



# examples/hash\_option.pql

by *Pequel*

---

[sample@youraddress.com](mailto:sample@youraddress.com)

## Hash Option Example Script

2.2



# Table of Contents

## Hash Option Example Script

SCRIPT NAME	1
DESCRIPTION	1
1. PROCESS DETAILS	1
1.1 LOCATION	1
Description	1
1.2 MIN_COST_PRICE	1
Description	1
1.3 MAX_COST_PRICE	1
Description	1
1.5 SALES_CODE_1	1
Description	1
Aggregation condition	1
1.6 SALES_CODE_2	1
Description	1
Aggregation condition	1
1.7 SALES_CODE_3	2
Description	2
Aggregation condition	2
1.8 SALES_CODE_4	2
Description	2
Aggregation condition	2
1.9 SALES_CODE_5	2
Description	2
Aggregation condition	2
2. CONFIGURATION SETTINGS	3
2.1 prefix	3
2.2 pequeldoc	3
2.3 detail	3
2.4 script_name	3
2.5 header	3
2.6 optimize	3
2.7 hash	3
2.8 doc_title	3
2.9 doc_email	3
2.10 doc_version	3
3. TABLES	4
4. TABLE INFORMATION SUMMARY	5
4.1 Table List Sorted By Table Name	5
5. EXAMPLES/HASH_OPTION.PQL	6
options	6
description	6
input section	6
group by	6
output section	6
6. PEQUEL GENERATED PROGRAM	7
7. ABOUT PEQUEL	9
COPYRIGHT	9



## SCRIPT NAME

examples/hash\_option.pql

## DESCRIPTION

This example demonstrates the use of the 'hash' option. With the 'hash' option input data sorting is not required — the data will be aggregated in memory. For this reason the 'hash' option should only be used when the total number of groups is small depending on the amount of available memory.

## 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 **9** fields. Fields are delimited by the '|' character.

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

### 1.1 LOCATION

Output Field

#### **Description**

Set to input field **LOCATION**

### 1.2 MIN\_COST\_PRICE

Output Field

#### **Description**

**Min** aggregation on input field **COST\_PRICE**.

### 1.3 MAX\_COST\_PRICE

Output Field

#### **Description**

**Max** aggregation on input field **COST\_PRICE**.

### 1.5 SALES\_CODE\_1

Output Field

#### **Description**

**First** aggregation on input field **SALES\_CODE**.

#### **Aggregation condition**

\_DISTINCT\_SALES\_CODE == 1;

### 1.6 SALES\_CODE\_2

Output Field

#### **Description**

**First** aggregation on input field **SALES\_CODE**.

#### **Aggregation condition**

\_DISTINCT\_SALES\_CODE == 2;

### 1.7 SALES\_CODE\_3

Output Field

**Description**

**First** aggregation on input field **SALES\_CODE**.

**Aggregation condition**

\_DISTINCT\_SALES\_CODE == 3;

### 1.8 SALES\_CODE\_4

Output Field

**Description**

**First** aggregation on input field **SALES\_CODE**.

**Aggregation condition**

\_DISTINCT\_SALES\_CODE == 4;

### 1.9 SALES\_CODE\_5

Output Field

**Description**

**First** aggregation on input field **SALES\_CODE**.

**Aggregation condition**

\_DISTINCT\_SALES\_CODE == 5;

## 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/hash\_option.pql

### 2.5 *header*

write header record to output.: 1

### 2.6 *optimize*

optimize generated code.: 1

### 2.7 *hash*

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

### 2.8 *doc\_title*

document title.: Hash Option Example Script

### 2.9 *doc\_email*

document email entry.: sample@youraddress.com

### 2.10 *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/HASH\_OPTION.PQL

### *options*

```

prefix(examples)
pequeldoc(pdf)
detail(1)
script_name(examples/hash_option.pql)
header(1)
optimize(1)
hash(1)
doc_title(Hash Option Example Script)
doc_email(sample@youraddress.com)
doc_version(2.2)

```

### *description*

This example demonstrates the use of the 'hash' option. With the 'hash' option input data sorting is not required -- the data will be aggregated in memory. For this reason the 'hash' option should only be used when the total number of groups is small depending on the amount of available memory.

### *input section*

```

PRODUCT_CODE
COST_PRICE
DESCRIPTION
SALES_CODE
SALES_PRICE
SALES_QTY
SALES_DATE
LOCATION

```

### *group by*

```

LOCATION string

```

### *output section*

string	LOCATION	LOCATION
numeric	MIN_COST_PRICE	min COST_PRICE
numeric	MAX_COST_PRICE	max COST_PRICE
numeric	_DISTINCT_SALES_CODE	distinct SALES_CODE
string	SALES_CODE_1	first SALES_CODE where _DISTINCT_SALES_CODE == 1
string	SALES_CODE_2	first SALES_CODE where _DISTINCT_SALES_CODE == 2
string	SALES_CODE_3	first SALES_CODE where _DISTINCT_SALES_CODE == 3
string	SALES_CODE_4	first SALES_CODE where _DISTINCT_SALES_CODE == 4
string	SALES_CODE_5	first SALES_CODE where _DISTINCT_SALES_CODE == 5

## 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 : hash_option.pql
#Created On  : Wed Nov 16 14:05:05 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/hash_option.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.
#doc_title(Hash Option 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_LOCATION              => int    1;
use constant _O_MIN_COST_PRICE        => int    2;
use constant _O_MAX_COST_PRICE        => int    3;
use constant _O_DISTINCT_SALES_CODE   => int    4;
use constant _O_SALES_CODE_1          => int    5;
use constant _O_SALES_CODE_2          => int    6;
use constant _O_SALES_CODE_3          => int    7;
use constant _O_SALES_CODE_4          => int    8;
use constant _O_SALES_CODE_5          => int    9;
local $_="\n";
local $,="|";
print STDERR '[examples/hash_option.pql ' . localtime() . "] Init";
use constant VERBOSE => int 10000;
use constant LAST_ICELL => int 7;
my @I_VAL;
my %O_VAL;
my $key;
my $_inprec=0;
my %DISTINCT;
&PrintHeader();
print STDERR '[examples/hash_option.pql ' . localtime() . "] Start";
use Benchmark;
my $benchmark_start = new Benchmark;
while (<STDIN>)
{
    ++$_inprec;
    print STDERR '[examples/hash_option.pql ' . localtime() . "] $_inprec records." if ($_inprec % VERBOSE =
= 0);
    chomp;
    @I_VAL = split("[|]", $_);
    $key = ( $I_VAL[_I_LOCATION] );
    $O_VAL{$key}{_O_LOCATION} = $I_VAL[_I_LOCATION];
    $O_VAL{$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_PRICE});
    $O_VAL{$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_PRICE});
    $O_VAL{$key}{_O_DISTINCT_SALES_CODE}++;
    if (defined($I_VAL[_I_SALES_CODE]) && ++$DISTINCT{$key}{_O_DISTINCT_SALES_CODE}{qq{$I_VAL[_I_SALES_CODE]}} == 1);

    if ($O_VAL{$key}{_O_DISTINCT_SALES_CODE} == 1) {
        $O_VAL{$key}{_O_SALES_CODE_1} = $I_VAL[_I_SALES_CODE] if (!defined($O_VAL{$key}{_O_SALES_CODE_1}));
    }
    elsif ($O_VAL{$key}{_O_DISTINCT_SALES_CODE} == 2) {
        $O_VAL{$key}{_O_SALES_CODE_2} = $I_VAL[_I_SALES_CODE] if (!defined($O_VAL{$key}{_O_SALES_CODE_2}));
    }
}
```

```

    elseif ($O_VAL{$key}{_O_DISTINCT_SALES_CODE} == 3) {
        $O_VAL{$key}{_O_SALES_CODE_3} = $I_VAL[_I_SALES_CODE] if (!defined($O_VAL{$key}{_O_SALES_CODE_3}));
    }
    elseif ($O_VAL{$key}{_O_DISTINCT_SALES_CODE} == 4) {
        $O_VAL{$key}{_O_SALES_CODE_4} = $I_VAL[_I_SALES_CODE] if (!defined($O_VAL{$key}{_O_SALES_CODE_4}));
    }
    elseif ($O_VAL{$key}{_O_DISTINCT_SALES_CODE} == 5) {
        $O_VAL{$key}{_O_SALES_CODE_5} = $I_VAL[_I_SALES_CODE] if (!defined($O_VAL{$key}{_O_SALES_CODE_5}));
    }
}

foreach $key (sort keys %O_VAL)
{
    print STDOUT
        $O_VAL{$key}{_O_LOCATION},
        $O_VAL{$key}{_O_MIN_COST_PRICE},
        $O_VAL{$key}{_O_MAX_COST_PRICE},
        $O_VAL{$key}{_O_SALES_CODE_1},
        $O_VAL{$key}{_O_SALES_CODE_2},
        $O_VAL{$key}{_O_SALES_CODE_3},
        $O_VAL{$key}{_O_SALES_CODE_4},
        $O_VAL{$key}{_O_SALES_CODE_5}
    ;
}

close(STDIN);
print STDERR "[examples/hash_option.pql ' . localtime() . "] $_inprec records.";
my $benchmark_end = new Benchmark;
my $benchmark_timediff = timediff($benchmark_start, $benchmark_end);
print STDERR "[examples/hash_option.pql ' . localtime() . "] Code statistics: @[["timestr($benchmark_timediff)"]";
#-----
sub PrintHeader
{
    local $\\="\n";
    local $,="|";
    print STDOUT
        'LOCATION',
        'MIN_COST_PRICE',
        'MAX_COST_PRICE',
        'SALES_CODE_1',
        'SALES_CODE_2',
        'SALES_CODE_3',
        'SALES_CODE_4',
        'SALES_CODE_5'
    ;
}

```

## 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

