examples/filter_regex.pql by *Pequel*

sample@youraddress.com

Filer Regex Example Script

2.2

Table of Contents
Filer Regex Example Script

SCRIPT NAME

examples/filter_regex.pql

DESCRIPTION

Demonstrates use of filter and Perl regular expressions. The reqular expression can contain Pequel field names macros and table names. This example also demonstrates the use of a simple ‘local’ table (LOC_DESCRIPT).

1. PROCESS DETAILS

Input records are read from standard input. The input record contains ***8*** fields. Fields are delimited by the ‘**|**’ character.

Output records are written to standard output. The output record contains ***11*** fields. Fields are delimited by the ‘**|**’ character.

Input stream is **sorted** by the input field ***SALES_CODE*** (*string*).

Input records are eliminated (**filtered**) unless ***LOCATION =~ /^NSW$|^WA$|^SA$/***.

Input records are **grouped** by the input field ***SALES_CODE*** (*string*).

1.1 *SALES_CODE*

Output Field

Description

Set to input field ***SALES_CODE***

1.2 *LOC_DESCRIPT*

Output Field

Description

Set to input field ***LDESCRIPT***

Derived Input Field Evaluation

```
=> %LOC_DESCRIPT(LOCATION)->1 . " in postcode " . %LOC_DESCRIPT(LOCATION)->2
```

1.3 *NUM_PRODUCTS*

Output Field

Description

Distinct aggregation on input field ***PRODUCT_CODE***.

1.5 *PROD_NUM*

Output Field

Description

Derived (calculated) field.

Derived Field Evaluation

1.6 *LOC_NSW*

Output Field

Description

Derived (calculated) field.

Derived Field Evaluation

1.7 *AVG_COST_PRICE_NSW*

Output Field

Description

Avg aggregation on input field ***COST_PRICE***.

Aggregation condition

LOCATION eq ‘NSW‘;

1.8 *LOC_WA*

Output Field

Description

Derived (calculated) field.

Derived Field Evaluation

1.9 *AVG_COST_PRICE_WA*

Output Field

Description

Avg aggregation on input field ***COST_PRICE***.

Aggregation condition

LOCATION eq ‘WA‘;

1.10 *LOC_SA*

Output Field

Description

Derived (calculated) field.

Derived Field Evaluation

1.11 *AVG_COST_PRICE_SA*

Output Field

Description

Avg aggregation on input field ***COST_PRICE***.

Aggregation condition

LOCATION eq ‘SA‘;

2. CONFIGURATION SETTINGS

2.1 *prefix*

directory pathname prefix.: examples

2.2 *pequeldoc*

generate pod / pdf pequel script Reference Guide.: pdf

2.3 *detail*

Include Pequel Generated Program chapter in Pequeldoc: 1

2.4 *script_name*

script filename: examples/filter_regex.pql

2.5 *header*

write header record to output.: 1

2.6 *optimize*

optimize generated code.: 1

2.7 *doc_title*

document title.: Filer Regex Example Script

2.8 *doc_email*

document email entry.: sample@youraddress.com

2.9 *doc_version*

document version for pequel script.: 2.2

3. TABLES

3.1 *LOC_DESCRIPT*

Table Type: ***local***

Data

NSW — New South Wales 2061 02
WA — Western Australia 5008 07
SA — South Australia 8078 08

4. TABLE INFORMATION SUMMARY

4.1 Table List Sorted By Table Name

LOC_DESCRIPT — ***1*** (*local*)

5. EXAMPLES/FILTER_REGEX.PQL

options

```
prefix(examples)
pequeldoc(pdf)
detail(1)
script_name(examples/filter_regex.pql)
header(1)
optimize(1)
doc_title(Filer Regex Example Script)
doc_email(sample@youraddress.com)
doc_version(2.2)
```

description

```
Demonstrates use of filter and Perl regular expressions. The reqular expression can
contain Pequel field names
macros and table names.
This example also demonstrates the use of a simple ’local’ table (LOC_DESCRIPT).
```

init table

```
LOC_DESCRIPT NSW New South Wales 2061 02
LOC_DESCRIPT WA Western Australia 5008 07
LOC_DESCRIPT SA South Australia 8078 08
```

input section

```
PRODUCT_CODE
COST_PRICE
DESCRIPTION
SALES_CODE
SALES_PRICE
SALES_QTY
SALES_DATE
LOCATION
LDESCRIPT => %LOC_DESCRIPT(LOCATION)->1 . " in postcode " . %LOC_DESCRIPT(LOCATION) \
->2
```

filter

```
LOCATION =~ /^NSW$|^WA$|^SA$/
```

sort by

```
SALES_CODE string
```

group by

```
SALES_CODE string
```

output section

```
string    SALES_CODE         SALES_CODE
string    LOC_DESCRIPT       LDESCRIPT
numeric   NUM_PRODUCTS       distinct PRODUCT_CODE
string    _PRODUCT_CODE      PRODUCT_CODE
string    PROD_NUM           = _PRODUCT_CODE . "-" . NUM_PRODUCTS
string    LOC_NSW            = %LOC_DESCRIPT(NSW)->1
numeric   AVG_COST_PRICE_NSW avg COST_PRICE where LOCATION eq ’NSW’
string    LOC_WA             = %LOC_DESCRIPT(WA)->1
numeric   AVG_COST_PRICE_WA  avg COST_PRICE where LOCATION eq ’WA’
string    LOC_SA             = %LOC_DESCRIPT(SA)->1
numeric   AVG_COST_PRICE_SA  avg COST_PRICE where LOCATION eq ’SA’
```

6. PEQUEL GENERATED PROGRAM

```
#!/usr/bin/perl
#-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
# vim: syntax=perl ts=4 sw=4
#-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
#Generated By: pequel Version 2.4-5, Build: Wednesday November 16 21:56:42 GMT 2005
#            : http://sourceforge.net/projects/pequel/
#Script Name : filter_regex.pql
#Created On  : Wed Nov 16 14:03:48 2005
#Perl Version: /usr/bin/perl 5.6.1 on solaris
#For         :
#-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
#Options:
#prefix(examples) directory pathname prefix.
#pequeldoc(pdf) generate pod / pdf pequel script Reference Guide.
#detail(1) Include Pequel Generated Program chapter in Pequeldoc
#script_name(examples/filter_regex.pql) script filename
#header(1) write header record to output.
#optimize(1) optimize generated code.
#doc_title(Filer Regex Example Script) document title.
#doc_email(sample@youraddress.com) document email entry.
#doc_version(2.2) document version for pequel script.
#-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
use strict;
use constant _I_PRODUCT_CODE          => int    0;
use constant _I_COST_PRICE            => int    1;
use constant _I_DESCRIPTION           => int    2;
use constant _I_SALES_CODE            => int    3;
use constant _I_SALES_PRICE           => int    4;
use constant _I_SALES_QTY             => int    5;
use constant _I_SALES_DATE            => int    6;
use constant _I_LOCATION              => int    7;
use constant _I_LDESCRIPT             => int    8;
use constant _O_SALES_CODE            => int    1;
use constant _O_LOC_DESCRIPT          => int    2;
use constant _O_NUM_PRODUCTS          => int    3;
use constant _O__PRODUCT_CODE         => int    4;
use constant _O_PROD_NUM              => int    5;
use constant _O_LOC_NSW               => int    6;
use constant _O_AVG_COST_PRICE_NSW    => int    7;
use constant _O_LOC_WA                => int    8;
use constant _O_AVG_COST_PRICE_WA     => int    9;
use constant _O_LOC_SA                => int   10;
use constant _O_AVG_COST_PRICE_SA     => int   11;
use constant _T_LOC_DESCRIPT_FLD_1   => int    0;
use constant _T_LOC_DESCRIPT_FLD_2   => int    1;
use constant _T_LOC_DESCRIPT_FLD_3   => int    2;
use constant _I_LOC_DESCRIPT_LOCATION_FLD_KEY => int    9;
use constant _I_LOC_DESCRIPT_LOCATION_FLD_1  => int   10;
use constant _I_LOC_DESCRIPT_LOCATION_FLD_2  => int   11;
use constant _I_LOC_DESCRIPT_LOCATION_FLD_3  => int   12;
local $\="\n";
local $,="|";
print STDERR '[examples/filter_regex.pql ' . localtime() . "] Init";
use constant VERBOSE => int 10000;
use constant LAST_ICELL => int 8;
my @I_VAL;
my @O_VAL;
my $_inprecs=0;
my %DISTINCT;
my %AVERAGE;
my $key__I_SALES_CODE;
my $previous_key__I_SALES_CODE = undef;
foreach my $f (1..11) { $O_VAL[$f] = undef; }
my $_TABLE_LOC_DESCRIPT = &InitLookupLOC_DESCRIPT; # ref to %$LOC_DESCRIPT hash
# Sort:SALES_CODE(asc:string)
open(DATA, q{cat  - | sort  -t'|' -y -k 4,4 2>/dev/null |}) || die "Cannot open input: $!";
&PrintHeader();
print STDERR '[examples/filter_regex.pql ' . localtime() . "] Start";
use Benchmark;
my $benchmark_start = new Benchmark;
while (<DATA>)
{
    ++$_inprecs;
    print STDERR '[examples/filter_regex.pql ' . localtime() . "] $_inprecs records." if ($_inprecs % VERBOSE
== 0);
    chomp;
    @I_VAL = split("[|]", $_);
    next unless ($I_VAL[_I_LOCATION] =~ /^NSW$|^WA$|^SA$/);
    $key__I_SALES_CODE = $I_VAL[_I_SALES_CODE];
    if (!defined($previous_key__I_SALES_CODE))
    {
        $previous_key__I_SALES_CODE = $key__I_SALES_CODE;
    }

    elsif ($previous_key__I_SALES_CODE ne $key__I_SALES_CODE)
    {
        $O_VAL[_O_PROD_NUM] = $O_VAL[_O__PRODUCT_CODE] . "-" . $O_VAL[_O_NUM_PRODUCTS];
        $O_VAL[_O_LOC_NSW] = ${$$_TABLE_LOC_DESCRIPT{qq{NSW}}}[_T_LOC_DESCRIPT_FLD_1];
        $O_VAL[_O_AVG_COST_PRICE_NSW] = ($AVERAGE{_O_AVG_COST_PRICE_NSW}{_COUNT} == 0 ? 0 : $AVERAGE{_O_AVG_CO
ST_PRICE_NSW}{_SUM} / $AVERAGE{_O_AVG_COST_PRICE_NSW}{_COUNT});
        $O_VAL[_O_LOC_WA] = ${$$_TABLE_LOC_DESCRIPT{qq{WA}}}[_T_LOC_DESCRIPT_FLD_1];
        $O_VAL[_O_AVG_COST_PRICE_WA] = ($AVERAGE{_O_AVG_COST_PRICE_WA}{_COUNT} == 0 ? 0 : $AVERAGE{_O_AVG_COST
_PRICE_WA}{_SUM} / $AVERAGE{_O_AVG_COST_PRICE_WA}{_COUNT});
        $O_VAL[_O_LOC_SA] = ${$$_TABLE_LOC_DESCRIPT{qq{SA}}}[_T_LOC_DESCRIPT_FLD_1];
        $O_VAL[_O_AVG_COST_PRICE_SA] = ($AVERAGE{_O_AVG_COST_PRICE_SA}{_COUNT} == 0 ? 0 : $AVERAGE{_O_AVG_COST
_PRICE_SA}{_SUM} / $AVERAGE{_O_AVG_COST_PRICE_SA}{_COUNT});
        print STDOUT
            $O_VAL[_O_SALES_CODE],
            $O_VAL[_O_LOC_DESCRIPT],
            $O_VAL[_O_NUM_PRODUCTS],
            $O_VAL[_O_PROD_NUM],
            $O_VAL[_O_LOC_NSW],
            $O_VAL[_O_AVG_COST_PRICE_NSW],
            $O_VAL[_O_LOC_WA],
            $O_VAL[_O_AVG_COST_PRICE_WA],
            $O_VAL[_O_LOC_SA],
            $O_VAL[_O_AVG_COST_PRICE_SA]
        ;
        $previous_key__I_SALES_CODE = $key__I_SALES_CODE;
        @O_VAL = undef;
        %DISTINCT = undef;
        %AVERAGE = undef;
    }

    $O_VAL[_O_SALES_CODE] = $I_VAL[_I_SALES_CODE];
    $I_VAL[_I_LDESCRIPT] = ${$$_TABLE_LOC_DESCRIPT{qq{$I_VAL[_I_LOCATION]}}}[_T_LOC_DESCRIPT_FLD_1] . " in pos
tcode " . ${$$_TABLE_LOC_DESCRIPT{qq{$I_VAL[_I_LOCATION]}}}[_T_LOC_DESCRIPT_FLD_2];
    $O_VAL[_O_LOC_DESCRIPT] = $I_VAL[_I_LDESCRIPT];
    $O_VAL[_O_NUM_PRODUCTS]++
        if (defined($I_VAL[_I_PRODUCT_CODE]) && ++$DISTINCT{_O_NUM_PRODUCTS}{qq{$I_VAL[_I_PRODUCT_CODE]}} == 1
);
    $O_VAL[_O__PRODUCT_CODE] = $I_VAL[_I_PRODUCT_CODE];

    if ($I_VAL[_I_LOCATION] eq ’NSW’) {
        $AVERAGE{_O_AVG_COST_PRICE_NSW}{_SUM} += $I_VAL[_I_COST_PRICE];
        $AVERAGE{_O_AVG_COST_PRICE_NSW}{_COUNT}++;
    }
    elsif ($I_VAL[_I_LOCATION] eq ’SA’) {
        $AVERAGE{_O_AVG_COST_PRICE_SA}{_SUM} += $I_VAL[_I_COST_PRICE];
        $AVERAGE{_O_AVG_COST_PRICE_SA}{_COUNT}++;
    }
    elsif ($I_VAL[_I_LOCATION] eq ’WA’) {
        $AVERAGE{_O_AVG_COST_PRICE_WA}{_SUM} += $I_VAL[_I_COST_PRICE];
        $AVERAGE{_O_AVG_COST_PRICE_WA}{_COUNT}++;
    }
}

$O_VAL[_O_PROD_NUM] = $O_VAL[_O__PRODUCT_CODE] . "-" . $O_VAL[_O_NUM_PRODUCTS];
$O_VAL[_O_LOC_NSW] = ${$$_TABLE_LOC_DESCRIPT{qq{NSW}}}[_T_LOC_DESCRIPT_FLD_1];
$O_VAL[_O_AVG_COST_PRICE_NSW] = ($AVERAGE{_O_AVG_COST_PRICE_NSW}{_COUNT} == 0 ? 0 : $AVERAGE{_O_AVG_COST_PRICE
_NSW}{_SUM} / $AVERAGE{_O_AVG_COST_PRICE_NSW}{_COUNT});
$O_VAL[_O_LOC_WA] = ${$$_TABLE_LOC_DESCRIPT{qq{WA}}}[_T_LOC_DESCRIPT_FLD_1];
$O_VAL[_O_AVG_COST_PRICE_WA] = ($AVERAGE{_O_AVG_COST_PRICE_WA}{_COUNT} == 0 ? 0 : $AVERAGE{_O_AVG_COST_PRICE_W
A}{_SUM} / $AVERAGE{_O_AVG_COST_PRICE_WA}{_COUNT});
$O_VAL[_O_LOC_SA] = ${$$_TABLE_LOC_DESCRIPT{qq{SA}}}[_T_LOC_DESCRIPT_FLD_1];
$O_VAL[_O_AVG_COST_PRICE_SA] = ($AVERAGE{_O_AVG_COST_PRICE_SA}{_COUNT} == 0 ? 0 : $AVERAGE{_O_AVG_COST_PRICE_S
A}{_SUM} / $AVERAGE{_O_AVG_COST_PRICE_SA}{_COUNT});
print STDOUT
    $O_VAL[_O_SALES_CODE],
    $O_VAL[_O_LOC_DESCRIPT],
    $O_VAL[_O_NUM_PRODUCTS],
    $O_VAL[_O_PROD_NUM],
    $O_VAL[_O_LOC_NSW],
    $O_VAL[_O_AVG_COST_PRICE_NSW],
    $O_VAL[_O_LOC_WA],
    $O_VAL[_O_AVG_COST_PRICE_WA],
    $O_VAL[_O_LOC_SA],
    $O_VAL[_O_AVG_COST_PRICE_SA]
;
close(DATA);
print STDERR ’[examples/filter_regex.pql ’ . localtime() . "] $_inprecs records.";
my $benchmark_end = new Benchmark;
my $benchmark_timediff = timediff($benchmark_start, $benchmark_end);
print STDERR ’[examples/filter_regex.pql ’ . localtime() . "] Code statistics: @{[timestr($benchmark_timediff)
]}";
```

```
#-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
#++++++ Table LOC_DESCRIPT --> Type :ETL::Pequel::Type::Table::Local ++++++
sub InitLookupLOC_DESCRIPT
{
    my %_TABLE_LOC_DESCRIPT;
    %_TABLE_LOC_DESCRIPT =
    (
        'NSW' => ['New South Wales', '2061', '02'],
        'SA' => ['South Australia', '8078', '08'],
        'WA' => ['Western Australia', '5008', '07']
    );
    return \%_TABLE_LOC_DESCRIPT;
}

sub PrintHeader
{
    local $\="\n";
    local $,="|";
    print STDOUT
        'SALES_CODE',
        'LOC_DESCRIPT',
        'NUM_PRODUCTS',
        'PROD_NUM',
        'LOC_NSW',
        'AVG_COST_PRICE_NSW',
        'LOC_WA',
        'AVG_COST_PRICE_WA',
        'LOC_SA',
        'AVG_COST_PRICE_SA'
        ;
}
```

7. ABOUT PEQUEL

This document was generated by Pequel.

https://sourceforge.net/projects/pequel/

COPYRIGHT