



array_fields.pql

by *Pequel*

sample@youraddress.com

Array Fields Example Script

2.2

Table of Contents

Array Fields Example Script

SCRIPT NAME	1
DESCRIPTION	1
1. PROCESS DETAILS	1
1.1 location	1
Description	1
1.2 product_code	1
Description	1
1.3 salesman_list	1
Description	1
1.4 num_salesmen	1
Description	1
Derived Input Field Evaluation	1
1.5 salesmen_sorted	1
Description	1
Derived Input Field Evaluation	1
1.6 salesmen_sorted_2	2
Description	2
Derived Input Field Evaluation	2
1.7 salesmen_uniq	2
Description	2
Derived Input Field Evaluation	2
1.8 salesmen_uniq_2	2
Description	2
Derived Input Field Evaluation	2
1.9 salesmen_reverse	2
Description	2
Derived Input Field Evaluation	2
2. CONFIGURATION SETTINGS	3
2.1 pequeldoc	3
2.2 detail	3
2.3 script_name	3
2.4 header	3
2.5 optimize	3
2.6 nulls	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. ARRAY_FIELDS.PQL	6
options	6
description	6
sort by	6
input section	6
output section	6
6. PEQUEL GENERATED PROGRAM	7
7. ABOUT PEQUEL	9
COPYRIGHT	9

SCRIPT NAME

array_fields.pql

DESCRIPTION

Demonstrates the use of array-fields. An array-field is denoted by the preceding '@' character. The 'salesman_list' field in this example is an 'array field' delimited by the default array field delimiter ','. Array type macros (&arr_...) will expect all arguments to be array-fields. Array macros can also be called as a method following the array-field.

1. PROCESS DETAILS

Input records are read from standard input. The input record contains **9** 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 stream is **sorted** by the input field **product_code** (*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 salesman_list

Output Field

Description

Set to input field **salesman_list**

1.4 num_salesmen

Output Field

Description

Set to input field **num_salesmen**

Derived Input Field Evaluation

```
=> &arr_size(@salesman_list)
```

1.5 salesmen_sorted

Output Field

Description

Set to input field **salesmen_sorted**

Derived Input Field Evaluation

```
=> &arr_sort(salesman_list)
```

1.6 salesmen_sorted_2

Output Field

DescriptionSet to input field **salesmen_sorted_2****Derived Input Field Evaluation**

```
=> @salesman_list->sort
```

1.7 salesmen_uniq

Output Field

DescriptionSet to input field **salesmen_uniq****Derived Input Field Evaluation**

```
=> &arr_values_uniq(@salesman_list)
```

1.8 salesmen_uniq_2

Output Field

DescriptionSet to input field **salesmen_uniq_2****Derived Input Field Evaluation**

```
=> @salesman_list->values_uniq
```

1.9 salesmen_reverse

Output Field

DescriptionSet to input field **salesmen_reverse****Derived Input Field Evaluation**

```
=> &arr_reverse(&arr_sort(@salesman_list))
```

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 *script_name*

script filename: array_fields.pql

2.4 *header*

write header record to output.: 1

2.5 *optimize*

optimize generated code.: 1

2.6 *nulls*

print zero for null numeric/decimal.: 1

2.7 *doc_title*

document title.: Array Fields 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. ARRAY_FIELDS.PQL

options

```

pequeldoc(pdf)
detail(1)
script_name(array_fields.pql)
header(1)
optimize(1)
nulls(1)
doc_title(Array Fields Example Script)
doc_email(sample@youraddress.com)
doc_version(2.2)

```

description

Demonstrates the use of array-fields. An array-field is denoted by the preceding '@' character. The 'salesman_list' field in this example is an 'array field' delimited by the default array field delimiter ','. Array type macros (&arr_...) will expect all arguments to be array-fields. Array macros can also be called as a method following the array-field.

sort by

```
product_code string
```

input section

```

product_code
cost_price
description
sales_code
sales_price
sales_qty
sales_date
location
salesman_list
num_salesmen => &arr_size(@salesman_list)

salesmen_sorted => &arr_sort(salesman_list)

salesmen_sorted_2 => @salesman_list->sort

salesmen_uniq => &arr_values_uniq(@salesman_list)

salesmen_uniq_2 => @salesman_list->values_uniq

salesmen_reverse => &arr_reverse(&arr_sort(@salesman_list))

```

output section

string	location	location
string	product_code	product_code
string	salesman_list	salesman_list
numeric	num_salesmen	num_salesmen
string	salesmen_sorted	salesmen_sorted
string	salesmen_sorted_2	salesmen_sorted_2
string	salesmen_uniq	salesmen_uniq
string	salesmen_uniq_2	salesmen_uniq_2
string	salesmen_reverse	salesmen_reverse

6. PEQUEL GENERATED PROGRAM

```
# vim: syntax=perl ts=4 sw=4
#-----
#Generated By: pequel Version 2.2-9, Build: Tuesday September 13 08:43:08 BST 2005
#           : https://sourceforge.net/projects/pequel/
#Script Name : array_fields.pql
#Created On  : Tue Sep 13 10:17:55 2005
#For         :
#-----
#Options:
#pequeldoc(pdf) generate pod / pdf pequel script Reference Guide.
#detail(1) Include Pequel Generated Program chapter in Pequeldoc
#script_name(array_fields.pql) script filename
#header(1) write header record to output.
#optimize(1) optimize generated code.
#nulls(1) print zero for null numeric/decimal.
#doc_title(Array Fields Example Script) document title.
#doc_email(sample@youraddress.com) document email entry.
#doc_version(2.2) document version for pequel script.
#-----
use strict;
local $\\="\\n"; local $,="|";
print STDERR "[array_fields.pql ' . localtime() . "] Init";
use constant VERBOSE => int 10000;
use constant LAST_ICELL => int 14;
my @I_VAL;
my @O_VAL;
foreach my $f (1..9) { $O_VAL[$f] = undef; }
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_salesman_list     => int 8;
use constant _I_num_salesmen      => int 9;
use constant _I_salesmen_sorted   => int 10;
use constant _I_salesmen_sorted_2 => int 11;
use constant _I_salesmen_uniq     => int 12;
use constant _I_salesmen_uniq_2   => int 13;
use constant _I_salesmen_reverse  => int 14;
use constant _O_location          => int 1;
use constant _O_product_code      => int 2;
use constant _O_salesman_list     => int 3;
use constant _O_num_salesmen      => int 4;
use constant _O_salesmen_sorted   => int 5;
use constant _O_salesmen_sorted_2 => int 6;
use constant _O_salesmen_uniq     => int 7;
use constant _O_salesmen_uniq_2   => int 8;
use constant _O_salesmen_reverse  => int 9;
open(DATA, q{cat - | sort -t'|' -y -k 1,1 |}) || die "Cannot open input: $!";
&PrintHeader();
print STDERR "[array_fields.pql ' . localtime() . "] Start";
use Benchmark;
my $benchmark_start = new Benchmark;
while (<DATA>)
{
    print STDERR "[array_fields.pql ' . localtime() . "] $. records." if ($. % VERBOSE == 0);
    chomp;
    @I_VAL = split("[|]", $_);
    $O_VAL[_O_location] = $I_VAL[_I_location];
    $O_VAL[_O_product_code] = $I_VAL[_I_product_code];
    $O_VAL[_O_salesman_list] = $I_VAL[_I_salesman_list];
    $I_VAL[_I_num_salesmen] = int(split(/\s*,\s*/, $I_VAL[_I_salesman_list], -1));
    $O_VAL[_O_num_salesmen] = $I_VAL[_I_num_salesmen];
    $I_VAL[_I_salesmen_sorted] = join(',', sort(split(/\s*,\s*/, $I_VAL[_I_salesman_list], -1)));
    $O_VAL[_O_salesmen_sorted] = $I_VAL[_I_salesmen_sorted];
    $I_VAL[_I_salesmen_sorted_2] = join(',', sort(split(/\s*,\s*/, $I_VAL[_I_salesman_list], -1)));
    $O_VAL[_O_salesmen_sorted_2] = $I_VAL[_I_salesmen_sorted_2];
    $I_VAL[_I_salesmen_uniq] = &{sub { my %uniq; foreach (split(/\s*,\s*/, $I_VAL[_I_salesman_list], -1)) { $uni
q{$_}++; } return join(',', keys %uniq); }};
    $O_VAL[_O_salesmen_uniq] = $I_VAL[_I_salesmen_uniq];
    $I_VAL[_I_salesmen_uniq_2] = &{sub { my %uniq; foreach (split(/\s*,\s*/, $I_VAL[_I_salesman_list], -1)) { $u
niq{$_}++; } return join(',', keys %uniq); }};
    $O_VAL[_O_salesmen_uniq_2] = $I_VAL[_I_salesmen_uniq_2];
    $I_VAL[_I_salesmen_reverse] = join(',', reverse(split(/\s*,\s*/, join(',', sort(split(/\s*,\s*/, $I_VAL[_I_s
alesman_list], -1))), -1));
    $O_VAL[_O_salesmen_reverse] = $I_VAL[_I_salesmen_reverse];
    print
}
```

```

        $O_VAL[_O_location],
        $O_VAL[_O_product_code],
        $O_VAL[_O_salesman_list],
        $O_VAL[_O_num_salesmen],
        $O_VAL[_O_salesmen_sorted],
        $O_VAL[_O_salesmen_sorted_2],
        $O_VAL[_O_salesmen_uniq],
        $O_VAL[_O_salesmen_uniq_2],
        $O_VAL[_O_salesmen_reverse]
    ;
}

print STDERR '[array_fields.pql ' . localtime() . "] $. records.>";
my $benchmark_end = new Benchmark;
my $benchmark_timediff = timediff($benchmark_start, $benchmark_end);
print STDERR '[array_fields.pql ' . localtime() . "] Code statistics: @{[timestr($benchmark_timediff)]}";
#-----
sub PrintHeader
{
    local $\="\n";
    local $,="| ";
    print
        'location',
        'product_code',
        'salesman_list',
        'num_salesmen',
        'salesmen_sorted',
        'salesmen_sorted_2',
        'salesmen_uniq',
        'salesmen_uniq_2',
        'salesmen_reverse'
    ;
}

```

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

