# Exercises

Your Name: STEPHEN OMONDI  
Your SUID: 946934043  
Your Email: SOOMONDI@SYR.EDU  
Date Due: 10/13/2019  
Homework #: LABWORK 2

Complete each of the following exercises. If you are unsure how to accomplish the task, please consult the coursework videos where there are explanations and demos.

1. Use built in SQL functions to write an SQL Select statement on **fudgemart\_products** which derives a **product\_category** column by extracting the last word in the product name. For example
   1. for a product named ‘Leather Jacket’ the product category would be ‘Jacket’
   2. for a product named ‘Straight Claw Hammer’ the category would be ‘Hammer’

Your select statement should include product id, product name, product category and product department.

**SQL QUERY:**

---set the database to use

USE fudgemart\_v3

GO

SELECT product\_id,

product\_name,

CASE(CHARINDEX(' ', (product\_name)))

WHEN 0 THEN product\_name

ELSE RIGHT(product\_name, CHARINDEX(' ', REVERSE(product\_name))-1)

END

AS product\_category

FROM dbo.fudgemart\_products

GO

1. Write a user defined function called **f\_total\_vendor\_sales** which calculates the sum of the wholesale price \* quantity of all products sold for that vendor. There should be one number associated with each vendor id, which is the input into the function. Demonstrate the function works by executing an SQL select statement over all vendors calling the function.

**SQL QUERY:**

USE fudgemart\_v3

GO

----drop the function if it already exists

DROP FUNCTION IF EXISTS f\_total\_vendor\_sales

GO

CREATE FUNCTION dbo.f\_total\_vendor\_sales

(

@vendor\_id int

)

RETURNS int

AS

BEGIN

declare @total int

set @total = (

SELECT SUM(PR.[product\_wholesale\_price] \* ORD.[order\_qty])

FROM [dbo].[fudgemart\_products] AS PR

INNER JOIN [dbo].[fudgemart\_order\_details] ORD ON ORD.product\_id = PR.product\_id

INNER JOIN [dbo].[fudgemart\_vendors] AS VND ON VND.vendor\_id = PR.product\_vendor\_id

WHERE PR.product\_vendor\_id = @vendor\_id);

RETURN @total

END

GO

--call the function

SELECT vendor\_name,

dbo.f\_total\_vendor\_sales(vendor\_id) AS Total\_Sales

FROM dbo.fudgemart\_vendors

ORDER BY vendor\_name ASC

GO

1. Write a stored procedure called **p\_write\_vendor** which when given a required vendor name, phone and optional website, will look up the vendor by name first. If the vendor exists, it will update the phone and website. If the vendor does not exist, it will add the info to the table. Write code to demonstrate the procedure works by executing the procedure twice so that it adds a new vendor and then updates that vendor’s information.

**SQL QUERY:**

---which database to use--

USE fudgemart\_v3

GO

----check if the procedure already exists, and drop it

DROP PROCEDURE IF EXISTS dbo.p\_write\_vendor

GO

--create the procedure

CREATE PROCEDURE dbo.p\_write\_vendor(

@vendor\_name varchar(max),

@vendor\_phone int

)AS

BEGIN

IF EXISTS(SELECT vendor\_name, vendor\_phone, vendor\_website FROM dbo.fudgemart\_vendors WHERE vendor\_name = @vendor\_name)

UPDATE dbo.fudgemart\_vendors

SET vendor\_website = @vendor\_name + '.com'

WHERE vendor\_name = @vendor\_name

ELSE

INSERT INTO dbo.fudgemart\_vendors(vendor\_name, vendor\_phone, vendor\_website) VALUES (

@vendor\_name,@vendor\_phone,@vendor\_name+'.com')

END

GO

--call the procedure---

exec dbo.p\_write\_vendor 'Johnson', 123455

GO

--verify--

SELECT \* FROM dbo.fudgemart\_vendors

1. Create a view based on the logic you completed in question 1 or 2. Your SQL script should be programmed so that the entire script works every time, dropping the view if it exists, and then re-creating it.

**SQL QUERY:**

USE fudgemart\_v3

GO

--does the exist already

DROP VIEW IF EXISTS dbo.vendor\_total\_sales

GO

CREATE VIEW dbo.vendor\_total\_sales

AS

SELECT SUM(PR.[product\_wholesale\_price] \* ORD.[order\_qty]) AS total\_sales

FROM [dbo].[fudgemart\_products] AS PR

INNER JOIN [dbo].[fudgemart\_order\_details] ORD ON ORD.product\_id = PR.product\_id

INNER JOIN [dbo].[fudgemart\_vendors] AS VND ON VND.vendor\_id = PR.product\_vendor\_id

WHERE PR.product\_vendor\_id = 10;

GO

1. Write a table valued function **f\_employee\_timesheets** which when provided an employee\_id will output the employee id, name, department, payroll date, hourly rate on the timesheet, hours worked, and gross pay (hourly rate times hours worked).

**SQL QUERY**

USE fudgemart\_v3

GO

DROP FUNCTION IF EXISTS dbo.f\_employees\_timesheets

GO

---create the function---

CREATE FUNCTION f\_employees\_timesheets(

@employee\_id int

) RETURNS TABLE AS

RETURN(

SELECT timesheet\_employee\_id,dbo.fudgemart\_employees.employee\_lastname, dbo.fudgemart\_employees.employee\_department,

timesheet\_payrolldate,timesheet\_hourlyrate, timesheet\_hours, timesheet\_hourlyrate \* timesheet\_hours AS gross\_pay

FROM dbo.fudgemart\_employee\_timesheets

JOIN dbo.fudgemart\_employees ON dbo.fudgemart\_employees.employee\_id = dbo.fudgemart\_employee\_timesheets.timesheet\_employee\_id

JOIN dbo.fudgemart\_departments\_lookup ON dbo.fudgemart\_departments\_lookup.department\_id = dbo.fudgemart\_employees.employee\_department

WHERE timesheet\_employee\_id = @employee\_id

)

GO

--call the function with employee\_id equal to 1 ---

SELECT \* FROM f\_employees\_timesheets(1)

GO