Курс "Новые возможности для разработчиков в Oracle 10G"

Регулярные выражения

ORAMASTER

Регулярные выражения

- Регулярное выражение – выражение, задающее шаблон для поиска и замены в текстовых данных
- Применение - проверка данных, поиск дубликатов или лишних пробелов, разбор строки, проверки формата телефонного номера, email-адресов, IP-адресов, имен файлов и директорий и т.д.
- Можно задать как простой, так и очень сложный поиск
- Реализация стандарта POSIX ERE плюс некоторые расширения и многоязычность
- Условие: **`regexp_like`** (в PL/SQL – булевская функция)
- Функции:
  - **`Regexp_substr`**
  - **`Regexp_instr`**
  - **`Regexp_replace`**

ORAMASTER

Начнём-с

- Метасимволы – много. Очень :)
- Один и тот же метасимвол в разных местах может иметь разное значение:

```
select last_name from employees
  where regexp_like(last_name, '^O')
```

```
select last_name from employees
  where regexp_like(last_name, 'a[^j]s')
```

- Метасимвол «любой символ» - точка
- Отличие от оператора like – не нужно обрамлять знаками %, если ищем подстроку:
  - найти фамилии с буквой «о»
    - `where last_name like '%o%'`
    - `where regexp_like(last_name,'o')`

Выражение	Эквивалент	Что ищем	да	нет
[аб]	**(а|б)**	а или б	барс лаг	кот
н+	**н{1,}**	букву н один или более раз	ночь длинна	утро
(та)		слог та	тра-та-та капуста	парурам
(та){2}	**тата**	слог та 2 раза подряд	татами	ом тат сат
поч?ёт	**по(ч)?ёт** **по[ч]?ёт** **(почёт|поёт)**	почёт или поёт (букву ч ноль или один раз)	почёт поёт	полёт
ет$		**ет** в конце строки	свет	Света
ро		**ро** в любом месте строки	ров кров перо	кора
^ро		**ро** в начале строки	рост	корова
[^опрст] **???**		фамилии, в которых нет указанных букв	Бах Ким	Петров

Выражение в квадратных скобках (Bracket Expression)

Внутри квадратных скобок точка, круглые скобки – просто символы. Чтобы найти в тексте], поставьте её вслед за открывающей скобкой

[ ]abc] [ ][abc]

- [аб] а или б
- [абву] а или б или в или у
- [1234567] либо [1-7] - цифры с 1 по 7
- [а-я] все символы с а по я, согласно NLS_SORT
- [а-яА-Я] все символы с а по я, с А по Я
- [а-вжис-я] все символы с а по в, ж, и, символы с – я
- [ ] пробел
- [-] пробел или тире (тире, как символ, а не разделитель диапазона)
- [^ы] не «ы» (крышка в первой позиции означает отрицание)
- [^чёт] не «ч» и не «ё» и не «т»
- [^a-zA-z] искать всё, кроме a-z A-Z
- [^-] искать пробел, крышку или минус

ORAMASTER

Выражение в круглых скобках (Subexpression)

- (привет)
- пру(т|д)
- лет(и(т)?|(ел)|(аю))
- back referencing

Метасимвол «ИЛИ» (alternation symbol)

- | - Вертикальная черта

Пятач(ок|ка|ком)

Квантификаторы (Quantifiers)

a?	Совпадает 0 или 1 раз
a*	Совпадает 0 или более раз
a+	Совпадает один или более раз
a{m}	Совпадает в точности m раз
a{m,}	Совпадает m раз или более
a{m,n}	Совпадает не менее m раз и не более n раз

- **[ ]{2,}** найти два и более пробела подряд
- **[4678] {3}** найти в строке три цифры подряд из данного списка
- **рука?** найти строку «рук», возможно с «а» в конце

Якорные метасимволы (anchors)

- Крышка доллару
- ^при – в начале строки (строк, при многострочном аргументе)
- кол$ - в конце строки (строк, при многострочном аргументе)

```
select * from employees
  where regexp_like(first_name, '^сте')
```

Escape метасимвол – «\»

- Поиск точки как символа - «\.»

```
select regexp_substr('Mr S.Crashen, PhD',
'.+\.[A-Za-z]+')
  from dual;

-> Mr S.Crashen
```

ORAMASTER

Класс символов Posix (Character Class)

[:alpha:]	Буквы
[:lower:] [:upper:]	Буквы в нижнем регистре/в верхнем регистре
[:digit:] [:xdigit:]	Цифры в 10-ти/16-ти разрядной системе счисления
[:alnum:]	Буквы и цифры
[:blank:]	Пустые символы (пробел, табуляция)
[:space:]	Пробелы (не печатаемые символы), такие как перевод каретки, новая строка, вертикальная табуляция и подача страницы
[:punct:]	Знаки препинания
[:cntrl:]	Управляющие символы (не печатаемые)
[:print:]	Печатаемые символы
[:graph:]	Символы из классов [:punct:], [:upper:], [:lower:], [:digit:]

[^[:alpha:]] - найти не буквы
[^ [:digit:]] - найти не цифры

ORAMASTER

Класс равнозначности (Equivalence Class)

- Класс равнозначности – поиск всех символов у которых один и тот же базовый символ. Зависит от параметра NLS_SORT. Перед сравнением исходный текст и текст поиска конвертируются к базовым символам.

```
select * from t
  where regexp_like(z,'[[=е=]лка]')
```

- Поиск [=л=] найдёт л и Л
- Для NLS_SORT russian [=е=] найдёт е и Е
- Для других NLS_SORT найдёт еЕёЁ
- Для NLS_SORT russian [=и=] найдёт и и И , для других найдёт иИйЙ

Collation class

- **Необходим в тех языках, где одна буква может состоять из нескольких символов**
- **Установки NLS_SORT, в которых имеются такие буквы:**
- **Синтаксис [.zzz.]**
- **Поиск в диапазоне для XSPANISH: [a-[.ch.]]**
- **Поиск буквы ll в XSPANISH:**

XDANISH	aa AA Aaoe OE Oe
XSPANISH	ch CH Chll LL Ll
XHUNGARIAN	cs CS Csgy GY Gyly LY Lyny NY Nysz SZ Szty TY Tyzs ZS Zs
XCZECH	ch CH Ch
XCZECH_PUNCTUATION	ch CH Ch
XSLOVAK	dz DZ Dz ch CH Ch dž DŽ Dž
XCROATIAN	lj LJ Lj nj Nj NJ dž DŽ Dž

```
select * from t
  where regexp_like(z,'[[.ll.]]')
```

ORAMASTER

Расширения Perl

Оператор	Описание
\d	Цифра
\D	Не цифра
\w	A word character. Алфавитно-цифровые символы и знак подчёркивания «_»
\W	A nonword character.
\s	Пробелы. Эквивалент [[:space:]]
\S	Не пробелы. Эквивалент [^[:space:]]
\A	Совпадение только в начале строки.
\Z	Совпадение только в конце текста или перед символом “new line” в конце строки в многострочном тексте
\z	Совпадение только в конце строки
*?	Совпадение 0 или более раз (nongreedy).
+?	Совпадение 1 или более раз (nongreedy).
??	Совпадение 0 или 1 раз (nongreedy).
{n}?	Совпадение точно n раз (nongreedy).
{n,}?	Совпадение n и более раз (nongreedy).
{n,m}?	Совпадение от n до m раз (nongreedy).

Жадность (Greediness)

- Жадность (в отношении регулярного выражения) — характеристика, указывающая на поведение регулярного выражения при обработке шаблона.
- *Жадное* регулярное выражение «стремится» захватить максимально возможный текст (например, указание «один или более символов» трактуется как «один или более, насколько возможно»).
- *Нежадное* регулярное выражение «стремится» захватить минимально возможный текст (например, указание «один или более символов» будет трактоваться как «один символ»).

ru.wikipedia.org

ORAMASTER

Син-так-сис

Regexp_like (s, p, match_parameter)

Regexp_substr (s, p, position, occurrence, match_parameter)

Regexp_instr(s, p, position, occurrence, return_option, match_parameter)

Regexp_replace (s, p, replace_str, position, occurrence, match_parameter)

match_parameter – параметр сопоставления, это один или несколько подряд текстовых литералов:

'**i**' – поиск без учёта регистра

'**c**' – поиск с учётом регистра

'**n**' – разрешает точке (.), которая является символом match-any, совпадать также и с символом newline. Если не указать этот параметр, то точка не будет соответствовать символу newline.

'**m**' – указывает, что исходный текст - многострочный. Oracle рассматривает ^ и $ как начало и конец для любой строки, а не начало первой и конец последней. Если не указать этот параметр, Oracle считает, что входной текст состоит из единственной строки.

'**x**' пропускает пробелы. По умолчанию, пробел соответствует самому себе.

пример: **where regexp_like('Строка Поиска', 'По', 'im')**

Если не указывать **match_parameter** , то:

Учитывать или нет регистр определяется параметром NLS_SORT.

Точка (.) не совпадает с символом newline.

Исходный текст рассматривается как однострочный

Опция «n» в match_parameter

- Выражение вернёт только совпадения в одной из строк многострочного исходного текста

```
select REGEXP_SUBSTR
    ('all you need'||chr(10)||'is love', 'all.*', 1,1,'')
  from dual

-> all you need
```

- Поиск совпадений продолжится и на след.строках

```
select REGEXP_SUBSTR
    ('all you need'||chr(10)||'is love', 'all.*', 1,1,'n')
  from dual

-> all you need
   is love
```

Опция «m» в match_parameter

- m - не установлено - якорные метасимволы работают только для начала и конца исходного текста

```
select REGEXP_SUBSTR
    ('all you need'||chr(10)||'is love', '^i.*', 1,1,'')
  from dual

-> NULL
```

- Поиск совпадений с якорями в начале и конце каждой строки текста

```
select REGEXP_SUBSTR
    ('all you need'||chr(10)||'is love', '^i.*', 1,1,'m')
  from dual

-> is love
```

Опция «x» в match_parameter

- Пропускает пробелы в регулярном выражении

```
select REGEXP_SUBSTR
     ('qwerty', 'q w .+', 1,1,'x')
  from dual;

-> qwerty
```

Особенности Regexp_instr

Regexp_instr(s, p, position, occurrence, return_option, match_parameter)

return_option – если указать 0, то возвращается номер символа, первого из совпадающих с шаблоном в строке. Если указать 1 – то номер символа, следующего за найденным шаблоном.

- **Совпадение с пустой строкой**

```
SQL> select regexp_instr('exa', 'm?') from dual;

-> 1
```

- **Последний символ**

```
SQL> select regexp_instr('Чека','ека',1,1,1)
       from dual;

-> 5
```

ORAMASTER

Смещение (Offset)

- У функций есть параметр position, позволяющий задать смещение. Символ ^ указывает на начало исходной строки, а не начало смещения.

```
SQL> select regexp_instr('dddd','^d', 2) from dual;

-> 0
```

- Такой запрос может вернуть непустой ответ, только если текст многострочный и match_parameter установлен в «m»

```
SQL> select
regexp_instr('dddd'||chr(10)||'dada','^d.+',2,1,'m')
from dual;

-> dada
```

Регистр и акценты

- Если не указан match_parameter, то чувствительность к регистру определяется параметром NLS_SORT
- Особенности:
  - Для NLS_SORT russian e<>E e<> ё
  - При NLS_SORT binary_ai или generic_m_ai выполняется сравнение без учёта регистра и диакритических знаков => эквивалентны е Е ё Ё
  - Для NLS_SORT russian: и<> й И<>Й, для других и=й И=Й

Ссылка на выражение (backreferencing)

- Результат выражения в круглых скобках сохраняется во временном буфере
- Можно обращаться повторно к результатам выражения с помощью \n – где n от 1 до 9.
- Выражения нумеруются слева направо
- Используется в основном в regexp_replace

```
SQL> select regexp_replace('да да я сказал ей ей',
'(\w+) \1','\1') from dual;

-> да я сказал ей
```

Использование в DDL

- Ограничение Check

```
alter table t add constraint t_email_format_chk
check (regexp_like(email,
'[[:graph:]]+@([[:graph:]]+\.*)+[[:alpha:]]$'))
```

- Индекс по функции

```
create index t_email_dom_fbi on t(
regexp_substr(email, '[[:alpha:]]{2,4}$'))
```

- Представление

```
create view v as
select ... regexp_substr(address,
'(у|ул|пр|пер|пл)\.?( )?.+') as street_addr
From emp_locations
```

Тип	Символ	Описание
Anchors (якоря)	^/$	Совпадает с началом / с концом строки
	|	Оператор для разделения альтернативных значений
Bracket expression	[ab] [^cd]	Совпадение с любым элементом списка, допускает диапазон значений, если крышка в начале – искать символы, кроме перечисленных
Match-any	.	Совпадение с любым символом
	\	1. Оператор escape 2. Вместе с цифрой обозначает backreference 3. Сам себя - \\ 4. Ничего не делает
Subexpression	()	Выражение в скобках
Счётчики (quantifiers)	? - 0 или 1 совпадение * - 0 или более совпадений +1 или более совпадений {m} {m,} {m,n}	
Nongreedy quantifiers	После обычного счётчика ставим знак вопроса: ?? *? +? {m}? {m,}? {m,n}?	
Backreference	\n	Ссылка на результат выражения в скобках
Класс символов	**[:alpha:] [:lower:] [:upper:] [:digit:] [:xdigit:] [:alnum:] [:blank:] [:space:] [:punct:] [:print:] [:graph:] [:cntrl:]**	
Расширения Perl	**\d цифра \D не цифра \w word chars \W nonword chars \s пробелы \S не пробелы \A якорь начала строки \Z \z якоря конца строки (\Z в т.ч. перед newline)**	
Equivalence Class	[=e=]	Поиск символов у которых один и тот же базовый символ, зависит от nls_sort
match_parameter в функциях	**'i' 'c'** – поиск без учёта / с учётом регистра **'n'** – точке можно совпадать с newline **'m'** –исходный текст – многострочный, якоря заработают для каждой строки. **'x'** – пропускать пробелы	
return_option в функции regexp_instr	Если указать 0, то возвращается номер символа, первого из совпадающих с шаблоном в строке. Если указать 1 – то номер символа, следующего за найденным шаблоном.	

Литература

- Introducing Oracle Regular Expressions. An Oracle White Paper. September 2003
- Статья *Writing Better SQL Using Regular Expressions, by Alice Rischert* - http://www.oracle.com/global/ru/oramag/dec2008/w_gen_10g_regular.html
- http://psoug.org/reference/regexp.html
- Using Regular Expressions. An Oracle By Example Series Lesson.
- Mastering Oracle SQL, 2nd Edition By Alan Beaulieu, Sanjay Mishra
- Documentation - Oracle SQL Reference *(10.2)*- B14200-01
- Documentation – Oracle Application Developer's Guide – Fundamentals 10*g Release 2* , B14251-01
- Documentation – Oracle Globalization Support Guide – 10*g Release 2* , B14225-02
- Книга «Регулярные выражения» –Дж.Фридл, O'Reilli-Питер, 2003
- Сайт Википедии – ru.wikipedia.org
- Стандарт POSIX RE: http://www.opengroup.org/onlinepubs/007908799/xbd/re.html

ORAMASTER

To be continued…

В Oracle 11G добавлены:

Функция regexp_count

Новый аргумент в функциях regexp_instr и regexp_substr

www.oramaster.ru