examples/copy_output.pql by Pequel

sample@youraddress.com

Copy Output Record Example Script

examples/copy_output.pql	Copy Output Record Example Script

Table of Contents Copy Output Record Example Script

SCRIPT NAME	1
DESCRIPTION	1
1. PROCESS DETAILS	1
1.1 LOCATION	1
Description	1
Derived Input Field Evaluation	1
1.2 DESCRIPTION	1
Description	1
Derived Input Field Evaluation	1
1.3 SALES_TOTAL	1
Description	1
2. CONFIGURATION SETTINGS	2
2.1 prefix	2
2.2 pequeldoc	2
2.3 detail	2
2.4 script_name	2
2.5 input_file	2
2.6 optimize	2
2.7 doc_title	2
2.8 doc_email	2
2.9 doc_version	2
3. TABLES	3
3.1 LOC_DESCRIPT	3
Data	3
4. TABLE INFORMATION SUMMARY	4
4.1 Table List Sorted By Table Name	4
5. EXAMPLES/COPY_OUTPUT.PQL	5
options	5
init table	5
input section	5
divert record(pequel:copy_output_WA.pql)	5
divert record(pequel:copy_output_SA.pql)	5
divert record(pequel:copy_output_NSW.pql)	5
divert record(pequel:copy_output_VIC.pql)	5
divert record(pequel:copy_output_NT.pql)	5
filter	5
group by	5
output section	5
having	5
sort output	5
6. PEQUEL GENERATED PROGRAM	6
7. ABOUT PEQUEL	17
COPYRIGHT	17

ii

SCRIPT NAME

examples/copy_output.pql

DESCRIPTION

1. PROCESS DETAILS

Input records are read from chain_pequel_pt1.pql. The input record contains **3** fields. Fields are delimited by the '|' character.

Output records are written to standard output. The output record contains **3** fields. Fields are delimited by the '|' character.

Input records are eliminated (filtered) unless LOCATION eq 'WA' || LOCATION eq 'SA' || LOCATION eq 'NSW' || LOCATION eq 'VIC' || LOCATION eq 'NT'.

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

Output aggregated records are eliminated unless having SALES_TOTAL 0.

1.1 LOCATION

Output Field

Description

Set to input field **LOCATION_DESC**

Derived Input Field Evaluation

=> %LOC_DESCRIPT(LOCATION)

1.2 DESCRIPTION

Output Field

Description

Set to input field **DESCRIPTION**

Derived Input Field Evaluation

=> 'State Total'

1.3 SALES_TOTAL

Output Field

Description

Sum aggregation on input field **SALES_TOTAL**.

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/copy_output.pql

2.5 input file

input data filename: chain_pequel_pt1.pql

2.6 optimize

optimize generated code.: 1

2.7 doc_title

document title.: Copy Output Record Example Script

2.8 doc_email

document email entry.: sample@youraddress.com

2.9 doc_version

document version for pequel script.: 2.3

3. TABLES

3.1 LOC_DESCRIPT

Table Type: Iocal

Data

NSW — New South Wales

WA — Western Australia

SA — South Australia

NT — Northern Territory

 $\mathsf{QLD}-\mathsf{Queensland}$

VIC — Victoria

4. TABLE INFORMATION SUMMARY

4.1 Table List Sorted By Table Name LOC_DESCRIPT — *1 (local)*

5. EXAMPLES/COPY_OUTPUT.PQL

options

```
prefix(examples)
pequeldoc(pdf)
detail(1)
script_name(examples/copy_output.pql)
input_file(chain_pequel_pt1.pql)
optimize(1)
doc_title(Copy Output Record Example Script)
doc_email(sample@youraddress.com)
doc_version(2.3)
```

init table

```
LOC_DESCRIPT NSW New South Wales
LOC_DESCRIPT WA Western Australia
LOC_DESCRIPT SA South Australia
LOC_DESCRIPT NT Northern Territory
LOC_DESCRIPT QLD Queensland
LOC_DESCRIPT VIC Victoria
```

input section

```
LOCATION
PRODUCT_CODE
SALES_TOTAL
LOCATION_DESC => %LOC_DESCRIPT(LOCATION)
DESCRIPTION => 'State Total'
```

divert record(pequel:copy_output_WA.pql)

```
LOCATION eq 'WA'
```

divert record(pequel:copy_output_SA.pql)

```
LOCATION eq 'SA'
```

divert record(pequel:copy_output_NSW.pql)

```
LOCATION eq 'NSW'
```

divert record(pequel:copy_output_VIC.pql)

```
LOCATION eq 'VIC'
```

divert record(pequel:copy_output_NT.pql)

```
LOCATION eq 'NT'
```

filter

```
\texttt{LOCATION eq 'WA' || LOCATION eq 'SA' || LOCATION eq 'NSW' || LOCATION eq 'VIC' || LOCATION eq 'NT' ||
```

group by

LOCATION string

output section

```
string LOCATION LOCATION_DESC string DESCRIPTION DESCRIPTION decimal SALES_TOTAL sum SALES_TOTAL
```

having

SALES_TOTAL > 0

sort output

```
LOCATION string
SALES_TOTAL numeric des
```

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 : copy_output.pql
#Created On : Wed Nov 16 13:57:33 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/copy_output.pql) script filename
#input_file(chain_pequel_pt1.pql) input data filename
#optimize(1) optimize generated code.
#doc_title(Copy Output Record Example Script) document title.
\verb|#doc_email(sample@youraddress.com)| document email entry.
#doc_version(2.3) document version for pequel script.
#-----
                                                                                       use strict;
use Fcntl ':flock';
use constant \_I\_LOCATION
                                                   => int
                                                                   0;
use constant _I_PRODUCT_CODE
                                                   => int.
                                                                   1;
use constant _I_SALES_TOTAL
                                                    => int
                                                                     2:
use constant _I_LOCATION_DESC => int
                                                                     3;
use constant _I_DESCRIPTION
                                                   => int
                                                                     4;
use constant _O_LOCATION
                                                   => int.
                                                                     1;
use constant _O_DESCRIPTION
                                                 => int
                                                                     2;
use constant _O_SALES_TOTAL
                                                   => int
                                                                     3;
use constant _T_LOC_DESCRIPT_FLD_1
                                                          => int
                                                                          0;
use constant _I_LOC_DESCRIPT_LOCATION_FLD_KEY => int
                                                                                         5;
use constant _I_LOC_DESCRIPT_LOCATION_FLD_1 => int
local $\="\n";
local $,="|";
print STDERR '[examples/copy_output.pql ' . localtime() . "] Init";
use constant VERBOSE => int 10000;
use constant LAST ICELL => int 4;
my @I VAL;
my @O_VAL;
my $_inprecs=0;
my $key__I_LOCATION;
my $previous_key__I_LOCATION = undef;
foreach my $f (1..3) { $O_VAL[$f] = undef; }
\verb|my $$_TABLE\_LOC\_DESCRIPT = \&InitLookupLOC\_DESCRIPT; # ref to $$LOC\_DESCRIPT hash | lookupLoC\_DESCRIPT | lookup
if (open(READ_CHAIN_PEQUEL_PT1, '-|') == 0) # Fork -- read from child
{
      &p_read_chain_pequel_pt1::read_chain_pequel_pt1;
      exit(0);
}
open(STDOUT, '|-', q{sort -t'|' -y -k 1,1 -k 3nr,3nr 2>/dev/null});
if (open(DIVERT INPUT COPY OUTPUT WA, ' | -') == 0) # Fork -- write to child
      &p_divert_input_copy_output_wa::divert_input_copy_output_wa;
      exit(0);
}
if (open(DIVERT_INPUT_COPY_OUTPUT_SA, '|-') == 0) # Fork -- write to child
{
      &p_divert_input_copy_output_sa::divert_input_copy_output_sa;
      exit(0);
if (open(DIVERT_INPUT_COPY_OUTPUT_NSW, '|-') == 0) # Fork -- write to child
{
      &p divert input copy output nsw::divert input copy output nsw;
      exit(0);
}
if (open(DIVERT_INPUT_COPY_OUTPUT_VIC, '|-') == 0) # Fork -- write to child
      &p_divert_input_copy_output_vic::divert_input_copy_output_vic;
      exit(0);
if (open(DIVERT_INPUT_COPY_OUTPUT_NT, '|-') == 0) # Fork -- write to child
```

```
&p_divert_input_copy_output_nt::divert_input_copy_output_nt;
    exit(0);
print STDERR '[examples/copy_output.pql ' . localtime() . "] Start";
use Benchmark;
my Sbenchmark start = new Benchmark;
while (<READ_CHAIN_PEQUEL_PT1>)
    ++$ inprecs;
   print STDERR '[examples/copy_output.pql ' . localtime() . "] $_inprecs records." if ($_inprecs % VERBOSE =
    chomp;
    @I_VAL = split("[|]", $_);
next unless ($I_VAL[_I_LOCATION] eq 'WA' || $I_VAL[_I_LOCATION] eq 'SA' || $I_VAL[_I_LOCATION] eq 'NSW' || $I_VAL[_I_LOCATION] eq 'VIC' || $I_VAL[_I_LOCATION] eq 'NT');
    if (($I_VAL[_I_LOCATION] eq 'WA'))
        print DIVERT INPUT COPY OUTPUT WA $ ;
       next;
    }
    if (($I VAL[ I LOCATION] eq 'SA'))
        print DIVERT_INPUT_COPY_OUTPUT_SA $_;
       next;
    if (($I_VAL[_I_LOCATION] eq 'NSW'))
        print DIVERT_INPUT_COPY_OUTPUT_NSW $_;
        next;
    }
    if (($I_VAL[_I_LOCATION] eq 'VIC'))
    {
        print DIVERT_INPUT_COPY_OUTPUT_VIC $_;
        next;
    if (($I_VAL[_I_LOCATION] eq 'NT'))
        print DIVERT_INPUT_COPY_OUTPUT_NT $_;
    $key__I_LOCATION = $I_VAL[_I_LOCATION];
    if (!defined($previous_key__I_LOCATION))
        $previous_key__I_LOCATION = $key__I_LOCATION;
    elsif ($previous_key__I_LOCATION ne $key__I_LOCATION)
        flock(STDOUT, LOCK_EX);
        print STDOUT
           $O_VAL[_O_LOCATION],
            $0_VAL[_O_DESCRIPTION],
            $0_VAL[_O_SALES_TOTAL]
        if
            $0_VAL[_O_SALES_TOTAL] > 0
        flock(STDOUT, LOCK_UN);
        $previous_key__I_LOCATION = $key__I_LOCATION;
        @O_VAL = undef;
    $0_VAL[_O_LOCATION] = $I_VAL[_I_LOCATION_DESC];
    $I_VAL[_I_DESCRIPTION] = 'State Total';
    $0_VAL[_O_DESCRIPTION] = $I_VAL[_I_DESCRIPTION];
    $0_VAL[_O_SALES_TOTAL] += $1_VAL[_I_SALES_TOTAL] unless ($1_VAL[_I_SALES_TOTAL] eq '');
flock(STDOUT, LOCK_EX);
print STDOUT
    $0_VAL[_O_LOCATION];
    $0_VAL[_O_DESCRIPTION],
    $0_VAL[_O_SALES_TOTAL]
if
    $0_VAL[_O_SALES_TOTAL] > 0
flock(STDOUT, LOCK_UN);
```

8

```
close(DIVERT INPUT COPY OUTPUT NT);
close(DIVERT INPUT COPY OUTPUT VIC);
close(DIVERT INPUT COPY OUTPUT NSW);
close(DIVERT INPUT COPY OUTPUT SA);
close(DIVERT INPUT COPY OUTPUT WA);
close(STDOUT);
close(READ CHAIN PEOUEL PT1);
print STDERR '[examples/copy_output.pql ' . localtime() . "] $_inprecs records.";
my $benchmark end = new Benchmark;
my $benchmark_timediff = timediff($benchmark_start, $benchmark_end);
print STDERR '[examples/copy_output.pql ' . localtime() . "] Code statistics: @{[timestr($benchmark_timediff)]
#+++++ Table LOC_DESCRIPT --> Type :ETL::Pequel::Type::Table::Local +++++
sub InitLookupLOC DESCRIPT
   my %_TABLE_LOC_DESCRIPT;
    % TABLE LOC DESCRIPT =
       'NSW' => 'New South Wales',
        'NT' => 'Northern Territory',
       'QLD' => 'Queensland',
       'SA' => 'South Australia',
'VIC' => 'Victoria',
       'WA' => 'Western Australia'
    );
   return \%_TABLE_LOC_DESCRIPT;
}
{
   package p_read_chain_pequel_pt1;
   sub read_chain_pequel_pt1
    !/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 : chain_pequel_pt1.pql
    Created On : Wed Nov 16 13:57:22 2005
    Perl Version: /usr/bin/perl 5.6.1 on solaris
    Options:
       input_file(sample.data) input data filename
        {\tt optimize}(1) optimize generated code.
        doc_title(Pequel Chaining Part-1 Example Script) document title.
        doc_email(sample@youraddress.com) document email entry.
        doc_version(2.3) document version for pequel script.
       use strict;
       use constant _I_PRODUCT_CODE
                                      => int
       use constant _I_COST_PRICE
                                      => int
       use constant _I_DESCRIPTION
                                       => int
       use constant _I_SALES_CODE
                                      => int
       use constant _I_SALES_PRICE
                                      => int
       use constant _I_SALES_QTY
                                      => int
                                      => int
       use constant _I_SALES_DATE
       use constant _I_LOCATION
                                      => int
       use constant _I_SALES_TOTAL
                                      => int
       use constant _O_LOCATION
                                      => int
                                                1;
       use constant _O_PRODUCT_CODE
       use constant _O_SALES_TOTAL
                                      => int
       local $\="\n";
       local $,="|";
       print STDERR '[examples/chain_pequel_pt1.pql ' . localtime() . "] Init";
       use constant VERBOSE => int 10000;
       use constant LAST_ICELL => int 8;
       my @I_VAL;
       my @O_VAL;
       my $_inprecs=0;
       my $key__I_LOCATION;
       my $previous_key__I_LOCATION = undef;
       my $key__I_PRODUCT_CODE;
       my $previous_key__I_PRODUCT_CODE = undef;
       foreach my $f (1..3) { $0_VAL[$f] = undef; }
     Sort:LOCATION(asc:string) PRODUCT_CODE(asc:string)
open(DATA, q{sort -t'|' -y -k 8,8 -k 1,1 examples/sample.data 2>/dev/null |});
open(STDOUT, '|-', q{sort -t'|' -y -k 1,1 2>/dev/null});
print STDERR '[examples/chain_pequel_pt1.pql ' . localtime() . "] Start";
       use Benchmark;
       my $benchmark_start = new Benchmark;
       while (<DATA>)
```

```
++$ inprecs;
                     print \ STDERR \ '[examples/chain\_pequel\_pt1.pql \ ' . \ local time() \ . \ "] \ \$\_inprecs \ records." \ if \ (\$\_inprecs \ records.") \ for \ inprecs \ records.
s % VERBOSE == 0);
                    chomp;
                    @I VAL = split("[|]", $ );
                    $key_I_LOCATION = $I_VAL[_I_LOCATION];
$key_I_PRODUCT_CODE = $I_VAL[_I_PRODUCT_CODE];
                    if (!defined($previous_key__I_NOCATION) || !defined($previous_key__I_PRODUCT_CODE))
                            $previous_key__I_LOCATION = $key__I_LOCATION;
                            $previous_key__I_PRODUCT_CODE = $key__I_PRODUCT_CODE;
                    }
                    elsif ($previous key I LOCATION ne $key I LOCATION | | $previous key I PRODUCT CODE ne $key I PRODUCT CODE ne $key
RODUCT_CODE)
                     {
                           print STDOUT
                                  $0_VAL[_O_LOCATION],
                                  $0_VAL[_O_PRODUCT_CODE],
                                  SO VALI O SALES TOTAL1
                            $previous_key__I_LOCATION = $key__I_LOCATION;
                            $previous_key__I_PRODUCT_CODE = $key__I_PRODUCT_CODE;
                            @O_VAL = undef;
                     }
                    $0_VAL[_O_LOCATION] = $I_VAL[_I_LOCATION];
                    $O_VAL[_O_PRODUCT_CODE] = $I_VAL[_I_PRODUCT_CODE];
                     $I_VAL[_I_SALES_TOTAL] = $I_VAL[_I_SALES_QTY] * $I_VAL[_I_SALES_PRICE];
                    $0_VAL[_0_SALES_TOTAL] += $I_VAL[_I_SALES_TOTAL] unless ($I_VAL[_I_SALES_TOTAL] eq '');
              print STDOUT
                    $0_VAL[_O_LOCATION],
                    $0_VAL[_O_PRODUCT_CODE],
                    $0_VAL[_O_SALES_TOTAL]
             close(STDOUT);
             close(DATA);
             print STDERR '[examples/chain_pequel_pt1.pql ' . localtime() . "] $_inprecs records.";
              my $benchmark_end = new Benchmark;
              my $benchmark_timediff = timediff($benchmark_start, $benchmark_end);
             print STDERR '[examples/chain_pequel_pt1.pql ' . localtime() . "] Code statistics: @{[timestr($benchma
rk_timediff)]}";
}
{
      package p_divert_input_copy_output_sa;
      sub divert_input_copy_output_sa
        !/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 : copy_output_SA.pql
        Created On : Wed Nov 16 13:57:27 2005
        Perl Version: /usr/bin/perl 5.6.1 on solaris
optimize(1) optimize generated code.
               hash(1) Generate in memory. Input data can be unsorted.
               doc_title(Copy Output Record Example Script) document title.
               doc_email(sample@youraddress.com) document email entry.
               {\tt doc\_version(2.3)} document version for pequel script.
     use strict;
              use Fcntl ':flock';
              use constant _I_LOCATION
                                                                      => int
                                                                                      0;
              use constant _I_PRODUCT_CODE
                                                                      => int
                                                                                        1;
              use constant _I_SALES_TOTAL
                                                                      => int
                                                                                       2;
              use constant _I_LOCATION_NAME
                                                                       => int
                                                                                         3;
              use constant _O_LOCATION_NAME
                                                                       => int
                                                                                        1;
              use constant _O_PRODUCT_CODE
                                                                       => int
                                                                                        2;
              use constant _O_SALES_TOTAL
                                                                      => int
              local \= \n";
              local $,="|";
              print STDERR '[examples/copy_output_SA.pql ' . localtime() . "] Init";
              use constant VERBOSE => int 10000;
              use constant LAST_ICELL => int 3;
```

```
my @I_VAL;
      my %O_VAL;
      mv škev;
      my $ inprecs=0;
       if (open(COPY_OUTPUT_COPY_OUTPUT_COMBINER, '|-') == 0) # Fork -- write to child
       {
          &p_copy_output_copy_output_combiner::copy_output_copy_output_combiner;
          exit(0);
       }
       print STDERR '[examples/copy_output_SA.pql ' . localtime() . "] Start";
       use Benchmark;
       my $benchmark start = new Benchmark;
       while (<STDIN>)
          ++$ inprecs;
          print STDERR '[examples/copy_output_SA.pql ' . localtime() . "] $_inprecs records." if ($_inprecs
% VERBOSE == 0);
          chomp;
          @I_VAL = split("[|]", $_);
          $key = ( $I_VAL[_I_PRODUCT_CODE] );
          $I_VAL[_I_LOCATION_NAME] = 'South Australia';
          $O_VAL{$key}{_O_LOCATION_NAME} = $I_VAL[_I_LOCATION_NAME];
$O_VAL{$key}{_O_PRODUCT_CODE} = $I_VAL[_I_PRODUCT_CODE];
          $O_VAL{$key}{_O_SALES_TOTAL} += $I_VAL[_I_SALES_TOTAL] unless ($I_VAL[_I_SALES_TOTAL] eq '');
       }
       foreach $key (sort keys %O_VAL)
          flock(STDOUT, LOCK_EX);
          print STDOUT
              $O_VAL{$key}{_O_LOCATION_NAME},
              $O_VAL{$key}{_O_PRODUCT_CODE},
              $O_VAL{$key}{_O_SALES_TOTAL}
          flock(STDOUT, LOCK_UN);
          if ($O_VAL{$key}{_O_SALES_TOTAL} > 0)
              flock(COPY_OUTPUT_COPY_OUTPUT_COMBINER, LOCK_EX);
              print COPY_OUTPUT_COPY_OUTPUT_COMBINER
                 O_VAL{skey}_{O_LOCATION_NAME},
                 $0_VAL{$key}{_O_PRODUCT_CODE},
                 $0_VAL{$key}{_O_SALES_TOTAL}
              flock(COPY_OUTPUT_COPY_OUTPUT_COMBINER, LOCK_UN);
          }
       }
       close(COPY_OUTPUT_COPY_OUTPUT_COMBINER);
       print STDERR '[examples/copy_output_SA.pql ' . localtime() . "] $_inprecs records.";
       my $benchmark_end = new Benchmark;
       my $benchmark_timediff = timediff($benchmark_start, $benchmark_end);
      print STDERR '[examples/copy_output_SA.pql ' . localtime() . "] Code statistics: @{[timestr($benchmark
_timediff)]}";
              }
{
   package p_divert_input_copy_output_wa;
   sub divert_input_copy_output_wa
#
    !/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 : copy_output_WA.pql
    Created On : Wed Nov 16 13:57:25 2005
    Perl Version: /usr/bin/perl 5.6.1 on solaris
#
    Options:
       optimize(1) optimize generated code.
       hash(1) Generate in memory. Input data can be unsorted.
       doc_title(Copy Output Record Example Script) document title.
       doc_email(sample@youraddress.com) document email entry.
       doc_version(2.3) document version for pequel script.
                                                    use strict;
       use Fcntl ':flock';
```

```
use constant \_I\_LOCATION
                                        => int
                                                  0;
       use constant _I_PRODUCT_CODE
                                       => int
                                                  1;
       use constant _I_SALES_TOTAL use constant _I_LOCATION_NAME
                                       => int.
                                                  2;
                                       => int
                                                  3;
       use constant _O_LOCATION_NAME
                                       => int
                                                  1;
       use constant _O_PRODUCT_CODE
                                       => int
                                                  2;
       use constant _O_SALES_TOTAL
                                       => int
                                                  3;
       local $\="\n";
       local $,="|";
       print STDERR '[examples/copy_output_WA.pql ' . localtime() . "] Init";
       use constant VERBOSE => int 10000;
       use constant LAST_ICELL => int 3;
       my @I VAL;
       my %O_VAL;
       my $key;
       my $ inprecs=0;
       if (open(COPY_OUTPUT_COPY_OUTPUT_COMBINER, '|-') == 0) # Fork -- write to child
       {
           &p copy output copy output combiner::copy output copy output combiner;
           exit(0);
       }
       print STDERR '[examples/copy output WA.pql ' . localtime() . "] Start";
       use Benchmark;
       my $benchmark_start = new Benchmark;
       while (<STDIN>)
            ++$_inprecs;
           print STDERR '[examples/copy_output_WA.pql ' . localtime() . "] $_inprecs records." if ($_inprecs
% VERBOSE == 0);
           chomp;
           @I_VAL = split("[|]", $_);
           $key = ( $I_VAL[_I_PRODUCT_CODE] );
            $I_VAL[_I_LOCATION_NAME] = 'Western Australia';
            $0_VAL{$key}{_O_LOCATION_NAME} = $I_VAL[_I_LOCATION_NAME];
            $0_VAL{$key}{_0_PRODUCT_CODE} = $I_VAL[_I_PRODUCT_CODE];
            $O_VAL{$key}{_O_SALES_TOTAL} += $I_VAL[_I_SALES_TOTAL] unless ($I_VAL[_I_SALES_TOTAL] eq '');
       }
       foreach $key (sort keys %O_VAL)
            flock(STDOUT, LOCK_EX);
           print STDOUT
               $O_VAL{$key}{_O_LOCATION_NAME},
               $0_VAL{$key}{_O_PRODUCT_CODE},
               $0_VAL{$key}{_O_SALES_TOTAL}
           flock(STDOUT, LOCK_UN);
            if ($O_VAL{$key}{_O_SALES_TOTAL} > 0)
               flock(COPY_OUTPUT_COPY_OUTPUT_COMBINER, LOCK_EX);
               print COPY_OUTPUT_COPY_OUTPUT_COMBINER
                   $0_VAL{$key}{_O_LOCATION_NAME},
$0_VAL{$key}{_O_PRODUCT_CODE},
                   $0_VAL{$key}{_O_SALES_TOTAL}
               flock(COPY_OUTPUT_COPY_OUTPUT_COMBINER, LOCK_UN);
            }
       }
       close(COPY OUTPUT COPY OUTPUT COMBINER);
       print STDERR '[examples/copy_output_WA.pql ' . localtime() . "] $_inprecs records.";
       my $benchmark_end = new Benchmark;
       my $benchmark_timediff = timediff($benchmark_start, $benchmark_end);
       print STDERR '[examples/copy_output_WA.pql ' . localtime() . "] Code statistics: @{[timestr($benchmark
_timediff)]}";
                #-+-+-+-+
}
{
   package p_divert_input_copy_output_nt;
    sub divert_input_copy_output_nt
#
     !/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 : copy_output_NT.pql
    Created On : Wed Nov 16 13:57:32 2005
```

```
Perl Version: /usr/bin/perl 5.6.1 on solaris
    For
#
    Options:
#
       optimize(1) optimize generated code.
       hash(1) Generate in memory. Input data can be unsorted.
       doc title(Copy Output Record Example Script) document title.
       doc email(sample@youraddress.com) document email entry.
       doc version(2.3) document version for pequel script.
#-+-+-+-+-+-+-+-+-
                                                       use strict;
      use Fcntl ':flock';
       use constant _I_LOCATION
                                    => int
                                             0;
       use constant _I_PRODUCT_CODE
                                   => int
                                             1;
      2;
                                             3;
       use constant _O_LOCATION_NAME
                                   => int
                                             1;
       use constant _O_PRODUCT_CODE
                                   => int
                                             2;
       use constant O SALES TOTAL
                                   => int
                                             3;
       local $\="\n";
       local $,="|";
       print STDERR '[examples/copy_output_NT.pql ' . localtime() . "] Init";
       use constant VERBOSE => int 10000;
       use constant LAST_ICELL => int 3;
       my @I_VAL;
       my %O_VAL;
       my $key;
       my $ inprecs=0;
       if (open(COPY_OUTPUT_COPY_OUTPUT_COMBINER, '|-') == 0) # Fork -- write to child
       {
          &p_copy_output_copy_output_combiner::copy_output_copy_output_combiner;
          exit(0);
       }
       print STDERR '[examples/copy_output_NT.pql ' . localtime() . "] Start";
       use Benchmark;
       my $benchmark_start = new Benchmark;
       while (<STDIN>)
           ++$_inprecs;
          print STDERR '[examples/copy_output_NT.pql ' . localtime() . "] $_inprecs records." if ($_inprecs
% VERBOSE == 0);
          chomp;
          @I_VAL = split("[|]", $_);
          $key = ( $I_VAL[_I_PRODUCT_CODE] );
          $I_VAL[_I_LOCATION_NAME] = 'Northern Territory';
          $O_VAL{$key}{_O_LOCATION_NAME} = $I_VAL[_I_LOCATION_NAME];
          $0_VAL{$key}{_O_PRODUCT_CODE} = $I_VAL[_I_PRODUCT_CODE];
           $O_VAL{$key}{_O_SALES_TOTAL} += $I_VAL[_I_SALES_TOTAL] unless ($I_VAL[_I_SALES_TOTAL] eq '');
       }
       foreach $key (sort keys %O_VAL)
          flock(STDOUT, LOCK_EX);
              $O_VAL{$key}{_O_LOCATION_NAME},
              $O_VAL{$key}{_O_PRODUCT_CODE},
              $0_VAL{$key}{_O_SALES_TOTAL}
          flock(STDOUT, LOCK_UN);
          if ($O_VAL{$key}{_O_SALES_TOTAL} > 0)
              flock(COPY_OUTPUT_COPY_OUTPUT_COMBINER, LOCK_EX);
              print COPY_OUTPUT_COPY_OUTPUT_COMBINER
                 O_VAL{skey}_{O_LOCATION_NAME},
                  $0_VAL{$key}{_O_PRODUCT_CODE},
                 $0_VAL{$key}{_0_SALES_TOTAL}
              flock(COPY_OUTPUT_COPY_OUTPUT_COMBINER, LOCK_UN);
          }
       }
       close(COPY_OUTPUT_COPY_OUTPUT_COMBINER);
       print STDERR '[examples/copy_output_NT.pql ' . localtime() . "] $_inprecs records.";
       my $benchmark_end = new Benchmark;
       my $benchmark_timediff = timediff($benchmark_start, $benchmark_end);
       print STDERR '[examples/copy_output_NT.pql ' . localtime() . "] Code statistics: @{[timestr($benchmark
_timediff)]}";
#-+-+-+-+
               }
```

```
{
   package p_divert_input_copy_output vic;
   sub divert_input_copy_output_vic
    !/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 : copy output VIC.pgl
    Created On : Wed Nov 16 13:57:30 2005
    Perl Version: /usr/bin/perl 5.6.1 on solaris
    For
Options:
        optimize(1) optimize generated code.
        hash(1) Generate in memory. Input data can be unsorted.
        doc title(Copy Output Record Example Script) document title.
       doc email(sample@vouraddress.com) document email entry.
       doc_version(2.3) document version for pequel script.
       use strict;
       use Fcntl ':flock';
       use constant _I_LOCATION
                                      => int
                                                0;
       use constant _I_PRODUCT_CODE => int
                                                1;
       use constant _I_SALES_TOTAL use constant _I_LOCATION_NAME
                                      => int
                                     => int
                                                 3;
       use constant _O_LOCATION_NAME => int
use constant _O_PRODUCT_CODE => int
                                      => int
                                                 1;
                                                 2;
       use constant _O_SALES_TOTAL
                                      => int
       local $\="\n";
       local $,="|";
       print STDERR '[examples/copy_output_VIC.pql ' . localtime() . "] Init";
       use constant VERBOSE => int 10000;
       use constant LAST_ICELL => int 3;
       my @I_VAL;
       my %O_VAL;
       my $key;
       my $_inprecs=0;
       if (open(COPY\_OUTPUT\_COPY\_OUTPUT\_COMBINER, '|-') == 0) # Fork -- write to child
       {
           &p_copy_output_copy_output_combiner::copy_output_copy_output_combiner;
           exit(0);
       }
       print STDERR '[examples/copy_output_VIC.pql ' . localtime() . "] Start";
       use Benchmark;
       my $benchmark_start = new Benchmark;
       while (<STDIN>)
           ++$_inprecs;
           print STDERR '[examples/copy_output_VIC.pql ' . localtime() . "] $_inprecs records." if ($_inprecs
 % VERBOSE == 0);
           chomp;
           @I_VAL = split("[|]", $_);
           $key = ( $I_VAL[_I_PRODUCT_CODE] );
           $I_VAL[_I_LOCATION_NAME] = 'Victoria';
           $0_VAL{$key}{_O_LOCATION_NAME} = $I_VAL[_I_LOCATION_NAME];
           $O_VAL{$key}{_O_PRODUCT_CODE} = $I_VAL[_I_PRODUCT_CODE];
           $O_VAL{$key}{_O_SALES_TOTAL} += $I_VAL[_I_SALES_TOTAL] unless ($I_VAL[_I_SALES_TOTAL] eq '');
       foreach $key (sort keys %O_VAL)
           flock(STDOUT, LOCK_EX);
           print STDOUT
               $0_VAL{$key}{_O_LOCATION_NAME},
$0_VAL{$key}{_O_PRODUCT_CODE},
               O_VAL{skey}{O_SALES_TOTAL}
           flock(STDOUT, LOCK_UN);
           if ($O_VAL{$key}{_O_SALES_TOTAL} > 0)
               flock(COPY_OUTPUT_COPY_OUTPUT_COMBINER, LOCK_EX);
               print COPY_OUTPUT_COPY_OUTPUT_COMBINER
                   $O_VAL{$key}{_O_LOCATION_NAME},
                   $0_VAL{$key}{_O_PRODUCT_CODE},
                   $0_VAL{$key}{_O_SALES_TOTAL}
               flock(COPY_OUTPUT_COPY_OUTPUT_COMBINER, LOCK_UN);
           }
       }
```

```
close(COPY OUTPUT COPY OUTPUT COMBINER);
       close(STDIN);
       print STDERR '[examples/copy_output_VIC.pql ' . localtime() . "] $_inprecs records.";
       my $benchmark_end = new Benchmark;
       my $benchmark_timediff = timediff($benchmark_start, $benchmark_end);
       print STDERR '[examples/copy_output_VIC.pql ' . localtime() . "] Code statistics: @{[timestr($benchmar
k timediff) | } ";
#-----
}
{
   package p divert input copy output nsw;
   sub divert_input_copy_output_nsw
    !/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 : copy output NSW.pgl
    Created On : Wed Nov 16 13:57:29 2005
    Perl Version: /usr/bin/perl 5.6.1 on solaris
   Options:
       optimize(1) optimize generated code.
       hash(1) Generate in memory. Input data can be unsorted.
       doc_title(Copy Output Record Example Script) document title.
       doc_email(sample@youraddress.com) document email entry.
       doc_version(2.3) document version for pequel script.
#-+-+-+-+-
       use strict;
       use Fcntl ':flock';
       use constant _I_LOCATION
                                    => int
       use constant _I_PRODUCT_CODE => int
                                             1;
       use constant _I_SALES_TOTAL => int
use constant _I_LOCATION_NAME => int
                                    => int
                                             3;
       use constant _O_LOCATION_NAME => int
use constant _O_PRODUCT_CODE => int
       use constant _O_SALES_TOTAL
                                    => int
       local \= \n'';
       local $,="|";
       print STDERR '[examples/copy_output_NSW.pql ' . localtime() . "] Init";
       use constant VERBOSE => int 10000;
       use constant LAST_ICELL => int 3;
       my @I_VAL;
       my %O_VAL;
       my $key;
       my $_inprecs=0;
       if (open(COPY\_OUTPUT\_COPY\_OUTPUT\_COMBINER, '|-') == 0) # Fork -- write to child
          &p_copy_output_copy_output_combiner::copy_output_copy_output_combiner;
       }
       print STDERR '[examples/copy_output_NSW.pql ' . localtime() . "] Start";
       my $benchmark_start = new Benchmark;
       while (<STDIN>)
           ++$_inprecs;
          print STDERR '[examples/copy_output_NSW.pql ' . localtime() . "] $_inprecs records." if ($_inprecs
% VERBOSE == 0);
          chomp;
          @I_VAL = split("[|]", $_);
           $key = ( $I_VAL[_I_PRODUCT_CODE] );
           $I_VAL[_I_LOCATION_NAME] = 'New South Wales';
           $O_VAL{$key}{_O_LOCATION_NAME} = $I_VAL[_I_LOCATION_NAME];
           $O_VAL{$key}{_O_PRODUCT_CODE} = $I_VAL[_I_PRODUCT_CODE];
           $O_VAL{$key}{_O_SALES_TOTAL} += $I_VAL[_I_SALES_TOTAL] unless ($I_VAL[_I_SALES_TOTAL] eq '');
       foreach $key (sort keys %O_VAL)
           flock(STDOUT, LOCK_EX);
          print STDOUT
              $O_VAL{$key}{_O_LOCATION_NAME},
              $0_VAL{$key}{_O_PRODUCT_CODE},
              $0_VAL{$key}{_O_SALES_TOTAL}
           flock(STDOUT, LOCK_UN);
           if ($O_VAL{$key}{_O_SALES_TOTAL} > 0)
```

```
{
                             flock(COPY_OUTPUT_COPY_OUTPUT_COMBINER, LOCK_EX);
                             print COPY_OUTPUT_COPY_OUTPUT_COMBINER
                                     $O_VAL{$key}{_O_LOCATION_NAME},
                                     $0_VAL{$key}{_O_PRODUCT_CODE},
                                    O_VAL{skey}{O_SALES_TOTAL}
                             flock(COPY OUTPUT COPY OUTPUT COMBINER, LOCK UN);
                      }
              }
              close(COPY OUTPUT COPY OUTPUT COMBINER);
              close(STDIN);
              print STDERR '[examples/copy_output_NSW.pql ' . localtime() . "] $_inprecs records.";
              my $benchmark end = new Benchmark;
              my $benchmark_timediff = timediff($benchmark_start, $benchmark_end);
               \texttt{print STDERR '[examples/copy\_output\_NSW.pql ' . local time() . "] Code statistics: @\{[\texttt{timestr(\$benchmarrow}] | \texttt{timestr(\$benchmarrow}] | \texttt{timestr(\$
k timediff)]}";
                                         }
}
{
       package p_copy_output_copy_output_combiner;
       sub copy_output_copy_output_combiner
        !/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 : copy_output_combiner.pql
        Created On : Wed Nov 16 13:57:25 2005
        Perl Version: /usr/bin/perl 5.6.1 on solaris
       For
#-----
        Options:
                optimize(1) optimize generated code.
                doc_title(Copy Output Record Example Script) document title.
                doc_email(sample@youraddress.com) document email entry.
              doc_version(2.3) document version for pequel script.
   use strict;
              use Fcntl ':flock';
              use constant _I_LOCATION_NAME => int
                                                                                              1;
              use constant _I_DESCRIPTION
                                                                          => int
                                                                                              3;
              use constant _O_LOCATION_NAME => int
                                                                                              1;
              use constant _O_DESCRIPTION => int
use constant _O_SALES_TOTAL => int
              local \= \n'';
              local $,="|";
              print STDERR '[examples/copy_output_combiner.pql ' . localtime() . "] Init";
              use constant VERBOSE => int 10000;
              use constant LAST_ICELL => int 3;
              my @I_VAL;
              my @O_VAL;
              my $_inprecs=0;
              my $key__I_LOCATION_NAME;
              my $previous_key__I_LOCATION_NAME = undef;
              foreach my $f (1..3) { $O_VAL[$f] = undef; }
          Sort:LOCATION_NAME(asc:string)
              open(DATA, q{cat - | sort -t'|' -y -k 1,1 2>/dev/null |}) || die "Cannot open input: !";
              print STDERR '[examples/copy_output_combiner.pql ' . localtime() . "] Start";
              use Benchmark;
              my $benchmark_start = new Benchmark;
              while (<DATA>)
                      print STDERR '[examples/copy_output_combiner.pql ' . localtime() . "] $_inprecs records." if ($_in
precs % VERBOSE == 0);
                      chomp;
                      @I_VAL = split("[|]", $_);
                      $key__I_LOCATION_NAME = $I_VAL[_I_LOCATION_NAME];
                      if (!defined($previous_key__I_LOCATION_NAME))
                             $previous_key__I_LOCATION_NAME = $key__I_LOCATION_NAME;
                      }
                      elsif ($previous_key__I_LOCATION_NAME ne $key__I_LOCATION_NAME)
```

```
flock(STDOUT, LOCK_EX);
               print STDOUT
                   $0_VAL[_O_LOCATION_NAME],
                   $0_VAL[_O_DESCRIPTION],
                  $0_VAL[_O_SALES_TOTAL]
               flock(STDOUT, LOCK_UN);
               $previous_key__I_LOCATION_NAME = $key__I_LOCATION_NAME;
               @O_VAL = undef;
           }
           $0_VAL[_O_LOCATION_NAME] = $I_VAL[_I_LOCATION_NAME];
           $\[ \VAL[ \] \] DESCRIPTION] = 'State Total';
$\[ \VAL[ \] O_DESCRIPTION] = $\[ \VAL[ \] \] DESCRIPTION];
           $0_VAL[_O_SALES_TOTAL] += $I_VAL[_I_SALES_TOTAL] unless ($I_VAL[_I_SALES_TOTAL] eq '');
       flock(STDOUT, LOCK_EX);
       print STDOUT
           $0_VAL[_O_LOCATION_NAME],
           $0_VAL[_O_DESCRIPTION],
           $O_VAL[_O_SALES_TOTAL]
       flock(STDOUT, LOCK_UN);
       close(DATA);
       print STDERR '[examples/copy_output_combiner.pql ' . localtime() . "] $_inprecs records.";
       my $benchmark_end = new Benchmark;
       my $benchmark_timediff = timediff($benchmark_start, $benchmark_end);
       print STDERR '[examples/copy_output_combiner.pql ' . localtime() . "] Code statistics: @{[timestr($ben
chmark_timediff)]}";
}
}
```

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