PostgreSQLでの正規表現によるテキストデータ抽出方法

[**【前提】**](#_j1f77vp13uwf) **1**

[**【基本】**](#_6ldxwrnur52u) **2**

[1.PostgreSQLでのパターンマッチ](#_f0i8tgo1i22v) 2

[1.1.POSIX正規表現](#_pgs096ob5uyc) 2

[1.2.正規表現マッチ演算子](#_9n3nta5tboa9) 2

[2.正規表現の記述方法](#_n6lax6b53h6x) 3

[2.1.量化アトムと量識別子](#_kti2odeemaye) 3

[2.2.正規表現クラス省略エスケープ](#_oidgrdgcqtoj) 4

[3.正規表現制約と関数](#_v3vpummvdf9o) 5

[3.1.正規表現制約](#_vf7hzqg51em5) 5

[3.2.正規表現制約エスケープ](#_3x9k1tt6l9ev) 5

[3.3.substring関数](#_wqch0prrxc7t) 6

[3.4.regexp\_match関数](#_dyihwxd70s9g) 6

[4.テストの仕方](#_oq4wcr11grks) 8

[**【ユースケース】**](#_s83wf1l8zuxu) **12**

[1.対象レコード検索](#_a1mhulp2psh9) 12

[1.1.単一条件検索](#_njnuwm4pavho) 12

[1.2.複数条件検索（項目追加）](#_hdpe87hxqgci) 13

[1.3.複数条件検索（対象除外）](#_cvhyv3d1ntt0) 18

[1.4.前後X文字検索](#_wa23hzd3lm6) 19

[2.対象文字列抽出](#_2y79a91pu24) 23

[2.1.substring関数による文字列抽出](#_m0lxvzgmpvls) 23

[2.2.regexp\_match関数による文字列抽出](#_cdt7nps6t747) 24

[2.3.複数の抽出条件での抽出](#_3xm59c6b0oc) 27

[**【参照元一覧】**](#_y0zuq0mqpm2h) **28**

## 

## 【前提】

当資料では以下を前提に説明する。

* SQLの基本的な構文（SELECT文やWHERE句、AND、OR、LIKE演算子）は利用できる程度のレベルを想定
* 正規表現制約（先行/後方検索制約（(?=re)など））や正規表現量指定子（最短マッチ（\*?など））を極力利用しない
* プロシージャは利用しない

## 

## 【基本】

### 1.PostgreSQLでのパターンマッチ

#### 1.1.POSIX正規表現

PostgreSQLでパターンマッチを行うには、LIKE演算子、SIMILAR TO演算子、POSIX正規表現の3パターンが存在する。POSIX正規表現とはUnixのsedやawkと類似したパターンマッチ言語を使用した正規表現で、LIKE演算子、SIMILAR TO演算子より複雑なパターンマッチが可能となる。PostgreSQLでのPOSIX正規表現はPOSIX標準以外にPerlやTclなどのプログラミング言語由来の拡張もサポートしている。

（参考）SIMILAR TOによる正規表現パターンマッチ

標準SQLの正規表現定義を使用してパターンマッチする方法としてSIMILAR TOが存在するが、拡張機能に該当するPOSIX正規表現で説明するため、これについては記述しない。

#### 1.2.正規表現マッチ演算子

正規表現マッチ演算子を用いたパターンマッチの記述方法は、「[検索対象文字列] [演算子] [正規表現]」の順で記述する。

| 演算子 | 説明 | 例（True条件） | 例（False条件） |
| --- | --- | --- | --- |
| ~ | 正規表現にマッチ、大文字小文字の区別あり | 'abc' ~ 'a' | 'abc' ~ 'A' |
| ~\* | 正規表現にマッチ、大文字小文字の区別なし | 'abc' ~\* 'A' | 'abc' ~\* 'd' |
| !~ | 正規表現にマッチしない、大文字小文字の区別あり | 'abc' !~ 'A' | 'abc' !~ 'a' |
| !~\* | 正規表現にマッチしない、大文字小文字の区別なし | 'abc' !~\* 'd' | 'abc' !~\* ‘A' |

※正規表現マッチ演算子の基本は「~」であり、

　大文字小文字を区別したくない場合は「\*」を付与する。

　否定する場合は「!」を付与すればよい。

### 2.正規表現の記述方法

正規表現は「|」で区切られた1つまたは複数のブランチとして定義される。ブランチの何れか1つにマッチすればマッチしたことになる。ブランチはゼロ個以上の量化アトムまたは正規表現制約の連結である。

#### 2.1.量化アトムと量識別子

量化アトムとは単一の量識別子が後ろにつくアトムのこと。量識別子がない場合、アトムにマッチするものがマッチしたことになる。

・正規表現のアトム

| アトム | 説明 |
| --- | --- |
| (re) | （ここでre は任意の正規表現で、）reとのマッチに適合するものです。 マッチは可能である報告用と意味づけられます。 |
| (?:re) | 上と同じ。ただし、マッチは報告用と意味づけられません。（「捕捉されない」括弧の集合）（AREのみ） |
| . | 任意の1文字にマッチします。 |
| [chars] | ブラケット式。 charsのいずれか1つにマッチします。  否定形は「[^chars]」で記載する。 |
| \k | ここでkは英数字以外です。  普通の文字として指定した文字にマッチします。 |
| \c | ここでcは英数字です （おそらく他の文字が後に続きます）。  エスケープです。 |
| { | 直後に数字以外がある場合、左中括弧{にマッチします。  直後に数字が続く場合、バウンド（後述）の始まりです。 |
| x | ここでxは他に意味を持たない1文字です。 xにマッチします。 |

※この中で理解すべき内容は、「.」（ピリオド）が任意の1文字のマッチを示す

　アトムであるということだけで良い。その他は適宜後述する。

・正規表現量識別子

| 量識別子 | 説明 |
| --- | --- |
| \* | アトムの0回以上の並びにマッチ |
| + | アトムの1回以上の並びにマッチ |
| ? | アトムの0回もしくは1回の並びにマッチ |
| {m} | アトムの正確なm回の並びにマッチ |
| {m,} | アトムのm回以上の並びにマッチ |
| {m,n} | アトムのm回以上かつn回以下の並びにマッチ |
| \*? | \*の最短マッチを行う |
| +? | +の最短マッチを行う |
| ?? | ?の最短マッチを行う |
| {m}? | {m}の最短マッチを行う |
| {m,}? | {m,}の最短マッチを行う |
| {m,n}? | {m,n}の最短マッチを行う |

※{...}を使用する形式はバウンドと呼ばれ、バウンド内のmとnは符号なし10進整数であり、

　0以上255以下の値を取ることができます。

#### 2.2.正規表現クラス省略エスケープ

| エスケープ | 説明 |
| --- | --- |
| \d | 数字（全角を含む） |
| \D | 数字（全角を含む）以外 |
| \s | スペース（全角を含む） |
| \S | スペース（全角を含む）以外 |
| \w | 文字と数字それにアンダースコア |
| \W | 文字と数字それにアンダースコア以外 |

それぞれを否定する場合、大文字にすればよい。

### 3.正規表現制約と関数

正規表現による主に文字列抽出時に有用な正規表現制約と関数について、以下の通り説明する。

1. 正規表現制約
2. 正規表現制約エスケープ
3. regexp\_match関数

#### 3.1.正規表現制約

| 制約 | 説明 |
| --- | --- |
| ^ | 文字列の先頭にマッチ |
| $ | 文字列の末尾にマッチ |
| (?=re) | 先行肯定検索：reにマッチする部分文字列が始まる任意の場所にマッチ  ※以下「re」は正規表現を示す |
| (?!re) | 先行否定検索：reにマッチしない部分文字列が始まる任意の場所にマッチ |
| (?<=re) | 後方肯定検索：reにマッチする部分文字列が終わる任意の場所にマッチ |
| (?<!re) | 後方否定検索：reにマッチしない部分文字列が終わる任意の場所にマッチ |

#### 3.2.正規表現制約エスケープ

| 制約 | 説明 |
| --- | --- |
| \A | 文字列の先頭にのみマッチ |
| \m | 単語の先頭にのみマッチ |
| \M | 単語の末尾にのみマッチ |
| \y | 単語の先頭もしくは末尾にのみマッチ |
| \Y | 単語の先頭もしくは末尾以外の場所にのみマッチ |
| \Z | 文字列の末尾にのみマッチ |

#### 

#### 3.3.substring関数

| 構文 | substring([文字列] from [正規表現]) |
| --- | --- |

regexp\_match関数はPOSIX正規表現パターンにマッチする部分文字列を抽出することができる。

#### 3.4.regexp\_match関数

| 構文 | regexp\_match([文字列], [正規表現], [フラグ])  ※[フラグ]は任意。詳細は「ARE埋め込みオプション文字」で後述。 |
| --- | --- |

regexp\_match関数はPOSIX正規表現パターンを文字列にマッチさせた結果、捕捉された最初の部分文字列のテキスト配列を返却する。フラグは「i」で大文字小文字を区別しないということは理解しておくこと。

（参考）regexp\_matches関数との違い

フラグに「g」を指定することで全ての部分文字列のテキスト配列を返却する点が異なる。

・ARE埋め込みオプション文字

| オプション | 説明 |
| --- | --- |
| b | 残りのreはBRE |
| c | 大文字小文字を区別するマッチ  （演算子で規定される大文字小文字の区別よりこの指定が優先されます） |
| e | 残りのreはERE |
| i | 大文字小文字を区別しないマッチ  （演算子で規定される大文字小文字の区別よりこの指定が優先されます） |
| m | 改行を区別するマッチ（nの歴史的な同義語） |
| n | 改行を区別するマッチ |
| p | 部分的な改行を区別するマッチ |
| q | 残りのreはリテラル（「引用符付けされた」）文字列、全て普通の文字 |
| s | 改行を区別しないマッチ（デフォルト） |
| t | 厳しめの構文（デフォルト） |
| w | 部分的な改行区別の逆（「ワイアード」）マッチ |
| x | 拡張構文 |

### 

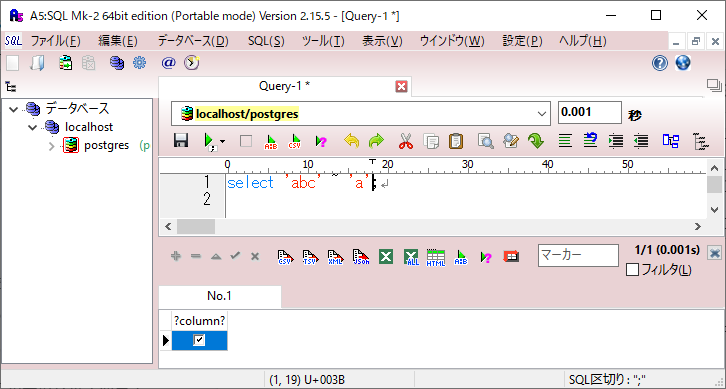
### 4.テストの仕方

select文でテストしたい正規表現を記述し、テストデータとして文字列を順次記述して確認を行う。

（例）「A」という文字列を大文字小文字区別せず検索する正規表現の作成、

　　　テストの仕方の参考例

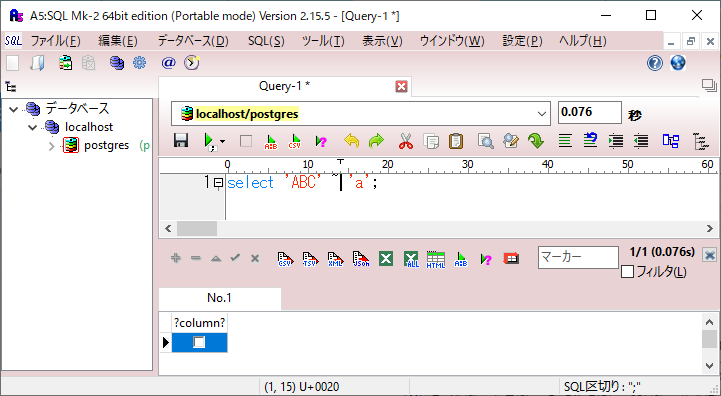
①「'abc' ~ 'a'」で正しく検索されることを確認する。



※A5SQLでTrueは上記のようにチェックボックスにチェックありで表記される。

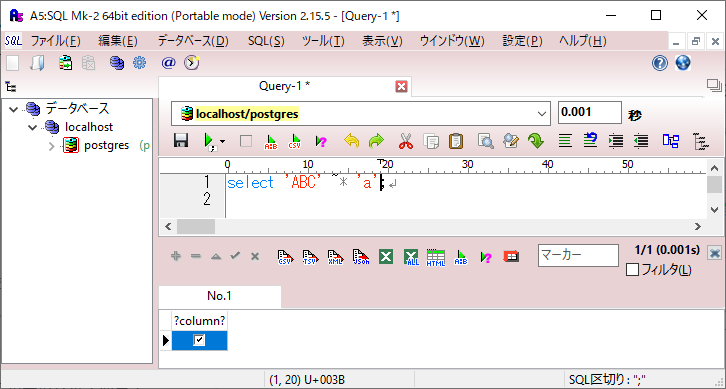
| select 'abc' ~ 'a'; |
| --- |

②「'ABC' ~ 'a'」だと検索されないため、正規表現が誤っていることを認識する。



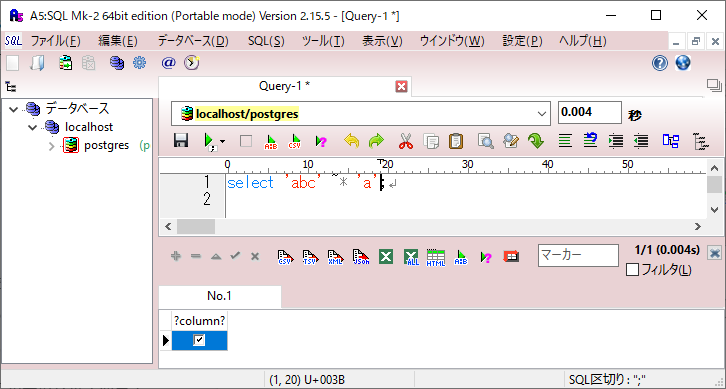
| select 'ABC' ~ 'a'; |
| --- |

③「'ABC' ~\* 'a'」で正しく検索されることを確認する。



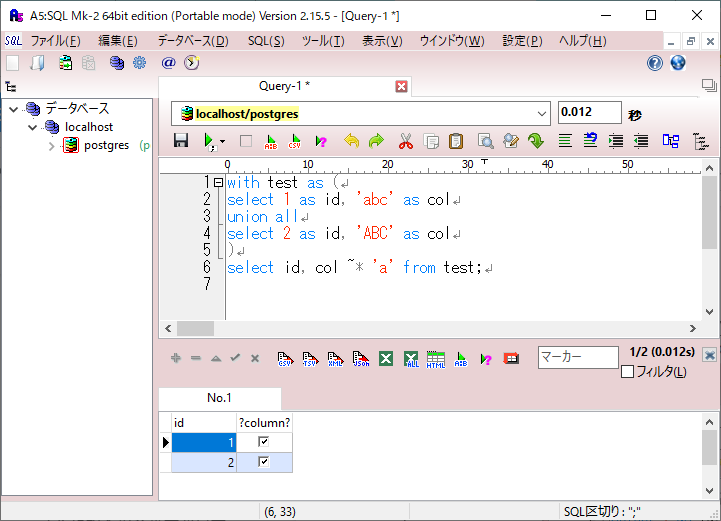
| select 'ABC' ~\* 'a'; |
| --- |

④「'abc' ~\* 'a'」でも正しく検索されることを確認する。



| select 'abc' ~\* 'a'; |
| --- |

①´テストデータをまとめてWITH句で作成して実施することもできる。



※「id」のカラムは結果を見やすくしているだけのため、記載しなくても問題ない。

| with test as (  select 1 as id, 'abc' as col  union all  select 2 as id, 'ABC' as col  )  select id, col ~\* 'a' from test; |
| --- |

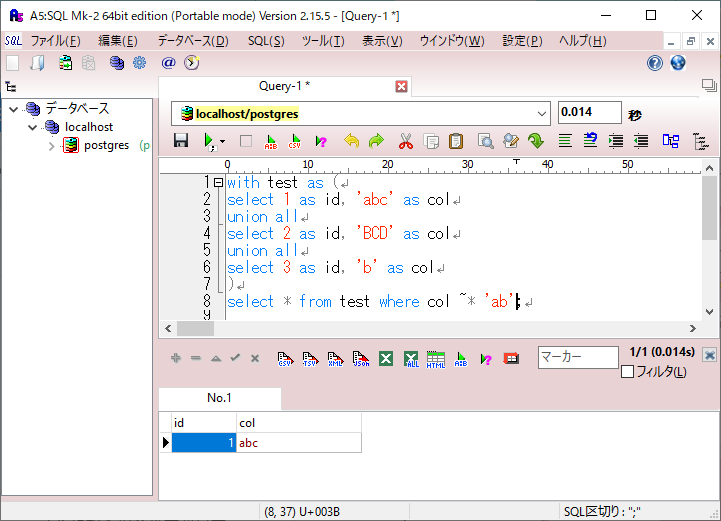
## 

## 【ユースケース】

### 1.対象レコード検索

#### 1.1.単一条件検索

WHERE句で正規表現マッチ演算子を用いて、検索を行う。



※id:1が正しく検索されている。

| with test as (  select 1 as id, 'abc' as col  union all  select 2 as id, 'BCD' as col  union all  select 3 as id, 'b' as col  )  select \* from test where col ~\* 'ab'; |
| --- |

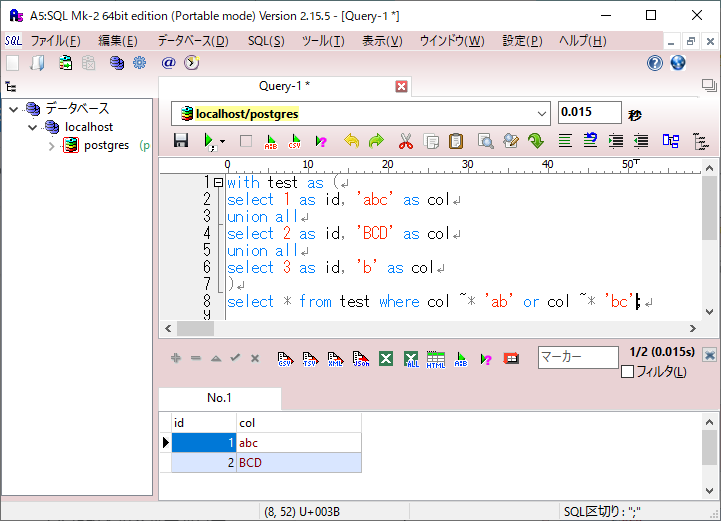
#### 

#### 1.2.複数条件検索（項目追加）

WHERE句での条件検索に項目を追加する方法としては、以下が挙げられる。

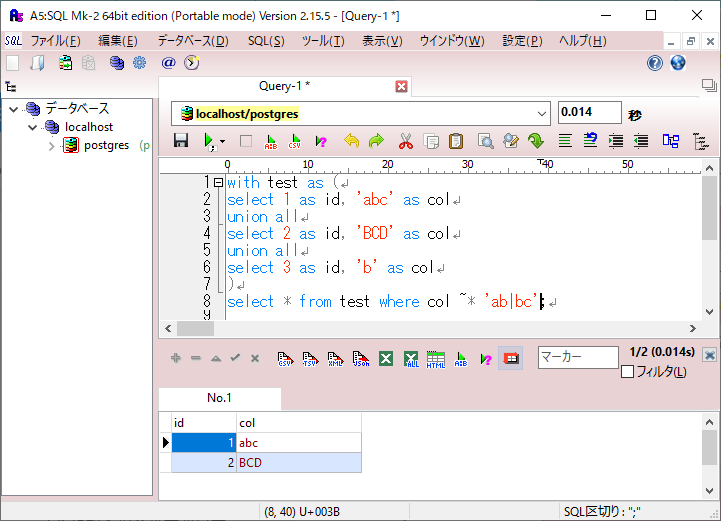
ただし（パフォーマンスを考慮しなければ、）全てOR条件で記載することも可能である。

1. OR条件
2. ブランチ「|」
3. グループ化「()」
4. OR条件で記載する。



| with test as (  select 1 as id, 'abc' as col  union all  select 2 as id, 'BCD' as col  union all  select 3 as id, 'b' as col  )  select \* from test where col ~\* 'ab' or col ~\* 'bc'; |
| --- |

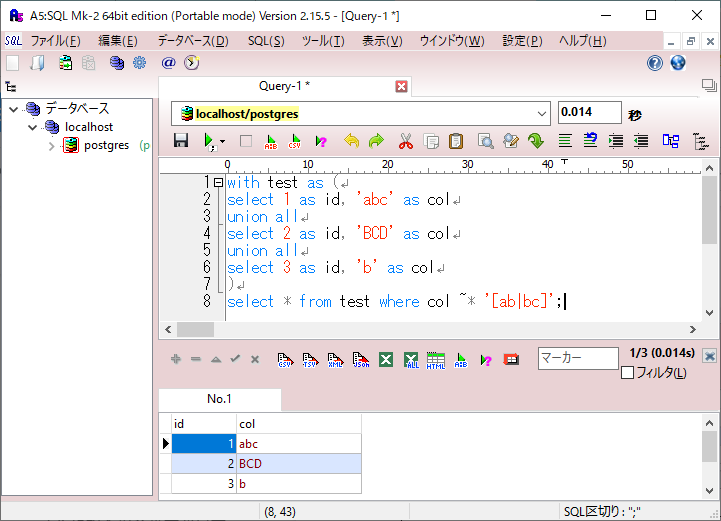
1. ブランチ「|」を用いる。



| with test as (  select 1 as id, 'abc' as col  union all  select 2 as id, 'BCD' as col  union all  select 3 as id, 'b' as col  )  select \* from test where col ~\* 'ab|bc'; |
| --- |

（注意）ブラケット式の適用誤り

ブラケット式は1文字の複数パターンをまとめたものなので注意が必要。

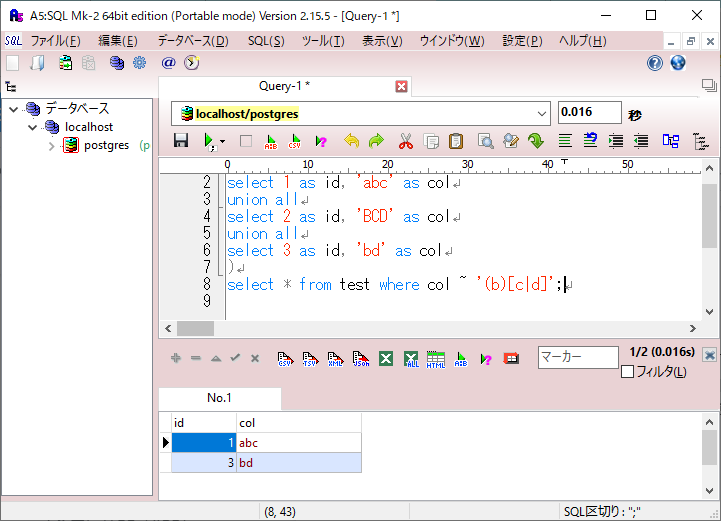


※「ab」または「bc」で検索するつもりが、id:3の「b」も抽出されてしまっている。

　上記正規表現は「a」「b」「c」「|」を検索する正規表現となっている。

| with test as (  select 1 as id, 'abc' as col  union all  select 2 as id, 'BCD' as col  union all  select 3 as id, 'b' as col  )  select \* from test where col ~\* '[ab|bc]'; |
| --- |

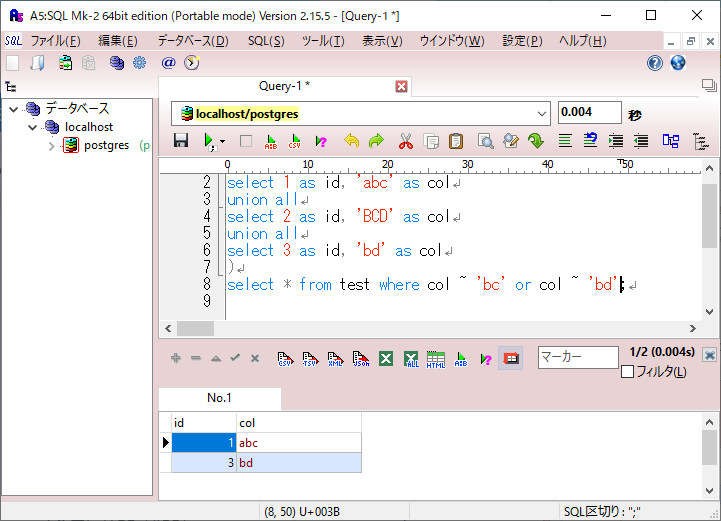
1. グループ化「()」を用いる。



※「bc」と「bd」の共通部分「b」をグループ化でまとめて記載した形式である。

| with test as (  select 1 as id, 'abc' as col  union all  select 2 as id, 'BCD' as col  union all  select 3 as id, 'bd' as col  )  select \* from test where col ~ '(b)[c|d]'; |
| --- |

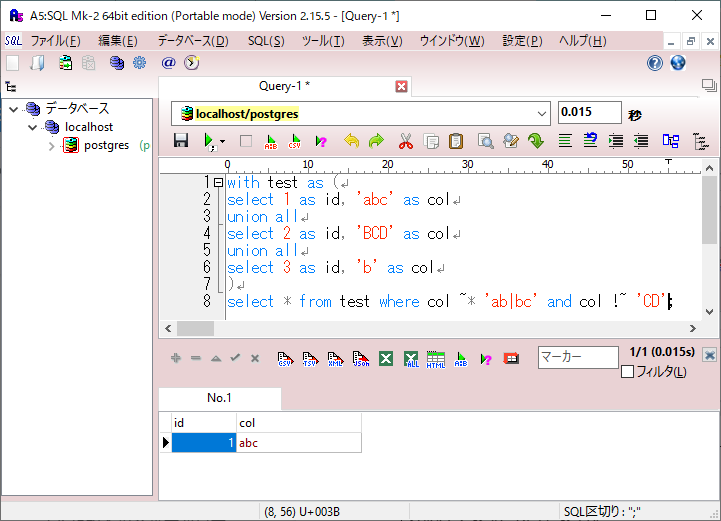
前記の内容は、以下の通り書き換えることができる。



| with test as (  select 1 as id, 'abc' as col  union all  select 2 as id, 'BCD' as col  union all  select 3 as id, 'bd' as col  )  select \* from test where col ~ 'bc' or col ~ 'bd'; |
| --- |

#### 1.3.複数条件検索（対象除外）

AND条件で否定を利用して除外する。



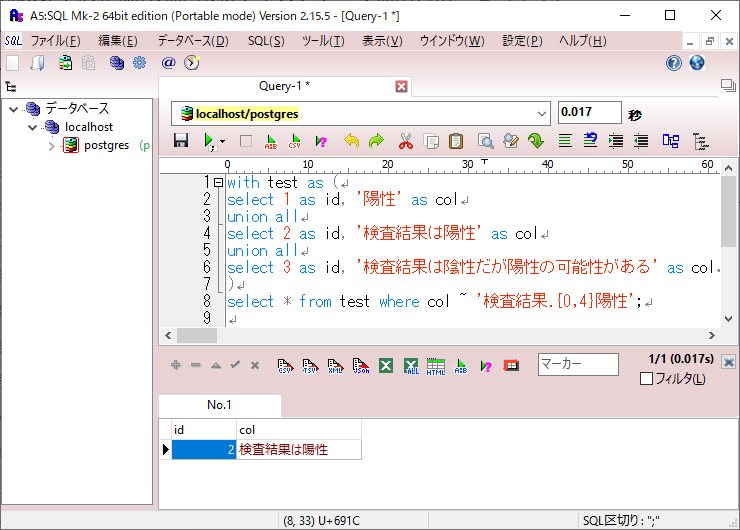
※大文字の「CD」だけは除きたい場合、AND条件で「CD」の否定形で除外している。

| with test as (  select 1 as id, 'abc' as col  union all  select 2 as id, 'BCD' as col  union all  select 3 as id, 'b' as col  )  select \* from test where col ~\* 'ab|bc' and col !~ 'CD'; |
| --- |

#### 

#### 1.4.前後X文字検索

「特定の文字列の前後X文字以内に別の特定の文字列が存在する」というような条件を検索する際は、量識別子を用いる方法が挙げられる。

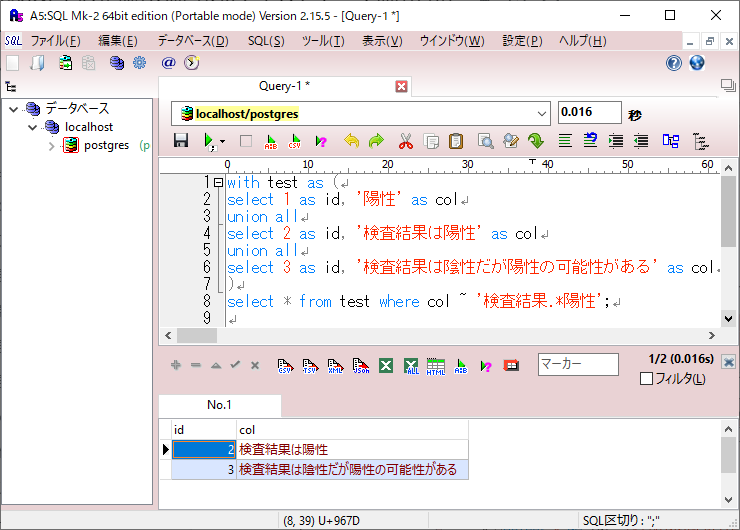


| with test as (  select 1 as id, '陽性' as col  union all  select 2 as id, '検査結果は陽性' as col  union all  select 3 as id, '検査結果は陰性だが陽性の可能性がある' as col  )  select \* from test where col ~ '検査結果.{0,4}陽性'; |
| --- |

※「検査結果」の直後の「.{0,4}」で「任意の4文字以内の文字列」が存在した後に

　「陽性」が存在するという条件になる。

（注意）バウンドで文字数を指定不要の場合



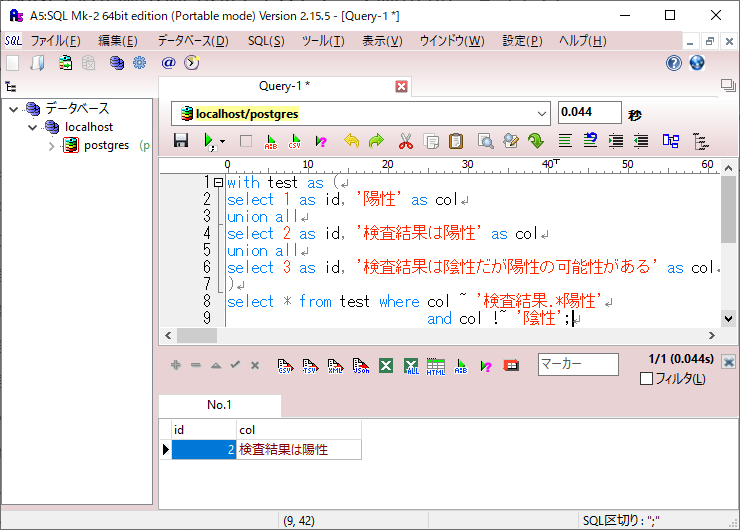
※量識別子「\*」を用いて任意の0文字以上の文字列を指定したため、

　後続の「陽性の可能性がある」という文言の「陽性」によりパターンマッチしてしまった

| with test as (  select 1 as id, '陽性' as col  union all  select 2 as id, '検査結果は陽性' as col  union all  select 3 as id, '検査結果は陰性だが陽性の可能性がある' as col  )  select \* from test where col ~ '検査結果.\*陽性'; |
| --- |

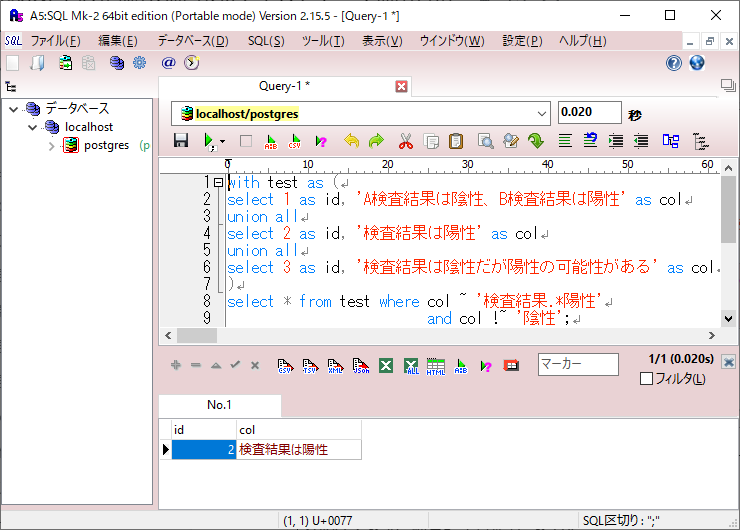
対策として、以下が挙げられる。

1. 複数条件検索（対象除外）
2. 抽出時に判定（後述）  
   ※抽出時に判定する方法が柔軟に対応できるため推奨
3. 複数条件検索（対象除外）：AND条件を利用



| with test as (  select 1 as id, '陽性' as col  union all  select 2 as id, '検査結果は陽性' as col  union all  select 3 as id, '検査結果は陰性だが陽性の可能性がある' as col  )  select \* from test where col ~ '検査結果.\*陽性' and col !~ '陰性'; |
| --- |

（注意）文章全体に1つでも「陰性」という文言が存在すると除外されてしまう点に留意



※id:1の条件も抽出したい場合は、この条件では抽出できないことになる

| with test as (  select 1 as id, 'A検査結果は陰性、B検査結果は陽性' as col  union all  select 2 as id, '検査結果は陽性' as col  union all  select 3 as id, '検査結果は陰性だが陽性の可能性がある' as col  )  select \* from test where col ~ '検査結果.\*陽性' and col !~ '陰性'; |
| --- |

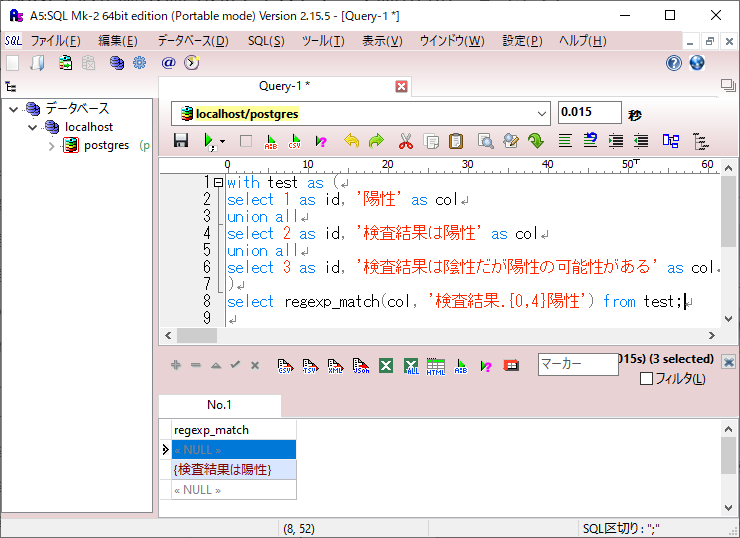
### 2.対象文字列抽出

substring関数やregexp\_match関数を利用すると、正規表現でマッチした特定の文字列を抽出することができる。

#### 2.1.substring関数による文字列抽出

| with test as (  select 1 as id, '陽性' as col  union all  select 2 as id, '検査結果は陽性' as col  union all  select 3 as id, '検査結果は陰性だが陽性の可能性がある' as col  )  select substring(col from '検査結果.{0,4}陽性') from test; |
| --- |

#### 2.2.regexp\_match関数による文字列抽出



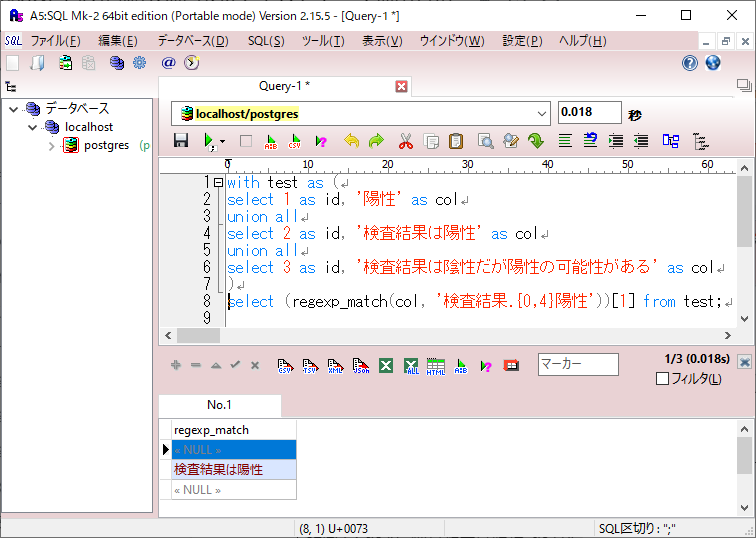
regexp\_match関数でマッチした場合、マッチした結果が配列型（{}で表示）で返却される。

マッチしなかった場合はNullが返却される。

| with test as (  select 1 as id, '陽性' as col  union all  select 2 as id, '検査結果は陽性' as col  union all  select 3 as id, '検査結果は陰性だが陽性の可能性がある' as col  )  select regexp\_match(col, '検査結果.{0,4}陽性') from test; |
| --- |

返却時の形式を文字列にしたい場合は、配列の一番目を指定するため、

以下の通り記述する必要がある。

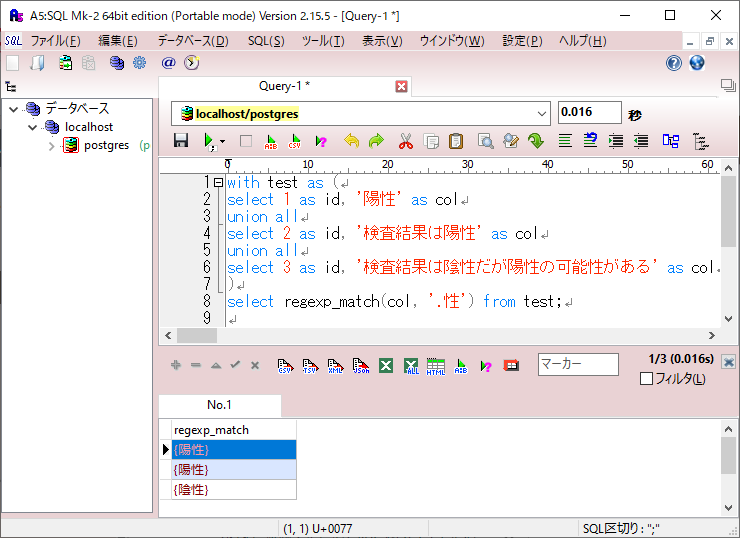


※regexp\_match関数全体を「()」で囲んだ上で、「[1]」と指定する必要がある点に留意

| with test as (  select 1 as id, '陽性' as col  union all  select 2 as id, '検査結果は陽性' as col  union all  select 3 as id, '検査結果は陰性だが陽性の可能性がある' as col  )  select (regexp\_match(col, '検査結果.{0,4}陽性'))[1] from test; |
| --- |

（注意）複数マッチした場合

substring関数、regexp\_match関数ともに最初にマッチした結果を返却する仕様となっている。両方記述があるかも判定したい場合は、複数の抽出条件での抽出（後述）を参照。

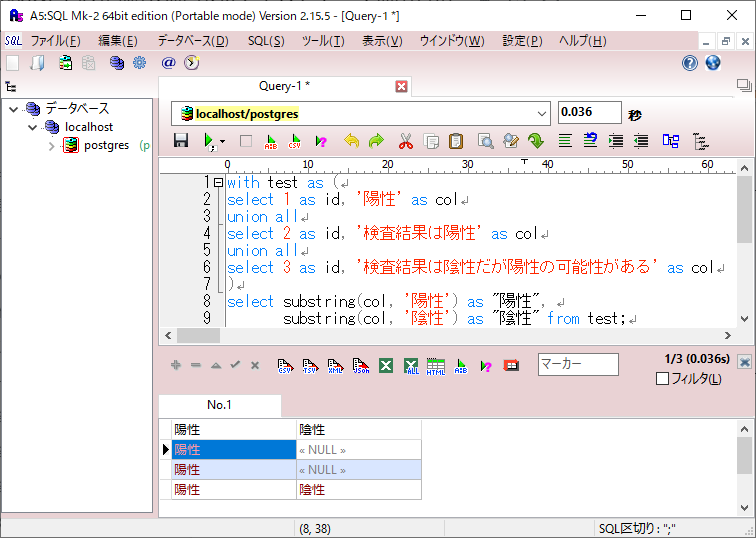


※id:3について、最初にマッチした「陰性」が返却されている。

| with test as (  select 1 as id, '陽性' as col  union all  select 2 as id, '検査結果は陽性' as col  union all  select 3 as id, '検査結果は陰性だが陽性の可能性がある' as col  )  select regexp\_match(col, '.性') from test; |
| --- |

#### 2.3.複数の抽出条件での抽出

select句で複数カラムにそれぞれ抽出したい条件を分けることで、複数の抽出条件での抽出結果を確認することができる。



| with test as (  select 1 as id, '陽性' as col  union all  select 2 as id, '検査結果は陽性' as col  union all  select 3 as id, '検査結果は陰性だが陽性の可能性がある' as col  )  select substring(col, '陽性') as "陽性",  substring(col, '陰性') as "陰性" from test; |
| --- |

## 

## 【参照元一覧】

* PostgreSQL 10.5文書 第9章 関数と演算子 9.7. パターンマッチ  
  <https://www.postgresql.jp/docs/10/functions-matching.html>
* PostgreSQLで正規表現  
  <https://qiita.com/y_ito/items/a0fb46a618b0316617c8>
* 【SQL超入門講座】19.正規表現｜複雑なパターンマッチングを条件に抽出する方法

<https://kino-code.com/sql19/>