

# Databázové systémy a SQL

---

## Lekce 9

Daniel Klimeš

- Operátor LIKE

- zástupné znaky
  - `_` = 1 libovolný znak
  - `%` = 0 nebo n libovolných znaků
  - `ESCAPE '\'`

- Příklad:

- Pracoviště Ústí

- `SELECT * FROM sites WHERE site LIKE '%Ústí%'`

- Text obsahující znak procento

- `SELECT * FROM eav_string WHERE value LIKE '%\%%' ESCAPE '\';`

- Jednoznakové texty

- `SELECT * FROM eav_string WHERE value LIKE '_';`

- Text podobný datumu kdekoli v textu

- `SELECT * FROM eav_string WHERE value LIKE '%__._.____%';`

Regulární výraz = šablona/vzor (pattern)

- Pochází z programovacích jazyků pro zpracování textu
- Nejen pro databáze

Skládá se:

- z hledaných znaků, textu
- zástupných znaků
- kvantifikátorů
- modifikátory
- operátory

Oracle funkce:

- WHERE REGEXP\_LIKE(sloupec, 'reg. výraz')
- WHERE REGEXP\_LIKE(first\_name, '^Ste(v|ph)en\$')

Znak	Význam
.	Jakýkoliv znak
^	Začátek řetězce
\$	Konec řetězce
\d	Číslice
\D	Vše kromě číslice
\w	Písmeno, číslice, podtržítko
\W	Doplňěk k \w
\s	Bílý znak - mezera, tabulátor
\S	Doplňěk k \s

Hledání datumu:

```
SELECT value FROM eav_string
WHERE REGEXP_LIKE(value, '\d\d\.\d\d\.\d\d\d\d')
```

Znak	Význam
*	0 - n opakování
+	1 - n opakování
?	0 nebo 1 opakování
{m}	Přesně m opakování
{m,}	m nebo více opakování
{m,n}	Minimálně m, maximálně n opakování

```
SELECT value FROM eav_string
WHERE REGEXP_LIKE(value, '\d{1,2}\.\d{1,2}\.\d{2,4}')
```

Znak	Význam
i	Case insensitive hledání
c	Case sensitive hledání

```
SELECT value FROM eav_string
WHERE REGEXP_LIKE(value, 'operace','i')
AND NOT REGEXP_LIKE(value, 'operace','c')
```

Znak	Význam
[abc]	Jeden z uvedených znaků (a nebo b nebo c)
[^abc]	Libovolný znak kromě uvedených (vše kromě a b c)
(abc)	Uzavření skupiny znaků-blok
	nebo
\1	Odkaz na první blok
\	Ruší speciální význam znaku např.: „\.“ = tečka

```
SELECT value FROM eav_string
WHERE REGEXP_LIKE(value, '[0123]?\d\.[01]?\d\.\d{2,4}')
```

Dvě stejné číslice za sebou (11, 22, 33,...)

```
SELECT value FROM eav_string
WHERE REGEXP_LIKE(value, '(\d)\1')
```

Extrakce subřetězce:

REGEXP\_SUBSTR(sloupec, pattern, hledat\_od, vyskyt, modifikator)

Extrakce pozice subřetězce:

REGEXP\_INSTR(sloupec, pattern, hledat\_od, vyskyt,  
navratova\_hodnota, modifikator)

**Hledat\_od** – pořadí znaku, od kterého hledat, 1 = od začátku (default)

**Vyskyt** – kolikátý výskyt vrátit, 1 = první (default)

**Modifikátor** – c = case sensitive, i= case insensitive

**Návratová\_hodnota** – 0 = vrátí pořadí prvního znaku nalezeného  
vzoru, 1 = vrátí pořadí prvního znaku za nalezeným vzorem

```
SELECT REGEXP_SUBSTR(value, '[0123]?\d\.[01]?\d\.\d{2,4}') FROM
eav_string
WHERE REGEXP_LIKE(value, '[0123]?\d\.[01]?\d\.\d{2,4}')
```

## Konverze na datum:

```
SELECT TO_DATE(datum, 'dd.mm.yyyy') FROM (
SELECT REGEXP_SUBSTR(value, '[0123]?\d\.[01]?\d\.\d{2,4}') datum
FROM eav_string
WHERE REGEXP_LIKE(value, '[0123]?\d\.[01]?\d\.\d{2,4}'))
```

**Pokus o konverzi může selhat, pokud nejde o platné datum**

ORACLE nemá funkci, která by testovala, zda lze text konvertovat na datum, ale...



```

CREATE OR REPLACE FUNCTION STUDENT.jetodatum
    (p_str IN VARCHAR2 ,format_datumu IN VARCHAR2)
    RETURN DATE
IS
BEGIN
    RETURN TO_DATE(p_str, format_datumu);
EXCEPTION
    WHEN OTHERS
    THEN
        RETURN NULL;
END;
/

```

- PLSQL procedura/funkce může obsahovat blok výjimek (exception) , který odchyťává chyby při běhu programu

```
SELECT TO_DATE(datum, 'dd.mm.yy'), value FROM (
  SELECT REGEXP_SUBSTR(value, '[0123]?\d\.[01]?\d\.\d{2,4}') datum, value
  FROM eav_string
  WHERE REGEXP_LIKE(value, '[0123]?\d\.[01]?\d\.\d{2,4}'))
WHERE jetodatum(datum, 'dd.mm.yyyy') IS NOT NULL
```

- Lépe zpracovat zvlášť dvojciferné a 4-ciferné roky

```
SELECT
  REGEXP_SUBSTR(REGEXP_SUBSTR(value,
  '[0123]?\d\.[01]?\d\.\d{2}(\D|$)'), '[0123]?\d\.[01]?\d\.\d{2}'), value
  FROM eav_string
  WHERE REGEXP_LIKE(value, '[0123]?\d\.[01]?\d\.\d{2}(\D|$)');
```

```
SELECT
  REGEXP_SUBSTR(value, '[0123]?\d\.[01]?\d\.\d{4}'), value
  FROM eav_string
  WHERE REGEXP_LIKE(value, '[0123]?\d\.[01]?\d\.\d{4}');
```

Vrací počet výskytů vzoru:

**REGEXP\_COUNT(sloupec, pattern, hledat\_od, modifikator)**

```
SELECT REGEXP_COUNT(value, '[0123]?\\d\\. [01]?\\d\\. \\d{4}') datum,
value
FROM eav_string
WHERE REGEXP_LIKE(value, '[0123]?\\d\\. [01]?\\d\\. \\d{4}')
```

Nahrazení nalezeného vzoru za jiný text:

**REGEXP\_REPLACE(sloupec, pattern, novy\_text, hledat\_od, vyskyt, modifikator)**

vyskyt – kolikátý výskyt nahradit, 0 = všechny

```
SELECT REGEXP_REPLACE(value, '([0123]?\\d)\\. ([01]?\\d)\\. (\\d{4})', '\\3-\\2-\\1') datum,
value
FROM eav_string
WHERE REGEXP_LIKE(value, '[0123]?\\d\\. [01]?\\d\\. \\d{4}')
```

```
SELECT value,
REGEXP_SUBSTR(value, '\d.*\d') greedy,
REGEXP_SUBSTR(value, '\d.*?\d') non_greedy
FROM eav_string WHERE REGEXP_LIKE (value, '\d.*\d')
```

Znak	Význam
*	0 - n opakování
+	1 - n opakování
?	0 nebo 1 opakování
{m,}	m nebo více opakování
{m,n}	Minimálně m, maximálně n opakování

C:\Program Files\PostgreSQL\9.1\doc\postgresql\html\functions-matching.html

<http://www.postgresql.org/docs/9.1/static/functions-matching.html>

Operator	Description	Example
<code>~</code>	Matches regular expression, case sensitive	'thomas' ~ '*.thomas.*'
<code>~*</code>	Matches regular expression, case insensitive	'thomas' ~* '*.Thomas.*'
<code>!~</code>	Does not match regular expression, case sensitive	'thomas' !~ '*.Thomas.*'
<code>!~*</code>	Does not match regular expression, case insensitive	'thomas' !~* '*.vadim.*'

• `regexp_replace(string text, pattern text, replacementtext [, flags text])`