

---

## EXERCISE 8

# *Loading Loads and BC's and Element Verification Parameters*

The image shows a screenshot of the 'Finite Elements' dialog box in PATRAN. The dialog has a title bar 'Finite Elements' with a close button. It contains several fields and checkboxes:

- Action: Verify
- Object: Quad
- Test: All
- Normalize
- Analysis Code: Miracle
- Reliability Threshold
- Aspect Ratio: 4.
- Warp Angle: 5.

### Objectives:

- Edit *exercise\_08.pcl* to add loads and element verification thresholds.



**Problem Description:**

In this Exercise, you will modify *exercise\_08.template* file discussed in the class to include temperature and pressure boundary conditions. We will also modify the element verification thresholds as follows:

**Table 1: Element Verification Thresholds**

Test	Threshold
aspect ratio, triangular elements	4.0
skew, triangular elements	25.0
aspect ratio, quadrilateral elements	4.0
skew, quadrilateral elements	25.0
warp, quadrilateral elements	5.0
taper, quadrilateral elements	.8
aspect ratio, tetrahedral elements	25.0
face skew, tetrahedral elements	4.0
collapse, tetrahedral elements	0.15
edge angle, tetrahedral elements	25.0
aspect ratio, pentahedral elements	4.0
face skew, pentahedral elements	25.0
face taper, pentahedral elements	.8
face warp, pentahedral elements	5.0
twist, pentahedral elements	45.0
edge angle, pentahedral elements	25.0
face skew, hexahedral elements	25.0
face warp, hexahedral elements	5.0
face taper, hexahedral elements	0.8
edge angle hexahedral elements	25.0
aspect ratio, hexahedral elements	4.0

**Table 1: Element Verification Thresholds**

Test	Threshold
twist, hexahedral elements	45.0
mid-side node normal offset, all elements	0.30
mid-side node tangent offset, all elements	0.30

## Suggested Exercise Steps:

- Edit *exercise\_08.template* file and make modifications to add the temperature and pressure boundary conditions.
- Edit *exercise\_08.template* file and make modifications to add the specified element threshold parameters
- After renaming the file to **exercise\_08.pcl**, compile the PCL program. Be sure to use the C pre-processor.
- Open a new database using the *base.db* template.
- Run `load_miracle()` and quit *p3*
- Verify the form.
- Verify the database relations/attributes.
- After the forms have been verified, copy *exercise\_08.pcl* to *load\_miracle.pcl*.

## Exercise Procedure:

1. Either use `vi` or `jot` as the text editing tool. Open the file named *exercise\_08.template*. It should already exist in your directory. Edit the file and rename it to *exercise\_08.pcl*.
2. Compile the function.

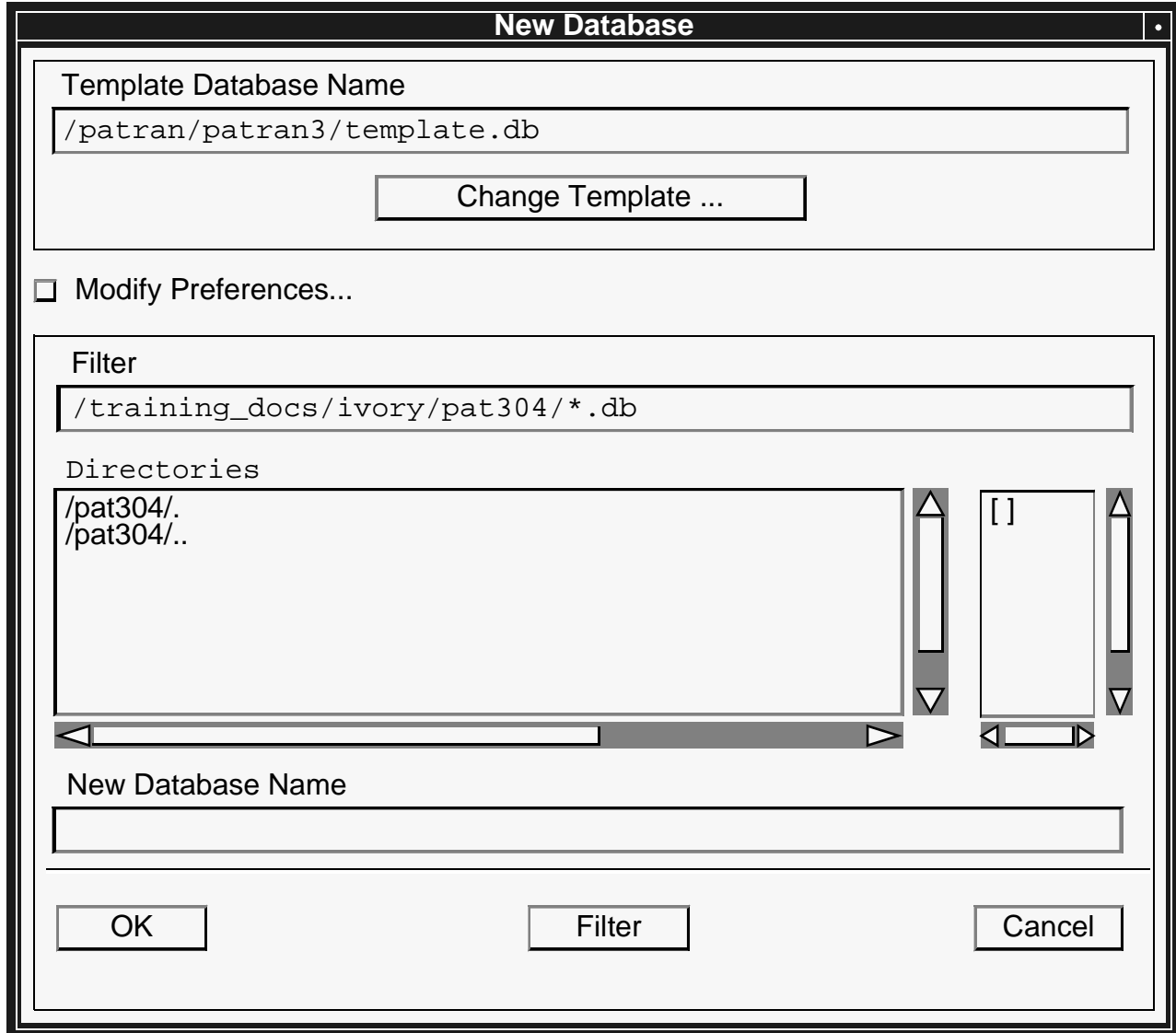
Type in the command line:

```
% /usr/lib/cpp -P exercise_08.pcl exercise_08.cpp  
% p3
```

Type `p3` in your xterm. At the command line type:

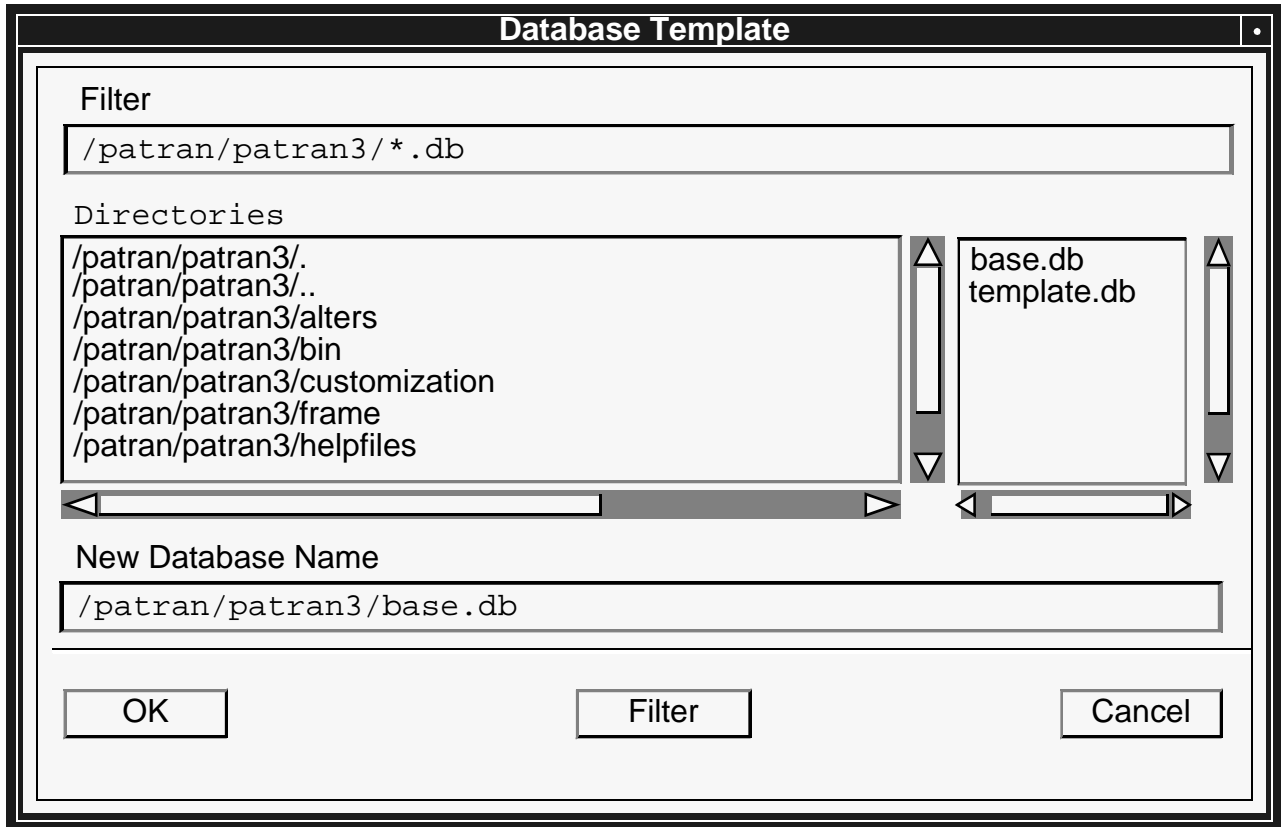
```
!!input exercise_08.cpp
```

Select **File** from the top menu bar. Select **New Database...** from the pull down menu.



Click the **Change Template...** button.

In the *Database Template* form, select **base.db** from the Database Template List listbox.



Click the **OK** button.

In the *New Database* form, click in the New Database Name databox and enter the name **miracle\_template.db**.

**New Database**

Template Database Name  
/patran/patran3/base.db  
Change Template ...

Modify Preferences...

Filter  
/training\_docs/ivory/pat304/\*.db

Directories  
/pat304/  
/pat304/..

New Database Name  
miracle\_template

OK Filter Cancel

Click the **OK** button.

Now type **load\_miracle()** in the command line and hit return.

Close the database.

Quit p3.

3. Test your modifications to *exercise\_08.template*.

Begin PATRAN again by entering p3 at the prompt.

Select the **File** in the *top menu bar*. Select **New Database...** from the pull-down menu. In the *New Database* form, change the Template Database Name to **miracle\_template.db**. Change the New Database Name to **exercise\_8.db**.

The image shows a 'New Database' dialog box with the following fields and controls:

- Template Database Name:** A text field containing 'miracle\_template.db' and a 'Change Template ...' button below it.
- Modify Preferences...**
- Filter:** A text field containing '/training\_docs/ivory/pat304/\*.res'.
- Directories:** Two list boxes. The left list box contains '/pat304/.' and '/pat304/..'. The right list box contains '[miracle\_template.db ]'.
- New Database Name:** A text field containing 'exercise\_8.db'.
- Buttons:** 'OK', 'Filter', and 'Cancel' buttons at the bottom.

Click the **OK** button.

4. Verify the result.

Select the **Finite Elements** radio button on the *Control Panel*:



Click here

You should see the following form appear:

Change to "Verify"

Change to "Quad"

---

Now, change the *Action* option menu to **Verify** and the *Object* option menu to **Quad**.

The form should look like the one below:

The image shows a dialog box titled "Finite Elements". It contains several controls: three dropdown menus for "Action" (set to "Verify"), "Object" (set to "Quad"), and "Test" (set to "All"); an unchecked checkbox for "Normalize"; a text field for "Analysis Code" containing "Miracle"; a section titled "Reliability Threshold" containing several text fields: "Aspect Ratio" (4.), "Warp Angle" (5.), "Skew Angle" (25.), "Taper" (0.8000001), "Normal Offset" (0.3000001), and "Tangent Offset" (0.3000001); another unchecked checkbox for "Write to Report"; and an "Apply" button at the bottom.

Check to make sure that the values are correct. To check the values of other element types (e.g. triangular, hexahedral, ...) simply change the Object option menu to the appropriate element name.

5. Using QLI, inspect the entries you have just created in the database. The following are some of the relations (i.e. Tables) you may want to examine.

- element\_verification\_parms

To list all of the relations in the p3 database, use the following command

```
QLI> show relations
```

We have included a sample QLI session for your reference. However, feel free to let your curiosity take you to unfamiliar relations!

```
dresden_%qli
Welcome to QLI
Query Language Interpreter
QLI> ready miracle_template.db
QLI> print element_verification_parms
```

```
ANALYSIS
CODE TEST
ID ID THRESHOLD
=====
1 1 4
1 2 25
1 3 4
1 4 25
1 5 5
1 6 0.8
1 7 4
1 8 25
1 9 0.15
1 10 25
1 11 4
1 12 25
1 13 0.8
1 14 5
1 15 45
1 16 25
1 17 25
1 18 5
1 19 0.8
1 20 25
1 21 4
1 22 45
1 23 0.3
1 24 0.3
```

## Sample Solution

```

#include "app_ep_defn_ids.i"
#include "app_ep_prop_ids.i"
#include "mat_ind_cons.i"
#include "mat_words.p"
#include "elem_tests.p"

#define _YES 0
#define _NO 1
#define _NOT_VALID 0
#define _SCALAR 1
#define _VECTOR 2
#define _INTEGER 3
#define _STRING 4
#define _MATERIAL 5
#define _LIST 6
#define _FIELD 7
#define _NODAL 8
#define _COORD 9
#define _STRUCTURAL 1
#define _THERMAL 2

FUNCTION load_miracle()

    INTEGER status, bar_code, plate_code, solid_code, anal_code_id

    /*
    * Fetch the first unused analysis code id
    */
        •
        •
        •

!`status`
$ Assign the already existing definitions for FORCE and DISPLACEMENT
$ to Miracle
status = db_add_lbc_type_for_ac( anal_code_id, 6 )
!`status`
status = db_add_lbc_type_for_ac( anal_code_id, 7 )
!`status`

$ Define TEMPRATURE & PRESSURE boundary condition
$ to Miracle
***** 1 *****
!`status`
***** 1 *****
!`status`
s
$ Define MPC types Explicit, Rigid(Fixed), Rigid(Pinned)
status = db_create_mpc_type_def ( 41, "Explicit", FALSE, FALSE, TRUE, "Constant Term", @
    FALSE, FALSE, TRUE, 1, 1, 1, 1, TRUE, TRUE, 0, 1, 1, 1 )
!`status`
status = db_create_mpc_type_def ( 42, "Rigid(Fixed)", FALSE, FALSE, FALSE, " ", @
    FALSE, FALSE, FALSE, 0, 1, 0, 1, FALSE, FALSE, 1, 1, 0, 1 )
!`status`
status = db_create_mpc_type_def ( 43, "Rigid(Pinned)", FALSE, FALSE, FALSE, " ", @
    FALSE, FALSE, FALSE, 0, 1, 0, 1, FALSE, FALSE, 1, 1, 0, 1 )
!`status`
$ Associate MPC types with the new analysis code
status = db_create_valid_mpc_type ( 41, anal_code_id, 1, 6, [1, 2, 3, 4, 5, 6] )
!`status`
status = db_create_valid_mpc_type ( 42, anal_code_id, 1, 0, [0] )

```



## Solutions

```

1)status = db_add_lbc_type_for_ac( anal_code_id, 8 )
   status = db_add_lbc_type_for_ac( anal_code_id, 9 )

2)status = db_set_elem_verification_params ( "Miracle", _TRI_ASPECT, 4.0 )
   status = db_set_elem_verification_params ( "Miracle", _TRI_SKWM, 25.0 )
   status = db_set_elem_verification_params ( "Miracle", _QUAD_ASPECT, 4.0 )
   status = db_set_elem_verification_params ( "Miracle", _QUAD_SKWM, 25.0 )
   status = db_set_elem_verification_params ( "Miracle", _QUAD_WARP, 5.0 )
   status = db_set_elem_verification_params ( "Miracle", _QUAD_TAPER, 0.8 )
   status = db_set_elem_verification_params ( "Miracle", _TFT_ASPECT, 4.0 )
   status = db_set_elem_verification_params ( "Miracle", _TFT_SKWM, 25.0 )
   status = db_set_elem_verification_params ( "Miracle", _TFT_COLLAPS, 0.15 )
   status = db_set_elem_verification_params ( "Miracle", _TFT_EDANGLE, 25.0 )
   status = db_set_elem_verification_params ( "Miracle", _WEDGE_ASPECT, 4.0 )
   status = db_set_elem_verification_params ( "Miracle", _WEDGE_FSKWM, 25.0 )
   status = db_set_elem_verification_params ( "Miracle", _WEDGE_FWARP, 0.8 )
   status = db_set_elem_verification_params ( "Miracle", _WEDGE_FWARP, 5.0 )
   status = db_set_elem_verification_params ( "Miracle", _WEDGE_TWIST, 45.0 )
   status = db_set_elem_verification_params ( "Miracle", _WEDGE_EDANGLE, 25.0 )
   status = db_set_elem_verification_params ( "Miracle", _HEX_FSKWM, 25.0 )
   status = db_set_elem_verification_params ( "Miracle", _HEX_FWARP, 5.0 )
   status = db_set_elem_verification_params ( "Miracle", _HEX_FTAPER, 0.8 )
   status = db_set_elem_verification_params ( "Miracle", _HEX_EDANGLE, 25.0 )
   status = db_set_elem_verification_params ( "Miracle", _HEX_ASPECT, 4.0 )
   status = db_set_elem_verification_params ( "Miracle", _HEX_TWIST, 45.0 )
   status = db_set_elem_verification_params ( "Miracle", _HOE_NORMOFF, 0.30 )
   status = db_set_elem_verification_params ( "Miracle", _HOE_TANOFF, 0.30 )

```