# IST769 SQL Programming

Your Name: Tajudeen Abdulazeez   
Your SUID: 69687-7373-0   
Your Email: toabdula@syr.edu   
Date Due:   
Homework #: 2

1. 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.

SELECT product\_id

,product\_name

,product\_department

,CASE

WHEN CHARINDEX(' ', product\_name) = 0 THEN product\_name

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

END AS product\_category

FROM fudgemart\_products

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.

CREATE FUNCTION f\_total\_vendor\_sales (

@vendor\_id float

)

RETURNS float AS

BEGIN

declare @ret float

set @ret = (

SELECT sum(t.product\_wholesale\_price \* o.order\_qty)

FROM fudgemart\_vendors as f

FULL OUTER JOIN fudgemart\_products as t

ON (f.vendor\_id = t.product\_vendor\_id)

FULL OUTER JOIN fudgemart\_order\_details as o

ON (o.product\_id = t.product\_id)

WHERE f.vendor\_id = @ret

)

RETURN @ret

END

GO

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

FROM fudgemart\_vendors

ORDER BY vendor\_name

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.

GO

IF OBJECT\_ID('bo.p\_write\_vendor') IS NOT NULL

DROP PROCEDURE dbo.p\_write\_vendor

GO

CREATE PROCEDURE dbo.p\_write\_vendor(

@vendor\_name VARCHAR,

@phone VARCHAR,

@website VARCHAR

)

AS

BEGIN

IF EXISTS( SELECT \* FROM fudgemart\_vendors WHERE vendor\_name = @vendor\_name)

UPDATE dbo.fudgemart\_vendors

SET vendor\_phone = @phone,

vendor\_website = @website

WHERE vendor\_name = @vendor\_name

ELSE

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

VALUES (@vendor\_name, @phone, @website)

END

GO

EXEC dbo.p\_write\_vendor 'Vendor name', '92234555', 'www.syr.edu'

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.

GO

DROP VIEW IF EXISTS dbo.vw\_vendor\_sales

GO

CREATE VIEW dbo.vw\_vendor\_sales

AS(

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

FROM fudgemart\_vendors

)

GO

SELECT \* FROM dbo.vw\_vendor\_sales

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).

GO

DROP FUNCTION IF EXISTS dbo.f\_employee\_timesheets

GO

CREATE FUNCTION dbo.f\_employee\_timesheets(

@employee\_id INT

)

RETURNS TABLE

AS

RETURN (

SELECT e.employee\_id, e.employee\_lastname, e.employee\_firstname, e.employee\_department, t.timesheet\_payrolldate,

t.timesheet\_hours \* t.timesheet\_hourlyrate AS grosspay

FROM fudgemart\_employee\_timesheets AS t

INNER JOIN fudgemart\_employees AS e ON e.employee\_id = t.timesheet\_employee\_id

WHERE e.employee\_id = @employee\_id

);

GO