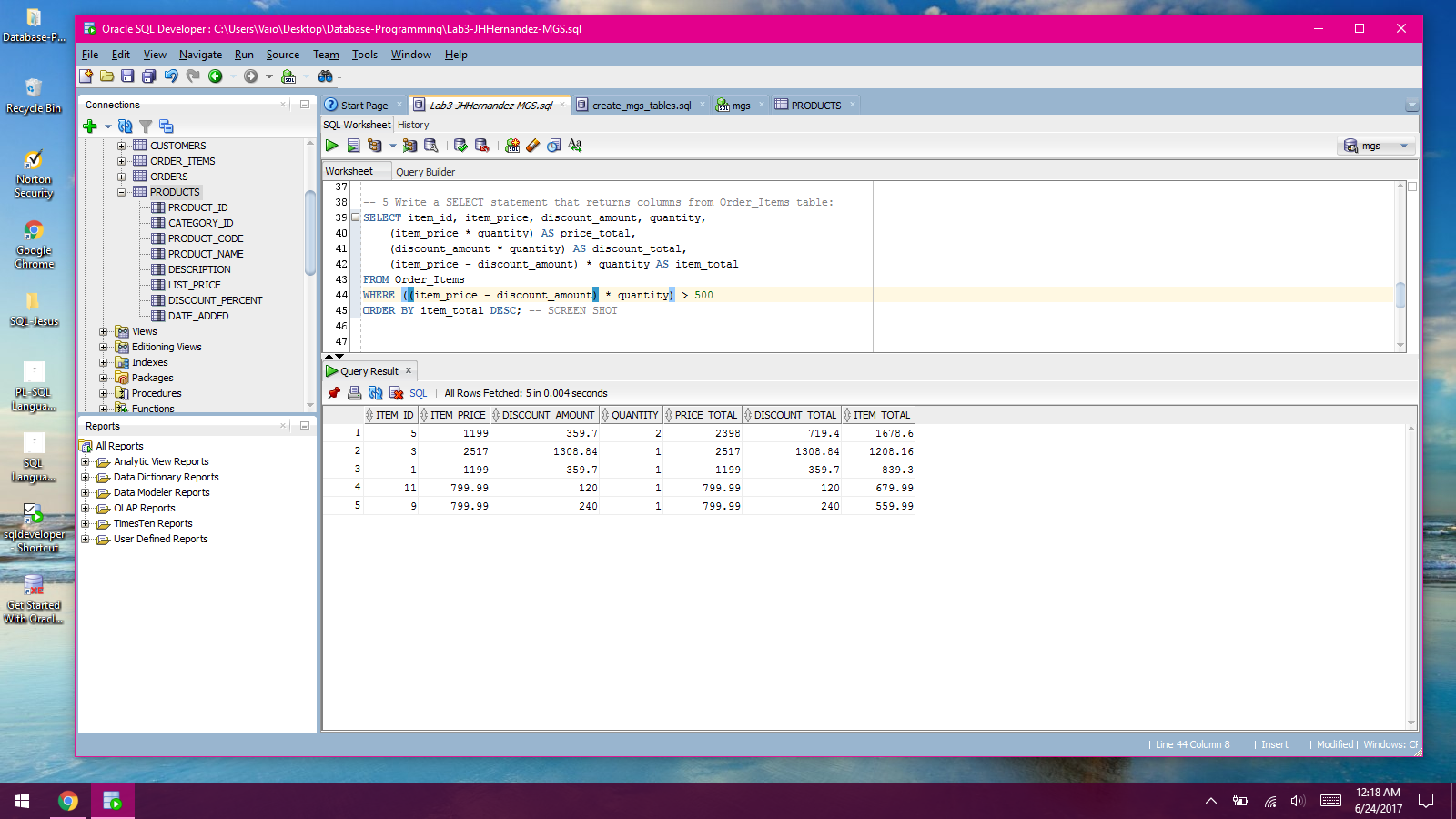
(discount\_amount \* quantity) AS discount\_total,

(item\_price - discount\_amount) \* quantity AS item\_total

FROM Order\_Items

WHERE ((item\_price - discount\_amount) \* quantity) > 500

ORDER BY item\_total DESC; -- SCREEN SHOT



-- WORK WITH NULLS AND TEST EXPRESSIONS --

-- WORK WITH NULLS AND TEST EXPRESSIONS --

-- WORK WITH NULLS AND TEST EXPRESSIONS –

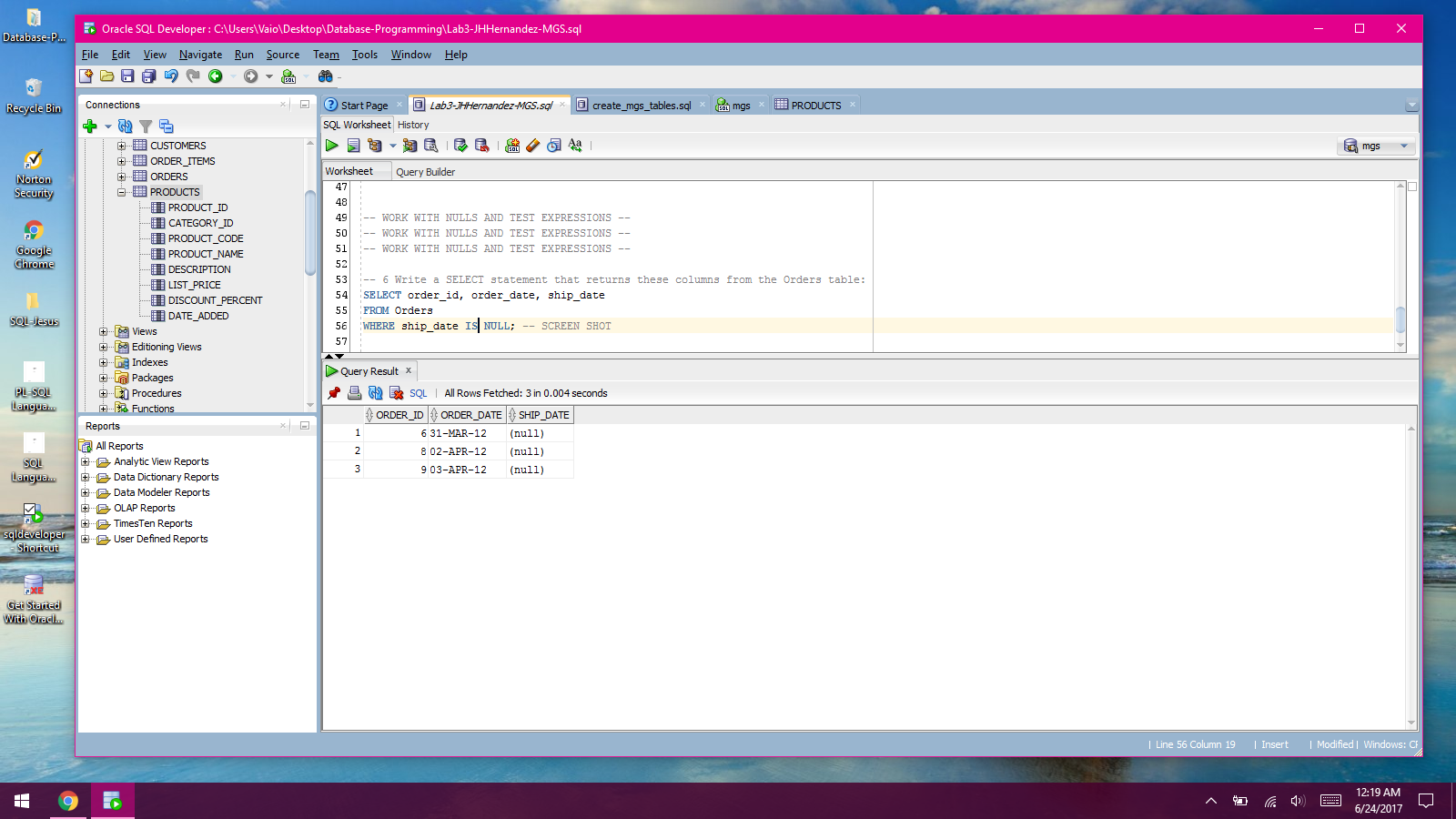
-- 6 Write a SELECT statement that returns these columns from the Orders table:

-- Return only the rows where the ship\_date column contains a null value.

SELECT order\_id, order\_date, ship\_date

FROM Orders

WHERE ship\_date IS NULL; -- SCREEN SHOT



-- 7 Write a SELECT statement that uses the SYSDATE function to create a row with these columns:

-- This displays a number for the month, a number for the day, and a four-digit year.

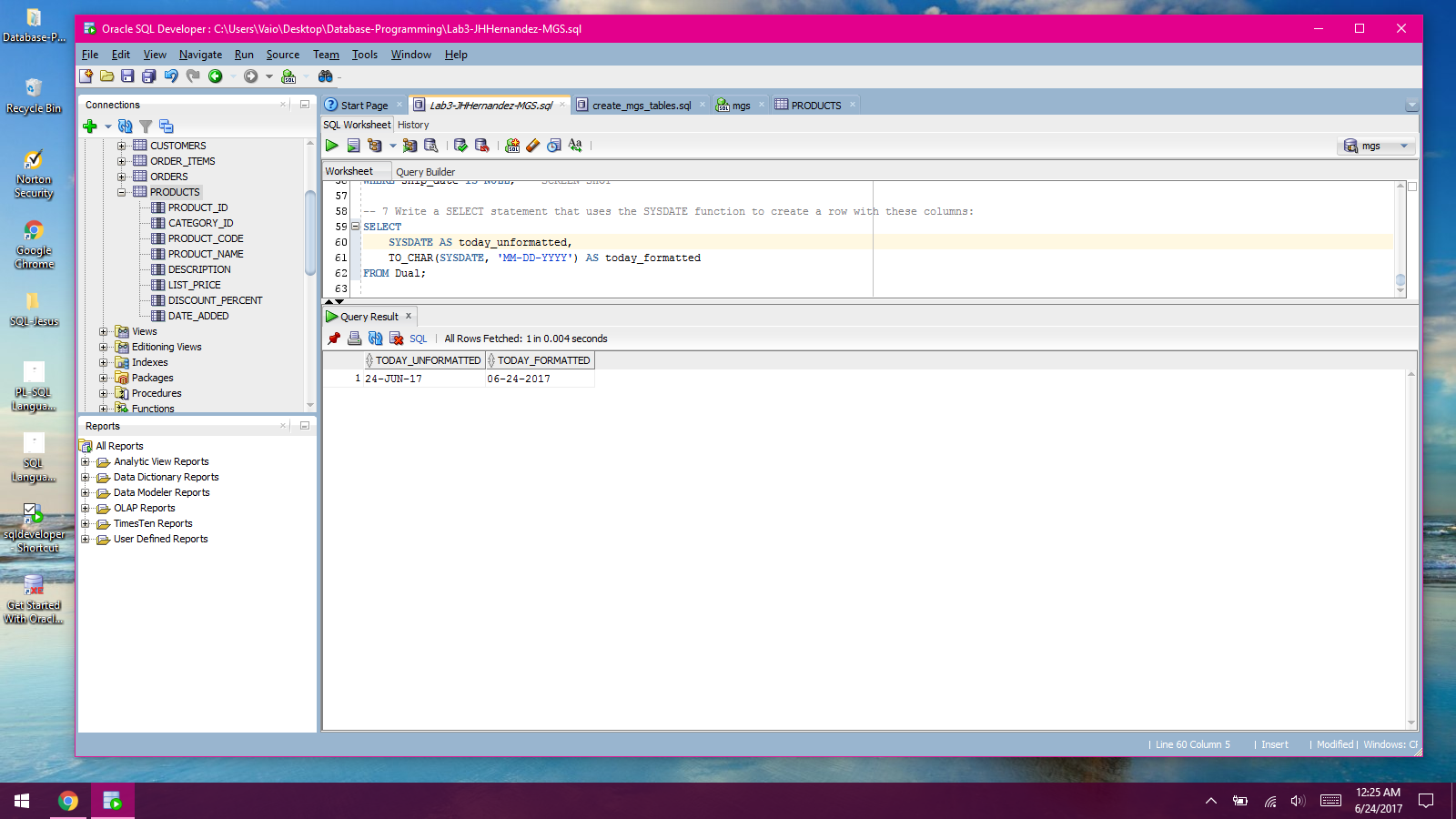
-- Use a FROM clause that specifies the Dual table.

SELECT

SYSDATE AS today\_unformatted,

TO\_CHAR(SYSDATE, 'MM-DD-YYYY') AS today\_formatted

FROM Dual; -- SCREEN SHOT



-- 8 Write a SELECT statement that creates a row with these columns;

-- To calculate the fourth column, add the expressions you used for the first and third columns.

-- Use a FROM clause that specifies the Dual table.

SELECT

100 AS price,

.07 AS tax\_rate,

(100 \* .07) AS tax\_amount,

(100 + (100 \* .07)) AS total

FROM Dual; -- SCREEN SHOT

