Goal:

Use Eventus software version 8.0 or greater to construct a mini-database of data obtained from any source, and run one or more event studies using the mini-database.

Data:

The request file listed below, `sasnoncrspdemo-request.dat`, summarizes the sample of three firm-events. The first column contains a ticker symbol. In a non-CRSP event study, the user decides how to identify securities. In this example the ticker is the security identifier. The event date is constant across firms here, but could just as easily vary across firms.

```
MSFT    19990407
IBM     19990407
CSCO    19990407
```

The user downloads weekly data from 29 December 1997 through 29 October 2001 for the three securities and the Russell 1000 index in spreadsheet (CSV) format from Yahoo!® Finance (http://chart.yahoo.com/d). Eventus can accommodate daily, weekly, monthly, quarterly or annual data. The first few lines of the downloaded file for Cisco Systems are:

```
Week of,Open,High,Low,Close,Volume
29-Oct-01,17.23,17.70,16.06,17.26,78482900
22-Oct-01,16.52,17.83,16.15,17.29,77071600
```

The data include prices but not returns, and the observations are in reverse date order. The downloaded prices are already adjusted for dividends, splits and spinoffs.

Solution:

Use SAS to sort the data and transform prices to returns. The following code and partial data illustrate reading the download file for one security and sorting the data set by date. The statement numbers do not appear in the actual code but are added here for reference.

```
❶ data csco;
❷  infile datalines firstobs=2 dsd truncover;
❸  input caldt:date9. (open high low close) (:7.) volume:9.0;
❹  ticker='CSCO';
❺  format caldt date9.;
❻ datalines;
Week of,Open,High,Low,Close,Volume
29-Oct-01,17.23,17.70,16.06,17.26,78482900
22-Oct-01,16.52,17.83,16.15,17.29,77071600
15-Oct-01,16.65,17.45,15.47,16.72,72493100
8-Oct-01,14.40,17.40,14.35,16.95,87359300

proc sort;
 by caldt;
run;
```

Statement 1 begins a data step and creates a SAS data set, also called a table, named `csco`. Statement 2 designates the data file to read. The keyword `datalines` indicates that the data appear just below the data step statements. The data could instead be in a separate file.[1] In that case, `datalines` in statement 2 would be replaced by the quoted file specification (drive, folder and file name in Windows, or just the file name in Unix if the file is in the current working directory). Because the first line of the download file contains labels to be ignored by SAS, the option `firstobs=2` causes the input to start with the second line. The option `dsd` calibrates the input operation for comma separated data. The `truncover` option prevents an error condition if some data fields are less than their maximum width. Statement 3 designates the columns to read and their input formats. The `date9.` informat causes the `caldt` column to be defined as a SAS date variable. SAS stores dates as the number of days since January 1, 1960. Statement 4 creates the column named `ticker` and sets its value on every row to CSCO. Statement 5 specifies an output format so that `caldt` will appear as a recognizable date. This is for user convenience when examining the data set using the SAS Explorer or Proc Print; it is optional and has no effect on Eventus operation. Statement 6 flags the beginning of in-line input data. If the data were stored in a separate file, this statement would be omitted.

After reading and sorting the data for all three firms, the user runs the following steps to combine the firms and create returns.

```
data ret;
 set ibm msft csco;
 by ticker caldt;
 ret=dif(close)/lag(close);
 if first.ticker then ret=.;
 keep ticker caldt ret;
run;
```

The `set` statement brings the three existing data sets into the new data set `ret`. The `by` statement tells SAS to bring the data in by ticker, then date. The `by` statement also creates `first.ticker`, a dummy variable for the first observation of each distinct ticker.

The steps to create the index data set are similar.

```
data russell_1000;
 infile datalines firstobs=2 dsd truncover;
 input caldt:date9. (open high low close volume) (:7.);
 format caldt yymmddn8.;
```

[1] The download format is CSV, a text format recognized by Excel and other spreadsheet software. Do not convert the file to Excel Workbook or other proprietary format before use with the code in this paper.

```
datalines;
Week of,Open,High,Low,Close,Volume
29-Oct-01,580.57,580.57,553.52,570.69,0
22-Oct-01,564.10,583.50,560.19,580.57,0
15-Oct-01,573.63,582.15,555.65,564.10,0
8-Oct-01,561.82,577.63,552.46,573.63,0
run;

proc sort;
 by caldt;
run;

data russell_1000(label='Weekly Russell 1000Returns‘);
 set russell_1000;
 by caldt;
 vwretd=dif(close)/lag(close);
 keep caldt vwretd;
run;
```

The parenthesized `label` option in the second data step, like the `format` statement in the first data step, is optional and has no effect on Eventus. The Russell 1000 return column name in the above code is `vwretd`. Eventus expects this name for a value-weighted index return with dividends. An event study program invokes the value-weighted index through the `Value` option of the `EvtStudy` statement. Index returns are expected to include dividends by default. This example involves only one index series, but an equal-weighted index return series could appear in the same data set under the column name `ewretd`. The complete code to read and process the downloaded data is available from www.eventstudy.com/sasnoncrspdemo-minidatabase8.sas.

The user now has a mini-database in the format that Eventus expects. The mini-database can be used for multiple event studies with different choices of event date, event period and estimation period. The Eventus statements below run an event study.

```
filename request 'F:\Folder\sasnoncrspdemo-request.dat';
Eventus weekly sasnoncrsp
 indexds=work.russell_1000
 returnds=work.ret;
 Request autodate est=11 pool estlen=52
  issuekey=ticker issuefmt=$4.;
  Windows (-10,-1) (0,+3) (0,+5) (-10,+10);
  EvtStudy value CSectErr pre=10 post=10;
```

The `Eventus` statement option `sasnoncrsp` invokes the mini-database method. The options `indexds` and `returnds` point to mini-database components. The `Request` statement options `issuekey` and `issuefmt` define the security identifier name and format, respectively. All other options in the example are common to both CRSP and non-CRSP event studies. The `CSectErr` option selects a cross-sectional test that is robust to calendar and industry clustering. The `value` option selects the value-weighted index. The mini-database contains no other market index, but the `value` option still is required.

The results appear in the SAS Output window or `*.lst` file; see the appendix below.

Final remarks:

- Check the SAS Log window or `*.log` file for error or warning messages.
- The mini-database should contain at least enough data, for each security, to accommodate the estimation and event periods desired, but the mini-database need not be limited to the data needed for one run.
- The mini-database in this example contains only the three securities in the sample. In general, however, the mini-database may contain more securities than used in any one study run from it.
- This example creates temporary SAS data sets, referenced in the `Eventus` statements using the `work.` prefix. The prefix is omitted in the SAS code because `work` is the default, but cannot be omitted in `Eventus`. To create a data set that persists beyond the current SAS session, define a SAS library name in the SAS Explorer or with a `libname` statement, pointing to a particular folder or directory. Use the SAS library name as the explicit prefix in place of the implied or explicit "`work.`".
- SAS offers many date informats (input formats), so it is rarely necessary to convert dates before reading into SAS. It is essential to specify the right informat on the `infile` or separate `informat` statement. SAS stores dates internally in the same format regardless of the informat or output format (specified by a format statement).
- Many researchers use Excel Workbook files to manage data. SAS 9.1 or greater for Windows includes a File, Import menu command that is easy to use and generally imports spreadsheet data into SAS data sets accurately, including the conversion of dates to SAS format. This interactive import command could replace the data steps that contain an `infile` statement above; the remaining code steps still would be needed. Some older versions of SAS include an importer that is not as reliable.
- Terminate every SAS or Eventus statement with a semicolon. Continuation lines need nothing special because only a semicolon, not a line break, terminates a statement.
- Eventus 7 licensees that have not yet upgraded to Eventus 8 or greater can and should use the mini-database method. In Eventus 7, the user must create a header data set in addition to the security-return and index data sets, and each data set must be indexed — that is, the user must use SAS programming techniques to define a SAS data set index meeting certain requirements. A version of this example specific to Eventus 7 appears on www.eventstudy.com. Eventus 8 includes several automated format and data checks to help ensure that the mini-database meets specifications.

Appendix: Eventus Output

```
                    Eventus (R) Software from Cowan Research, L.C.                    1


  Eventus (R) software is produced by Cowan Research, L.C.
  http://www.eventstudy.com/


  ESTIMATION PERIOD:
                        52 weeks in length,
                        split before and after the event.

  TOTAL EVENTS IN REQUEST FILE:                              3

  EVENTS DROPPED:                                            0
  EVENTS WITH USEABLE RETURNS:                               3


  MINIMUM RETURN DATA REQUIRED FOR ESTIMATION:                 3

  NOTE: Any non-trading dates were converted to the next trading date.



  STATISTICAL SIGNIFICANCE LEVELS:  1 tailed



  NOTE:  Useable returns means all nonmissing returns except the
        first week after a missing estimation period return.
```

<page break in original output removed for this example>

```
                     Eventus (R) Software from Cowan Research, L.C.                   2

                     Results of Weekly Security Return Data Input

                                                                      Esti-
                                                                     mation
                                                                     Period
                                                                     Returns
          ticker    Name on Event Date                  Event Week     <=52

           CSCO                                         04/07/1999       52
           IBM                                          04/07/1999       52
           MSFT                                         04/07/1999       52


                      Event
                     Period
                     Returns
          ticker      <=21      Reason if no useable returns

           CSCO         21
           IBM          21
           MSFT         21
```

Eventus (R) Software from Cowan Research, L.C. 3

Parameter Estimates and Estimation Period Statistics

---------- Market Index=Value ----------

ticker	Event Date	Mean Total Return	% of Raw Returns >0	Alpha	Beta	Market Model Res-iduals>0	Total Return Variance	Residual Standard Deviation	Autocor-relation*
MSFT	07APR1999	0.01703	53.85%	0.01094	1.21	40.38%	0.00371	0.04911	-0.0281
IBM	07APR1999	0.00498	53.85%	0.00123	0.75	51.92%	0.00231	0.04288	-0.1351
CSCO	07APR1999	0.02316	59.62%	0.01516	1.59	50.00%	0.00401	0.04171	-0.0704
MEAN		0.01506	55.77%	0.00911	1.18	47.44%	0.00334	0.04457	-0.0779
MEDIAN		0.01703	53.85%	0.01094	1.21	50.00%	0.00371	0.04288	-0.0704

<page break in original output removed for this example>

Eventus (R) Software from Cowan Research, L.C. 4

Market Model, Value Weighted Index

Week	N	Mean Abnormal Return	Positive: Negative	Patell Z	CSectErr t	Generalized Sign Z
-10	3	-6.27%	0:3<	-2.350**	-13.546***	-1.645*
-9	3	0.29%	1:2	0.127	0.132	-0.489
-8	3	-4.81%	0:3<	-1.765*	-2.373**	-1.645*
-7	3	-0.42%	1:2	-0.178	-0.800	-0.489
-6	3	-0.57%	1:2	-0.211	-0.328	-0.489
-5	3	-0.65%	1:2	-0.270	-1.002	-0.489
-4	3	-0.22%	1:2	-0.180	-0.069	-0.489
-3	3	2.82%	3:0>	1.034	2.891**	1.823*
-2	3	1.74%	3:0>	0.658	15.372***	1.823*
-1	3	-0.84%	1:2	-0.259	-0.454	-0.489
0	3	-7.80%	0:3<	-2.961**	-11.628***	-1.645*
+1	3	4.81%	2:1	1.981*	0.821	0.667
+2	3	-0.46%	1:2	-0.106	-0.146	-0.489
+3	3	-2.99%	1:2	-1.122	-0.933	-0.489
+4	3	4.11%	2:1	1.688*	1.026	0.667
+5	3	-1.93%	1:2	-0.771	-1.645*	-0.489
+6	3	1.99%	2:1	0.677	0.934	0.667
+7	3	-1.76%	1:2	-0.606	-1.072	-0.489
+8	3	-0.19%	2:1	-0.082	-0.424	0.667
+9	3	2.22%	3:0>	0.808	2.617**	1.823*
+10	3	2.99%	3:0>	1.162	3.257***	1.823*

The symbols $,*,**, and *** denote statistical significance at the 0.10, 0.05, 0.01 and 0.001 levels, respectively, using a 1-tail test. The symbols (,< or),> etc. correspond to $,* and show the significance and direction of the generalized sign test.

Market Model, Value Weighted Index

Weeks	N	Mean Cumulative Abnormal Return	Precision Weighted CAAR	Positive: Negative
(-10,-1)	3	-8.94%	-9.04%	0:3<
(0,+3)	3	-6.44%	-5.84%	1:2
(0,+5)	3	-4.25%	-3.42%	1:2
(-10,+10)	3	-7.95%	-7.24%	1:2

The symbols $,*,**, and *** denote statistical significance at the 0.10, 0.05, 0.01 and 0.001 levels, respectively, using a 1-tail test. The symbols (,< or),> etc. correspond to $,* and show the significance and direction of the generalized sign test.

Market Model, Value Weighted Index

Weeks	Patell Z	CSectErr t	Generalized Sign Z
(-10,-1)	-1.073	-2.191*	-1.645*
(0,+3)	-1.104	-0.551	-0.489
(0,+5)	-0.527	-0.297	-0.489
(-10,+10)	-0.595	-0.439	-0.489

The symbols $,*,**, and *** denote statistical significance at the 0.10, 0.05, 0.01 and 0.001 levels, respectively, using a 1-tail test. The symbols (,< or),> etc. correspond to $,* and show the significance and direction of the generalized sign test.