examples/aggregates_1.pql by *Pequel*

sample@youraddress.com

Aggregates Example Script

2.2

Table of Contents
Aggregates Example Script

SCRIPT NAME

examples/aggregates_1.pql

DESCRIPTION

Demonstrates aggregation and use of various aggregate function.

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 records are **grouped** by the input fields ***PRODUCT_CODE*** (*string*) and ***LOCATION*** (*string*).

1.1 *LOCATION*

Output Field

Description
Set to input field ***LOCATION***

1.2 *PRODUCT_CODE*

Output Field

Description
Set to input field ***PRODUCT_CODE***

1.3 *MIN_COST_PRICE*

Output Field

Description
Min aggregation on input field ***COST_PRICE***.

1.4 *MAX_COST_PRICE*

Output Field

Description
Max aggregation on input field ***COST_PRICE***.

1.5 *AVG_SALES_PRICE*

Output Field

Description
Mean aggregation on input field ***SALES_PRICE***.

1.7 *SALES_TOTAL*

Output Field

Description
Sum aggregation on input field ***SALES_TOTAL***.

1.8 *SALES_TOTAL_2*

Output Field

Description

Sum aggregation on input field ***SALES_TOTAL***.

1.9 *RANGE_COST*

Output Field

Description

Range aggregation on input field ***COST_PRICE***.

1.10 *MODE_SALES_CODE*

Output Field

Description

Mode aggregation on input field ***SALES_CODE***.

1.11 *AVGS*

Output Field

Description

Derived (calculated) field.

Derived Field Evaluation

2. CONFIGURATION SETTINGS

2.1 *pequeldoc*

generate pod / pdf pequel script Reference Guide.: pdf

2.2 *detail*

Include Pequel Generated Program chapter in Pequeldoc: 1

2.3 *noverbose*

do not progress counter: 1

2.4 *prefix*

directory pathname prefix.: examples

2.5 *script_name*

script filename: examples/aggregates_1.pql

2.6 *header*

write header record to output.: 1

2.7 *optimize*

optimize generated code.: 1

2.8 *hash*

Generate in memory. Input data can be unsorted.: 1

2.9 *nulls*

print zero for null numeric/decimal.: 1

2.10 *doc_title*

document title.: Aggregates Example Script

2.11 *doc_email*

document email entry.: sample@youraddress.com

2.12 *doc_version*

document version for pequel script.: 2.2

3. TABLES

4. TABLE INFORMATION SUMMARY

4.1 Table List Sorted By Table Name

5. EXAMPLES/AGGREGATES_1.PQL

options

```
pequeldoc(pdf)
detail(1)
noverbose(1)
prefix(examples)
script_name(examples/aggregates_1.pql)
header(1)
optimize(1)
hash(1)
nulls(1)
doc_title(Aggregates Example Script)
doc_email(sample@youraddress.com)
doc_version(2.2)
```

description

```
Demonstrates aggregation and use of various aggregate function.
```

input section

```
PRODUCT_CODE
COST_PRICE
DESCRIPTION
SALES_CODE
SALES_PRICE
SALES_QTY
SALES_DATE
LOCATION
SALES_TOTAL => SALES_QTY * SALES_PRICE
```

group by

```
PRODUCT_CODE string
LOCATION string
```

output section

```
string    LOCATION        LOCATION
string    PRODUCT_CODE    PRODUCT_CODE
decimal   MIN_COST_PRICE  min COST_PRICE
decimal   MAX_COST_PRICE  max COST_PRICE
decimal   AVG_SALES_PRICE mean SALES_PRICE
numeric   _AVG_SALES_QTY  mean SALES_QTY
decimal   SALES_TOTAL     sum SALES_TOTAL
decimal   SALES_TOTAL_2   sum SALES_TOTAL
decimal   RANGE_COST      range COST_PRICE
numeric   MODE_SALES_CODE mode SALES_CODE
numeric   AVGS            = _AVG_SALES_QTY * 2
```

6. PEQUEL GENERATED PROGRAM

```
#!/usr/bin/perl
#-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
# vim: syntax=perl ts=4 sw=4
#-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
#Generated By: pequel Version 2.3-6, Build: Wednesday October  26 23:16:49 BST 2005
#            : http://sourceforge.net/projects/pequel/
#Script Name : examples/aggregates_1.pql
#Created On  : Wed Oct 26 14:21:36 2005
#Perl Version: /usr/bin/perl 5.6.1 on solaris
#For         :
#-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
#Options:
#pequeldoc(pdf) generate pod / pdf pequel script Reference Guide.
#detail(1) Include Pequel Generated Program chapter in Pequeldoc
#noverbose(1) do not progress counter
#prefix(examples) directory pathname prefix.
#script_name(examples/aggregates_1.pql) script filename
#header(1) write header record to output.
#optimize(1) optimize generated code.
#hash(1) Generate in memory. Input data can be unsorted.
#nulls(1) print zero for null numeric/decimal.
#doc_title(Aggregates 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_SALES_TOTAL        => int    8;
use constant _O_LOCATION           => int    1;
use constant _O_PRODUCT_CODE       => int    2;
use constant _O_MIN_COST_PRICE     => int    3;
use constant _O_MAX_COST_PRICE     => int    4;
use constant _O_AVG_SALES_PRICE    => int    5;
use constant _O__AVG_SALES_QTY     => int    6;
use constant _O_SALES_TOTAL        => int    7;
use constant _O_SALES_TOTAL_2      => int    8;
use constant _O_RANGE_COST         => int    9;
use constant _O_MODE_SALES_CODE    => int   10;
use constant _O_AVGS               => int   11;
local $\="\n";
local $,="|";
use constant VERBOSE => int 10000;
use constant LAST_ICELL => int 8;
my @I_VAL;
my %O_VAL;
my $key;
my %AVERAGE;
my %RANGE;
my %MODE;
&PrintHeader();
while (<STDIN>)
{
    chomp;
    @I_VAL = split("[|]", $_);
    $key = ( $I_VAL[_I_PRODUCT_CODE] ) . ’|’ . ( $I_VAL[_I_LOCATION] );
    $O_VAL{$key}{_O_LOCATION} = $I_VAL[_I_LOCATION];
    $O_VAL{$key}{_O_PRODUCT_CODE} = $I_VAL[_I_PRODUCT_CODE];
    $O_VAL{$key}{_O_MIN_COST_PRICE} = $I_VAL[_I_COST_PRICE]
        if (!defined($O_VAL{$key}{_O_MIN_COST_PRICE}) || $I_VAL[_I_COST_PRICE] < $O_VAL{$key}{_O_MIN_COST_PRIC
E});
    $O_VAL{$key}{_O_MAX_COST_PRICE} = $I_VAL[_I_COST_PRICE]
        if (!defined($O_VAL{$key}{_O_MAX_COST_PRICE}) || $I_VAL[_I_COST_PRICE] > $O_VAL{$key}{_O_MAX_COST_PRIC
E});
    $AVERAGE{$key}{_O_AVG_SALES_PRICE}{_SUM} += $I_VAL[_I_SALES_PRICE];
    $AVERAGE{$key}{_O_AVG_SALES_PRICE}{_COUNT}++;
    $AVERAGE{$key}{_O__AVG_SALES_QTY}{_SUM} += $I_VAL[_I_SALES_QTY];
    $AVERAGE{$key}{_O__AVG_SALES_QTY}{_COUNT}++;
    $I_VAL[_I_SALES_TOTAL] = $I_VAL[_I_SALES_QTY] * $I_VAL[_I_SALES_PRICE];
    $O_VAL{$key}{_O_SALES_TOTAL} += $I_VAL[_I_SALES_TOTAL] unless ($I_VAL[_I_SALES_TOTAL] eq ’’);
    $I_VAL[_I_SALES_TOTAL] = $I_VAL[_I_SALES_QTY] * $I_VAL[_I_SALES_PRICE];
    $O_VAL{$key}{_O_SALES_TOTAL_2} += $I_VAL[_I_SALES_TOTAL] unless ($I_VAL[_I_SALES_TOTAL] eq ’’);
    $RANGE{$key}{_O_RANGE_COST}{_MIN} = $I_VAL[_I_COST_PRICE]
        if
        (
            !defined($RANGE{$key}{_O_RANGE_COST}{_MIN})
            || $I_VAL[_I_COST_PRICE] < $RANGE{$key}{_O_RANGE_COST}{_MIN}
        );

    $RANGE{$key}{_O_RANGE_COST}{_MAX} = $I_VAL[_I_COST_PRICE]
        if
        (
            !defined($RANGE{$key}{_O_RANGE_COST}{_MAX})
            || $I_VAL[_I_COST_PRICE] > $RANGE{$key}{_O_RANGE_COST}{_MAX}
        );

    $MODE{$key}{_O_MODE_SALES_CODE}{qq{$I_VAL[_I_SALES_CODE]}}++;
}

foreach $key (sort  keys %O_VAL)
{
    $O_VAL{$key}{_O_AVG_SALES_PRICE} = ($AVERAGE{$key}{_O_AVG_SALES_PRICE}{_COUNT} == 0 ? 0 : $AVERAGE{$key}{_
O_AVG_SALES_PRICE}{_SUM} / $AVERAGE{$key}{_O_AVG_SALES_PRICE}{_COUNT});
    $O_VAL{$key}{_O__AVG_SALES_QTY} = ($AVERAGE{$key}{_O__AVG_SALES_QTY}{_COUNT} == 0 ? 0 : $AVERAGE{$key}{_O_
_AVG_SALES_QTY}{_SUM} / $AVERAGE{$key}{_O__AVG_SALES_QTY}{_COUNT});
    $O_VAL{$key}{_O_RANGE_COST} = $RANGE{$key}{_O_RANGE_COST}{_MAX} - $RANGE{$key}{_O_RANGE_COST}{_MIN};
    $O_VAL{$key}{_O_MODE_SALES_CODE} = join(’ ’, &{sub{my @top; foreach my $k (sort { $MODE{$key}{_O_MODE_SALE
S_CODE}{$b} <=> $MODE{$key}{_O_MODE_SALES_CODE}{$a} } keys %{$MODE{$key}{_O_MODE_SALES_CODE}} ){ last if ($MOD
E{$key}{_O_MODE_SALES_CODE}{$k} != $MODE{$key}{_O_MODE_SALES_CODE}{$_[0]}); push(@top, $k);} @top; }}((sort {
$MODE{$key}{_O_MODE_SALES_CODE}{$b} <=> $MODE{$key}{_O_MODE_SALES_CODE}{$a} } keys %{$MODE{$key}{_O_MODE_SALES
_CODE}} )[0]));
    $O_VAL{$key}{_O_AVGS} = $O_VAL{$key}{_O__AVG_SALES_QTY} * 2;
    print STDOUT
        $O_VAL{$key}{_O_LOCATION},
        $O_VAL{$key}{_O_PRODUCT_CODE},
        $O_VAL{$key}{_O_MIN_COST_PRICE},
        $O_VAL{$key}{_O_MAX_COST_PRICE},
        $O_VAL{$key}{_O_AVG_SALES_PRICE},
        $O_VAL{$key}{_O_SALES_TOTAL},
        $O_VAL{$key}{_O_SALES_TOTAL_2},
        $O_VAL{$key}{_O_RANGE_COST},
        $O_VAL{$key}{_O_MODE_SALES_CODE},
        $O_VAL{$key}{_O_AVGS}
    ;
}

#-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
sub PrintHeader
{
    local $\="\n";
    local $,="|";
    print
        ’LOCATION’,
        ’PRODUCT_CODE’,
        ’MIN_COST_PRICE’,
        ’MAX_COST_PRICE’,
        ’AVG_SALES_PRICE’,
        ’SALES_TOTAL’,
        ’SALES_TOTAL_2’,
        ’RANGE_COST’,
        ’MODE_SALES_CODE’,
        ’AVGS’
        ;
}
```

7. ABOUT PEQUEL

This document was generated by Pequel.

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

COPYRIGHT