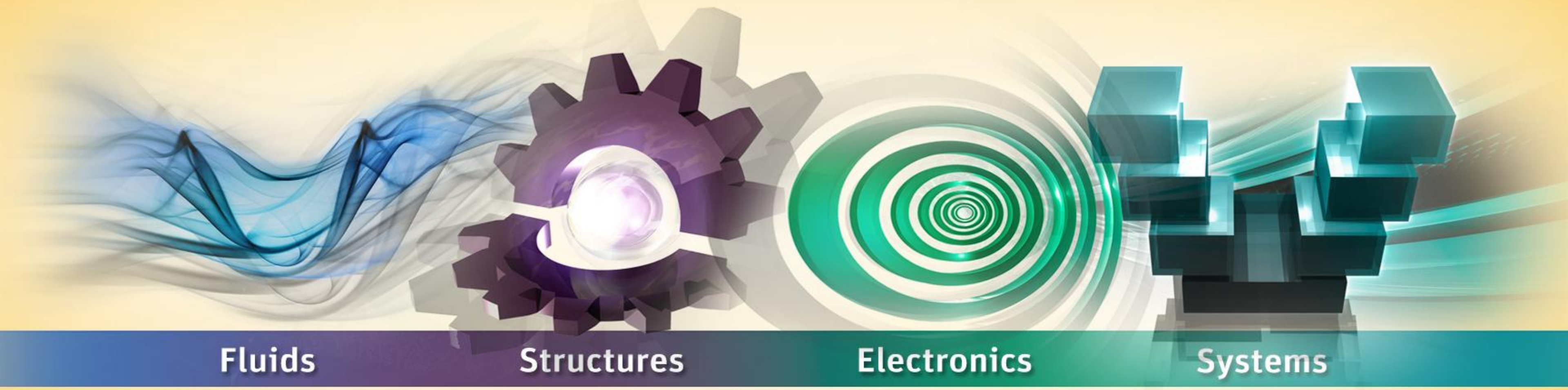
Realize Your Product Promise®

A (Very) Brief Primer on MAPDL Command Objects for Postprocessing



Will Goodrum

ANSYS, Inc.

v1 – April 2015

Objective

Provide ANSYS Mechanical users with a *brief* introduction to the use of Solution Command Objects for post-processing.

Disclaimer: The ANSYS Parametric Design Language (APDL) is an extensive programming language developed over more than 40 years. This document covers only **very minimal aspects of APDL**. For a more detailed introduction, users are referred to the Introduction to Mechanical APDL training course (ANSYS Customer Portal login required).

Part 1

Part 2

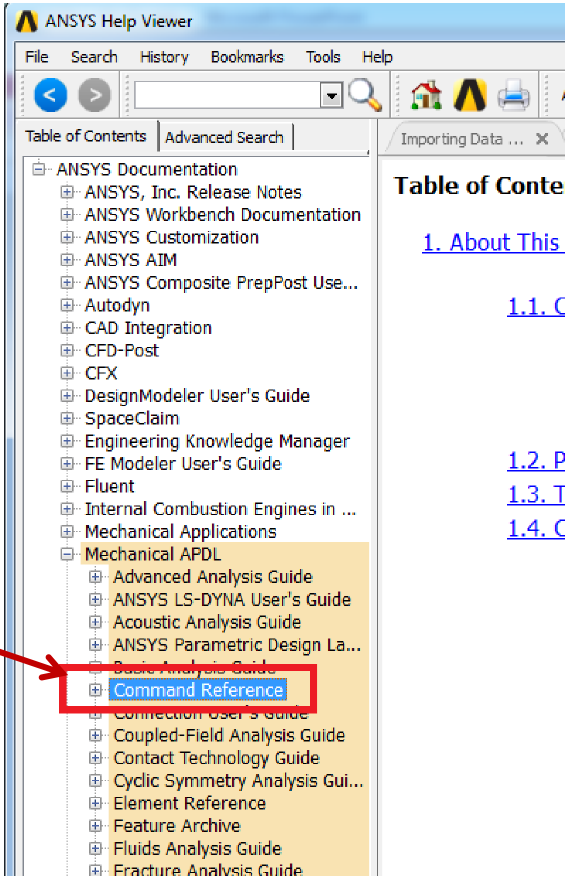
Additionally, the current release version of ANSYS at the time of writing was version **16.0**. All Help Documentation links provided in the notes are appropriate to that version. To the best extent possible, the advice given in these notes is generic and not specific to any version.

Help Documentation

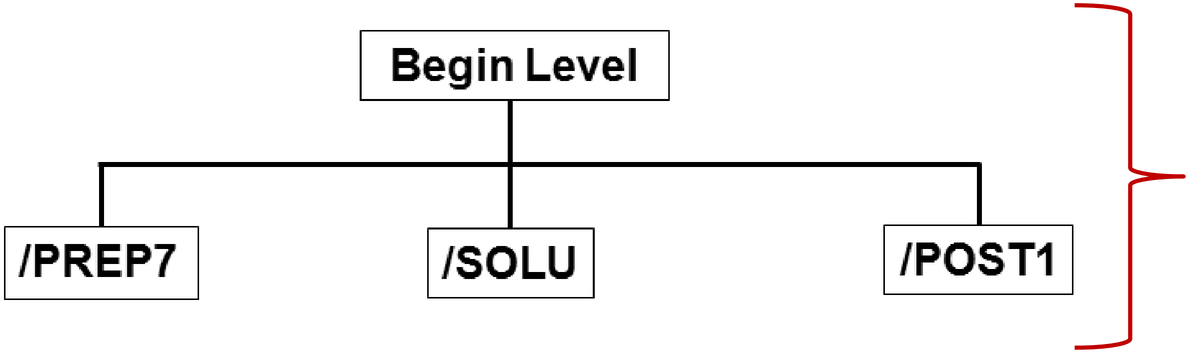
When it comes to learning Mechanical APDL, there is no more complete resource than the Mechanical APDL Help Documentation. It is recommended that users reading through these notes keep the Help Documentation open for further reference.

All of the commands mentioned in this document are described in detail in the Mechanical APDL Command Reference. The more familiar you become with the Command Reference, the more quickly you will pick up APDL!

The appendix of this document contains links to the Help Documentation page for every command that is referenced.



Brief notes on MAPDL – Structure



Begin Level

/PREP7

/SOLU

/POST1

These are the MAPDL “processors” that define the contexts for all APDL commands

Some commands are valid in any processor:

`/EXIT`, `NSEL`, etc.

Some commands are valid in multiple processors:

`D` command is valid in `/PREP7` and `/SOLU`

Some commands are only valid in one processor:

`ANTYPE` is only valid in `/SOLU`

Brief Notes on MAPDL – The Language

ANSYS Parametric Design Language (APDL) is the command language that controls **MAPDL**

MAPDL 16.0 contains approximately **2,000** documented commands

With each new Release:

- New commands are added
- Existing commands are enhanced
- Infrequently used commands are archived
- Obsolete commands are undocumented

Brief Notes on MAPDL – General Command Formatting

General Format: `COMMAND_NAME,Field1,Field2,…,Fieldn`

- APDL commands begin with a command name followed by a comma-separated list of arguments (or “fields”)
- Alphanumeric data is entered in the fields
- The number of fields and data required in each field is command dependent

  NOTE: sometimes the command name is referred to as “Field1” and the first data field is referred to as “Field2”

Examples:

- `N,100,1,2,3`
- `E,1,2,34`
- `F,1,FX,1000`

Brief Notes on MAPDL – Command Names

APDL command names are case insensitive:

- `ET`, `et`, `Et`, and `eT` are equivalent

Only the first 4 characters of the command name are relevant:

- `SECD` and `SECDATA` are equivalent (extra characters are ignored)

Carriage return or $ character ends a command

- Multiple commands can be included same line if separated by $:

```
ET,1,185   $    ET,2,187
```

Brief Notes on MAPDL – General Data Input

Data input is free-format:

- Mandatory spacing is not required
- Fields are as wide as the number of characters defined
- Comma defines end of the field:

```
NSEL, , loc, x,0 or NseL,     ,    loc,x,   0
```

Data input is nonrestrictive:

- APDL converts reals to integers as necessary:

```
N, 1.1, 2 is converted to N,1,2.0
```

- Exponential notation (e or d format) is permitted:

```
F1, 1, FY, 1e3 or F1, 1, FY, 1000
```

- Can include arithmetic expressions:

```
N, 2-1, 1-1, 3-2, 2*0.5+1 or N, 1, 0, 1, 2
```

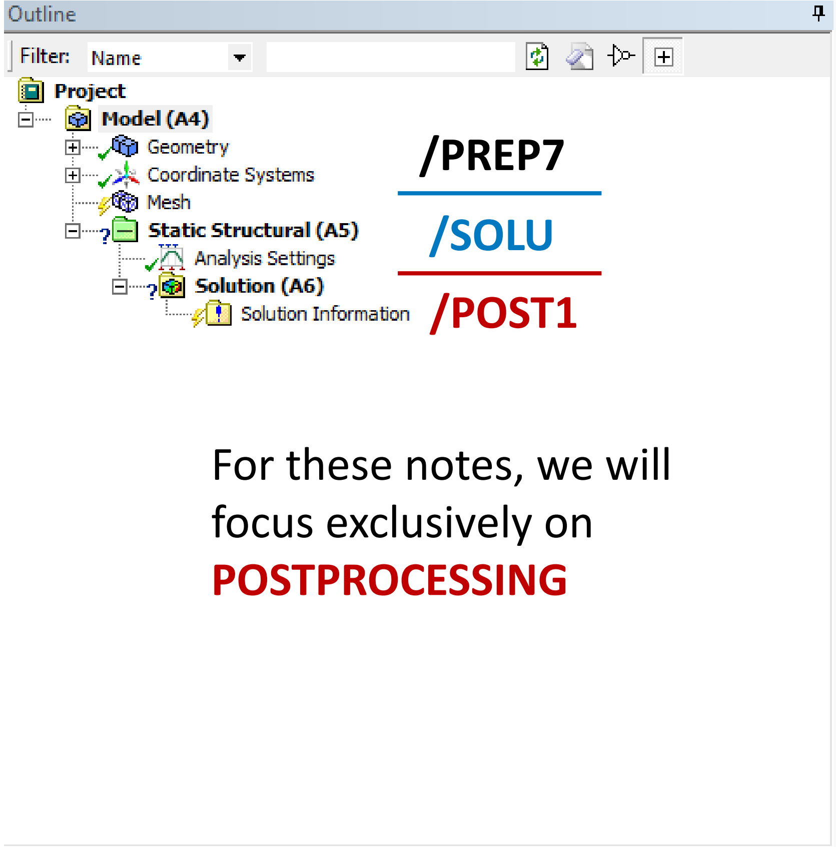
APDL Command Objects

APDL Command Objects may be inserted in different contexts in the ANSYS Mechanical Model Tree. These contexts correspond with the Mechanical APDL “processors”:

/PREP7 – Preprocessing

/SOLU – Solution

/POST1 – General Postprocessing



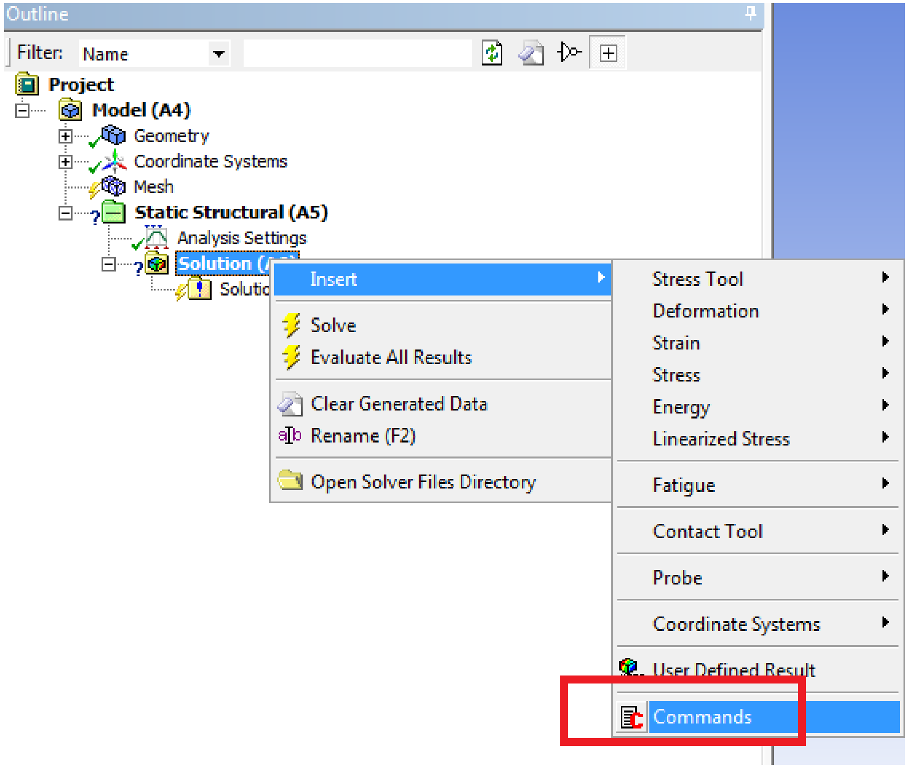
Processors are important, since certain commands are only valid in certain contexts!

For these notes, we will focus exclusively on **POSTPROCESSING**

APDL Command Objects

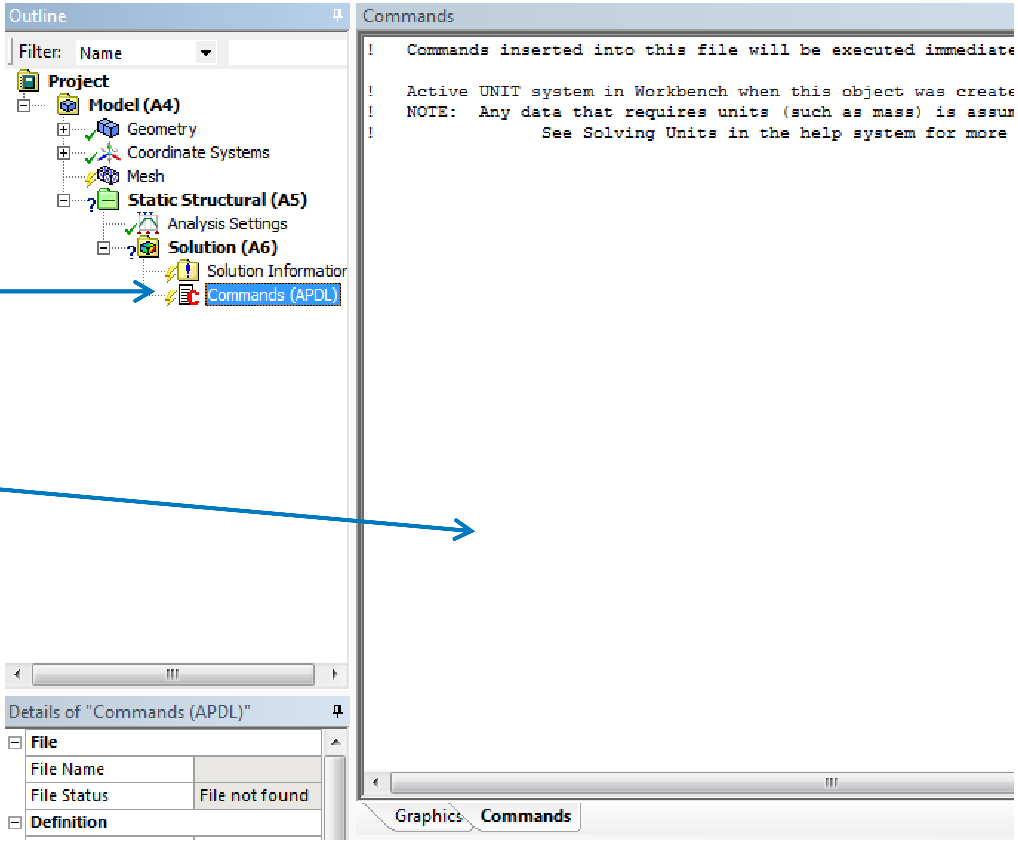
To insert a Solution Command Object:

1. Right Click on Solution
2. Insert > Commands



APDL Command Objects

This will place a Command Object under the Solution branch of the Model Tree. APDL Commands may now be entered in the Worksheet.



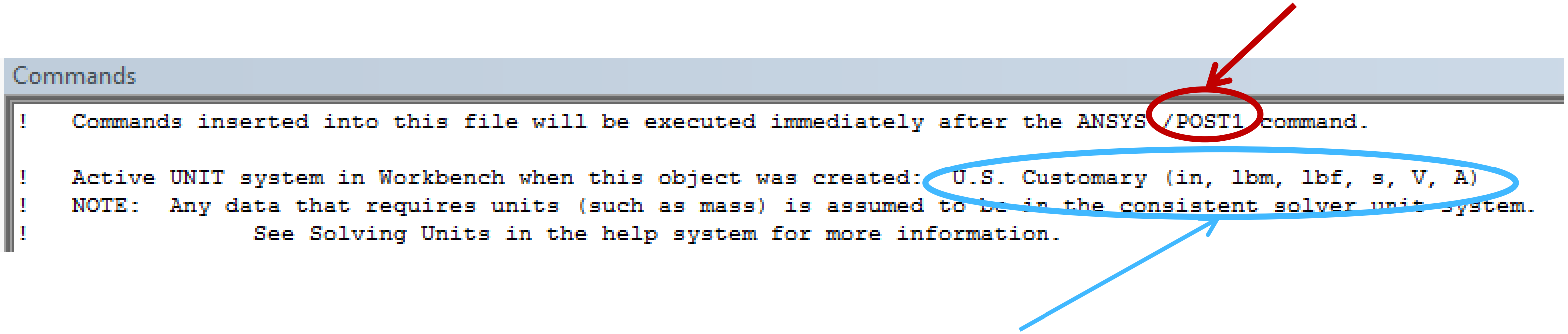
```
!   Commands inserted into this file will be executed immediate

!   Active UNIT system in Workbench when this object was create
!   NOTE:  Any data that requires units (such as mass) is assum
!                    See Solving Units in the help system for more
```

Details of "Commands (APDL)"

File	
File Name	
File Status	File not found
Definition	

The Command Object Header

Every Command Object inserted under Solution contains a header like the following:



Indicates postprocessing!

Commands

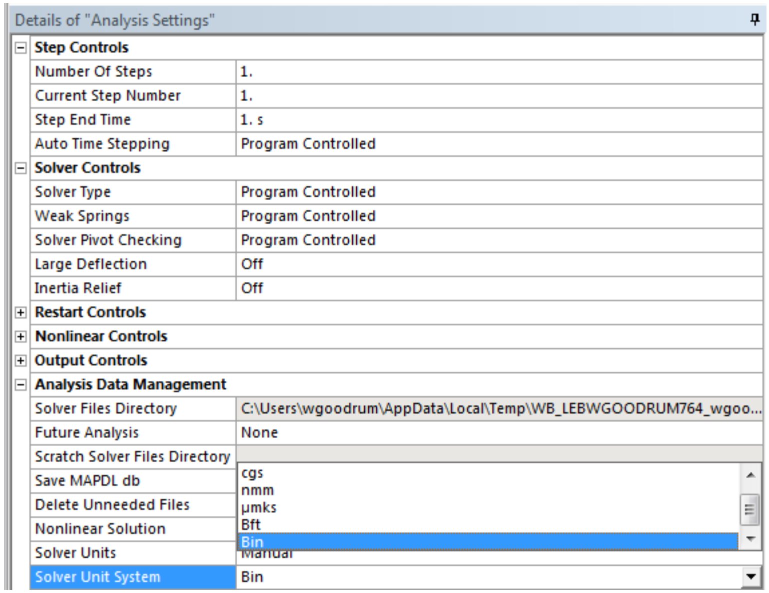
```
!   Commands inserted into this file will be executed immediately after the ANSYS /POST1 command.

!   Active UNIT system in Workbench when this object was created:  U.S. Customary (in, lbm, lbf, s, V, A)
!   NOTE:  Any data that requires units (such as mass) is assumed to be in the consistent solver unit system.
!                 See Solving Units in the help system for more information.
```

NOTE: any scalar values entered directly in a Command Object will be *hard-coded* with the units that were active **when the Command Object was created.** Use caution when changing the units and re-solving!

Units for Command Objects

By default the “**Solver Units**” under **Analysis Settings > Analysis Data Management** will be set to “**Active System**.” By changing this to “**Manual**,” the user may specify a fixed unit system to be used, regardless of the specification on the Units dropdown menu.

Specifying a fixed solver unit system may help to prevent any issues with executing Command Objects when the active unit system has changed after the Command Object was inserted.

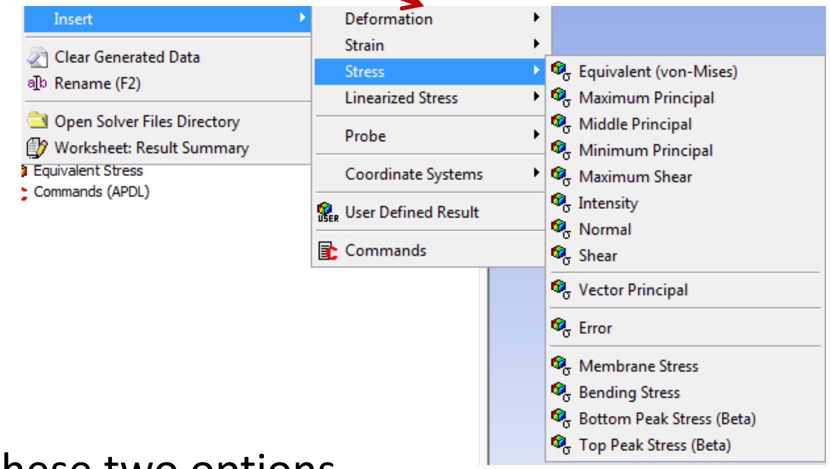


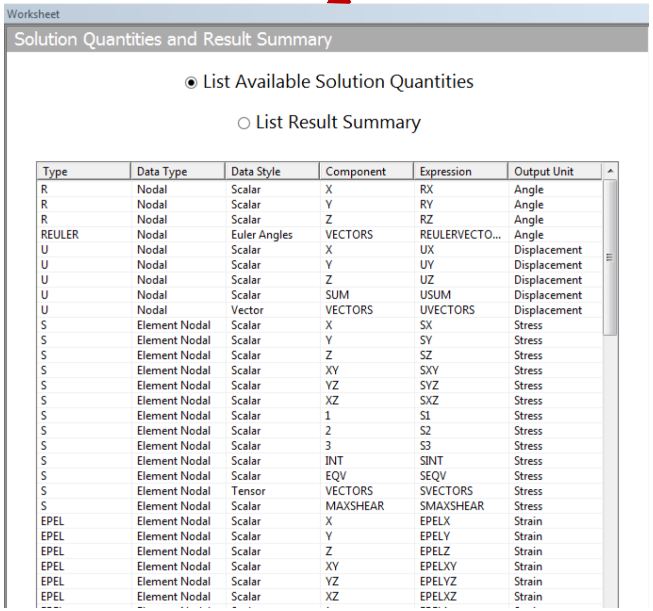
Details of "Analysis Settings"

Step Controls	
Number Of Steps	1.
Current Step Number	1.
Step End Time	1. s
Auto Time Stepping	Program Controlled
Solver Controls	
Solver Type	Program Controlled
Weak Springs	Program Controlled
Solver Pivot Checking	Program Controlled
Large Deflection	Off
Inertia Relief	Off
Restart Controls	
Nonlinear Controls	
Output Controls	
Analysis Data Management	
Solver Files Directory	C:\Users\wgoodrum\AppData\Local\Temp\WB_LEBWGOODRUM764_wgoo...
Future Analysis	None
Scratch Solver Files Directory	
Save MAPDL db	
Delete Unneeded Files	
Nonlinear Solution	
Solver Units	Manual
Solver Unit System	Bin

Why Command Objects for Post-processing?

ANSYS Mechanical (i.e., the Workbench Structural Mechanics application) has many quantities that we can post-process natively, both through pre-defined and user-defined results.





Worksheet

Solution Quantities and Result Summary

● List Available Solution Quantities

○ List Result Summary

Type	Data Type	Data Style	Component	Expression	Output Unit
R	Nodal	Scalar	X	RX	Angle
R	Nodal	Scalar	Y	RY	Angle
R	Nodal	Scalar	Z	RZ	Angle
REULER	Nodal	Euler Angles	VECTORS	REULERVECTO...	Angle
U	Nodal	Scalar	X	UX	Displacement
U	Nodal	Scalar	Y	UY	Displacement
U	Nodal	Scalar	Z	UZ	Displacement
U	Nodal	Scalar	SUM	USUM	Displacement
U	Nodal	Vector	VECTORS	UVECTORS	Displacement
S	Element Nodal	Scalar	X	SX	Stress
S	Element Nodal	Scalar	Y	SY	Stress
S	Element Nodal	Scalar	Z	SZ	Stress
S	Element Nodal	Scalar	XY	SXY	Stress
S	Element Nodal	Scalar	YZ	SYZ	Stress
S	Element Nodal	Scalar	XZ	SXZ	Stress
S	Element Nodal	Scalar	1	S1	Stress
S	Element Nodal	Scalar	2	S2	Stress
S	Element Nodal	Scalar	3	S3	Stress
S	Element Nodal	Scalar	INT	SINT	Stress
S	Element Nodal	Scalar	EQV	SEQV	Stress
S	Element Nodal	Tensor	VECTORS	SVECTORS	Stress
S	Element Nodal	Scalar	MAXSHEAR	SMAXSHEAR	Stress
EPEL	Element Nodal	Scalar	X	EPELX	Strain
EPEL	Element Nodal	Scalar	Y	EPELY	Strain
EPEL	Element Nodal	Scalar	Z	EPELZ	Strain
EPEL	Element Nodal	Scalar	XY	EPELXY	Strain
EPEL	Element Nodal	Scalar	YZ	EPELYZ	Strain
EPEL	Element Nodal	Scalar	XZ	EPELXZ	Strain

Between these two options,
you may not need APDL Commands!

(TIP: click on Solution and then select Worksheet to view all available user-defined result quantities!)

Why Command Objects for Post-processing?

Example scenarios when commands are required:

- Postprocessing radiative heat loss to an enclosure
- Rotational response of a remote point in a PSD analysis
- Average contact pressure in a contact pair (or any average quantity on a geometry)

General Procedure for Postprocessing with Solution Command Objects

1. Resume the database (DB) File
2. Load the results set from the results (RST) file
3. Access model information in the DB file
4. Apply selection logic to make entities active/inactive
5. Create Element Tables (ETABLEs)
6. Create Plots
7. Parameters

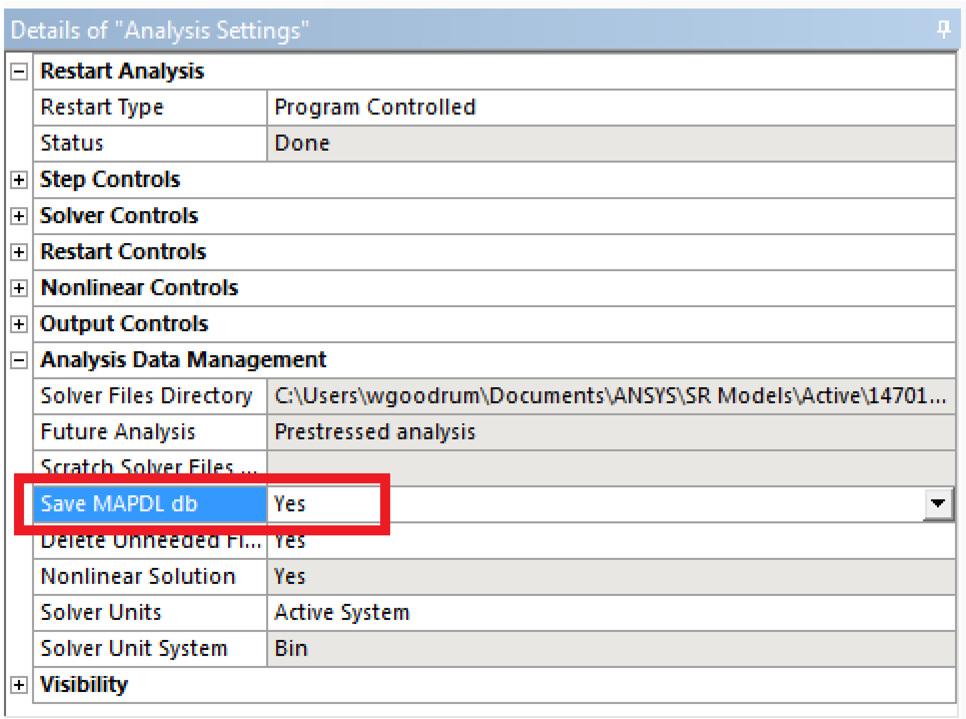
General Procedure for Postprocessing with Solution Command Objects

	APDL Commands
1. Resume the DB File	RESUME
2. Load the results set from the RST file	SET
3. Access model/result information in the DB file	*GET
4. Apply selection logic to make entities active/inactive	xSEL
5. Create Element Tables (ETABLEs)	ETABLE
6. Create Plots	PLNSOL/PLESOL
7. Parameters	

Not always in this order

1. Resume the DB File

When APDL postprocessing is required, it is highly-desirable to set **Analysis Settings > Analysis Data Management > Save MAPDL db? > Yes in ANSYS Mechanical**



Details of "Analysis Settings"

⊟ **Restart Analysis**	
Restart Type	Program Controlled
Status	Done
⊞ **Step Controls**	
⊞ **Solver Controls**	
⊞ **Restart Controls**	
⊞ **Nonlinear Controls**	
⊞ **Output Controls**	
⊟ **Analysis Data Management**	
Solver Files Directory	C:\Users\wgoodrum\Documents\ANSYS\SR Models\Active\14701...
Future Analysis	Prestressed analysis
Scratch Solver Files ...	
Save MAPDL db	Yes
Delete Unneeded Fi...	Yes
Nonlinear Solution	Yes
Solver Units	Active System
Solver Unit System	Bin
⊞ **Visibility**	

The DB file contains:

- All FE information (mesh, loads, boundary conditions)
- Named Selections/Components

NOTE: the DB file *does not* contain any geometry information!

1. Resume the DB File

The **`RESUME`** command loads the DB file when executed.

The syntax for **`RESUME`** is:

```
RESUME, <FILENAME>, <EXTENSION>
```

In ANSYS Mechanical, the DB file is almost always **"FILE.DB".** So, the first command issued will likely be:

```
RESUME, FILE, DB
```

2. Load a Results Set from the Results File

Results are loaded into the database by the **SET** command. **NOTE:** all Solution Command Objects should have a SET command. Without a SET command, the Command Object may not have the required information available!

For most analyses, the results database will be called **"FILE.RST".** Solution Command Objects automatically seek this file when the **SET** command is issued.

The general format of **SET** is:

SET,*<loadstep>*,*<substep>*

There are also special ways to call SET:

SET,FIRST → load the first results set on FILE.RST

SET,NEXT → load the next results set. . .

SET,LAST → load the last results set. . .

3. Access Model/Results Information in the DB File

One of the most powerful commands in APDL is **`*GET`** (pronounced “star-get”).

This command enables you to access almost all information that is available about your model in the DB file (including results information stored there)

Some common **`*GET`** operations are:

- Determine how many nodes are in the mesh
- Find the minimum/maximum node/element numbers
- Grab results of last force sum (FSUM) command
- Get results at a particular node

3. Access Model/Results Information in the DB File

The options available on ***GET** are far too numerous to summarize in these notes. Because of this, users are strongly advised to become familiar with searching the ***GET** documentation (16.0 Doc Link: help/ans_cmd/Hlp_C_GET.html)

The Documentation on ***GET** is organized based on Preprocessing, Solution, and Postprocessing values.

If any postprocessing operations require information about the model, then ***GET** is the best way to find it!

3. Access Model/Results Information in the DB File

There are certain ***GET** operations that are so common that they have been hard-coded into APDL as "**GET Functions**" (16.0 Doc Link: help/ans_apdl/Hlp_P_APDLget.html)

Some useful examples are:

- X(*N*) → get the X location of a node
- U*n*(*N*) → get the U*n* result at Node *N* (where *n* = X, Y, Z)
- ARNODE(*N*) → get the area at Node *N*
- NDNEXT(*N*) → get the ID of the next selected node

3. Access Model/Results Information in the DB File

In addition to **`*GET`** and the GET Function shortcuts, there is also the **`*VGET`** command (16.0 Doc Link: help/ans_cmd/Hlp_C_VGET_st.html).

This is a *vectorized* form of **`*GET`** that enables users to extract information for a set of entities into an array without needing to iterate using a looping structure (e.g., **`*DO`**). Working with vectors, when possible, is more computationally efficient!

4. Apply Selection Logic to make Entities Active/Inactive

APDL has an intuitive family of commands for selecting nodes, elements, and other entities for processing. These commands are generally referred to as the "**`xSEL` commands**."

For example:

- **`NSEL`** → select/unselect nodes
- **`ESEL`** → select/unselect elements
- **`CMSEL`** → select/unselect components (i.e., Named Selections)

4. Apply Selection Logic to make Entities Active/Inactive

The general format of the **`xSEL`** commands is:

```
xSEL,<select_type>,<item>,<component>,<min_value>,<max_value>
```

Examples:

- **`NSEL,S,NODE,,5,10`** → new selection (“S”) of node IDs 5-10
- **`ESEL,U,ENAME,,185`** → unselect (“U”) all SOLID185 elements
- **`NSEL,A,LOC,X,2,4`** → add (“A”) nodes from X = 2-4 to the current set

NOTE: if a field is not used then it may simply be left blank. For example, there is no “component” for a Node ID, so the component field is blank, above.

4. Apply Selection Logic to make Entities Active/Inactive

Some other special selection commands are:

- **`ESLN`** → select elements attached to the active nodal selection
- **`NSLE`** → select nodes attached to the active element selection
- **`ALLSEL,ALL`** → select all entities in the model

NOTE: **`ESLN`** and **`NSLE`** are very important, since the active nodal and element selection sets are *independent of one another.* So, even if you have selected only the nodes you want, all elements in the model remain active!

4. Apply Selection Logic to make Entities Active/Inactive

Components (i.e., Named Selections) are another way to make subsets of entities active. Named Selections created in ANSYS Mechanical are translated into APDL components as follows:

- Body Named Selections → ELEMENT components
- Any other entities → NODAL components

Also, the name given to a Named Selection in the Model Tree is used as the Component name in the DB file, and may be selected using **`CMSEL`**.

5. Create Element Tables (ETABLEs)

There are special postprocessing data structures available in Mechanical APDL called ***element tables*** (or, ETABLEs). ETABLEs allow users to:

- Perform arithmetic operations on results data
- Access element results that are not otherwise directly accessible (e.g., derived data for line elements, contact element results)

ETABLEs are like spreadsheets. Each row is an element, and each column is a particular data item for all elements. For example, one column might be SX, another might be element volumes, and yet another may be the Y-coordinate of each element centroid.

5. Create Element Tables (ETABLEs)

Element Table results quantities are summarized in the "Output Data" section of the Mechanical APDL Element Reference page for each element.

There are two primary types of data reported by ETABLEs:

1. **SMISC**: Summable Miscellaneous
2. **NMISC**: Nonsummable Miscellaneous

Data is extracted from ETABLEs via the ***GET command** (see Section 3) by referring to the **sequence number**

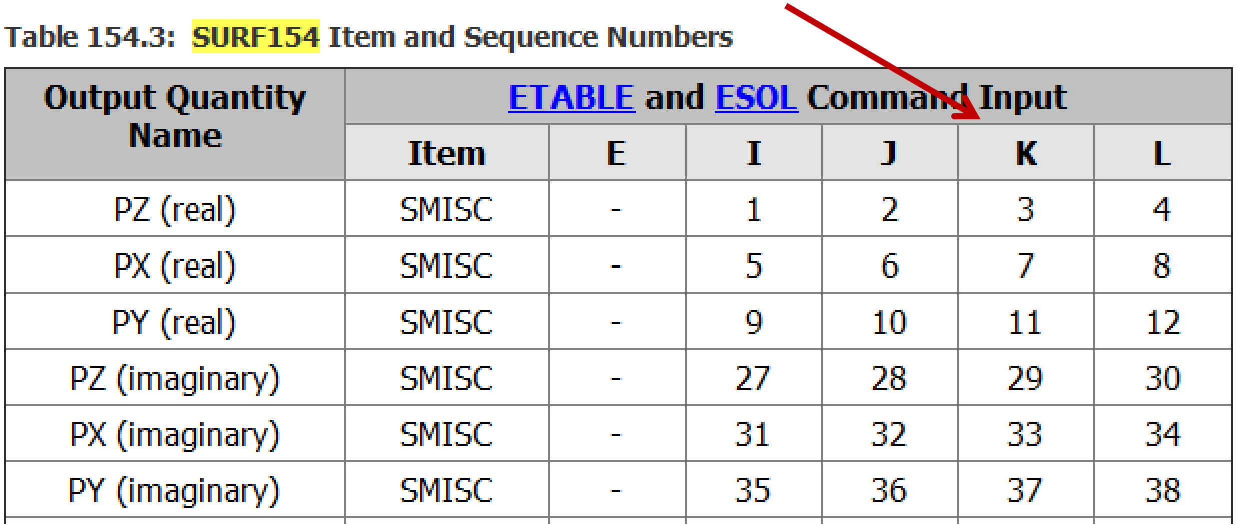


Table 154.3: SURF154 Item and Sequence Numbers

Output Quantity Name	ETABLE and ESOL Command Input					
	Item	E	I	J	K	L
PZ (real)	SMISC	-	1	2	3	4
PX (real)	SMISC	-	5	6	7	8
PY (real)	SMISC	-	9	10	11	12
PZ (imaginary)	SMISC	-	27	28	29	30
PX (imaginary)	SMISC	-	31	32	33	34
PY (imaginary)	SMISC	-	35	36	37	38

5. Create Element Tables

Some useful commands for working with Element Tables are:

- **SSUM** – calculate the sum of element table values (result can be accessed via ***GET**)
- **PRETAB** – print the element table values

5. Create Element Tables (ETABLEs)

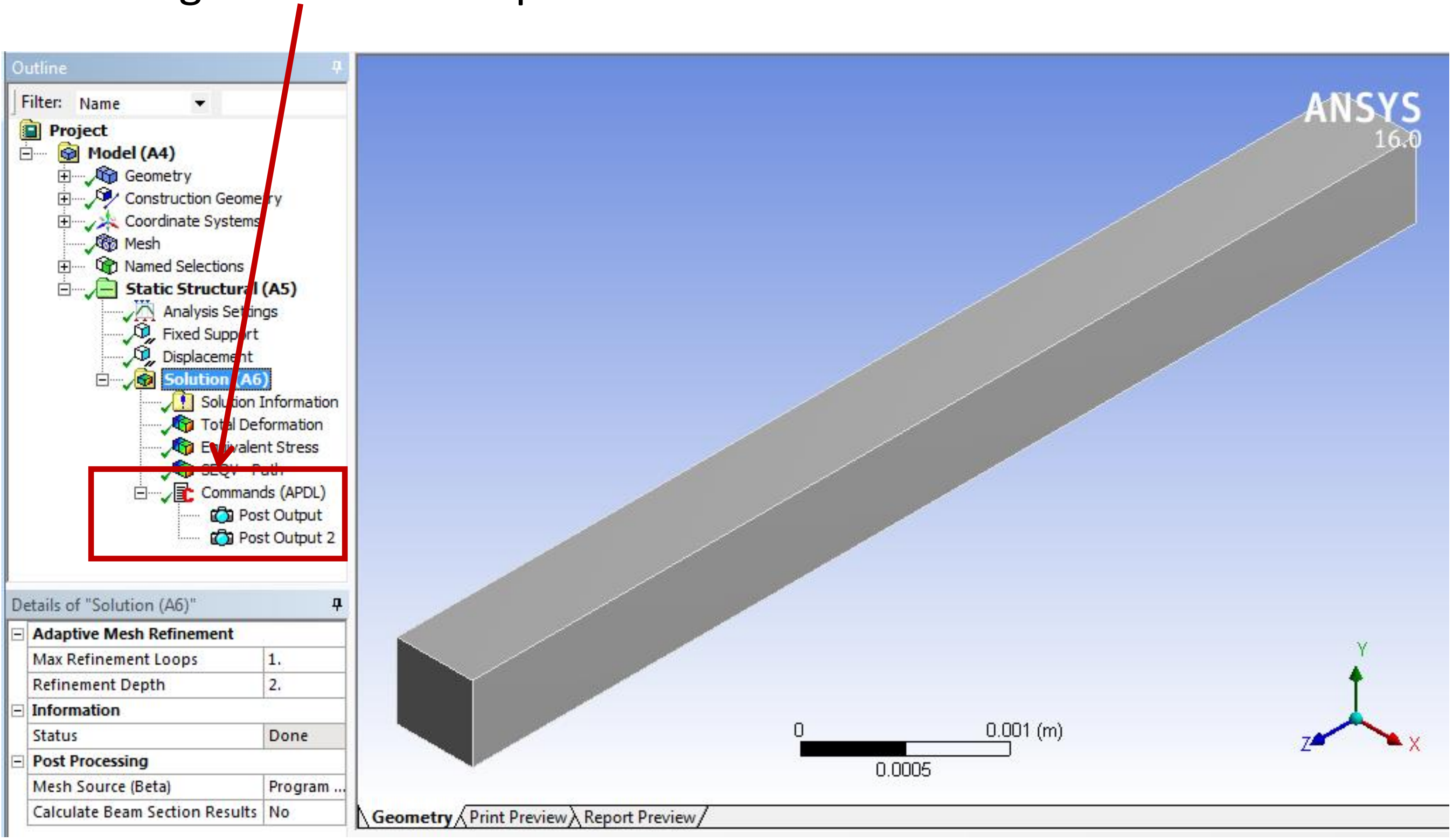
Some SMISC and NMISC values are accessible via User-Defined Results directly in Mechanical, meaning **Command Objects may not be necessary!** (e.g., CONTNMISC*n* or CONTSMISC*n*, where "*n*" is the **Sequence Number** of the results quantity in the results file).

Element Table values accessible by User-Defined Result are summarized in the following section of the ANSYS Help Documentation:

help/wb_sim/ds_user_defined_MAPDL.html

6. Create Plots

Results quantities extracted from Commands Objects cannot be plotted using the “Graph” in ANSYS Mechanical. However, it is possible to make visible, static figures of MAPDL plots.



Details of "Solution (A6)"

Adaptive Mesh Refinement	
Max Refinement Loops	1.
Refinement Depth	2.
Information	
Status	Done
Post Processing	
Mesh Source (Beta)	Program ...
Calculate Beam Section Results	No

6. Create Plots

In order to create a plot, the graphics must be redirected to an image file using the **`/SHOW`** command.

For example: **`/SHOW,PNG`** will redirect any subsequent MAPDL plotting commands to a .PNG image file.

The image buffer can be cleared by issuing **`/SHOW,CLOSE`** after all invoking all plotting commands.

6. Create Plots

Once plotting has been redirected to file, plotting commands may be issued. The most commonly used plotting commands in MAPDL begin with a “PL-” prefix. For example:

- **`PLNSOL`** → plot Nodal Solution (e.g., U,SUM)
- **`PLESOL`** → plot Element Solution (e.g., S,EQV)

The format of these commands is:

```
PLNSOL,Item,Comp
```

So, for example, if we wanted to plot total vector displacement, the command would be:

```
PLNSOL,U,SUM
```

6. Create Plots

It is also possible to create X-Y plots of Tables and Arrays using the **`*VPLOT`** command. Up to 8 arrays may be plot simultaneously using the format:

```
*VPLOT, ParX, ParY, Y2, …
```

NOTE: all arrays will be plotted against the same X-values. Arrays are plotted as discrete bars, while Tables are plotted as continuous curves.

6. Create Plots

A constraint of creating plots from MAPDL Command Objects is that they are ***static figures*** that cannot be manipulated directly in the ANSYS Mechanical GUI. All view and plotting specifications must be made by MAPDL command.

For example:

- **`/VIEW`**: change the orientation of the model in the figure
- **`/ANGLE`**: rotate the view about an axis
- **`/GRAPHICS`**: specify whether to use PowerGraphics or Full Graphics for element displays

7. Parameters

There are two broad types of parameters in APDL: **scalar** and **vector**.

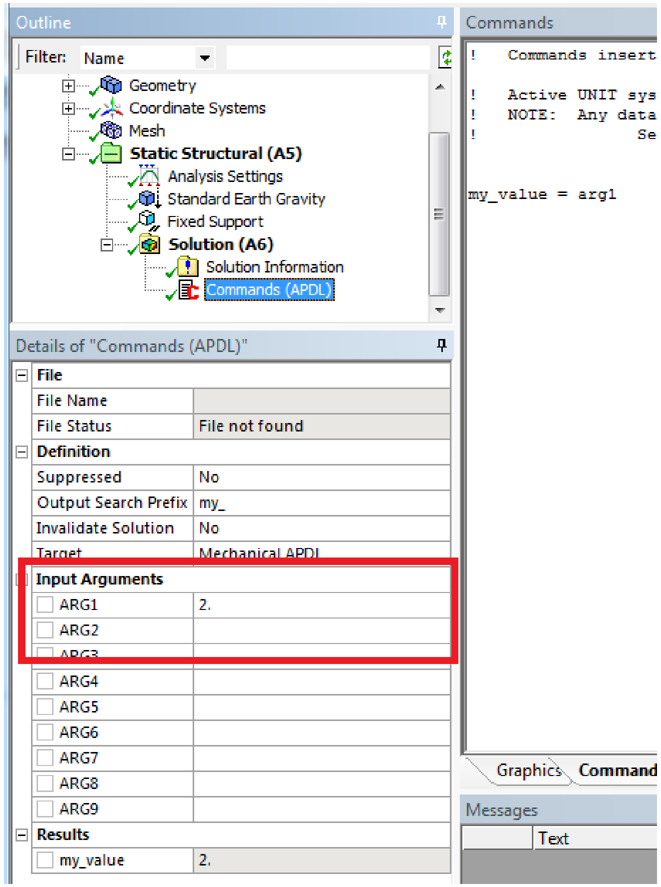
Scalar parameters are constant variables (e.g., number of nodes in the model, element type number). Scalar parameters can easily be passed into and displayed from Solution Command Objects.

Vector parameters are multi-dimensional arrays (e.g., UX for all nodes in model). Vector parameters can only be printed to or read from text files.

7. Parameters

Passing a scalar value into a Solution Command Object is easy. In the Details of the Command Object there are nine fields, ARG1-ARG9. These **Input Arguments** can then be used in the Command Object to avoid hard-coding in scalar values.

In the example at right, a value of “2” has been entered for ARG1. This value is then reassigned to a parameter “MY_VALUE,” which displays at the bottom of the Command Object that it has a value of “2”



File	
File Name	
File Status	File not found
Definition	
Suppressed	No
Output Search Prefix	my_
Invalidate Solution	No
Target	Mechanical APDL
Input Arguments	
□ ARG1	2.
□ ARG2	
□ ARG3	
□ ARG4	
□ ARG5	
□ ARG6	
□ ARG7	
□ ARG8	
□ ARG9	
Results	
□ my_value	2.

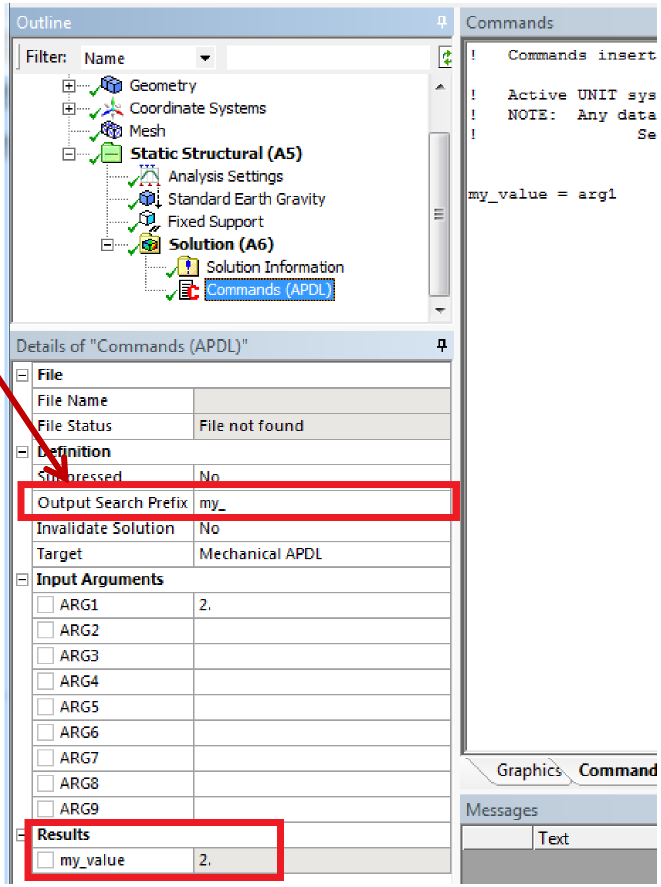
```
!    Commands insert

!    Active UNIT sys
!    NOTE:  Any data
!                  Se


my_value = arg1
```

7. Parameters

Similarly, scalar values may be passed out of a Command Object by using the **Output Search Prefix**.

Any parameters that are named starting with this prefix will be reported under **Results** at the bottom of the Details.

You may change the text in this field to whatever you like (the default is always "my_").



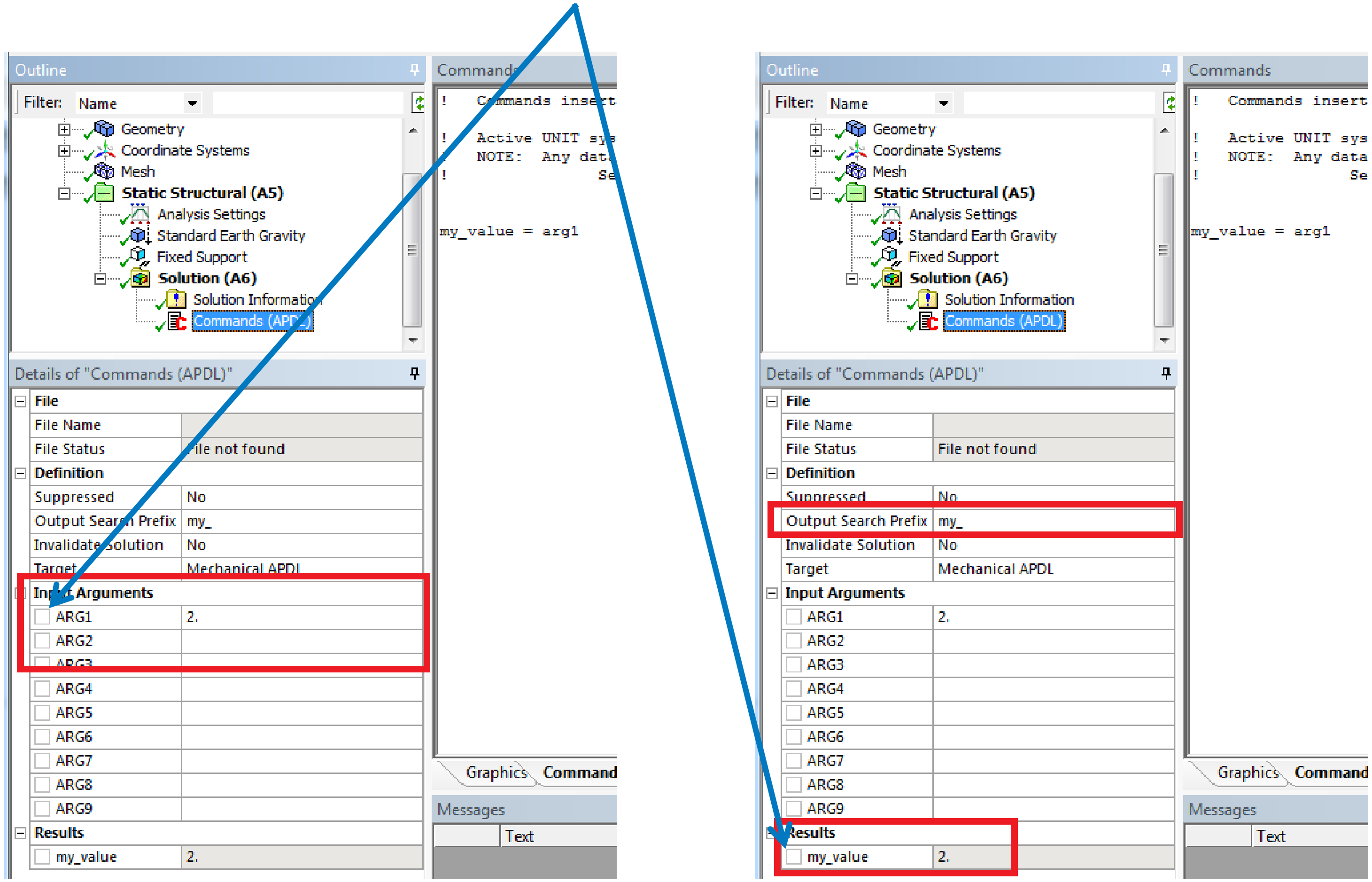
Details of "Commands (APDL)"

File	
File Name	
File Status	File not found
Definition	
Suppressed	No
Output Search Prefix	my_
Invalidate Solution	No
Target	Mechanical APDL
Input Arguments	
□ ARG1	2.
□ ARG2	
□ ARG3	
□ ARG4	
□ ARG5	
□ ARG6	
□ ARG7	
□ ARG8	
□ ARG9	
Results	
□ my_value	2.

```
!    Commands insert

!    Active UNIT sys
!    NOTE:  Any data
!                 Se


my_value = arg1
```

7. Parameters

Note that both Input and Output parameters can be elevated to be Workbench parameters!



```
!   Commands insert

!   Active UNIT sys
!   NOTE:  Any data
!               Se

my_value = arg1
```

Details of "Commands (APDL)"

File	
File Name	
File Status	File not found
Definition	
Suppressed	No
Output Search Prefix	my_
Invalidate Solution	No
Target	Mechanical APDL
Input Arguments	
□ ARG1	2.
□ ARG2	
□ ARG3	
□ ARG4	
□ ARG5	
□ ARG6	
□ ARG7	
□ ARG8	
□ ARG9	
Results	
□ my_value	2.

Details of "Commands (APDL)"

File	
File Name	
File Status	File not found
Definition	
Suppressed	No
Output Search Prefix	my_
Invalidate Solution	No
Target	Mechanical APDL
Input Arguments	
□ ARG1	2.
□ ARG2	
□ ARG3	
□ ARG4	
□ ARG5	
□ ARG6	
□ ARG7	
□ ARG8	
□ ARG9	
Results	
□ my_value	2.

7. Parameters

APDL has many tools for working with multi-dimensional (or vector) data.

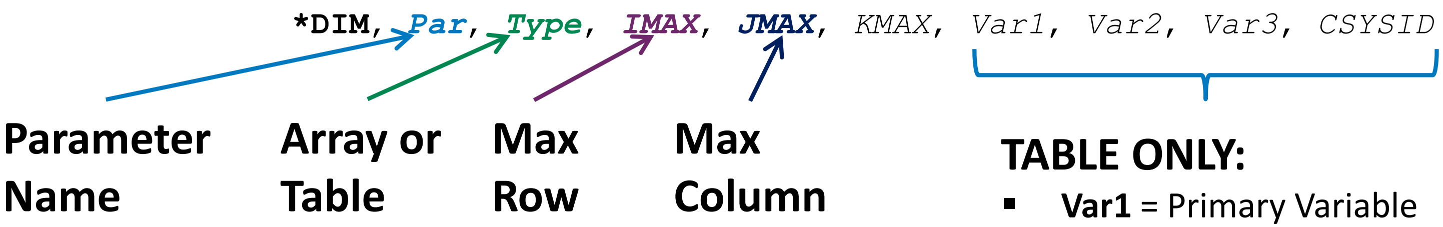
The two important structures to become familiar with are **arrays** and **tables**.

Arrays are similar to standard FORTRAN arrays, and are indexed, discrete values.

Tables are a special type of array. Values defined in a table are linearly interpolated by the MAPDL solver, based on the current value of “primary variables” (e.g., time, frequency, spatial coordinate) that are defined in the table.

7. Parameters

All arrays and tables must be pre-allocated before they are used in any downstream commands. This is accomplished with the **`*DIM`** command:



```
*DIM, Par, Type, IMAX, JMAX, KMAX, Var1, Var2, Var3, CSYSID
```

Parameter Name

Array or Table

Max Row

Max Column

TABLE ONLY:

- **Var1** = Primary Variable for Rows
- **Var2** = Primary Variable for Columns
- **CSYSID** = Coordinate System ID (if spatial coordinates in Var1/2)

7. Parameters

Arrays or Tables are typically populated by:

- Reading values from the database (**`*VGET`**)
- Reading values from text files (**`*VREAD`** or **`*TREAD`**)

Array or Tables are written to text file using the **`*VWRITE`** command.

<u>**NOTE**</u>: a common pitfall of the **`*VREAD`** and **`*VWRITE`** commands is the appropriate use of FORTRAN formatting. A good workaround for formatting is "**`%g`**":

```
*VWRITE,…
%g,…
```

This will let the solver automatically select an appropriate numerical format for the data (if it is numerical data).

7. Parameters

Additionally there are special functions for manipulating arrays and tables:

- **`*VSCFUN`** – takes in a vector and produces a scalar (e.g., sum all values in an array).
- **`*VOPER`** – take two vectors and produce a vector (e.g., add two vectors together).
- **`*VFILL`** – fill a vector with values using a variety of different functions.

Example: Determining Average Contact Pressure in a Contact Pair

GOAL: calculate average pressure in a contact pair

Problem: ANSYS Mechanical does not calculate average values across topology (e.g., faces or bodies)

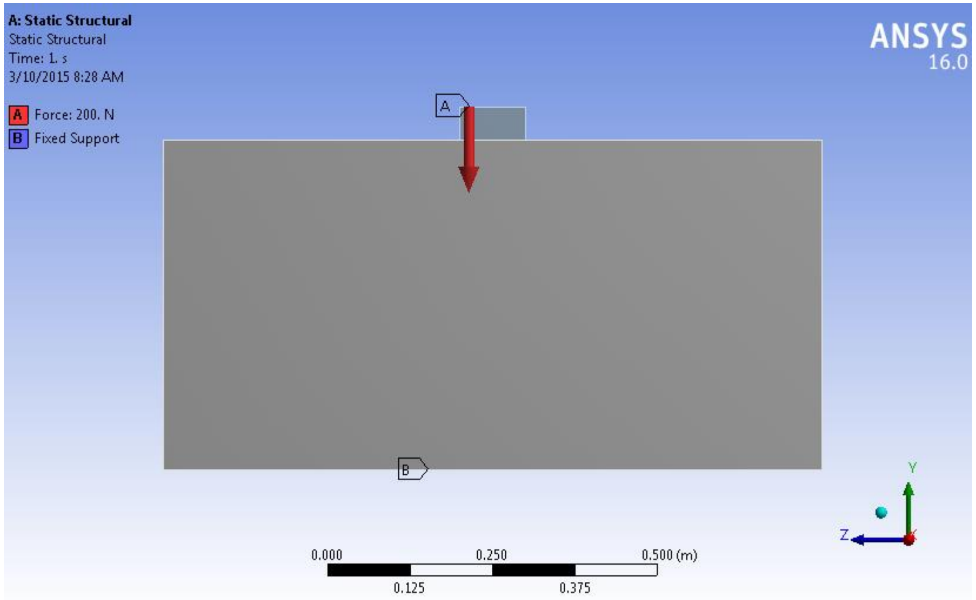
Solution: a Solution Command Object!

Example: Problem Setup

Loading/Constraints: A 200 N force **(A)** is applied to a small steel block resting on a large block of "soft" material, which is fixed at its base **(B)**.

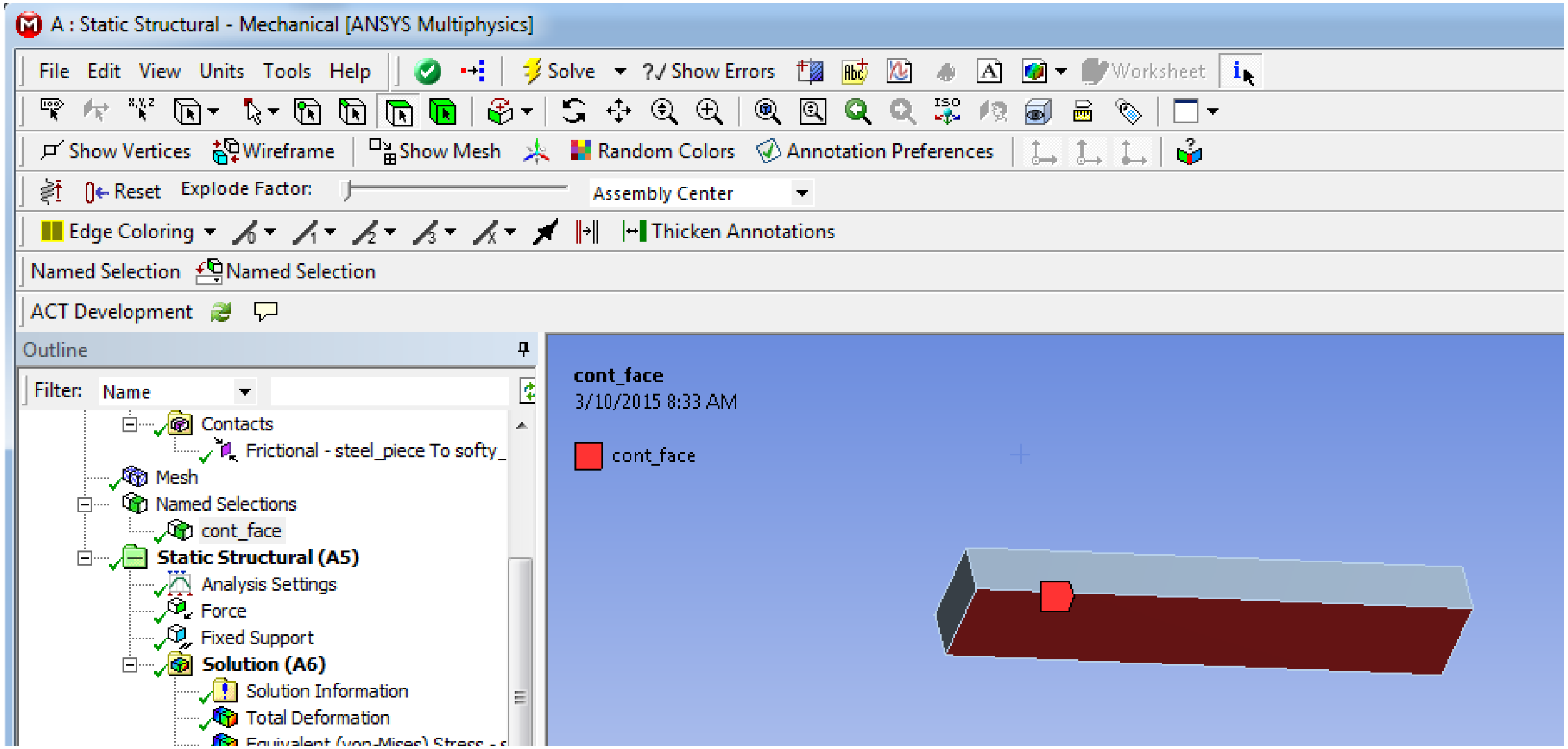
Expected Result:

200 N/0.05 m^2 = **4000 Pa**

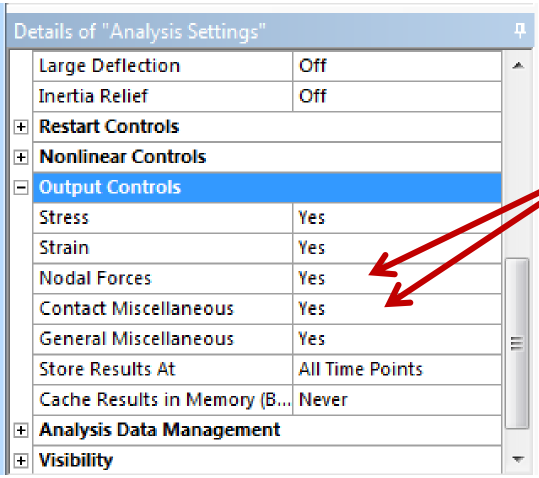


Example: Define Named Selection

A Named Selection "cont_face" is scoped to the face of the steel piece that is in contact. At solution, cont_face will be come a **nodal component** that we can select in our Solution Commands Object.



Example: Analysis Settings

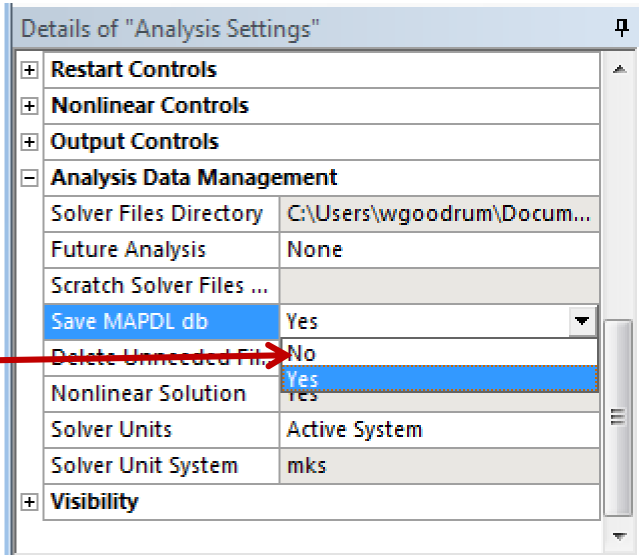


Details of "Analysis Settings"

Large Deflection	Off
Inertia Relief	Off
⊞ **Restart Controls**	
⊞ **Nonlinear Controls**	
⊟ **Output Controls**	
Stress	Yes
Strain	Yes
Nodal Forces	Yes
Contact Miscellaneous	Yes
General Miscellaneous	Yes
Store Results At	All Time Points
Cache Results in Memory (B...	Never
⊞ **Analysis Data Management**	
⊞ **Visibility**	

We must request “Contact Miscellaneous” and “Nodal Forces” > **YES** from Analysis Settings > Output Controls so that the required **area** and **force** data will be available in FILE.RST.

Analysis Settings > Analysis Data Management > Save MAPDL db is set to **YES** so that the Solution Command Object has full access to the model information (including the cont_face component!)



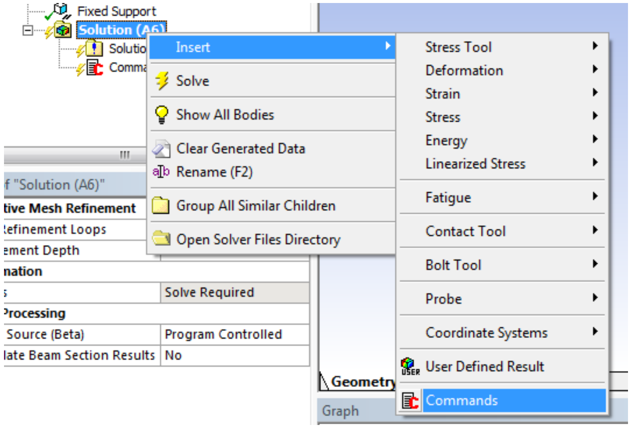
Details of "Analysis Settings"

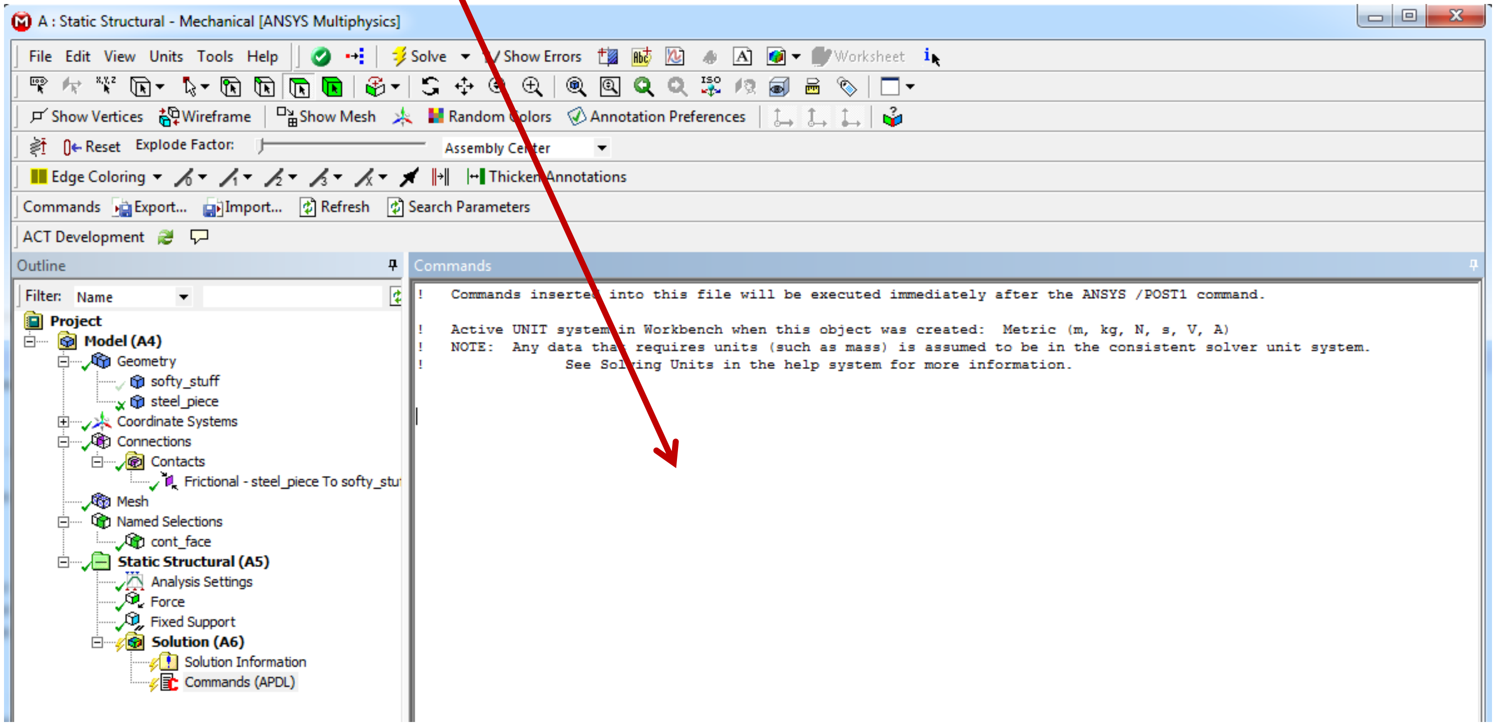
⊞ **Restart Controls**	
⊞ **Nonlinear Controls**	
⊞ **Output Controls**	
⊟ **Analysis Data Management**	
Solver Files Directory	C:\Users\wgoodrum\Docum...
Future Analysis	None
Scratch Solver Files ...	
Save MAPDL db	Yes No Yes
Delete Unneeded Fil...	
Nonlinear Solution	Yes
Solver Units	Active System
Solver Unit System	mks
⊞ **Visibility**	

Example: Commands Object

Right Click (Solution) > Insert > **Commands**

We'll write our commands in here.





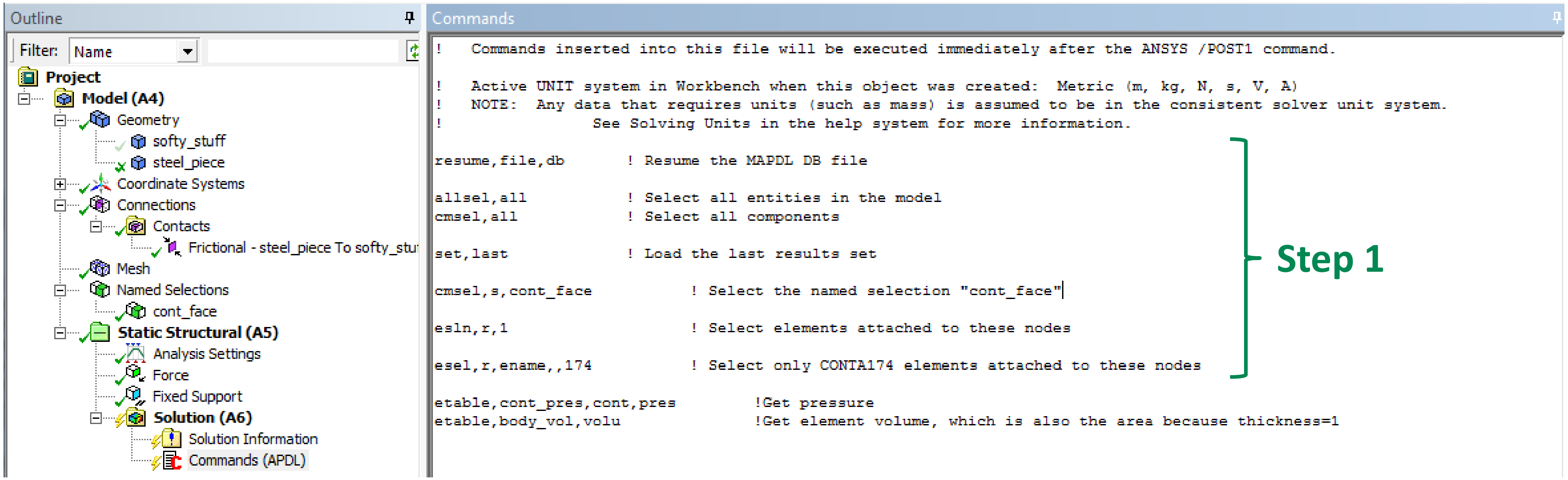
```
!   Commands inserted into this file will be executed immediately after the ANSYS /POST1 command.

!   Active UNIT system in Workbench when this object was created:  Metric (m, kg, N, s, V, A)
!   NOTE:  Any data that requires units (such as mass) is assumed to be in the consistent solver unit system.
!                See Solving Units in the help system for more information.
```

Example: The General Plan

The average contact pressure will be calculated by:

1. Selecting the contact elements attached to the nodes in the cont_face component
2. Extracting the area of these elements and the contact pressures on these elements from ETABLEs
3. Define arrays and put ETABLE values into arrays using **`*VGET`**
4. Calculating the weighted average of the element pressures via **`*VOPER`** and **`*VSCFUN`**

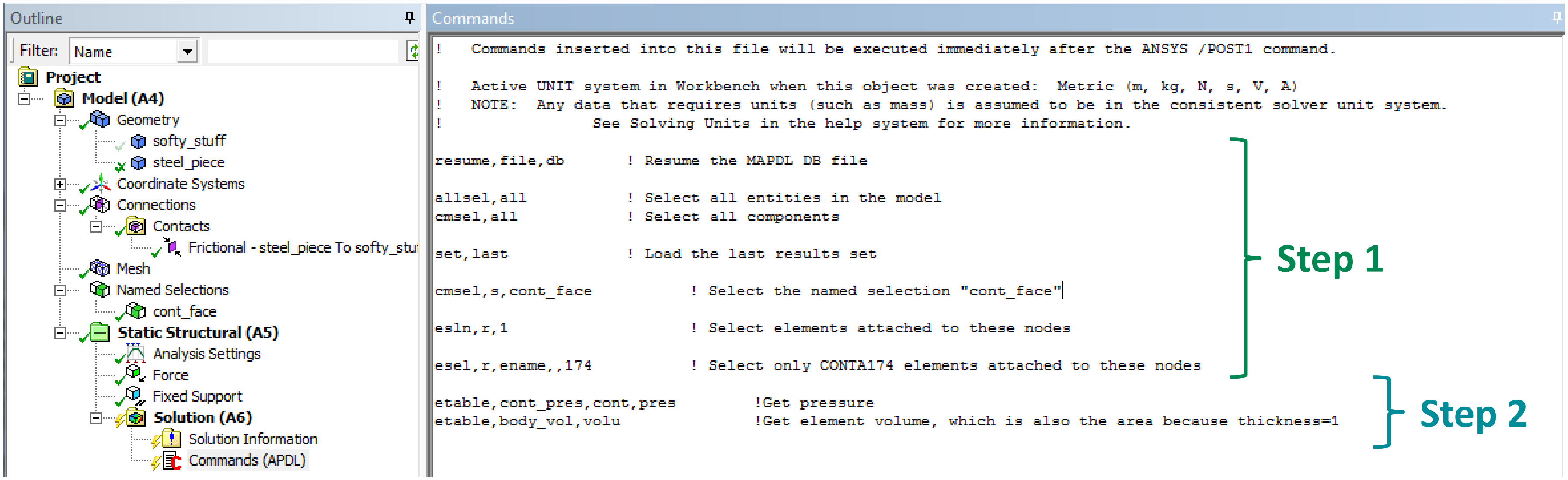
Step 1. Select the contact elements (CONTA174)



```
!   Commands inserted into this file will be executed immediately after the ANSYS /POST1 command.

!   Active UNIT system in Workbench when this object was created:  Metric (m, kg, N, s, V, A)
!   NOTE:  Any data that requires units (such as mass) is assumed to be in the consistent solver unit system.
!                See Solving Units in the help system for more information.

resume,file,db         ! Resume the MAPDL DB file

allsel,all             ! Select all entities in the model
cmsel,all              ! Select all components

set,last               ! Load the last results set

cmsel,s,cont_face             ! Select the named selection "cont_face"

esln,r,1                      ! Select elements attached to these nodes

esel,r,ename,,174             ! Select only CONTA174 elements attached to these nodes

etable,cont_pres,cont,pres           !Get pressure
etable,body_vol,volu                 !Get element volume, which is also the area because thickness=1
```

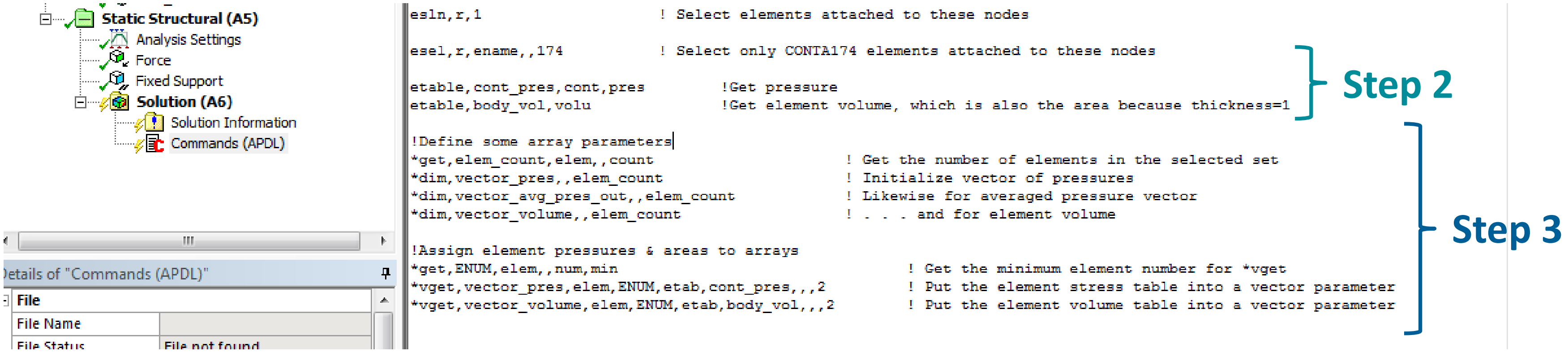
Step 1

Step 2. Define ETABLEs for contact pressure and area



```
!   Commands inserted into this file will be executed immediately after the ANSYS /POST1 command.

!   Active UNIT system in Workbench when this object was created:  Metric (m, kg, N, s, V, A)
!   NOTE:  Any data that requires units (such as mass) is assumed to be in the consistent solver unit system.
!                 See Solving Units in the help system for more information.

resume,file,db           ! Resume the MAPDL DB file

allsel,all               ! Select all entities in the model
cmsel,all                ! Select all components

set,last                 ! Load the last results set

cmsel,s,cont_face               ! Select the named selection "cont_face"

esln,r,1                        ! Select elements attached to these nodes

esel,r,ename,,174               ! Select only CONTA174 elements attached to these nodes

etable,cont_pres,cont,pres             !Get pressure
etable,body_vol,volu                   !Get element volume, which is also the area because thickness=1
```

Step 3. ETABLE → Array

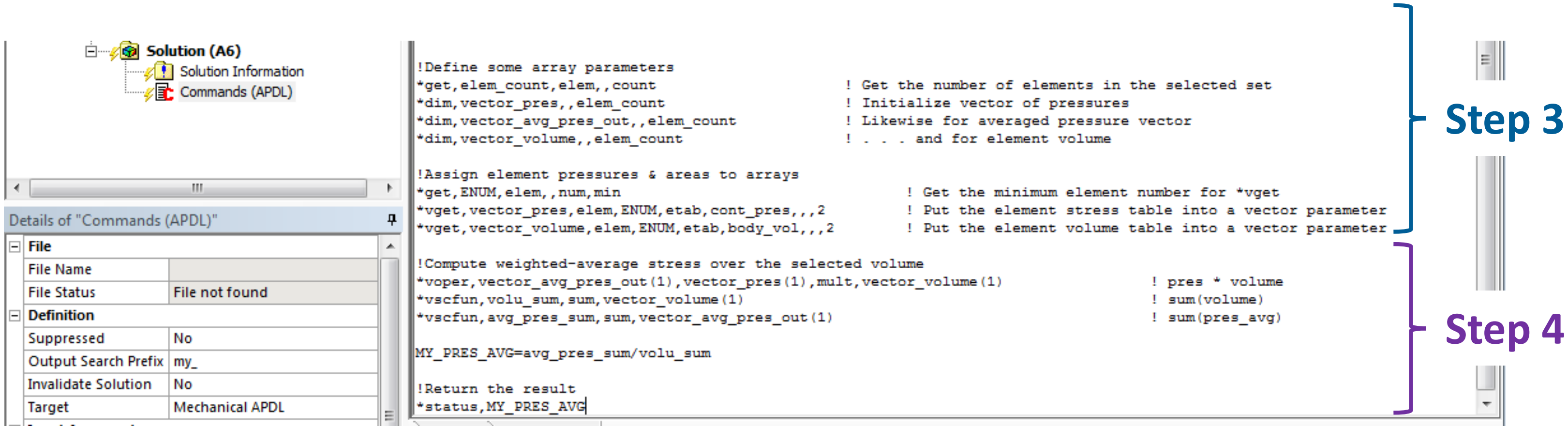


```
esln,r,1                        ! Select elements attached to these nodes

esel,r,ename,,174               ! Select only CONTA174 elements attached to these nodes

etable,cont_pres,cont,pres            !Get pressure
etable,body_vol,volu                  !Get element volume, which is also the area because thickness=1

!Define some array parameters
*get,elem_count,elem,,count                         ! Get the number of elements in the selected set
*dim,vector_pres,,elem_count                        ! Initialize vector of pressures
*dim,vector_avg_pres_out,,elem_count                ! Likewise for averaged pressure vector
*dim,vector_volume,,elem_count                      ! . . . and for element volume

!Assign element pressures & areas to arrays
*get,ENUM,elem,,num,min                                    ! Get the minimum element number for *vget
*vget,vector_pres,elem,ENUM,etab,cont_pres,,,2             ! Put the element stress table into a vector parameter
*vget,vector_volume,elem,ENUM,etab,body_vol,,,2            ! Put the element volume table into a vector parameter
```

Step 2

Step 3

Step 4. Calculate Weighted Average Contact Pressure



File	
File Name	
File Status	File not found
Definition	
Suppressed	No
Output Search Prefix	my_
Invalidate Solution	No
Target	Mechanical APDL

```
!Define some array parameters
*get,elem_count,elem,,count                     ! Get the number of elements in the selected set
*dim,vector_pres,,elem_count                    ! Initialize vector of pressures
*dim,vector_avg_pres_out,,elem_count            ! Likewise for averaged pressure vector
*dim,vector_volume,,elem_count                  ! . . . and for element volume

!Assign element pressures & areas to arrays
*get,ENUM,elem,,num,min                                ! Get the minimum element number for *vget
*vget,vector_pres,elem,ENUM,etab,cont_pres,,,2         ! Put the element stress table into a vector parameter
*vget,vector_volume,elem,ENUM,etab,body_vol,,,2        ! Put the element volume table into a vector parameter

!Compute weighted-average stress over the selected volume
*voper,vector_avg_pres_out(1),vector_pres(1),mult,vector_volume(1)          ! pres * volume
*vscfun,volu_sum,sum,vector_volume(1)                                       ! sum(volume)
*vscfun,avg_pres_sum,sum,vector_avg_pres_out(1)                             ! sum(pres_avg)

MY_PRES_AVG=avg_pres_sum/volu_sum

!Return the result
*status,MY_PRES_AVG
```

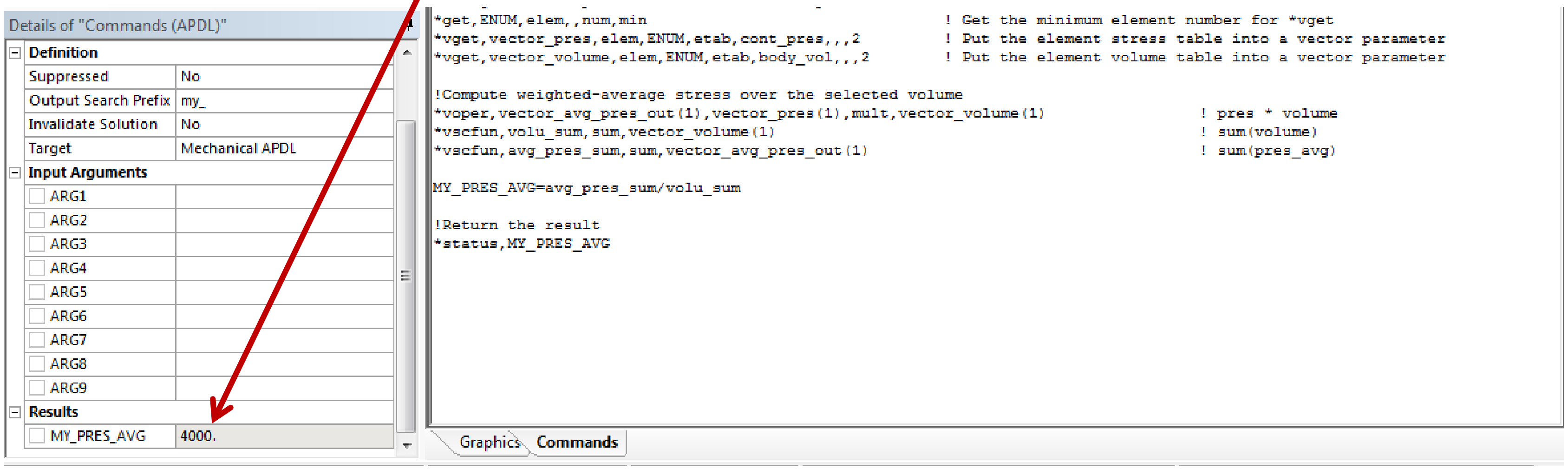
Step 3

Step 4

Example: Result

Expected Result:

200 N/0.05 m^2 = **4000 Pa**



Details of "Commands (APDL)"

Definition	
Suppressed	No
Output Search Prefix	my_
Invalidate Solution	No
Target	Mechanical APDL
Input Arguments	
□ ARG1	
□ ARG2	
□ ARG3	
□ ARG4	
□ ARG5	
□ ARG6	
□ ARG7	
□ ARG8	
□ ARG9	
Results	
□ MY_PRES_AVG	4000.

```
*get,ENUM,elem,,num,min                            ! Get the minimum element number for *vget
*vget,vector_pres,elem,ENUM,etab,cont_pres,,,2     ! Put the element stress table into a vector parameter
*vget,vector_volume,elem,ENUM,etab,body_vol,,,2    ! Put the element volume table into a vector parameter

!Compute weighted-average stress over the selected volume
*voper,vector_avg_pres_out(1),vector_pres(1),mult,vector_volume(1)              ! pres * volume
*vscfun,volu_sum,sum,vector_volume(1)                                           ! sum(volume)
*vscfun,avg_pres_sum,sum,vector_avg_pres_out(1)                                 ! sum(pres_avg)

MY_PRES_AVG=avg_pres_sum/volu_sum

!Return the result
*status,MY_PRES_AVG
```

Appendix – Help Documentation Links

1. **Resume the DB File**

   RESUME: help/ans_cmd/Hlp_C_RESUME.html

2. **Load Results File**

   SET: help/ans_cmd/Hlp_C_SET.html

3. **Access Model/Results Information in DB File**

   ***GET:** help/ans_cmd/Hlp_C_GET.html
   ***VGET:** help/ans_cmd/Hlp_C_VGET_st.html
   Get Function Shortcuts: help/ans_apdl/Hlp_P_APDLget.html

Appendix – Help Documentation Links

4. **Apply Selection Logic to Make Entities Active/Interactive**

   NSEL: help/ans_cmd/Hlp_C_NSEL.html
   ESEL: help/ans_cmd/Hlp_C_ESEL.html
   CMSEL: help/ans_cmd/Hlp_C_CMSEL.html
   ESLN: help/ans_cmd/Hlp_C_ESLN.html
   NSLE: help/ans_cmd/Hlp_C_NSLE.html
   ALLSEL: help/ans_cmd/Hlp_C_ALLSEL.html

5. **Create Element Tables**

   ETABLE: help/ans_cmd/Hlp_C_ETABLE.html
   SSUM: help/ans_cmd/Hlp_C_SSUM.html
   PRETAB: help/ans_cmd/Hlp_C_PRETAB.html

Appendix – Help Documentation Links

6. Create Plots

PLNSOL: help/ans_cmd/Hlp_C_PLNSOL.html
PLESOL: help/ans_cmd/Hlp_C_PLESOL.html
PLETAB: help/ans_cmd/Hlp_C_PLETAB.html

7. Parameters

***DIM:** help/ans_cmd/Hlp_C_DIM.html
***TREAD:** help/ans_cmd/Hlp_C_TREAD.html
***VFILL:** help/ans_cmd/Hlp_C_VFILL.html
***VGET:** help/ans_cmd/Hlp_C_VGET_st.html
***VOPER:** help/ans_cmd/Hlp_C_VOPER.html
***VSCFUN:** help/ans_cmd/Hlp_C_VSCFUN.html
***VREAD:** help/ans_cmd/Hlp_C_VREAD.html
***VWRITE:** help/ans_cmd/Hlp_C_VWRITE.html

Appendix – Release Notes

April 2015: version 1 – first release of notes