# Unit 13 Data Access

**Structure:**

## 13.1 Introduction

By this time you must be familiar with the design of model and its notions like process and reuse. We also explored to design the model with the controls and retrieve and display data from the database table. Also we explored the process templates with action and find. By unifying the definition of user interface, client-side behaviour, and server-side processing into a single visual language, ersus streamlines the process of creating online applications.

In this unit we are going to discuss the “Insert” template that are used to store the data in the database. We are also exploring the various action models that help to define the process in Tersus platform. The major focus of this unit is to discuss the various techniques that are available to display the data in the table format that are retrieved from the database including reuse of existing data structure action models that are populating the data list, how to define the sub process and display the data element using refresh requisition and the insert template.

**Objectives:**

After studying this unit, you will be able to:

- design a table display for data base data
- explain the role of Insert template

- develop application using the existing data structure
- define the sub process
- explain the techniques used to display data element

Let us first brush up on Visual Programming. It is either a programming language or a programming environment, Instead of being interpreted as text, the program is represented and manipulated graphically, It enables the programmer to think visually from the machine's point of view, It enables humans to explain the process with the help of illustrations. It allows people to include visual aids in their explanations of the procedure. The programmer may now view the world as the machine does.

Data Access in Visual Basics:

Now, let us understand Data Access in VB (Virtual Basic). It is a Visual Basic function that lets you manipulate any database from inside the program, Data access is made possible with the help of VB applications, It allows us to use a variety of databases, including MS-Access and Fox-Pro databases, also we can Using Visual Basic's Data Access Objects, we may alter our database's structure.We can use many types of databases, such as MS-Access and FoxPro.

Tersus Visual Programming:
To put it simply, it's a visual programming language that adapted to meet a wide variety of needs.

- Visual programming framework for application construction
- Nothing programmed or scripted
- It's a tool for developing apps for the web and mobile devices.
- Programming logic can be diagrammed using flowcharts.

**Components of tersus:**

There are three major components involved in tersus

1. Tersus Studio
2. Tersus Model
3. 3.Tersus Server

Tersus studio is basically considered to be modelers use them to graphically describe the application's features. It is an extension of the Eclipse program, the tarsus model libraries includes components for putting together applications and finally the tarsus server carries out the modeled solutions as well as the necessary updates.

1. Tersus Studio: Tersus Studio is more like Android Studio, where gadgets of the application can be dragged and fixed. The user will draw the diagram and the program will put it into action The diagram depicts the entire system, including the screen configuration, client actions and server process, There is no need for text coding or scripting. Data Access is made very efficient and fast. Drag and drop view elements such as buttons, tables, fields and other elements to build layouts Everything from the screen layout to client operations to server procedures are shown in the diagram.
2. Tersus Model: All the building blocks for programmes are included. A model is the sum of the parts that make up an application, including the systems, displays, processes, data structures, and data items.It's a runtime driver that makes it possible to run any given application model.

The expressive potential of a language is modelled by its syntax.Semantics of modelling languages describe the principles that govern the studio-generated models' validity.

3. Tersus Server: . They support many OSes, so you may choose the one that works best for you: Windows, Linux, UNIX, or Mac OS. It enables us to execute a simulated programme in parallel with itself. It logs and transmits information about the application's operation to the production office. It's a platform that can host both online and mobile applications.

Data Access Mechanism: there are three types first one is Data Access Object referred to as DAO, second one is Remote Data Object referred as RDO and the third one is Active Data Object refereed as ADO. DAO gives you access to the database in a particular way, the RDO was created with the intention of making communication easier with remote databases were as ADO is a combination of DAO and RDO

Let us understand the features of Data Access Object. Accessibility, It allows us to use the Jet database engine that comes with MS Access, DAO is supported in Visual Basic 3.0

- Second feature is interface It is extremely user-friendly and easy to use, The user interface is very interactive
- the third one Popularity , It rapidly grew in popularity as one of the most widely used databases

In comparison to its equivalents, it is the most widely used

Remote Data Object:

It was created to deal with the ODBC’s complexities

, It allows us to link to a wide range of databases with just one method , It enables the alteration of databases, It is a database that allows DAO users to connect to other databases

## Active Data Object Features:

Now, let us understand the different features of ADO. It's like getting the best of both DAO and RDO in one convenient bundle. To communicate with the database, it employs the Object Linking and Embedding Database (OLEDB) protocol. It can create virtual tunnels over the network to get to distant servers. It enables quicker and more efficient access to a variety of data sources, It has the ability to slice the internet in order to access various data storages

Some other data access technologies apart from what we have discussed are mentioned here.

- ODBC , Open Database Connectivity is a term that refers to the ability to link several databases together,The requested statement is translated into native text using driver manager
- SQL Access Group, this was Started in 1989, It establishes guidelines for transferring data between local and remote systems and the
- last one is OLED which was introduced in 1996 for databases, it stands for Object Linking and Embedding

Populating the process simply refers to populating/transmitting the data into the table from some dataset. The term ‘populating the process’ refers to the process of populating the table with data from the available dataset Filling out a table with information from a dataset is what is meant by "populating the process. "Hierarchical representation is the most effective approach for populating structured data, Data retrieval is performed in a meaningful organized way when populating, It defines the data type with the Display Data feature and then sends the data to the Tersus platform. The Display Data function classifies the information before transmitting it to the Tersus server. Information is retrieved and populated in a meaningful, ordered fashion.

## Action model for populating the process:

Mentioned above are the action models for populating the data to the table from dataset. Drag the Basic/Action template to the ‘Open Recognitions Model’ and drop it there, Define a sub-process to produce the memory table data, it retrieves data from the database, use the ‘Find Template’ and to The Add flow function is used to complete the process of creating the Requisition list table. Also To describe the data type and construct the table data element, use the Display data element, In order to generate the table data element and define the data type, the Display data element is used.

### Advantage of Data access:

Advantages of Data Access in Visual Programming are far reaching and outshine its shortcomings. Access to data is more robust and understandable, Better handling of queries and result sets, Better handling of queries and result sets, It is capable of handling both relational and non-relational databases in short we can say that it provides Enhanced management of searches and output, Improved data accessibility and readability, as well as more efficient query processing and output.

### Linitations of Data Access:

As of now, there are some major limitations in Data Access in Visual Programming. However, these are being rectified at a very high pace. Regrading accessibility and performance , it is slower than the textual coding, even a simpe process consumes more energy and heavy-duty applications such as gaming are not recommended, then with respect to compatibility the disadvantage part is that Only the Windows operating system is supported by the applications, transferring programmes is difficult and it mostly supports database programs that are not used on a daily basis.

On overall basis In comparison to textual coding, it is less accessible and has lower performance; even a simple process uses more energy, so it's not ideal for demanding tasks like video games. Only Windows is supported by the applications, programme transfers are cumbersome, and most of the supported software is database software that is rarely used.

## 13.2 Insert Template

We discussed the creation of data structure and populating the structure with the list of data in the previous units. Now we are going to insert the “Insert” template to store the data in the database. Below is the syntax to insert the template.

- Select the Database/Insert template (▮+) and drop it next to the Requisitiondata model.

<Record> - (▷) trigger in the Insert template helps to receive the information from the data structure and store it in the data base. When the

process starts executing the trigger points are acting as the entry point to receive the input data. Now we will create a flow to send the Requisition data structure to the “Insert”. To do this

- Select the Flow tool (→), then click on the Requisition data structure (anywhere outside the Id and Description fields) to define it as the source of the flow, and then click on the <Record> trigger of the Insert model to define it as the target of the flow.

After doing this the model looks as shown in figure 13.1.

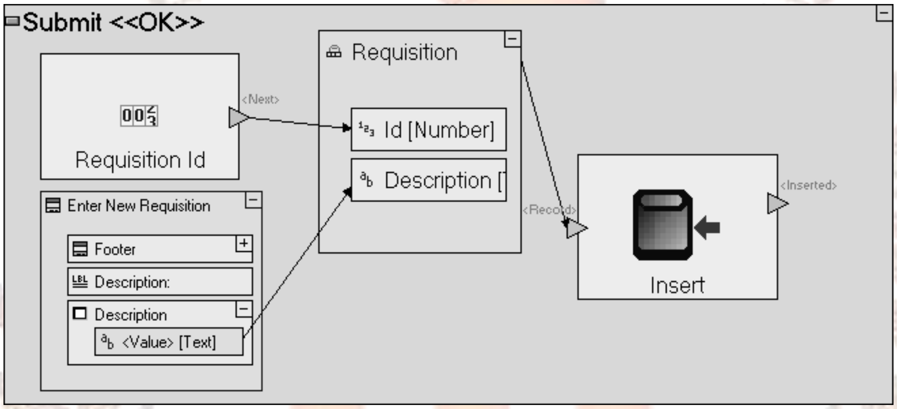


**Fig. 13.1: Model window with Insert template**

### Close window template

As the name indicates once the data are written to the database the window needs to be closed for this we needs to include the close window in to the model window.

- Select the Display Actions/Close Window template (⊠) and drop it into the Submit button.

After inserting the close window the submit model has the two option either you can save the data to the database through insert or you can close the window. In order to avoid the accidental errors that the user may click the close window before the insert operation we can fix the order. Basically, the order of execution of elements of a process (its sub-processes and data elements) is determined by the flows between them: an element is only executed (a sub-process starts or a data element obtains its value) after all the elements that have flows to it. Hence, an element that has no flows leading to it (typical examples are an initialization sub-process or a data constant) is executed immediately (if there are several such elements, the server is free to execute them in any order).

When there are flows leading to an element (i.e. the element or some of its sub-elements are the targets of a flow or several flows), then the element may not be executed before theses flows have been executed (or it becomes clear they will never be executed).

When a flow is executed, it passes data from its source to its target. Notice that in case the source is an empty slot, no data is passed through the flow (just a generic activation indication).

When repetitive sub-processes and data elements are involved, they are executed multiple times, each time a different instance. E.g. when there is a flow (or multiple flows) whose target is the trigger of a repetitive sub-process, then each time data arrives to the trigger, a new instance of the sub-process is created, receiving the triggering data object as input.

The mechanism that ensures these execution order rules is a dependencies system; an element is only executed after the elements on which it depends:

- (R1) A target of a flow depends on its source.
- (R2) When there are flows to a data element and to a sub-element of it, then the source of the flow to the sub-element depends on the source of the flow to the parent element.

Most of the time the application will be left free to the user choice without deciding the flow of control, but here it is better to fix the order in which the control should precede. If you want say that “Enter New Requisition popup window closes only after data has been saved” we need to define that the close window will be enabled after the successful insertion of “Insert” template. To make this conditional flow we need to add the trigger to the close window either by adding new trigger slot from palette taking the advantage of hiding close window.

- Right-click on the Close Window element, opens the Add Element sub-menu, and select the Control trigger element.

Also we need to insert all the data and when Insert becomes successful then the window can be closed.

- Select the Flow tool to link the <Inserted> exit of Insert to the Control trigger of Close Window.

Figure 13.2 exhibits the model window with the Insert and the Close window with the appropriate control on which order it needs to be executed.

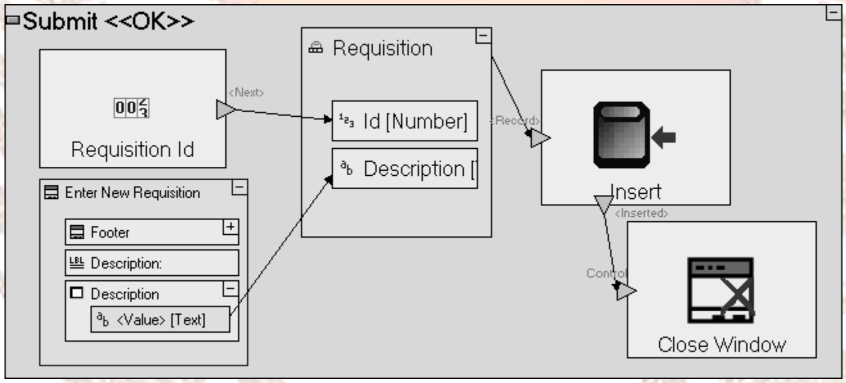


**Fig. 13.2: Model window with control enabled for flow**

Now you can create a project with the following controls on the model window and further we will be using this project for displaying data in table format as shown in figure 13.3.

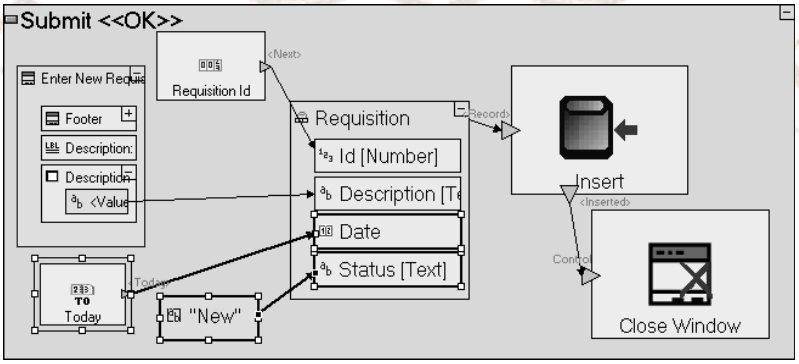
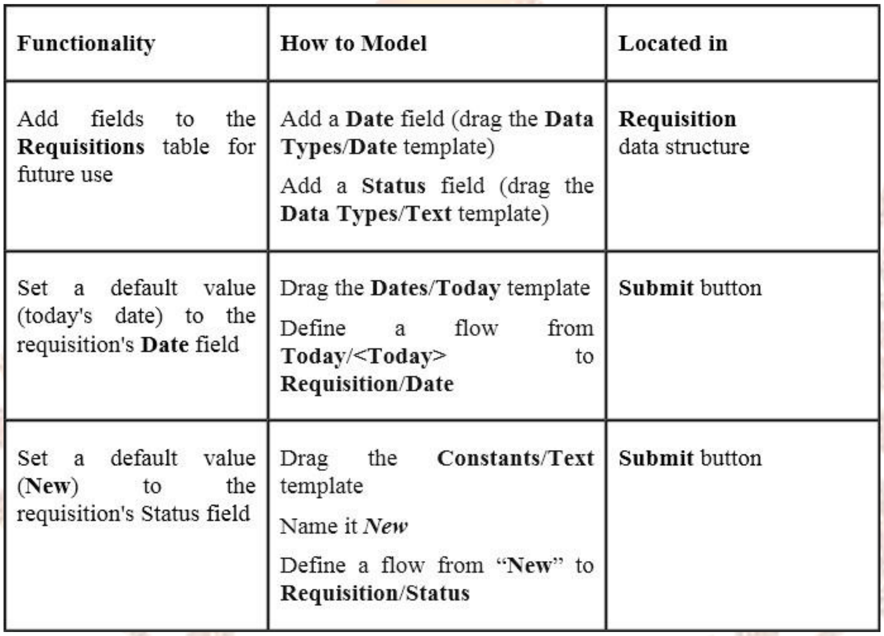


**Fig. 13.3: Model window for design**

Table 13.1 provide hints to design the model window, that has four requisition control , new request control , submit, Insert and Close window options with control and the control to display the current date.

**Table 13.1: Functionality with model design**



| **Functionality** | **How to Model** | **Located in** |
|---|---|---|
| Add fields to the **Requisitions** table for future use | Add a **Date** field (drag the **Data Types/Date** template)<br>Add a **Status** field (drag the **Data Types/Text** template) | **Requisition**<br>data structure |
| Set a default value (today's date) to the requisition's **Date** field | Drag the **Dates/Today** template<br>Define a flow from **Today/<Today>** to **Requisition/Date** | **Submit** button |
| Set a default value (**New**) to the requisition's Status field | Drag the **Constants/Text** template<br>Name it ***New***<br>Define a flow from "**New**" to **Requisition/Status** | **Submit** button |

**Self Assessment Questions**

1. ________________template is used to store data in the database.
2. ___________________window is used to close the window after the updation of data in the database.
3. ________________________ execution of elements helps to avoid the accident selection of close window before the database updation.
4. To achieve the conditional flow we need to add________________to the window.

## 13.3 Action model to define a process

The Tersus Modeling Language is used to fully define an application's functionality. It is rich enough and precise enough for the Tersus Server, acting as a runtime engine, to execute each application's model. This requires clear syntax (defining what can be modeled by the language) and clear semantics (defining what models mean).

### Syntax

For the modeler: The language's syntax defines the expressive power of the language. That is, what models can be created using Tersus Studio.

For the Tersus Platform: The syntax defines the validity rules that govern the models when created in the studio (and later as they are executed by the server).

### Semantics

For the modeler: The language's semantics define the meaning of each model. That is, what the application should do based on the way it has been modeled.

For the Tersus Platform: Since processes are executed by the server, the semantics of models is tightly related to the way the server works as a runtime engine. On one hand, the server should execute models exactly according to the semantics defined for the language (What You Define Is What It Does). On the other hand, the exact behavior of the server illustrates (and in a sense defines) the exact semantics of the language (What It Does is What You Have Defined).

### Basic Components

The model of a typical application consists of Systems, Displays, Processes, Data Structures and Data Items:

System is a high level module either the full application or a major part of it. The root of the model hierarchy is a system, which may contain sub-systems. Systems appear in Tersus Studio as yellow rectangles. A system typically contains multiple displays and/or multiple processes.

Display is a component that appears on the end user's screen (a GUI component). Typical displays are View, Button, Popup, Table, Label, etc. The model of a highly interactive application will contain many displays, while the model of a back-office server-side system will contain no displays or very few displays. Displays appear in Tersus Studio as blue rectangles (with various icons representing their types). Displays often contain lower level displays and/or processes (e.g. a Button contains the process that should take place when the button is pressed).

Process is a unit of activity. Currently all processes are Actions (synchronous processes) each is a set of activities carried out sequentially without interrupt (e.g. some kind of calculation). In the future, the Tersus Platform will also support Operations (asynchronous processes) processes that may receive inputs after starting. Processes appear in Tersus Studio as blue rectangles (some of them with various icons representing their types). Processes often contain lower level processes and/or data elements.

Data Structure and Data Item contain information used by the application. Data items are atomic building blocks (Text, Number, Date, etc.), while a data structure is composed of lower level data structures and/or data items. Data structures and data items appear in Tersus Studio as grey rectangles (usually with various icons representing their types).

**Slots and Flows**

Tersus implement Dataflow Programming. Processes (and in certain cases also systems and displays) are able to receive and send out data through Slots. The flow of data between processes, as well as the sequencing of processes, is governed by Flows.

Slot is either a Trigger (an entry point through which a process receives input data when its starts executing), or an Exit (an outlet through which a process exposes data it generates while executing). A trigger appears as a green, inward-pointing triangle on the frame of a process. An exit appears as a gray, outward-pointing triangle on the frame of a process.

Flow defines the order of execution (when should a process be executed) and/or data exchange (when and how should data be passed). A flow appears as an arrow between two model elements (Source and Target), each a slot or a data element. Each flow can define: (a) order of execution (the process containing its target slot may not start before the flow); (b) data flow from its source to its target; or (c) both order of execution and data flow.

**Self Assessment Questions**

5. ________________ and ___________________ are require for action model to define process.
6. Write any two components of model application.
7. Slot will act either as a trigger or as a exit process. State [True/False].

## 13.4 Table Display

So now we can proceed with the concept of how to display the data in the tabular format. Just recollect what we discussed in the previous unit to design the data entry model and to store the data in the database. Now we will see how to view the data that are previously entered in form of table here we are going to fix each record in a row. The table comes under the part of Open Requisition that we discussed previously. This can be done by Zooming to the Open Requisition by double clicking it on the editor or locate on the editor and then double click on it.

You can see the table will appear below the New Requisition button so need to make provision for new room where we will create the Open Requisition model. This can be achieved by

> Resize the New Requisition button and position it near the top of the Open Requisitions view.

### Create a Table Display

Now we can add the Table Display to the Open Requisition.

- Select the Display/Simple Table template (▦) from the palette, and insert it into Open Requisitions.
- Name it Requisition List.

By default the simple Table template consists of <Selected Row> about this we will discuss in the later stage. Figure 13.4 depicts the view model after the insertion of new requisition and the Requisition List.

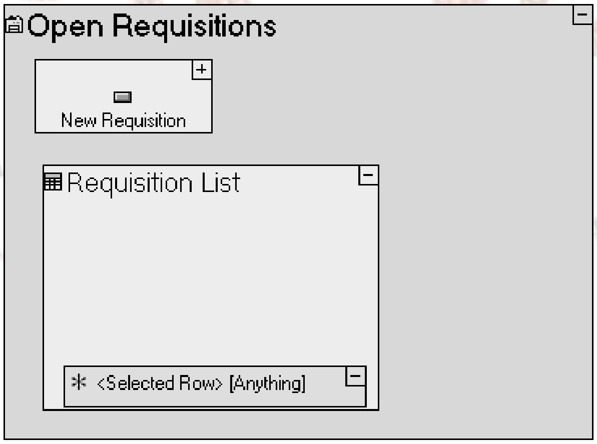


**Fig. 13.4: Open Requisition with list control**

### 13.4.1 Reuse data structure to define Table content

Now we will use the same model table which we used for the previous example (referred in figure 13.3). So now we are going to discuss the concept of reusability that is utilize the existing data structure. Reusing means using the models that are previously defined may be a part of modeling process ultimately improves the development time and clarity on the design. Whenever you use the existing model, the changes made in one place that impacts all the occurrences and shows the consistency of the model. Reusability is allowed in display, data and process models.

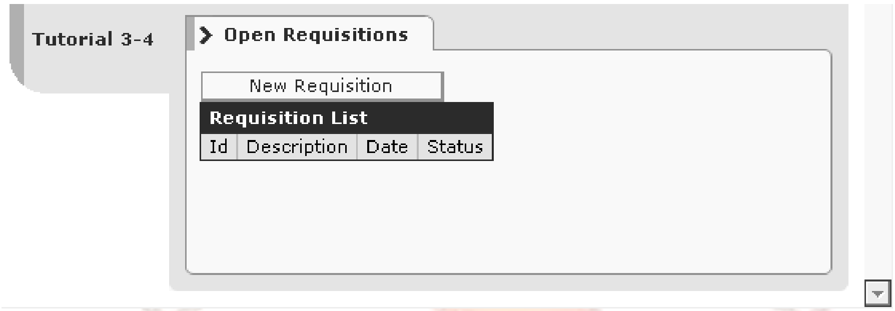
Now we will see how to get the existing data structure to the model, go to the repository view where you can see the Requisition data structure or you can find in outline view. From there Zoom in the Requisition List Table and drag them in to the model editor. And the dragged item can be dropped inside the Requisition List element. Now the model will appear as shown in figure 13.6. As we know the existing table has the multiple rows that means more than one record. In order to repeat the same data structure for more times we also needs to define the data structure in a repetitive way to do this execute the following action:

- Right-click on the Requisition element we have just created, and select repetitive from the menu.

You can verify that the Requisition element has been defined as repetitive by:

1. Right-clicking on it again, and verifying that repetitive has a check mark next to it.
2. Checking the repetitive property in the Properties pane.
3. Verifying that the Requisition element in the model editor, appears stacked.

Now you can save your work and execute the application in the browser it should appear as shown in figure 13.5.



Requisition List

| Id | Description | Date | Status |
|---|---|---|---|

**Fig. 13.5: Model with Requisition list**

In figure 13.6 you can observe that the Requisition table list is empty only the title of the columns is appearing. No data is been listed as rows. It shows that we need to do one more step to populate the data in the Requisition list by retrieving the data from the database or table.

### 13.4.2 Action models for populating process

Through this action we are going to populate the display table with data retrieved from the data base by doing the following step

- Select the Basic/Action template (▣), and drag it into the Open Requisitions model.
- Name it Populate Open Requisitions List.

This Action template helps in defining container with complex processes. So for Populate Open Requisitions List does not provide with the trigger it will be forced to execute automatically through its parent Open Requisitions when it is executed. You can see Populate Open Requisitions is not just a display model it is a process model. Because it is not only defined to display the content but also to do some process takes place to display it properly. We can call this process as initialization process because it will not be triggered but it will be executed automatically. This Open Requisition List will populate the Requisition List data element then it will be sent for display. The Requisition List data element will be updated to include the Open Requisition List, and the change will be communicated to the user interface.

### 13.4.3 Define sub process

Here this sub process is defined to generate the table data element in the memory.

- Zoom into Populate Open Requisitions List.
- Select the Basic/Actiontemplate (▣), and drag it into the Populate Open

Requisitions List model.

- Name it Generate Requisition List

Now we are going to use the Find template in order to retrieve data from table. Generate Requisition List, need to be done by starting the retrieving action of data from the table. This is possible through Find template.

- Select the Database/Findtemplate (🔍), and insert it into Generate Requisition List.

You can see the action model of Populate Open Requisite List in figure 13.6.

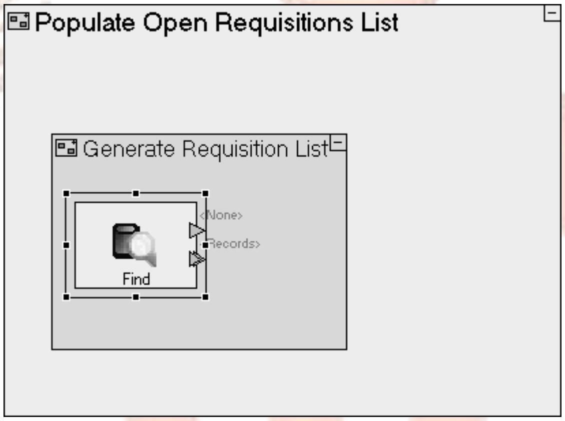


**Fig. 13.6: Populate Open Requisite List**

The Find template includes two exits, <None> &<Records>, which are activated and expose output depending on whether Find finds any records or not. If no record is found, the <None> exit is activated. If at least one record is found, the records are exposed through the<Records> exit.

If you look closely at the <Records> exit, you'll notice that it is marked as repetitive (▷), meaning that Find can output multiple records. We still need to define the database table from which Find Requisitions should retrieve records. This is defined through defining the Data Type of its <Records> exit the data structure that the exit outputs.

**Define data type to create table element**

The records that are retrieved through the Find needs to be stored in the proper data structure. In this case it needs to be the Requisition list that we have created already.

- Drag the Requisition List model from the outline, and drop it next to the Find Requisitionselement

This leads to creation of Requisition List that represents the data element. Next the add Flow will create the Requisition List, from the recorded received through the Find requisition.

- Expand the Requisition List element (by clicking the ⊞ sign at its upper right corner) to display its content.
- Select the Flow tool to link the <Records>trigger of Find to the repetitive Requisition data element in Requisition List.

We have done two things by creating the above flow First is it is explicitly defined that where the Find will go. Secondly implicit declaration of data type for records leads information since it is based on the record template Find will be clear about which database it needs to access.

### 13.4.4 Display data element

Here the mechanism that are used through Ancestor Reference display to receive input from user to display, have to processed in reverse to display the data here.

- Zoom to the Populate Open Requisitions List element.
- Right-click on the Populate Open Requisitions List element, select Add Ancestor Reference from the menu, and select Open Requisitions.

Now we can use the Open Requisitions ancestor reference we have created as the target to which Generate Requisition List will send its output table.

- Double-click on the Open Requisitions data element to zoom into it.
- Expand the Requisition List element (by clicking the ⊞ sign at its upper right corner) to display its content.
- Use the Flowtool to link the Generate Requisition List exit to Open Requisitions/Requisition List.

Figure 13.7 depicts the model window with the Populate Open Requisition List and the Generated Requisition List.

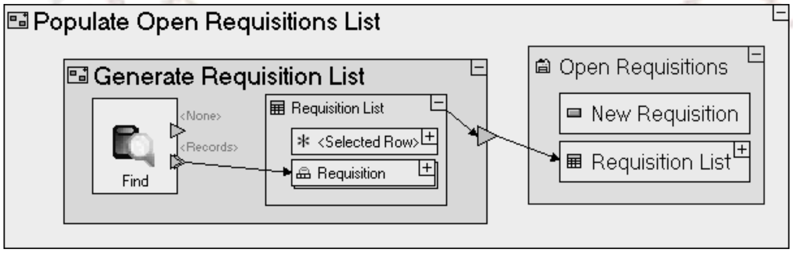


**Fig. 13.7: Populate Open Requisitions List**

If you execute the above application the result window will appear with 4 columns namely Id, Description, Date and Status with the number of rows populated from the data base. Now you can click on the Requisition button and enter the descriptions and click submit button to update in the database. But then these newly entered data or descriptions will not be available. May be user will try to refresh the list by clicking the refresh buttons but it is not allowed here. Now what we can do to refresh the list automatically whenever there is a new entry. Here refreshing we mean to say display the list with the updated data. Here we are going to use the “reuse” concept of using the “Populate Open Requisitions List”. Here the refreshing needs to be done while submitting the request, it needs to be modeled with in the Submit Requisition button. Here for the refreshing process we need to add the new button with action process.

- Select the Basic/Action template (▣), and drag it into Submit Requisition.
- Name it Refresh Requisition List.
- Select the Trigger slot (▷) from the palette and click on the frame of the Refresh Requisition List element.
- Select the Flow tool, and link the <Inserted>slot of Insert to the Trigger slot of Refresh Requisition List.

Adding a trigger to the Refresh requisition list is an alternative method for the controls that executes in order. You can also reuse the populate Open requisites by

- Drag the Populate Open Requisitions List process from the Repository (or Outline) and drop it into Refresh Requisition List.

When you do this process the model window look as shown in the figure 13.8.

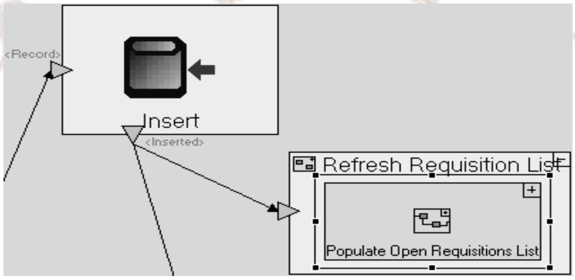


**Fig. 13.8: Model window with Request Requisition List**

Save the application and execute the same in the browser to see the output.

**Self Assessment Questions**

8. Table display used to display the data in the __________________ format.
9. _______________________ display needs to be inserted to display the data in the table format.
10. Find template is used to retrieve data from the data base. State [True/False].
11. Name the action require to display the content with the recent updation.

## 13.5 Summary

- "Insert" template is used to store the data in to the data base.
- Closed window template is used to close the window once the data is been written in to the database.
- Action models require a clear syntax and semantics to understand the allowable models for the language and what each model means.
- The basic model of Tersus platform consists of display, systems, data items, processes and data structure.
- The role of process is to receive and send the data between the slots and the flow takes care of sequence control of data (in which order the data should move).
- Table format display, displays the data from the database in the tabular format (in the row format).
- Requisition list model action template helps in defining container with complex processes.
- None and Records are two exists in Find templates will be activated accordingly whether the Find template find records or not.
- The data display needs to be refreshed to reflect the latest updates from the data base.
- Adding a trigger to the Refresh requisition list is an alternative method for the controls that executes in order

## 13.6 Terminal Questions

1. Explain the role of Insert template in the display.
2. Discuss the term "order of execution" with the example.

3. Explain the action model to define the process.
4. Create a model to display in table format for 4 records.
5. Discuss the concept of reusing of existing data structure for the new model.
6. Brief the role of Find template in the Open Requisite List.

## 13.7 Answers

### Self Assessment Questions

1. Insert
2. Close
3. Order of
4. Trigger
5. Syntax and semantics
6. Display and process.
7. True
8. Tabular.
9. Requisition list.
10. True
11. Refresh.

### Terminal Questions

1. Insert template is used to store data in the data base. How insert template in the model are discussed. For more details refer section 13.2.
2. In order to avoid the accidental errors that the user may click the close window before the insert operation we can fix the order. For more details refer section 13.2.
3. Action models require a clear syntax and semantics to understand the allowable models for the language and what each model means. For more details refer section 13.3.
4. In table format we can fix each record in a row. The table comes under the part of Open Requisition. For more details refer section 13.4 .
5. Reusing means using the models that are previously defined may be a part of modeling process ultimately. For more details refer sub section 13.4.1.

6. Reusing means using the models that are previously defined may be a part of modeling process ultimately. For more details refer section 13.4.3.

**E-Reference:**

- http://www.tersus.com/#Id=42
- http://infocenter.tersus.com/index.jsp?topic=/tersus.help/html/Tutorial/Stage 4/Stage 4.html