# IST769 Homework Submission #2

## Basic Information

Your Name: Thulasi Ram Ruppa Krishnan   
Your SUID: truppakr  
Your Email: truppakr@syr.edu  
Date Due: 460746269   
Homework #: HW2

## Instructions

For each answer, please include your answer as text, and any screenshot(s) which demonstrate your answer was executed. Most importantly, make sure to include evidence your answer is correct. This will most likely be a screenshot. If you had issues, problems, or had to make assumptions include them in your answer.

## Answers:

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.

**Solution:**

select

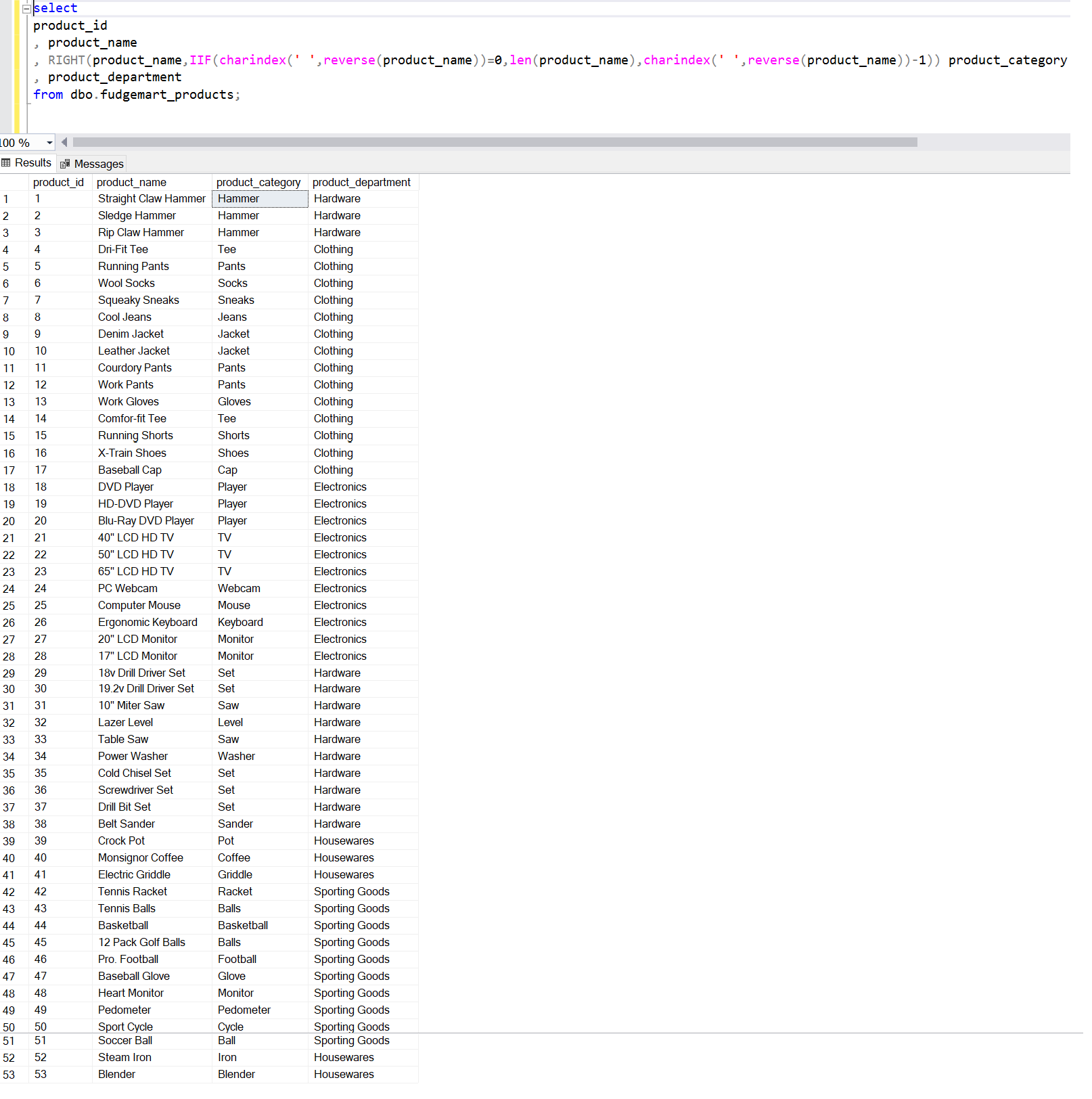
product\_id

, product\_name

, RIGHT(product\_name,IIF(charindex(' ',reverse(product\_name))=0,len(product\_name),charindex(' ',reverse(product\_name))-1)) product\_category

, product\_department

from dbo.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.

if exists (select \* from sys.objects where name ='f\_total\_vendor\_sales')

begin

drop function dbo.f\_total\_vendor\_sales

end

go

create function dbo.f\_total\_vendor\_sales(

@vendor\_id int

)

returns dec(12,2) as

begin

declare @total dec(12,2);

select @total= sum(p.product\_wholesale\_price\*od.order\_qty )

from dbo.fudgemart\_products p

join dbo.fudgemart\_order\_details od on p.product\_id=od.product\_id

where p.product\_vendor\_id=@vendor\_id ;

return @total;

end;

go

select \*,dbo.f\_total\_vendor\_sales(vendor\_id) sales\_amount from dbo.fudgemart\_vendors;



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.

if exists(select \* from sys.objects where name ='p\_write\_vendor')

begin

drop procedure dbo.p\_write\_vendor;

end

go

create procedure p\_write\_vendor(

@vendor\_name varchar(50), @phone varchar(20), @website varchar(1000)

)

as

begin

if exists(select \* from dbo.fudgemart\_vendors where vendor\_name=@vendor\_name)

begin

update dbo.fudgemart\_vendors set vendor\_phone=@phone,vendor\_website=@website where vendor\_name=@vendor\_name;

end

else

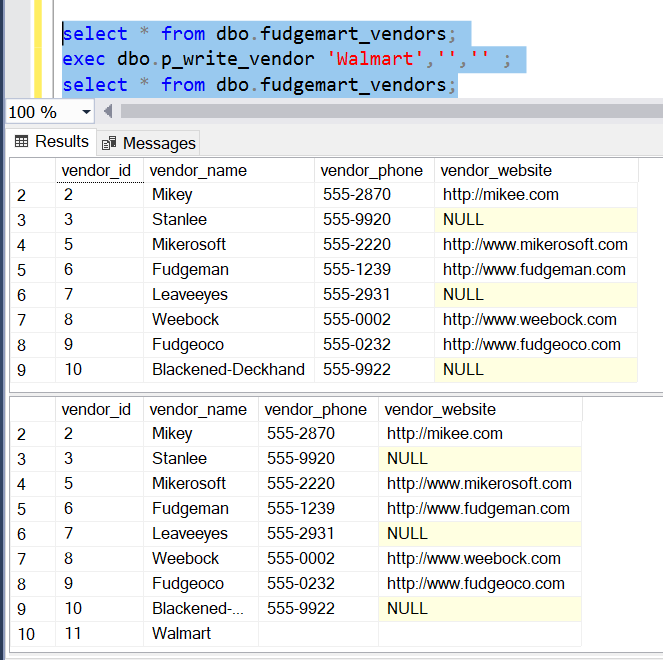
begin

insert into dbo.fudgemart\_vendors (vendor\_name,vendor\_phone,vendor\_website) values(@vendor\_name,@phone,@website);

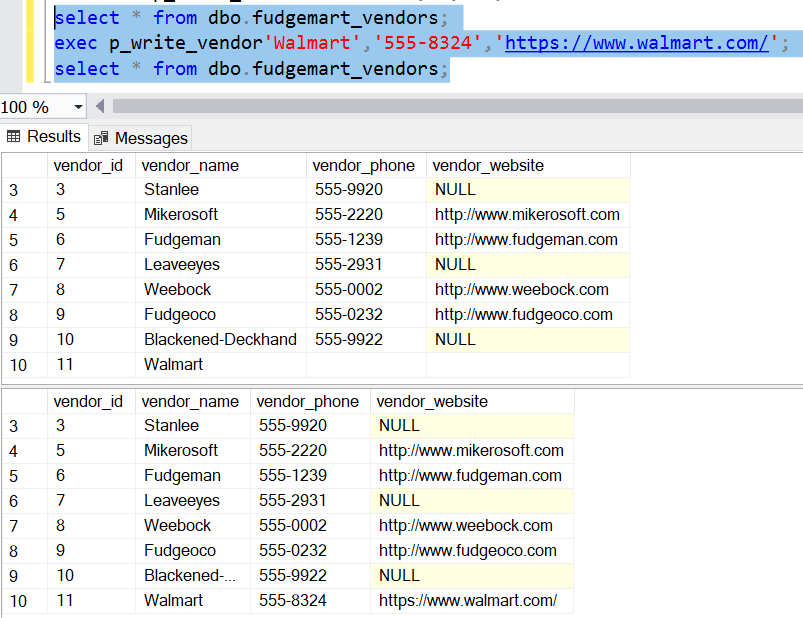
end

end ;

**Execution 1: Insert a new vendor “Walmart” with blank phone and website**



**Execution 2: Update existing vendor “Walmart” with phone and website**



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.

**Solution:**

if exists(select \* from sys.objects where name='vw\_fudgemart\_product\_category')

begin

drop view dbo.vw\_fudgemart\_product\_category;

end

go

create view dbo.vw\_fudgemart\_product\_category as

select

product\_id

, product\_name

, RIGHT(product\_name,IIF(charindex(' ',reverse(product\_name))=0,len(product\_name),charindex(' ',reverse(product\_name))-1)) product\_category

, product\_department

from dbo.fudgemart\_products;

go

if exists(select \* from sys.objects where name='vw\_fudgemart\_vendor\_sales')

begin

drop view dbo.vw\_fudgemart\_vendor\_sales;

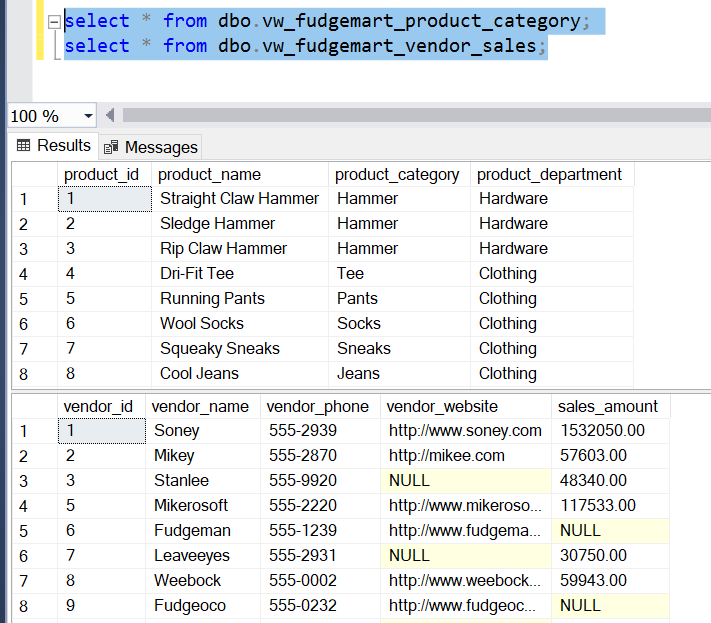
end

go

create view dbo.vw\_fudgemart\_vendor\_sales as

select vendor\_id,vendor\_name,vendor\_phone,vendor\_website,dbo.f\_total\_vendor\_sales(vendor\_id) sales\_amount from dbo.fudgemart\_vendors;

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).

**Solution:**

if exists(select \* from sys.objects where name ='f\_employee\_timesheets')

begin

drop function dbo.f\_employee\_timesheets;

end

go

create function f\_employee\_timesheets(

@employee\_id int

)

returns table as

return

select employee\_id

,concat(employee\_firstname,' ',employee\_lastname) name

,employee\_department department

,ts.timesheet\_payrolldate payroll\_date

,ts.timesheet\_hourlyrate hourly\_rate

,ts.timesheet\_hours hours\_worked

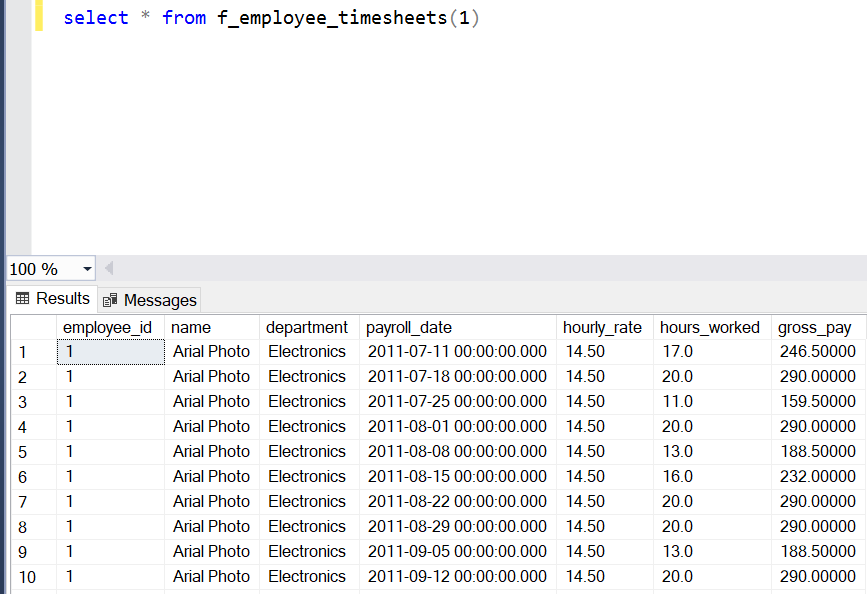
,ts.timesheet\_hourlyrate\*ts.timesheet\_hours gross\_pay

from dbo.fudgemart\_employees emp

left join dbo.fudgemart\_employee\_timesheets ts on emp.employee\_id=ts.timesheet\_employee\_id

where emp.employee\_id=@employee\_id;

go



# Tear-Down

When you are finished with the homework you should stop the environment:

1. From the terminal window where you typed docker-compose up -d type in the following:  
   docker-compose stop

