



examples/conditional_aggr.pql

by *Pequel*

sample@youraddress.com

Conditional Aggregation Example Script

2.2

Table of Contents

Conditional Aggregation Example Script

SCRIPT NAME	1
DESCRIPTION	1
1. PROCESS DETAILS	1
1.1 PRODUCT_CODE	1
Description	1
1.2 AVG_COST_PRICE	1
Description	1
1.3 MIN_COST_PRICE	1
Description	1
1.4 MAX_COST_PRICE	1
Description	1
1.5 SUM_COST_PRICE	1
Description	1
1.6 AVG_COST_PRICE_NSW	1
Description	2
Aggregation condition	2
1.7 MIN_COST_PRICE_NSW	2
Description	2
Aggregation condition	2
1.8 MAX_COST_PRICE_NSW	2
Description	2
Aggregation condition	2
1.9 SUM_COST_PRICE_NSW	2
Description	2
Aggregation condition	2
1.10 AVG_COST_PRICE_VIC	2
Description	2
Aggregation condition	2
1.11 MIN_COST_PRICE_VIC	2
Description	2
Aggregation condition	2
1.12 MAX_COST_PRICE_VIC	2
Description	2
Aggregation condition	3
1.13 SUM_COST_PRICE_VIC	3
Description	3
Aggregation condition	3
1.14 RANGE_COST_PRICE	3
Description	3
Derived Field Evaluation	3
2. CONFIGURATION SETTINGS	4
2.1 prefix	4
2.2 pequeldoc	4
2.3 detail	4
2.4 script_name	4
2.5 header	4
2.6 optimize	4
2.7 doc_title	4
2.8 doc_email	4
2.9 doc_version	4
3. TABLES	5
4. TABLE INFORMATION SUMMARY	6
4.1 Table List Sorted By Table Name	6

5. EXAMPLES/CONDITIONAL_AGGR.PQL	7
options	7
description	7
input section	7
sort by	7
group by	7
output section	7
6. PEQUEL GENERATED PROGRAM	8
7. ABOUT PEQUEL	11
COPYRIGHT	11

SCRIPT NAME

examples/conditional_aggr.pql

DESCRIPTION

Demonstrates the use of conditional aggregations. A conditional aggregate is done with the 'where' clause. This example analyses the **COST_PRICE** in various ways for the two states: NSW and VIC.

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 **14** fields. Fields are delimited by the '|' character.

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

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

1.1 **PRODUCT_CODE**

Output Field

Description

Set to input field **PRODUCT_CODE**

1.2 **AVG_COST_PRICE**

Output Field

Description

Avg aggregation on input field **COST_PRICE**.

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 **SUM_COST_PRICE**

Output Field

Description

Sum aggregation on input field **COST_PRICE**.

1.6 **AVG_COST_PRICE_NSW**

Output Field

Description

Avg aggregation on input field **COST_PRICE**.

Aggregation condition

LOCATION eq 'NSW';

1.7 MIN_COST_PRICE_NSW

Output Field

Description

Min aggregation on input field **COST_PRICE**.

Aggregation condition

LOCATION eq 'NSW';

1.8 MAX_COST_PRICE_NSW

Output Field

Description

Max aggregation on input field **COST_PRICE**.

Aggregation condition

LOCATION eq 'NSW';

1.9 SUM_COST_PRICE_NSW

Output Field

Description

Sum aggregation on input field **COST_PRICE**.

Aggregation condition

LOCATION eq 'NSW';

1.10 AVG_COST_PRICE_VIC

Output Field

Description

Avg aggregation on input field **COST_PRICE**.

Aggregation condition

LOCATION eq 'VIC';

1.11 MIN_COST_PRICE_VIC

Output Field

Description

Min aggregation on input field **COST_PRICE**.

Aggregation condition

LOCATION eq 'VIC';

1.12 MAX_COST_PRICE_VIC

Output Field

Description

Max aggregation on input field **COST_PRICE**.

Aggregation condition

LOCATION eq 'VIC';

1.13 SUM_COST_PRICE_VIC

Output Field

Description

Sum aggregation on input field **COST_PRICE**.

Aggregation condition

LOCATION eq 'VIC';

1.14 RANGE_COST_PRICE

Output Field

Description

Derived (calculated) field.

Derived Field Evaluation

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/conditional_aggr.pql

2.5 *header*

write header record to output.: 1

2.6 *optimize*

optimize generated code.: 1

2.7 *doc_title*

document title.: Conditional Aggregation 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

4. TABLE INFORMATION SUMMARY

4.1 Table List Sorted By Table Name

5. EXAMPLES/CONDITIONAL_AGGR.PQL

options

```

prefix(examples)
pequeldoc(pdf)
detail(1)
script_name(examples/conditional_aggr.pql)
header(1)
optimize(1)
doc_title(Conditional Aggregation Example Script)
doc_email(sample@youraddress.com)
doc_version(2.2)

```

description

Demonstrates the use of conditional aggregations. A conditional aggregate is done with the 'where' clause. This example analyses the COST_PRICE in various ways for the two states: NSW and VIC.

input section

```

PRODUCT_CODE
COST_PRICE
DESCRIPTION
SALES_CODE
SALES_PRICE
SALES_QTY
SALES_DATE
LOCATION

```

sort by

```
PRODUCT_CODE string
```

group by

```
PRODUCT_CODE string
```

output section

string	PRODUCT_CODE	PRODUCT_CODE
numeric	AVG_COST_PRICE	avg COST_PRICE
numeric	MIN_COST_PRICE	min COST_PRICE
numeric	MAX_COST_PRICE	max COST_PRICE
numeric	SUM_COST_PRICE	sum COST_PRICE
numeric	AVG_COST_PRICE_NSW	avg COST_PRICE where LOCATION eq 'NSW'
numeric	MIN_COST_PRICE_NSW	min COST_PRICE where LOCATION eq 'NSW'
numeric	MAX_COST_PRICE_NSW	max COST_PRICE where LOCATION eq 'NSW'
numeric	SUM_COST_PRICE_NSW	sum COST_PRICE where LOCATION eq 'NSW'
numeric	AVG_COST_PRICE_VIC	avg COST_PRICE where LOCATION eq 'VIC'
numeric	MIN_COST_PRICE_VIC	min COST_PRICE where LOCATION eq 'VIC'
numeric	MAX_COST_PRICE_VIC	max COST_PRICE where LOCATION eq 'VIC'
numeric	SUM_COST_PRICE_VIC	sum COST_PRICE where LOCATION eq 'VIC'
numeric	RANGE_COST_PRICE	= MAX_COST_PRICE - MIN_COST_PRICE

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 : conditional_aggr.pql
#Created On  : Wed Nov 16 13:56:08 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/conditional_aggr.pql) script filename
#header(1) write header record to output.
#optimize(1) optimize generated code.
#doc_title(Conditional Aggregation 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 _O_PRODUCT_CODE      => int    1;
use constant _O_AVG_COST_PRICE    => int    2;
use constant _O_MIN_COST_PRICE    => int    3;
use constant _O_MAX_COST_PRICE    => int    4;
use constant _O_SUM_COST_PRICE    => int    5;
use constant _O_AVG_COST_PRICE_NSW => int    6;
use constant _O_MIN_COST_PRICE_NSW => int    7;
use constant _O_MAX_COST_PRICE_NSW => int    8;
use constant _O_SUM_COST_PRICE_NSW => int    9;
use constant _O_AVG_COST_PRICE_VIC => int   10;
use constant _O_MIN_COST_PRICE_VIC => int   11;
use constant _O_MAX_COST_PRICE_VIC => int   12;
use constant _O_SUM_COST_PRICE_VIC => int   13;
use constant _O_RANGE_COST_PRICE  => int   14;
local $="\n";
local $,="|";
print STDERR "[examples/conditional_aggr.pql ' . localtime() . "] Init";
use constant VERBOSE => int 10000;
use constant LAST_ICELL => int 7;
my @I_VAL;
my @O_VAL;
my $_inprec=0;
my %AVERAGE;
my $key__I_PRODUCT_CODE;
my $previous_key__I_PRODUCT_CODE = undef;
foreach my $f (1..14) { $O_VAL[$f] = undef; }
# Sort:PRODUCT_CODE(asc:string)
open(DATA, q{cat - | sort -t'|' -y -k 1,1 2>/dev/null |}) || die "Cannot open input: $!";
&PrintHeader();
print STDERR "[examples/conditional_aggr.pql ' . localtime() . "] Start";
use Benchmark;
my $benchmark_start = new Benchmark;
while (<DATA>)
{
    ++$_inprec;
    print STDERR "[examples/conditional_aggr.pql ' . localtime() . "] $_inprec records." if ($_inprec % VERB
OSE == 0);
    chomp;
    @I_VAL = split("[|]", $_);
    $key__I_PRODUCT_CODE = $I_VAL[_I_PRODUCT_CODE];
    if (!defined($previous_key__I_PRODUCT_CODE))
    {
        $previous_key__I_PRODUCT_CODE = $key__I_PRODUCT_CODE;
    }

    elsif ($previous_key__I_PRODUCT_CODE ne $key__I_PRODUCT_CODE)
    {
        $O_VAL[_O_AVG_COST_PRICE] = ($AVERAGE{_O_AVG_COST_PRICE}{_COUNT} == 0 ? 0 : $AVERAGE{_O_AVG_COST_PRICE
}{_SUM} / $AVERAGE{_O_AVG_COST_PRICE}{_COUNT});
    }
}
```

```

    $O_VAL[_O_AVG_COST_PRICE_NSW] = ($AVERAGE{_O_AVG_COST_PRICE_NSW}{_COUNT} == 0 ? 0 : $AVERAGE{_O_AVG_COST_PRICE_NSW}{_SUM} / $AVERAGE{_O_AVG_COST_PRICE_NSW}{_COUNT});
    $O_VAL[_O_AVG_COST_PRICE_VIC] = ($AVERAGE{_O_AVG_COST_PRICE_VIC}{_COUNT} == 0 ? 0 : $AVERAGE{_O_AVG_COST_PRICE_VIC}{_SUM} / $AVERAGE{_O_AVG_COST_PRICE_VIC}{_COUNT});
    $O_VAL[_O_RANGE_COST_PRICE] = $O_VAL[_O_MAX_COST_PRICE] - $O_VAL[_O_MIN_COST_PRICE];
    print STDOUT
        $O_VAL[_O_PRODUCT_CODE],
        $O_VAL[_O_AVG_COST_PRICE],
        $O_VAL[_O_MIN_COST_PRICE],
        $O_VAL[_O_MAX_COST_PRICE],
        $O_VAL[_O_SUM_COST_PRICE],
        $O_VAL[_O_AVG_COST_PRICE_NSW],
        $O_VAL[_O_MIN_COST_PRICE_NSW],
        $O_VAL[_O_MAX_COST_PRICE_NSW],
        $O_VAL[_O_SUM_COST_PRICE_NSW],
        $O_VAL[_O_AVG_COST_PRICE_VIC],
        $O_VAL[_O_MIN_COST_PRICE_VIC],
        $O_VAL[_O_MAX_COST_PRICE_VIC],
        $O_VAL[_O_SUM_COST_PRICE_VIC],
        $O_VAL[_O_RANGE_COST_PRICE]
    ;
    $previous_key__I_PRODUCT_CODE = $key__I_PRODUCT_CODE;
    @O_VAL = undef;
    %AVERAGE = undef;
}

$O_VAL[_O_PRODUCT_CODE] = $I_VAL[_I_PRODUCT_CODE];
$AVERAGE{_O_AVG_COST_PRICE}{_SUM} += $I_VAL[_I_COST_PRICE];
$AVERAGE{_O_AVG_COST_PRICE}{_COUNT}++;
$O_VAL[_O_MIN_COST_PRICE] = $I_VAL[_I_COST_PRICE]
    if (!defined($O_VAL[_O_MIN_COST_PRICE]) || $I_VAL[_I_COST_PRICE] < $O_VAL[_O_MIN_COST_PRICE]);
$O_VAL[_O_MAX_COST_PRICE] = $I_VAL[_I_COST_PRICE]
    if (!defined($O_VAL[_O_MAX_COST_PRICE]) || $I_VAL[_I_COST_PRICE] > $O_VAL[_O_MAX_COST_PRICE]);
$O_VAL[_O_SUM_COST_PRICE] += $I_VAL[_I_COST_PRICE] unless ($I_VAL[_I_COST_PRICE] eq '');

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}++;
    $O_VAL[_O_MIN_COST_PRICE_NSW] = $I_VAL[_I_COST_PRICE]
        if (!defined($O_VAL[_O_MIN_COST_PRICE_NSW]) || $I_VAL[_I_COST_PRICE] < $O_VAL[_O_MIN_COST_PRICE_NSW]);
    $O_VAL[_O_MAX_COST_PRICE_NSW] = $I_VAL[_I_COST_PRICE]
        if (!defined($O_VAL[_O_MAX_COST_PRICE_NSW]) || $I_VAL[_I_COST_PRICE] > $O_VAL[_O_MAX_COST_PRICE_NSW]);
    $O_VAL[_O_SUM_COST_PRICE_NSW] += $I_VAL[_I_COST_PRICE] unless ($I_VAL[_I_COST_PRICE] eq '');
}
elseif ($I_VAL[_I_LOCATION] eq 'VIC') {
    $AVERAGE{_O_AVG_COST_PRICE_VIC}{_SUM} += $I_VAL[_I_COST_PRICE];
    $AVERAGE{_O_AVG_COST_PRICE_VIC}{_COUNT}++;
    $O_VAL[_O_MIN_COST_PRICE_VIC] = $I_VAL[_I_COST_PRICE]
        if (!defined($O_VAL[_O_MIN_COST_PRICE_VIC]) || $I_VAL[_I_COST_PRICE] < $O_VAL[_O_MIN_COST_PRICE_VIC]);
    $O_VAL[_O_MAX_COST_PRICE_VIC] = $I_VAL[_I_COST_PRICE]
        if (!defined($O_VAL[_O_MAX_COST_PRICE_VIC]) || $I_VAL[_I_COST_PRICE] > $O_VAL[_O_MAX_COST_PRICE_VIC]);
    $O_VAL[_O_SUM_COST_PRICE_VIC] += $I_VAL[_I_COST_PRICE] unless ($I_VAL[_I_COST_PRICE] eq '');
}

$O_VAL[_O_AVG_COST_PRICE] = ($AVERAGE{_O_AVG_COST_PRICE}{_COUNT} == 0 ? 0 : $AVERAGE{_O_AVG_COST_PRICE}{_SUM} / $AVERAGE{_O_AVG_COST_PRICE}{_COUNT});
$O_VAL[_O_AVG_COST_PRICE_NSW] = ($AVERAGE{_O_AVG_COST_PRICE_NSW}{_COUNT} == 0 ? 0 : $AVERAGE{_O_AVG_COST_PRICE_NSW}{_SUM} / $AVERAGE{_O_AVG_COST_PRICE_NSW}{_COUNT});
$O_VAL[_O_AVG_COST_PRICE_VIC] = ($AVERAGE{_O_AVG_COST_PRICE_VIC}{_COUNT} == 0 ? 0 : $AVERAGE{_O_AVG_COST_PRICE_VIC}{_SUM} / $AVERAGE{_O_AVG_COST_PRICE_VIC}{_COUNT});
$O_VAL[_O_RANGE_COST_PRICE] = $O_VAL[_O_MAX_COST_PRICE] - $O_VAL[_O_MIN_COST_PRICE];
print STDOUT
    $O_VAL[_O_PRODUCT_CODE],
    $O_VAL[_O_AVG_COST_PRICE],
    $O_VAL[_O_MIN_COST_PRICE],
    $O_VAL[_O_MAX_COST_PRICE],
    $O_VAL[_O_SUM_COST_PRICE],
    $O_VAL[_O_AVG_COST_PRICE_NSW],
    $O_VAL[_O_MIN_COST_PRICE_NSW],
    $O_VAL[_O_MAX_COST_PRICE_NSW],
    $O_VAL[_O_SUM_COST_PRICE_NSW],
    $O_VAL[_O_AVG_COST_PRICE_VIC],
    $O_VAL[_O_MIN_COST_PRICE_VIC],
    $O_VAL[_O_MAX_COST_PRICE_VIC],
    $O_VAL[_O_SUM_COST_PRICE_VIC],
    $O_VAL[_O_RANGE_COST_PRICE]
;
close(DATA);
print STDERR '[examples/conditional_aggr.pql ' . localtime() . "] $_inprec records.";
my $benchmark_end = new Benchmark;

```

```
my $benchmark_timediff = timediff($benchmark_start, $benchmark_end);
print STDERR "[examples/conditional_aggr.pql ' . localtime() . "] Code statistics: @{{timestr($benchmark_timediff)}}";
#-----
sub PrintHeader
{
    local $\="\n";
    local $,="| ";
    print STDOUT
        'PRODUCT_CODE',
        'AVG_COST_PRICE',
        'MIN_COST_PRICE',
        'MAX_COST_PRICE',
        'SUM_COST_PRICE',
        'AVG_COST_PRICE_NSW',
        'MIN_COST_PRICE_NSW',
        'MAX_COST_PRICE_NSW',
        'SUM_COST_PRICE_NSW',
        'AVG_COST_PRICE_VIC',
        'MIN_COST_PRICE_VIC',
        'MAX_COST_PRICE_VIC',
        'SUM_COST_PRICE_VIC',
        'RANGE_COST_PRICE'
    ;
}
```

7. ABOUT PEQUEL

This document was generated by Pequel.

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

COPYRIGHT

Copyright ©1999-2005, Mario Gaffiero. All Rights Reserved.

'Pequel' TM Copyright ©1999-2005, Mario Gaffiero. All Rights Reserved.

This program and all its component contents is copyrighted free software by Mario Gaffiero and is released under the GNU General Public License (GPL), Version 2, a copy of which may be found at <http://www.opensource.org/licenses/gpl-license.html>

Pequel is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

Pequel is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with Pequel; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

