**IS420 Homework 2**

**Due: 4/04/2018 11:59PM**

**What you need to submit in your assignment document (MS Word or PDF):**

**For every question submit:**

**1. SQL query/ PLSQL program unit**

**2. SCREENSHOT OF THE OUTPUT (wrong screenshots or missing screenshots will result in deduction of 50% of the points)**

Once your homework is graded please check the grades and let the TA know if there are any discrepancies within one week.

1-(25 points) related topics :view, predefined\_functions group by.

Create a **view** that contains statistics readings for invoices total in each state. This view is important for the decision makers. The view contains one row per state and contains seven columns (vendor\_state, total\_sum\_invoices, average, median\_value, mode\_value, max\_value, min\_value)

The columns description as follow:

* Vendor\_state: vendor\_state from vendors table
* Total\_sum\_invoices: is the SUM of all invoice\_total for each state. Use SUM() function
* Average : is the average of all invoice\_total for each state. Use AVG() function and round the result using ROUND() function.
* Median\_value: is the middle value of invoice\_total for all invoice\_total for each state. Use MEDIAN() function
* Mode\_value: is the most frequent invoice\_total value for each state. Use STATS\_MODE() function
* Max\_value: is the maximum invoice\_total for each state. USE MAX() function.
* Min\_value: is the minimum invoice\_total for each state. USE MIN() function.

Submit the following :

* The code to create the view
* A screenshot to show the data in the view.

“create or replace view state\_view

as

select vendor\_state,

sum(invoice\_total) as invoice\_sum, round(avg(invoice\_total)) as invoice\_avg,

median(invoice\_total) as invoice\_median, stats\_mode(invoice\_total) as invoice\_mode,

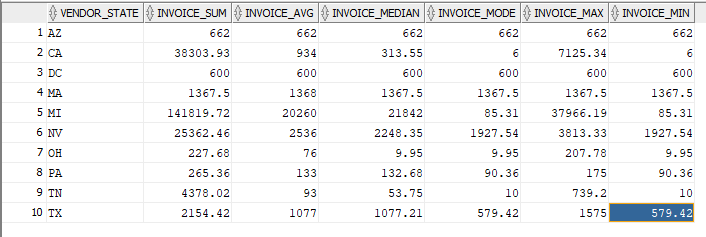
max(invoice\_total) as invoice\_max, min(invoice\_total) as invoice\_min

from vendors, invoices

where vendors.vendor\_id = invoices.vendor\_id

group by vendor\_state;”

“select \* from state\_view;”



2-(25 points) related topics :view, predefined\_functions, group by.

Create a **view** that list all vendors names and the average\_months\_between the invoice\_due\_date and the invoice\_date. Round the “average\_months\_between” to one decimal point e.g. (99.9). Filter your results to only show those vendors that the “average\_months\_between” is greater than or equal 1.5 months. Then, sort the result set in descending order by the average\_months\_between.

Hint: the round of 12.567 to one decimal point can be obtained using ROUND(12.567,1) = 12.6

Submit the following:

* The code to create the view
* A screenshot to show the data in the view.

“create or replace view Average\_Months

as

select vendor\_name, round(avg(months\_between(invoice\_due\_date, invoice\_date)),1) as average\_months\_between

from vendors, invoices

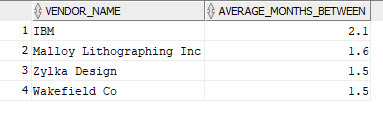
where vendors.vendor\_id = invoices.vendor\_id

and 1.5 <= (select round(avg(months\_between(invoice\_due\_date, invoice\_date)),1) from invoices where invoices.vendor\_id = vendors.vendor\_id)

group by vendor\_name

order by average\_months\_between desc;”

“select \* from average\_months”



3-(25 points) related topics : anonymous block program unit, implicit cursor, bind\_variables, user\_defined\_exception.

Write an **anonymous block program unit** that display the number of invoices in invoices table for a specific state which is entered by the user (bind variable). The number of invoices should be only for those invoices that are due ( payment\_date= null) and its invoice total exceeds $500 (like >$500). The printed message should be like **" # invoices in the state of % are due and exceed $500 dollars"** where # is 'count value' and the % is the user-input state.

You will define an exception called 'ZERO\_COUNT'. The program raises the exception"ZERO\_COUNT" if the count of invoices retrieved =0. and print the message **"No invoices that are due and exceeds $500 were found for the state of %"** where % is to be replaced with the user-input state.

Hint:

* Use implicit cursor only.
* Do not use "no\_data\_found" exception. It will not be triggered if you used it
* Declare only one variable to hold the count and one exception called "ZERO\_COUNT"

Try three cases:

State: “CA”

State: “MD”

State: “MI”

Submit the following:

* The code to create the program
* A screenshot to show the message for each test case.

“set serveroutput on;

declare

no\_data exception;

bind char(2);

ct number;

begin

bind := :bind;

select count(invoice\_id) into ct

from invoices , vendors

where vendor\_state = bind

and invoices.vendor\_id = vendors.vendor\_id

and payment\_date is null

and invoice\_total > 500;

if ct > 0 then

dbms\_output.put\_line(ct || ' invoices in the state of ' || bind ||' are due and exceed $500 dollar');

else

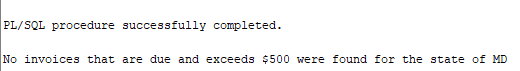
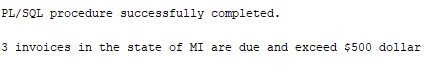
raise no\_data;

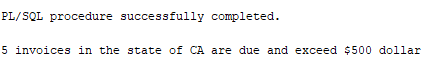
end if;

exception when no\_data then

dbms\_output.put\_line('No invoices that are due and exceeds $500 were found for the state of ' || bind);

end;”





4-(25 points) related topics : anonymous block program unit, explicit cursor, bind\_variable, reference variable rowtype, iterative logic , Exceptions.

Write an **anonymous block program unit** that display vendors names, full address (address1-city-state-zipcode) for those vendors who live in a user-input city.

Try two cases (**case sensitive**):

City: Chicago

City: Washington

Submit the following:

* The code to create the program
* A screenshot to show the results for each test case.

“set serveroutput on;

declare

no\_city exception;

city varchar(30) := :city;

ct number := 0;

cursor vendor\_cursor is

select vendor\_name, vendor\_address1, vendor\_address2, vendor\_city, vendor\_state, vendor\_zip\_code

from vendors

where vendor\_city = city;

vendor\_row vendor\_cursor%rowtype;

begin

for vendor\_row in vendor\_cursor

loop

dbms\_output.put\_line('Vendor Name: ' || vendor\_row.vendor\_name ||

' Vendor Address: ' || vendor\_row.vendor\_address1 || '

' || vendor\_row.vendor\_address2 || ' ' || vendor\_row.vendor\_city || '

' || vendor\_row.vendor\_state || ' ' || vendor\_row.vendor\_zip\_code);

ct := ct + 1;

end loop;

if ct = 0 then

raise no\_city;

end if;

exception when no\_city then

dbms\_output.put\_line('City ' || city || ' not found');

end;”

