Module Objectives

In this module, we will:

- Examine object relationships and hierarchies
- Define projects, jobs, work flows, data flows, and queries
- Examine the query editor and its components
- Validate and execute jobs
- View the logs generated by an executed job
Review: Object Relationships

Repository

Projects

Jobs

Work Flows

Data Flows

R/3 Data Flows

Datastores

Source Databases

Target Databases
Review: Object Relationships

- Project
- Job
- Work Flow
- Data Flow

Diagram showing relationships between objects in ActaWorks for data extraction.
Projects

- Used to group and organize related objects
- Not important to the actual movement of data
- Can contain any number of:
  - Jobs
  - Work flows
  - Data flows
Jobs

- Reusable object
- Next level of organization below a project
- Special work flow that can be executed:
  - Manually, or . . .
  - By scheduling it to run at a specific time(s)
- Contains work flows (optional), and data flows that have the actual data movement instructions
- Can call many work flows
- Should contain all the steps you want executed together
Saving and Deleting Objects

- From the Project menu, click:
  - Save All to save a project and all of its components
  - Save to save individual objects currently open in the workspace

- Projects and jobs can be deleted from the project area; however, their reusable objects are still available in the Object Library

- To permanently delete a reusable object, do it from the Object Library
Product Demo 4-1

In the demo that follows we will:

- Define a new project
- Define and delete a new job
Work Flows

- Reusable object
- Saved in the Object Library
- Executes only in the context of a job
- Prepares for data flow execution
Work Flows ...

- **A work flow does not:**
  - Operate on data
  - Provide more data to a data flow

- **A work flow can:**
  - Call data flows to perform data movement operations
  - Contain another work flow
  - Determine the order in which to execute data flows
  - Pass parameters to and from data flows
  - Define strategies for handling errors that occur during execution
  - Define conditions for executing sections of the project
## Work Flow Components

<table>
<thead>
<tr>
<th>Work Flow Component</th>
<th>Programming Analog</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work Flow</td>
<td>Procedure</td>
</tr>
<tr>
<td>Data Flow</td>
<td>Declarative SQL select statement</td>
</tr>
<tr>
<td>Script</td>
<td>Subset of lines in a procedure</td>
</tr>
<tr>
<td>Conditional</td>
<td>If/then/else logic</td>
</tr>
<tr>
<td>Try</td>
<td>Try block initiator</td>
</tr>
<tr>
<td>Catch</td>
<td>Try block terminator and exception handler</td>
</tr>
</tbody>
</table>
Product Demo 4-2

In the demo that follows we will:

- Define a new work flow
- Reuse an existing work flow
Data Flows

- Defines the basic task that ActaWorks accomplishes, which is to:
  - Identify the source(s) from which to extract data (SAP R/3 or non-SAP)
  - Define the transformations the data should undergo
  - Identify the target table
Data Flows ...

- Data flows are reusable objects
- Work flows and jobs call data flows to perform data movement operations
- From inside a work flow, a data flow can send and receive information through parameters
- From a work flow's perspective, a data flow looks like a table load
Product Demo 4-3

In the demo that follows we will:

- Define a data flow
- Add objects to the data flow
- Define the order of execution
Query Transform

- While mapping source columns to target columns, a query transform allows you to:
  - Select row sets
  - Create joins
  - Group and order by data values
  - Create data set filters
  - Execute functions

- Essentially . . . Implements a SQL Select statement on input row sets
The Query Transform Editor

Source Schema

Target Schema

Column Mapping Definition

Property Sheets

<table>
<thead>
<tr>
<th>Source Schema</th>
<th>Target Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Column Mapping</td>
<td>Definition</td>
</tr>
</tbody>
</table>

Property Sheets

(VBAK.VBELN = VBAK.VBELN) AND
(AUDAT >= '19960101') AND
(AUDAT <= '19961231')
The Table Editor

- Also called "table loader"
- After adding a target table or file to a data flow diagram, click the target name to open the editor
- The Target tab displays information on the selected target
The Table Editor (Target Tab)
The Table Editor (Options Tab)
Table Editor Options

- Batch size - Sets the number of rows sent to the target database all at once
- Column comparison:
  - Compare-by-name - Maps data to columns with matching names
  - Compare-by-position - Respects the order of columns in existing table
- Ignore case - Finds matches regardless of case
- Delete data from table before loading - Deletes all rows from existing table before loading new data set
Table Editor Options ...

- Auto correct load - Deletes duplicate rows before inserting new ones (used in recovery from a "partially loaded table" situation)
- Use input keys - Uses input schema primary keys in target
- Use overflow file - Writes an unloadable row to a predetermined file (recovery feature)
PreLoad/PostLoad Editing

- Specify SQL statements that will be executed in the target datastore before or after table loader execution
Product Demo 4-4

In the demo that follows we will:

- Define a query transform
- Examine various transform options
- Change column properties
- View the table loading options
Pop Quiz

Which table editor option deletes duplicate rows before inserting new ones?
Tip: Close Windows Periodically

• Your workspace is a window
• A tab marks every window you have open
• Close them periodically - they use resource!
• To close all windows:
  View - Close All Windows
Validation

- Validating, or testing, your data movement occurs in two phases:
  - Design time
  - Run time
- The Debug menu provides the following design-time options:
  - Validate
  - Display Language
  - Generate ABAP (SAP-related objects only)
Debug — Validate

This warning message is usually ok
Debug — Display Language

• The ATL may be useful to you in debugging

• Acta Customer Support is likely to ask for it when you call for help in solving problems
Validation ...

- **Design-time validation**
  - Only objects currently open in the workspace (Current View)
  - All objects (All Objects in View)

- **Design tips**
  - Build down the hierarchy and validate individual objects as you go
  - Validate the entire hierarchy before you execute

- **Run-time validation:**
  - Occurs when a job is executed
  - Produces logs that monitor job execution
Executing a Job

- If a job validates properly and you execute (right-click the job name and select Execute), the Execution Properties dialog box appears.
- This dialog box allows you to specify what and how much debugging information ActaWorks generates (for this execution only).

Note: Trace options are defined in module 16.
Setting Job Properties Through Multiple Executions

- To set a job's properties through multiple executions, right-click the job name and select Properties.
As a job executes, ActaWorks produces three log files:

- **Trace**, which itemizes the steps executed in a job
- **Statistics**, which displays details for each step
- **Error**, which displays a list of any errors

![Image of ActaWorks log files interface with Trace, Statistics, and Error highlighted]
Product Demo 4-5

In the demo that follows we will:

- Validate objects and jobs
- View error messages
- Execute jobs
- View trace and statistics logs
Pop Quiz

My job validated fine, but when I execute I find I have an error in the data flow.

How can this be true?
Summary

In this module, we have:

- Examined object relationships and hierarchies
- Defined projects, jobs, work flows, data flows, and queries
- Examined the query editor and its components
- Validated job components and a job
- Executed a job
- Viewed logs
For More Information . . .

- **Objects:** *User's Guide*, chapters 2 and 21, and *Reference* chapter 3
- **Projects and jobs:** *UG*, chapter 10
- **Error handling:** *UG*, chapter 22
Exercise

- Do chapter 3 of the ActaWorks tutorial:
  - Populate the Sales Organization dimension table from a flat file