



# examples/statistics\_aggr.pql

by *Pequel*

---

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

## Statistics Aggregates Example Script

2.2



# Table of Contents

## Statistics Aggregates Example Script

|                                     |    |
|-------------------------------------|----|
| SCRIPT NAME                         | 1  |
| DESCRIPTION                         | 1  |
| 1. PROCESS DETAILS                  | 1  |
| 1.1 SALES_CODE                      | 1  |
| Description                         | 1  |
| 1.2 COUNT_LOCATION                  | 1  |
| Description                         | 1  |
| 1.3 COUNT_PRODUCTS                  | 1  |
| Description                         | 1  |
| 1.4 TOTAL_SALES_PRICE               | 1  |
| Description                         | 1  |
| 1.5 TOTAL_SALES_QTY                 | 1  |
| Description                         | 1  |
| 1.6 MEDIAN_QTY                      | 1  |
| Description                         | 1  |
| 1.7 VARIANCE_QTY                    | 2  |
| Description                         | 2  |
| 1.8 STDDEV_QTY                      | 2  |
| Description                         | 2  |
| 1.9 RANGE_QTY                       | 2  |
| Description                         | 2  |
| 1.10 RANGE_QTY_2                    | 2  |
| Description                         | 2  |
| Derived Field Evaluation            | 2  |
| 1.11 RANGE_QTY_3                    | 2  |
| Description                         | 2  |
| Derived Field Evaluation            | 2  |
| 1.12 TEST_1                         | 2  |
| Description                         | 2  |
| Derived Field Evaluation            | 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 doc_title                       | 3  |
| 2.8 doc_email                       | 3  |
| 2.9 doc_version                     | 3  |
| 3. TABLES                           | 4  |
| 4. TABLE INFORMATION SUMMARY        | 5  |
| 4.1 Table List Sorted By Table Name | 5  |
| 5. EXAMPLES/STATISTICS_AGGR.PQL     | 6  |
| options                             | 6  |
| description                         | 6  |
| input section                       | 6  |
| sort by                             | 6  |
| group by                            | 6  |
| output section                      | 6  |
| 6. PEQUEL GENERATED PROGRAM         | 7  |
| 7. ABOUT PEQUEL                     | 10 |
| COPYRIGHT                           | 10 |



## SCRIPT NAME

examples/statistics\_aggr.pql

## DESCRIPTION

Demonstrate various statistical aggregates functions.

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

Input stream is **sorted** by the input field **SALES\_CODE** (*string*).

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 COUNT\_LOCATION

Output Field

**Description**

**Distinct** aggregation on input field **LOCATION**.

#### 1.3 COUNT\_PRODUCTS

Output Field

**Description**

**Distinct** aggregation on input field **PRODUCT\_CODE**.

#### 1.4 TOTAL\_SALES\_PRICE

Output Field

**Description**

**Sum** aggregation on input field **SALES\_PRICE**.

#### 1.5 TOTAL\_SALES\_QTY

Output Field

**Description**

**Sum** aggregation on input field **SALES\_QTY**.

#### 1.6 MEDIAN\_QTY

Output Field

**Description**

**Median** aggregation on input field **SALES\_QTY**.

#### 1.7 **VARIANCE\_QTY**

Output Field

**Description**

**Variance** aggregation on input field **SALES\_QTY**.

#### 1.8 **STDDEV\_QTY**

Output Field

**Description**

**Stddev** aggregation on input field **SALES\_QTY**.

#### 1.9 **RANGE\_QTY**

Output Field

**Description**

**Range** aggregation on input field **SALES\_QTY**.

#### 1.10 **RANGE\_QTY\_2**

Output Field

**Description**

Derived (calculated) field.

**Derived Field Evaluation**

#### 1.11 **RANGE\_QTY\_3**

Output Field

**Description**

Derived (calculated) field.

**Derived Field Evaluation**

#### 1.12 **TEST\_1**

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/statistics\_aggr.pql

### 2.5 *header*

write header record to output.: 1

### 2.6 *optimize*

optimize generated code.: 1

### 2.7 *doc\_title*

document title.: Statistics Aggregates 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/STATISTICS\_AGGR.PQL

### *options*

```

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

```

### *description*

Demonstrate various statistical aggregates functions.

### *input section*

```

PRODUCT_CODE
COST_PRICE
DESCRIPTION
SALES_CODE
SALES_PRICE
SALES_QTY
SALES_DATE
LOCATION

```

### *sort by*

```
SALES_CODE string
```

### *group by*

```
SALES_CODE string
```

### *output section*

|         |                   |                       |
|---------|-------------------|-----------------------|
| string  | SALES_CODE        | SALES_CODE            |
| numeric | COUNT_LOCATION    | distinct LOCATION     |
| numeric | COUNT_PRODUCTS    | distinct PRODUCT_CODE |
| decimal | TOTAL_SALES_PRICE | sum SALES_PRICE       |
| decimal | TOTAL_SALES_QTY   | sum SALES_QTY         |
| numeric | MEDIAN_QTY        | median SALES_QTY      |
| numeric | VARIANCE_QTY      | variance SALES_QTY    |
| numeric | STDDEV_QTY        | stddev SALES_QTY      |
| numeric | RANGE_QTY         | range SALES_QTY       |
| numeric | RANGE_QTY_2       | = RANGE_QTY * 2       |
| numeric | RANGE_QTY_3       | = RANGE_QTY * 3       |
| decimal | TEST_1            | = MEDIAN_QTY + 100    |

## 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 : statistics_aggr.pql
#Created On  : Wed Nov 16 14:20:40 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/statistics_aggr.pql) script filename
#header(1) write header record to output.
#optimize(1) optimize generated code.
#doc_title(Statistics 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 _O_SALES_CODE        => int    1;
use constant _O_COUNT_LOCATION    => int    2;
use constant _O_COUNT_PRODUCTS    => int    3;
use constant _O_TOTAL_SALES_PRICE => int    4;
use constant _O_TOTAL_SALES_QTY   => int    5;
use constant _O_MEDIAN_QTY        => int    6;
use constant _O_VARIANCE_QTY      => int    7;
use constant _O_STDDEV_QTY        => int    8;
use constant _O_RANGE_QTY         => int    9;
use constant _O_RANGE_QTY_2       => int   10;
use constant _O_RANGE_QTY_3       => int   11;
use constant _O_TEST_1            => int   12;

local $\\="\n";
local $,="|";
print STDERR "[examples/statistics_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 %DISTINCT;
my %MEDIAN;
my %MEDIAN_COUNT;
my %VARIANCE;
my %STDDEV;
my %RANGE;
my $key__I_SALES_CODE;
my $previous_key__I_SALES_CODE = undef;
foreach my $f (1..12) { $O_VAL[$f] = undef; }
# 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/statistics_aggr.pql ' . localtime() . "] Start";
use Benchmark;
my $benchmark_start = new Benchmark;
while (<DATA>)
{
    ++$_inprec;
    print STDERR "[examples/statistics_aggr.pql ' . localtime() . "] $_inprec records." if ($_inprec % VERBOSE
SE == 0);
    chomp;
    @I_VAL = split("[|]", $_);
    $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_MEDIAN_QTY] = $MEDIAN_COUNT[_O_MEDIAN_QTY] % 2 == 0 ? &{sub{($_[0] + $_[1]) / 2}}((( sort {$a <=> $b} keys %{$MEDIAN_COUNT[_O_MEDIAN_QTY]} )[$MEDIAN_COUNT[_O_MEDIAN_QTY]/2-1, $MEDIAN_COUNT[_O_MEDIAN_QTY]/2])[$0,1]) : (sort {$a <=> $b} keys %{$MEDIAN_COUNT[_O_MEDIAN_QTY]} )[(($MEDIAN_COUNT[_O_MEDIAN_QTY]+1)/2)-1];
    $O_VAL[_O_VARIANCE_QTY] = ($VARIANCE[_O_VARIANCE_QTY]{_SUM_SQUARES} / ($VARIANCE[_O_VARIANCE_QTY]{_COUNT} == 0 ? 1 : $VARIANCE[_O_VARIANCE_QTY]{_COUNT}))- (($VARIANCE[_O_VARIANCE_QTY]{_SUM} / $VARIANCE[_O_VARIANCE_QTY]{_COUNT}) * 2);
    $O_VAL[_O_STDDEV_QTY] = sqrt(($STDDEV[_O_STDDEV_QTY]{_SUM_SQUARES} / ($STDDEV[_O_STDDEV_QTY]{_COUNT} == 0 ? 1 : $STDDEV[_O_STDDEV_QTY]{_COUNT}))- (($STDDEV[_O_STDDEV_QTY]{_SUM} / $STDDEV[_O_STDDEV_QTY]{_COUNT}) * 2));
    $O_VAL[_O_RANGE_QTY] = $RANGE[_O_RANGE_QTY]{_MAX} - $RANGE[_O_RANGE_QTY]{_MIN};
    $O_VAL[_O_RANGE_QTY_2] = $O_VAL[_O_RANGE_QTY] * 2;
    $O_VAL[_O_RANGE_QTY_3] = $O_VAL[_O_RANGE_QTY] * 3;
    $O_VAL[_O_TEST_1] = $O_VAL[_O_MEDIAN_QTY] + 100;
    print STDOUT
        $O_VAL[_O_SALES_CODE],
        $O_VAL[_O_COUNT_LOCATION],
        $O_VAL[_O_COUNT_PRODUCTS],
        $O_VAL[_O_TOTAL_SALES_PRICE],
        $O_VAL[_O_TOTAL_SALES_QTY],
        $O_VAL[_O_MEDIAN_QTY],
        $O_VAL[_O_VARIANCE_QTY],
        $O_VAL[_O_STDDEV_QTY],
        $O_VAL[_O_RANGE_QTY],
        $O_VAL[_O_RANGE_QTY_2],
        $O_VAL[_O_RANGE_QTY_3],
        $O_VAL[_O_TEST_1]
    ;
    $previous_key_I_SALES_CODE = $key_I_SALES_CODE;
    @O_VAL = undef;
    %DISTINCT = undef;
    %MEDIAN = undef;
    %MEDIAN_COUNT = undef;
    %VARIANCE = undef;
    %STDDEV = undef;
    %RANGE = undef;
}

$O_VAL[_O_SALES_CODE] = $I_VAL[_I_SALES_CODE];
$O_VAL[_O_COUNT_LOCATION]++
    if (defined($I_VAL[_I_LOCATION]) && ++$DISTINCT[_O_COUNT_LOCATION]{qq{$I_VAL[_I_LOCATION]}} == 1);
$O_VAL[_O_COUNT_PRODUCTS]++
    if (defined($I_VAL[_I_PRODUCT_CODE]) && ++$DISTINCT[_O_COUNT_PRODUCTS]{qq{$I_VAL[_I_PRODUCT_CODE]}} == 1);
$O_VAL[_O_TOTAL_SALES_PRICE] += $I_VAL[_I_SALES_PRICE] unless ($I_VAL[_I_SALES_PRICE] eq '');
$O_VAL[_O_TOTAL_SALES_QTY] += $I_VAL[_I_SALES_QTY] unless ($I_VAL[_I_SALES_QTY] eq '');
$MEDIAN_COUNT[_O_MEDIAN_QTY]++ if (++$MEDIAN_COUNT[_O_MEDIAN_QTY]{qq{$I_VAL[_I_SALES_QTY]}} == 1);
$VARIANCE[_O_VARIANCE_QTY]{_SUM} += $I_VAL[_I_SALES_QTY];
$VARIANCE[_O_VARIANCE_QTY]{_SUM_SQUARES} += $I_VAL[_I_SALES_QTY] ** 2;
$VARIANCE[_O_VARIANCE_QTY]{_COUNT}++;
$STDDEV[_O_STDDEV_QTY]{_SUM} += $I_VAL[_I_SALES_QTY];
$STDDEV[_O_STDDEV_QTY]{_SUM_SQUARES} += $I_VAL[_I_SALES_QTY] ** 2;
$STDDEV[_O_STDDEV_QTY]{_COUNT}++;
$RANGE[_O_RANGE_QTY]{_MIN} = $I_VAL[_I_SALES_QTY]
    if
    (
        !defined($RANGE[_O_RANGE_QTY]{_MIN})
        || $I_VAL[_I_SALES_QTY] < $RANGE[_O_RANGE_QTY]{_MIN}
    );
$RANGE[_O_RANGE_QTY]{_MAX} = $I_VAL[_I_SALES_QTY]
    if
    (
        !defined($RANGE[_O_RANGE_QTY]{_MAX})
        || $I_VAL[_I_SALES_QTY] > $RANGE[_O_RANGE_QTY]{_MAX}
    );
}

$O_VAL[_O_MEDIAN_QTY] = $MEDIAN_COUNT[_O_MEDIAN_QTY] % 2 == 0 ? &{sub{($_[0] + $_[1]) / 2}}((( sort {$a <=> $b} keys %{$MEDIAN_COUNT[_O_MEDIAN_QTY]} )[$MEDIAN_COUNT[_O_MEDIAN_QTY]/2-1, $MEDIAN_COUNT[_O_MEDIAN_QTY]/2])[$0,1]) : (sort {$a <=> $b} keys %{$MEDIAN_COUNT[_O_MEDIAN_QTY]} )[(($MEDIAN_COUNT[_O_MEDIAN_QTY]+1)/2)-1];
$O_VAL[_O_VARIANCE_QTY] = ($VARIANCE[_O_VARIANCE_QTY]{_SUM_SQUARES} / ($VARIANCE[_O_VARIANCE_QTY]{_COUNT} == 0 ? 1 : $VARIANCE[_O_VARIANCE_QTY]{_COUNT}))- (($VARIANCE[_O_VARIANCE_QTY]{_SUM} / $VARIANCE[_O_VARIANCE_QTY]{_COUNT}) * 2);
$O_VAL[_O_STDDEV_QTY] = sqrt(($STDDEV[_O_STDDEV_QTY]{_SUM_SQUARES} / ($STDDEV[_O_STDDEV_QTY]{_COUNT} == 0 ? 1 : $STDDEV[_O_STDDEV_QTY]{_COUNT}))- (($STDDEV[_O_STDDEV_QTY]{_SUM} / $STDDEV[_O_STDDEV_QTY]{_COUNT}) * 2));
$O_VAL[_O_RANGE_QTY] = $RANGE[_O_RANGE_QTY]{_MAX} - $RANGE[_O_RANGE_QTY]{_MIN};
$O_VAL[_O_RANGE_QTY_2] = $O_VAL[_O_RANGE_QTY] * 2;
$O_VAL[_O_RANGE_QTY_3] = $O_VAL[_O_RANGE_QTY] * 3;
$O_VAL[_O_TEST_1] = $O_VAL[_O_MEDIAN_QTY] + 100;
print STDOUT
    $O_VAL[_O_SALES_CODE],
    $O_VAL[_O_COUNT_LOCATION],
    $O_VAL[_O_COUNT_PRODUCTS],

```

```

    $O_VAL[_O_TOTAL_SALES_PRICE],
    $O_VAL[_O_TOTAL_SALES_QTY],
    $O_VAL[_O_MEDIAN_QTY],
    $O_VAL[_O_VARIANCE_QTY],
    $O_VAL[_O_STDDEV_QTY],
    $O_VAL[_O_RANGE_QTY],
    $O_VAL[_O_RANGE_QTY_2],
    $O_VAL[_O_RANGE_QTY_3],
    $O_VAL[_O_TEST_1]
;
close(DATA);
print STDERR '[examples/statistics_aggr.pql ' . localtime() . "] $_inprec records.";
my $benchmark_end = new Benchmark;
my $benchmark_timediff = timediff($benchmark_start, $benchmark_end);
print STDERR '[examples/statistics_aggr.pql ' . localtime() . "] Code statistics: @{[timestr($benchmark_timediff)]}";
#-----
sub PrintHeader
{
    local $\="\n";
    local $,="| ";
    print STDOUT
        'SALES_CODE',
        'COUNT_LOCATION',
        'COUNT_PRODUCTS',
        'TOTAL_SALES_PRICE',
        'TOTAL_SALES_QTY',
        'MEDIAN_QTY',
        'VARIANCE_QTY',
        'STDDEV_QTY',
        'RANGE_QTY',
        'RANGE_QTY_2',
        'RANGE_QTY_3',
        'TEST_1'
    ;
}

```

## 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



