**CSIT 2520 – Lab 7 Practice with Simple Regular Expressions (25 pts.)**

This lab provides practice using regular expressions as described in Chapter 4 of your Price textbook. From the online course, you will also need to download the script that creates movie rental tables (Movie Rental Schema), extract the script from the zip file and run it in SQL Plus or SQL Developer. Review the script file to familiarize yourself with the new tables you’ve created (recommended: create an ERD of the relationship between tables).

**To receive credit: Submit responses in electronic form. Copy this document file to your class disk. Put your name at the top. Do all requested tasks; answer all questions in this document in bold and submit when completed.**

1. Using the Movie Rental Schema, run the following queries.

a.

select phone, regexp\_substr (phone, '([[:digit:]]{3})([[:digit:]]{3})([[:digit:]]{4})')

as "Phone Number" from video\_customer;

**PHONE Phone Number**

**-------------- --------------**

**(202) 234-3223**

**(202) 111-2222**

**(202) 222-1111**

**(202) 212-1212**

**7053567712 7053567712**

**7037213725 7037213725**

**7038547854 7038547854**

**7055435712 7055435712**

**7037213324 7037213324**

**7038542154 7038542154**

**7038659856 7038659856**

b.

select phone, regexp\_substr (phone, '705([[:digit:]]{3})([[:digit:]]{4})')

as "Phone Number" from video\_customer;

**PHONE Phone Number**

**-------------- --------------**

**(202) 234-3223**

**(202) 111-2222**

**(202) 222-1111**

**(202) 212-1212**

**7053567712 7053567712**

**7037213725**

**7038547854**

**7055435712 7055435712**

**7037213324**

**7038542154**

**7038659856**

14 rows selected

What is the difference in the result returned in b. versus a. (be very precise)?

**It writes the phone numbers that only have 10 numbers, no symbols & start with the numbers ‘705’ into the second column labeled Phone Number**

c. Now try

select regexp\_replace

(phone, '([[:digit:]]{3})([[:digit:]]{3})([[:digit:]]{4})', '(\1) (\2)-(\3)')

as "Modifed Phone#"

from video\_customer;

**Modified Phone #**

**----------------------------------------------------------------------------- (202) 111-2222**

**(202) 222-1111**

**(202) 212-1212**

**(705) (356)-(7712)**

**(703) (721)-(3725)**

**(703) (854)-(7854)**

**(705) (543)-(5712)**

**(703) (721)-(3324)**

**(703) (854)-(2154)**

**(703) (865)-(9856)**

**14 rows selected**

**> -- Replace in regexp\_replace is calling for the first argument to be replaced by the**

**> -- second argument/ 10 digit phone numbers without symbols get formatted**

**> -- using the backreferences in the second argument to put parenthesis around all three**

**> -- number references and to add a space between**

**> -- backreference 1 and 2 and a dash between backreference 2 and 3.**

Look at the end of the second line of this query. Which clause in the statement caused the replacement to occur?

Did this replacement change the address table in any way? Why or why not?

**> --I'm assuming you mean does it change the original table that this select statement is querying. If so, the answer is no. This is just a query, it is not a DML statement.**

Why do the first 4 phone numbers look different from the other modified phone numbers?

**> --The first 4 addresses do not get changed because they do not match the pattern of 10 digit phone number without characters, so the replace is not done.**

d. Modify query c. to replace the middle three numbers with the prefix 694.

Paste your query and results here. (hint: focus on the \2 part of the last regexp\_replace argument)

-- Sorry, my OCD got the best of me and I had to take the extra parenthesis

-- out too. I’m hoping my ability to manipulate the code will equal out with

-- my inability to follow instructions exactly.

**> select regexp\_replace**

**(phone, '([[:digit:]]{3})([[:digit:]]{3})([[:digit:]]{4})', '(\1) 694-\3')**

**as "Modified Phone #"**

**from video\_customer**

**Modified Phone #**

**----------------------------------------------------------------------------- (202) 234-3223**

**(202) 111-2222**

**(202) 222-1111**

**(202) 212-1212**

**(705) 694-7712**

**(703) 694-3725**

**(703) 694-7854**

**(705) 694-5712**

**(703) 694-3324**

**(703) 694-2154**

**(703) 694-9856**

**14 rows selected**

2. Now try the following:

a.

select phone from video\_customer

where regexp\_instr(Phone,'7',4,2) > 0;

PHONE

--------------

7053567712

7037213725

Explain what this query has done for us?

**> The query displays phone from video\_customer if the phone number contains two 7's after the third letter in string.**

b. This next query builds on example 2a.Try it.

select regexp\_replace(Phone, '7','#',4,2) "Match Test"

from video\_customer

where regexp\_instr(Phone, '7',4,2) > 0;

Paste a copy of your query results here. Explain what you are seeing in the results of this query?

Match Test

--------------------------------------------------------------------------

7053567#12

7037213#25

> -- If the Phone field has a second 7 that occurs after the 3rd character,

> -- that 7 is replaced with a '#' character and displayed in a column labeled ‘Match Test’ with the '#' replacement showing.

3. **Use the REGEXPR\_LIKE function** to perform the following queries. Paste your working query under each specification.

1. Display the last name and phone number of all video rental customers that have an area code of ‘202’. Be sure your query won’t accidentally change the value 202 if it appears in the middle of the phone number.

**> select cus.last\_name, cus.phone**

**from video\_customer cus**

**where regexp\_like (Phone, '^202|(202)')**

**LAST\_NAME PHONE**

**------------------------------ --------------**

**Moore (202) 234-3223**

**Campbell (202) 111-2222**

**Smith (202) 222-1111**

**Richard (202) 212-1212**

b. Display the last name, first name and phone number of all video rental customers who have the string ‘ON’ in their last name. Use a case insensitive match.

**> --b**

**> select cus.last\_name, cus.first\_name, cus.phone**

**from video\_customer cus**

**where regexp\_like (cus.last\_name, '^on|%on%|on$','i')**

**LAST\_NAME FIRST\_NAME PHONE**

**------------------------------ ------------------------------ --------------**

**Milton Henry 7037213725**

**Jackson Cindy 7038547854**

**Johnson Michelle 7038659856**

c. Referring to the example in the middle of page 123 of your Price text (REGEXP\_LIKE function example using dates), display the customer id, full name and date of birth for all video customers born between 1964 and 1978.

> select cus.id, (cus.first\_name || ' ' || cus.last\_name) as "FULL NAME", cus.DOB

from video\_customer cus

where regexp\_like (to\_char(dob,'YYYY'), '^196[4-9]|197[0-8]$')

ID FULL NAME DOB

---------- ----------------------- ---------

10 John Moore 12-MAR-65

30 Robert Smith 31-MAY-66

230 Rebecca Andrews 12-MAR-69

329 Denning Jeffrey 15-MAR-64

330 Arthur Mayer 11-JUN-73

331 Paul Baker 19-DEC-69

332 Michelle Johnson 21-MAR-66

7 rows selected