Regular Expressions

snowflake

ANALYTICS WITH ANAND

Agenda

- Simple Pattern Matching
- What are Regular Expressions
- META Characters
  - Examples
- Regular Expression Function
  - Examples

Simple Pattern Matching

LIKE Operator

%(multiple characters)

_ (single character)

Simple Pattern Matching

LIKE Operator with “%”

```
SELECT first_name, last_name
FROM employees
WHERE last_name LIKE 'Ba%'
```

	FIRST_NAME	LAST_NAME
1	Hermann	Baer
2	Shelli	Baida
3	Amit	Banda
4	Elizabeth	Bates

Simple Pattern Matching

LIKE Operator with '_'

```
SELECT first_name, last_name
FROM employees
WHERE last_name LIKE 'Ba_da'
```

	FIRST_...	LAST_NAME
1	Shelli	Baida
2	Amit	Banda

Simple Pattern Matching

What if you needed to find all words where every second character is a vowel?

Regular Expressions

What are Regular Expressions?

- Methods of describing both simple and complex patterns for searching and manipulating
- Uses META Characters for pattern matching
- Oracle's implementation is an extension of the POSIX (Portable Operating System for UNIX)

META Characters

Symbol	Description
^	Marks the start of a line
$	Marks the end of a line
[ ]	Matching list
|	Operator for specifying alternative matches (logical OR)
?	Matches zero or one occurrence
.	Matches any character except NULL
{m}	Matches exactly *m* times
{m,n}	Matches at least *m* times but no more than *n* times
[: :]	Specifies a character class and matches any character in the class
\	Escape character

META Characters

Symbol	Description
+	Matches one or more occurrences
*	Matches zero or more occurrences
()	Grouping for expression
\n	Back-reference expression

META Characters

Examples

^ Beginning of a line

Example: ^(Oracle)	Match
Oracle Open World	√
The Oracle at Delphi	X
Oracle	√

$ End of a line

Example: (Oracle)$	Match
Welcome to Oracle	√
The Oracle at Delphi	X
Oracle	√

(string1 | string2)logical OR

Example: Ste(v|ph)en	Match
Stephen	√
Stefan	X
Steven	√

.(dot) Single character match

Example: re.d	Match
read	√
rear	X
reed	√

. Breakdown (re.d)

Character	Check	Success
r	**r**ead **r**ear **r**eed	√ √ √
e	r**e**ad r**e**ar r**e**ed	√ √ √
. (A-Z,a-z)	re**a**d re**a**r re**e**d	√ √ √
d	rea**d** rea**r** ree**d**	√ X √

{m} Matches exactly *m* times

Example: s{2}	Match
password	√
sister	X
essential	√

{m} Breakdown (s{2})

password	Check	sister	Check
p	X	s	√
a	X	i	X
s	√	s	√
s	√	t	X
w	X	e	X
o	X	r	X
r	X		
d	X		

*(star) Matches zero or more

Example: ab*c	Match
abc	√
acc	X
ac	√

* Breakdown (ab*c)

abc	Check	acc	Check
a	√	a	√
b	√	c	X
c	√	c	X

Regular Expression Functions

- Set of SQL functions used to search and manipulate strings using Regular Expressions
- These functions can be used on any data type that holds character data (CHAR, VARCHAR, CLOB, etc)
- The Regular Expression must be enclosed in single quote marks

Regular Expression Functions

Function name	Description
REGEXP_LIKE	Similar to the LIKE operator but allows for the use of regular expressions in matching
REGEXP_REPLACE	Search and replace text using regular expression pattern
REGEXP_INSTR	Searches for a string using regular expression pattern and returns the position when match is found
REGEXP_SUBSTR	Searches for a string using regular expression pattern and returns the matched substring
REGEXP_COUNT	Returns the number of times a pattern appears in the string.

REGEXP_LIKE

Similar to the LIKE operator but allows for the use of regular expressions in matching

```
SELECT first_name as "First Name"
, last_name as "Last Name"
FROM hr.employees
where first_name LIKE 'S%'
```

	First Name	Last Name
1	Sundar	Ande
2	Shelli	Baida
3	Sarah	Bell
4	Shelley	Higgins
5	Steven	King
10	Sarath	Sewall
11	Stephen	Stiles
12	Sigal	Tobias
13	Shanta	Vollman

REGEXP_LIKE

All employees first name of Steven or Stephen

```
SELECT first_name as "First Name"
, last_name as "Last Name"
, hire_date as "Hire Date"
FROM hr.employees
WHERE REGEXP_LIKE (first_name, '^Ste(v|ph)en$');
```

	First Name	Last Name	Hire Date
1	Steven	King	17-JUN-03
2	Steven	Markle	08-MAR-08
3	Stephen	Stiles	26-OCT-05

REGEXP_LIKE

All employees first name of Steven or Stephen

```
WHERE REGEXP_LIKE (first_name, '^Ste(v|ph)en$');
```

Meta Character	Description
^	Start of the string
Ste	Beginning letters of the string
(	Starts the group
v	Is next character a 'v'
|	OR
ph	Are next characters 'ph'
)	End the group
en	Ending letters of string
$	End of the string

REGEXP_REPLACE

Search and replace text using regular expression pattern

```
SELECT PH_NUM
FROM phone_number;
```

	PH_NUM
1	404.777.9311
2	404.867.5309
3	404.436.3566
4	505.555.5555

REGEXP_REPLACE

Reformat phone number from ###.###.#### to
1 (###)-###-####

```
SELECT ph_num, REGEXP_REPLACE(ph_num,
'([[:digit:]]{3})\.([[:digit:]]{3})\.([[:digit:]]{4})',
'1 (\1)-\2-\3') RESULT
FROM phone_number
```

	PH_NUM	RESULT
1	404.777.9311	1 (404)-777-9311
2	404.867.5309	1 (404)-867-5309
3	404.436.3566	1 (404)-436-3566
4	505.555.5555	1 (505)-555-5555

REGEXP_REPLACE

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

Meta Character	Description
[[:digit:]]{3}	Three digits (group 1)
\.	Then a '.' (Since the '.' is a META Character, we have to use the \ to 'escape' it)
[[:digit:]]{3}	Three digits (group 2)
\.	Then a '.'
[[:digit:]]{4}	Four digits (group 3)

REGEXP_REPLACE

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

Replacements	Description (sample 404.777.9311)
1	Start with a 1
(\1)	Enclose group 1 in () -> (404)
-	Add a '-'
\2	Group 2 -> 777
-	Add a '-'
\3	Group 3 -> 9311
RESULT	1 (404)-777-9311

REGEXP_INSTR

Searches for a string using regular expression pattern and returns the position when match is found

```
SELECT address
, city
FROM address
```

	ADDRESS	CITY
1	Woodshire Trail	Atlanta
2	2809 Elkmont Ridge	Atlanta
3	2900 Elkmont Ridge	Atlanta
4	Elkmont Ridge	Atlanta
5	100 Green Valley Cir	Atlanta
6	720 Green Valley Cir	Atlanta

REGEXP_INSTR

Search for addresses that don't start with a number and list the position of the first non-alpha character

```
SELECT address,
REGEXP_INSTR (address, '[^[:alpha:]]') as "RegExp Location"
FROM hr.address
where REGEXP_INSTR (address, '[^[:alpha:]]')>1
```

	ADDRESS	RegExp Location
1	Woodshire Trail	10
2	Elkmont Ridge	8

snowflake

ANALYTICS WITH ANAND

REGEXP_INSTR

Search for addresses that don't start with a number and list the position of the first non-alpha character

```
where REGEXP_INSTR (address, '[^[:alpha:]]')
```

Meta Character	Description
[	Start of the expression
^	Starts with
[:alpha:]	Alpha characters
]	End of the expression

REGEXP_SUBSTR

Searches for a string using regular expression pattern and returns the matched substring

```
SELECT * FROM positions
```

	POSITION
1	10.Administration.1700
2	20.Marketing.1800
3	30.Purchasing.1700
4	40.Human Resources.2400
5	50.Shipping.1500
6	60.IT.1400
7	70.Public Relations.2700
8	80.Sales.2500
9	90.Executive.1700
10	100.Finance.1700
11	110.Accounting.1700
12	120.Treasury.1700
13	130.Corporate Tax.1700
14	140.Control And Credit.1700

REGEXP_SUBSTR

Only return POSITION department name

```
SELECT REGEXP_SUBSTR (position, '(\.)([A-z-]+)(\.)'
,1 ,1 ,'i' ,2) as "Pos Dept"  from positions
```

	Pos Dept
1	Administration
2	Marketing
3	Purchasing
4	Human Resources
5	Shipping
6	IT
7	Public Relations
8	Sales
9	Executive
10	Finance
11	Accounting
12	Treasury
13	Corporate Tax
14	Control And Credit

REGEXP_SUBSTR

Only return POSITION department name

```
REGEXP_SUBSTR (position, '(\.)([A-z-]+)(\.)' ,1 ,1 ,'i' ,2)
```

Meta Character	Description
\.	Escape the '.'
[A-z-]+	Matches one or more occurrences of alpha characters
\.	Escape the '.'
1	Starting position
1	Nth occurrences
'i'	Ignore case
2	Sub-expression to return

REGEXP_COUNT

Returns the number of times a pattern appears in the string.

- Scenario 1: DNA sequence of a mouse. Need to find the number of times the sequence of Cytosine, Adenine, Thymine (cat) proteins occur.

- Scenario 2: All rows where there is an ‘i’ in the first name

REGEXP_COUNT

Scenario 1

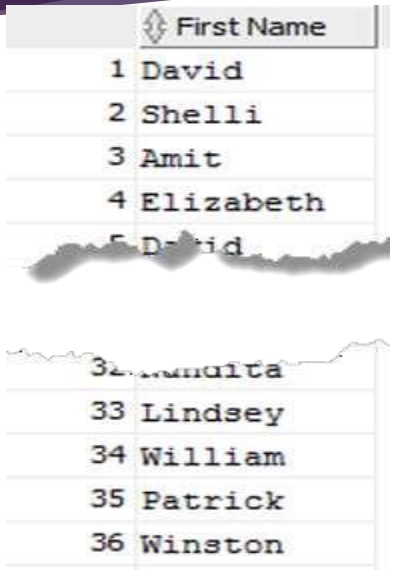
```
SELECT REGEXP_COUNT ('ccacctttccctccactcagttctcacctgtaaagcgtccctccctcatc
ccatgccccctttaccctgcagggtagagtaggctagaaaccagaga
gctccaagctccatctgtggagaggtgccatccttgggctgcgagaga
ggagaatttgcccaaagctgcctgtttgaacgatggagacatgattgc
ccgtaaagggtcctgaatgcatgagatgtctttcgagagtaccggttac
gggttaaaaggtcatgagacttcgatcattacgatcgtggttaacacac
atatgagtatagagacacattggccaagagttgagattgagag', 'cat') as "cat count"
from dual;
```

	cat count
1	9

REGEXP_COUNT

Scenario 2

```
SELECT first_name as "First Name"
FROM hr.employees
WHERE REGEXP_COUNT (first_name, 'i') > 0;
```

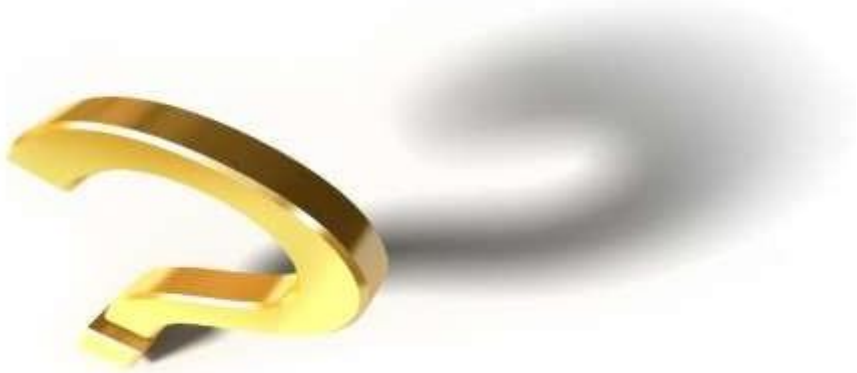


	First Name
1	David
2	Shelli
3	Amit
4	Elizabeth
5	[illegible]
32	[illegible]
33	Lindsey
34	William
35	Patrick
36	Winston

snowflake

ANALYTICS WITH ANAND

Q/A



snowflake

ANALYTICS WITH ANAND