How to Implement Technical Support Views for CM System 7.0.3

This document outlines the steps necessary for deploying the Technical Support Views on a Rhythmyx 7.0.3 environment.

Files Included in this ZIP Archive

The ZIP archive contains the following files (including this PDF):

1. Rx_7.0.3_How_to_Implement_Technical_Support_Views.pdf
2. CREATE-ts_cxViews_MSSQL2000.sql
3. CREATE-ts_cxViews_MSSQL2005+.sql
4. CREATE-ts_cxViews_ORACLE.sql
5. TechnicalSupportViews.ppkg
6. TechnicalSupportViewsReport.xltx

SQL Views Overview

There are 28 SQL views total (16 Content Explorer Views, 12 Report Views).

Types of SQL Views

There are two types of SQL views:

1. **Content Explorer View**
    i. A Content Explorer View is a SQL view that will be used to load content items in the content explorer using an accompanying Rhythmyx View, which you'll install using the supplied Package. These are the views that will more than likely help resolve many content/relationship related issues.

    NOTE: All Content Explorer Views will be filtered by community when viewed in the Content Explorer. You will have to switch to each and every community to see if these views return results.

2. **Report View**
    i. A Report View is a SQL view that is mainly for the purposes of running against the database periodically to aggregate statistics to see at a high level if the customer has any issues in the database. There are 4 parent Content Explorer SQL views (TS_ORPHANED_DEP_ITEMS, TS_ORPHANED_DEP_PUB_ITEMS, TS_ORPHANED_ITEMS, TS_ORPHANED_PUB_ITEMS) that each have 3 child Report SQL Views (<ParentView>_COM, <ParentView>_CT, <ParentView>_WF).

SQL Views Defined

1. **TS_INVALID_CONTENTTYPEID** *(Content Explorer View – "TS_InvalidContenttypeid")*
    i. Finds all items in the CONTENTSTATUS table, that have an invalid CONTENTTYPEID.
2. **TS_INVALID_WORKFLOWAPPID** *(Content Explorer View – "TS_InvalidWorkflowappid")*
    i. Finds all items in the CONTENTSTATUS table, that have an invalid WORKFLOWAPPID.
3. **TS_MULTIPLE_PARENT_NAVONS** *(Content Explorer View – "TS_MultipleParentNavons")*
    i. Finds all publishable navons with more than one parent navon.
4. **TS_NON_PUB_LANDING_PAGE** *(Content Explorer View – "TS_NonPublishableLandingPage")*
    i. Finds all non-publishable landing pages of publishable navons.
5. **TS_ORPHANED_DEP_ITEMS** *(Content Explorer View – "TS_OrphanedDependentItems")*
    i. Finds all items that are not in a folder and have AA relationship(s) to other items.
    A. **TS_ORPHANED_DEP_ITEMS_COM** *(Report View)*
        i. Counts TS_ORPHANED_DEP_ITEMS grouped by Community.
    B. **TS_ORPHANED_DEP_ITEMS_CT** *(Report View)*
        i. Counts TS_ORPHANED_DEP_ITEMS grouped by Content Type.
    C. **TS_ORPHANED_DEP_ITEMS_WF** *(Report View)*
        i. Counts TS_ORPHANED_DEP_ITEMS grouped by Workflow.
6. **TS_ORPHANED_DEP_NAVONS** *(Content Explorer View – "TS_OrphanedDependentNavons")*
    i. Finds all Navons that are not in a folder and have AA relationship(s) to other items.
7. **TS_ORPHANED_DEP_PUB_ITEMS** *(Content Explorer View – "TS_OrphanedDependentPublishableItems")*

i. Finds all items that are not in a folder, have AA relationship(s) to other items, and are in a publishable state.

A. **TS_ORPHANED_DEP_PUB_ITEMS_COM** *(Report View)*

i. Counts TS_ORPHANED_DEP_PUB_ITEMS grouped by Community.

B. **TS_ORPHANED_DEP_PUB_ITEMS_CT** *(Report View)*

i. Counts TS_ORPHANED_DEP_PUB_ITEMS grouped by Content Type.

C. **TS_ORPHANED_DEP_PUB_ITEMS_WF** *(Report View)*

i. Counts TS_ORPHANED_DEP_PUB_ITEMS grouped by Workflow.

8. **TS_ORPHANED_FOLDERS** *(Content Explorer View – "TS_OrphanedFolders")*
   i. Finds all folders that are not in a folder.
9. **TS_ORPHANED_ITEMS** *(Content Explorer View – "TS_OrphanedItems")*
   i. Finds all items that are not in a folder.
   A. **TS_ORPHANED_ITEMS_COM** *(Report View)*
      i. Counts TS_ORPHANED_ITEMS grouped by Community.
   B. **TS_ORPHANED_ITEMS_CT** *(Report View)*
      i. Counts TS_ORPHANED_ITEMS grouped by Content Type.
   C. **TS_ORPHANED_ITEMS_WF** *(Report View)*
      i. Counts TS_ORPHANED_ITEMS grouped by Workflow.
10. **TS_ORPHANED_NAVONS** *(Content Explorer View – "TS_OrphanedNavons")*
    i. Finds all Navons that are not in a folder.
11. **TS_ORPHANED_PUB_ITEMS** *(Content Explorer View – "TS_OrphanedPublishableItems")*
    i. Finds all items that are not in a folder and are in a publishable state.
    A. **TS_ORPHANED_PUB_ITEMS_COM** *(Report View)*
       i. Counts TS_ORPHANED_PUB_ITEMS grouped by Community.
    B. **TS_ORPHANED_PUB_ITEMS_CT** *(Report View)*
       i. Counts TS_ORPHANED_PUB_ITEMS grouped by Content Type.
    C. **TS_ORPHANED_PUB_ITEMS_WF** *(Report View)*
       i. Counts TS_ORPHANED_PUB_ITEMS grouped by Workflow.
12. **TS_PUB_NAVONS_NO_LANDING_PAGE** *(Content Explorer View – "TS_PublishableNavonsNoLandingPage")*
    i. Finds all publishable navons without a landing page.
13. **TS_RXPUBDOCS_NO_FILENAME** *(Content Explorer View – "TS_RXPUBDOCS_NoFilename")*
    i. Finds all published content items that end in a '/' (e.g. don't have a filename or file extension) for their publocation.
14. **TS_RXPUBDOCS_NO_FILENAME_ORPH** *(Content Explorer View – "TS_RXPUBDOCS_NoFilenameOrphaned")*
    i. Finds all published content items that end in a '/' (e.g. don't have a filename or file extension) for their publocation, and are also in the TS_ORPHANED_ITEMS view result set.
15. **TS_SHAREDBINARY_NO_FILENAME** *(Content Explorer View – "TS_RXPUBDOCS_NoFilename")*
    i. Finds all items in the RXS_CT_SHAREDBINARY table, that have a null ITEM_FILENAME, but ITEM is not null for the item's current public revision.
16. **TS_SHAREDIMAGE_NO_FILENAME** *(Content Explorer View – "TS_SHAREDIMAGE_NoIMG1Filename")*
    i. Finds all items in the RXS_CT_SHAREDIMAGE table, that have a null IMG1_FILENAME, but IMG1 is not null for the item's current public revision.

Installation Instructions

1. Run the supplied SQL for your RDBMS (ORACLE or MS SQL Server 2000 or 2005).
   a. CREATE-ts_cxViews_ORACLE.sql
   b. CREATE-ts_cxViews_MSSQL2000.sql
   c. CREATE-ts_cxViews_MSSQL2005+.sql *(*also works in MSSQL2008)*

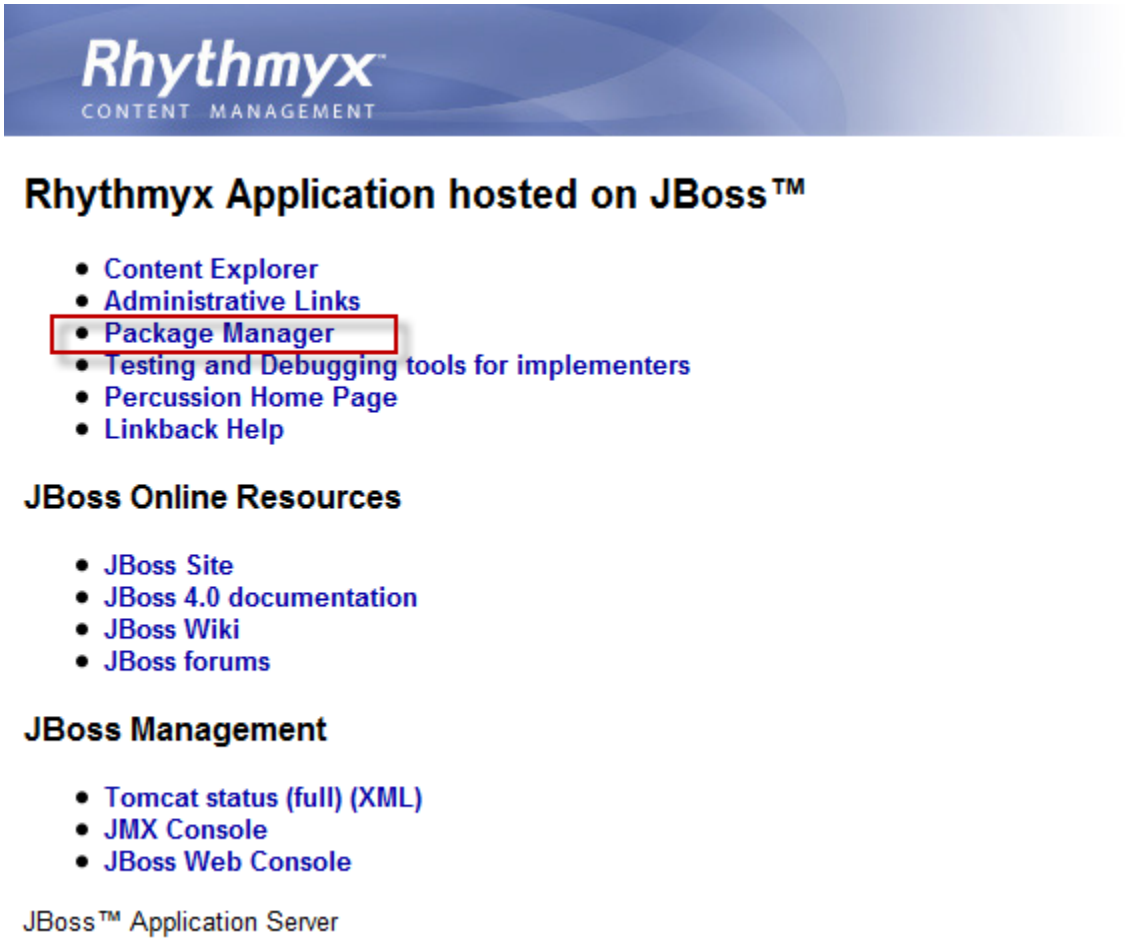
2. Deploy the Technical Support Views Package using the "Percussion Package Installer"

   NOTE: See "Detailed Package Installation Instructions" below for step-by-step screenshots of installing the Technical Support Views Package.

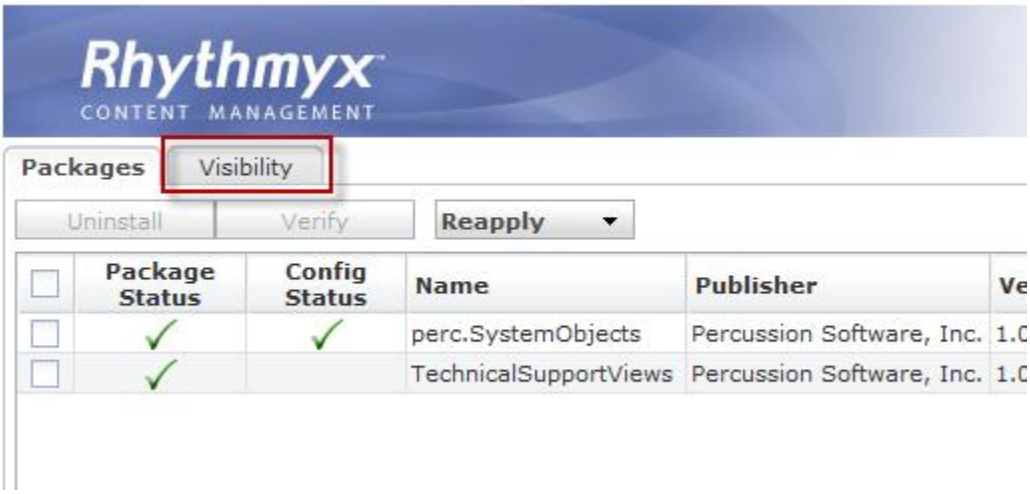
3. Launch a browser and navigate to your Percussion CM Server (e.g. http://localhost:9992/>)

   NOTE: Do not include the word "Rhythmyx" after the port.

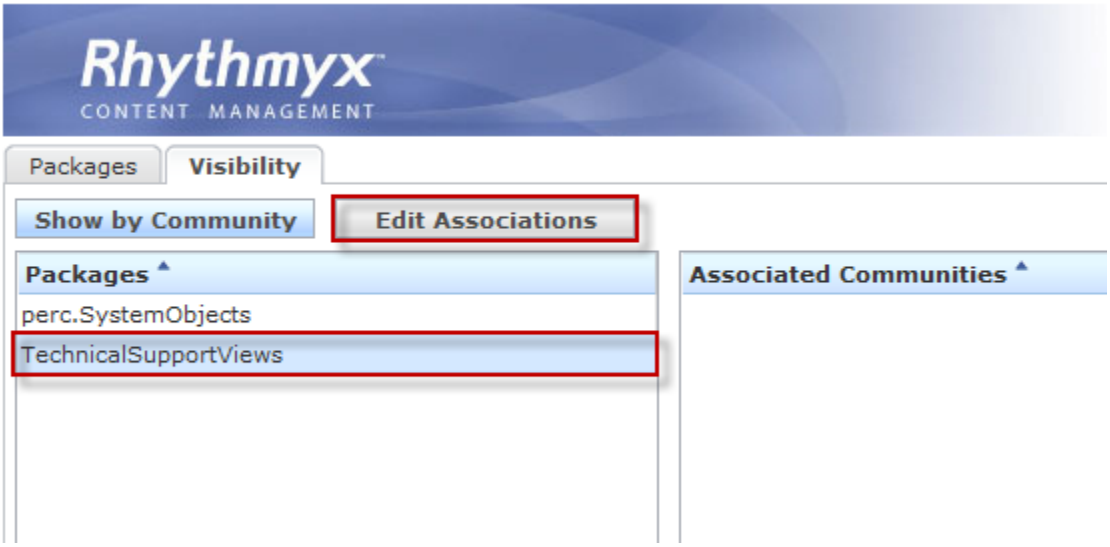
4. You should see a page that looks like the following:



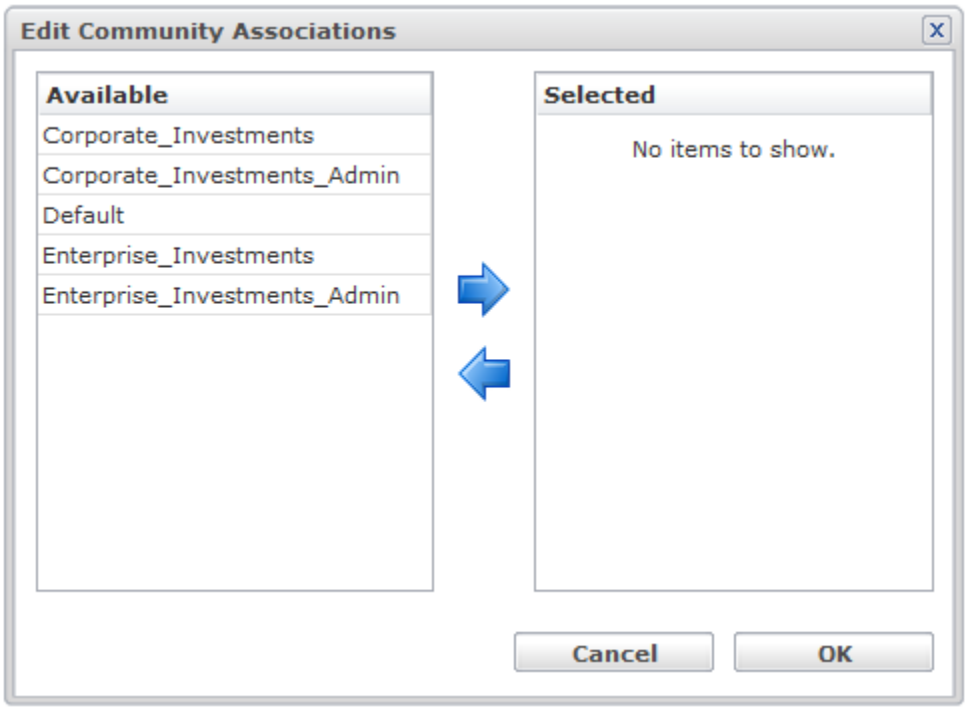
5. Click on the “Package Manager” link.



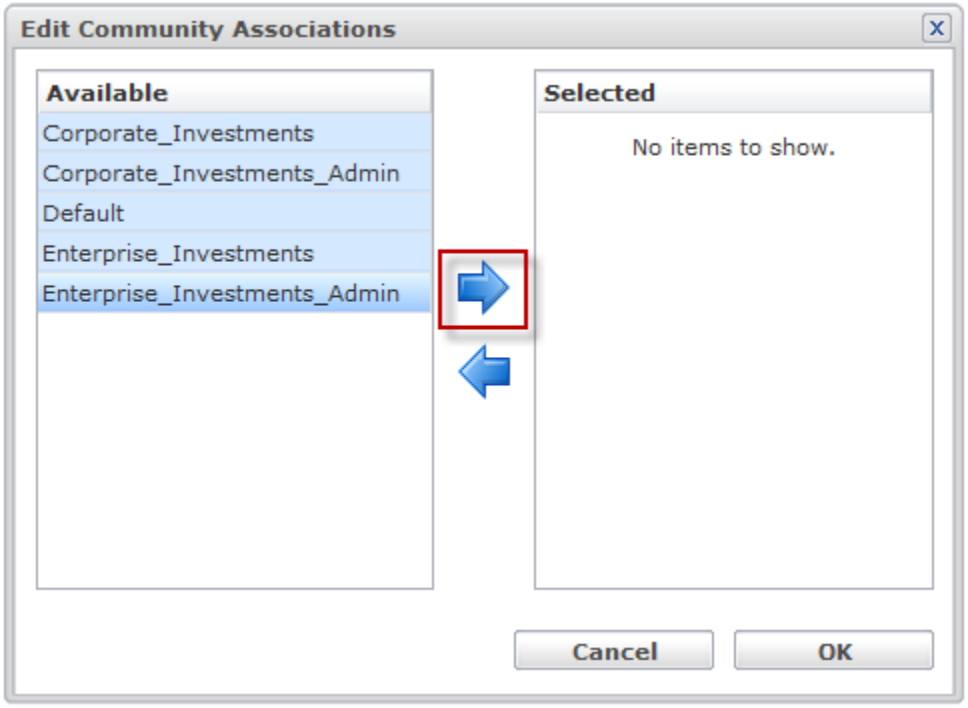
6. Click on the “Visibility” tab.
7. Highlight the “TechnicalSupportViews” package, and then click the “Edit Associations” button:

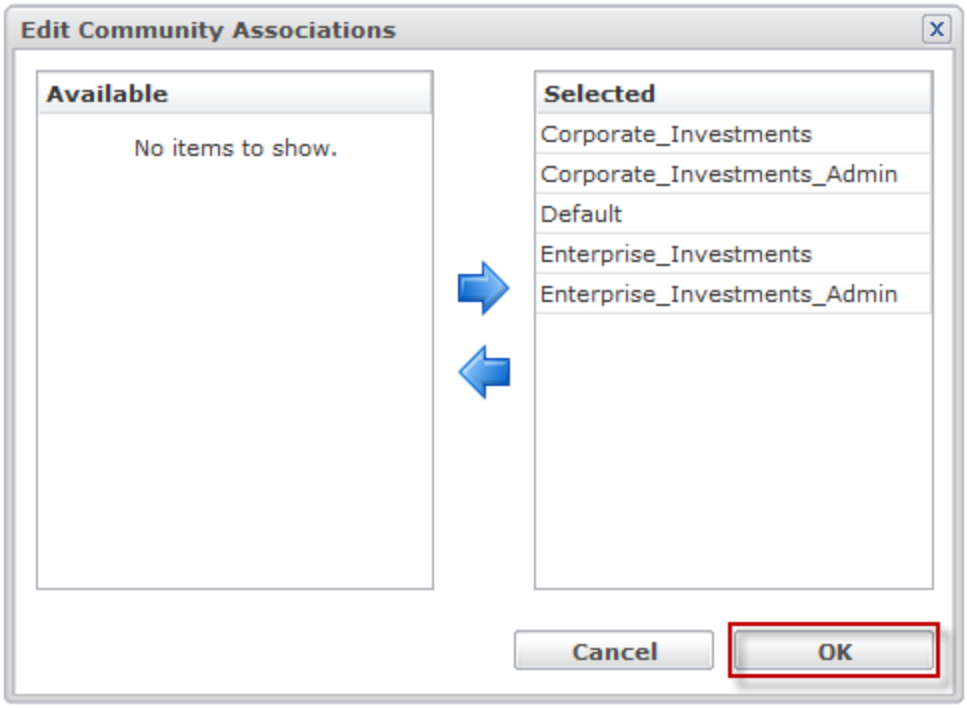


8. A modal window will pop open:



9. Select the Communities that you want the Percussion Technical Support Views to be visible:



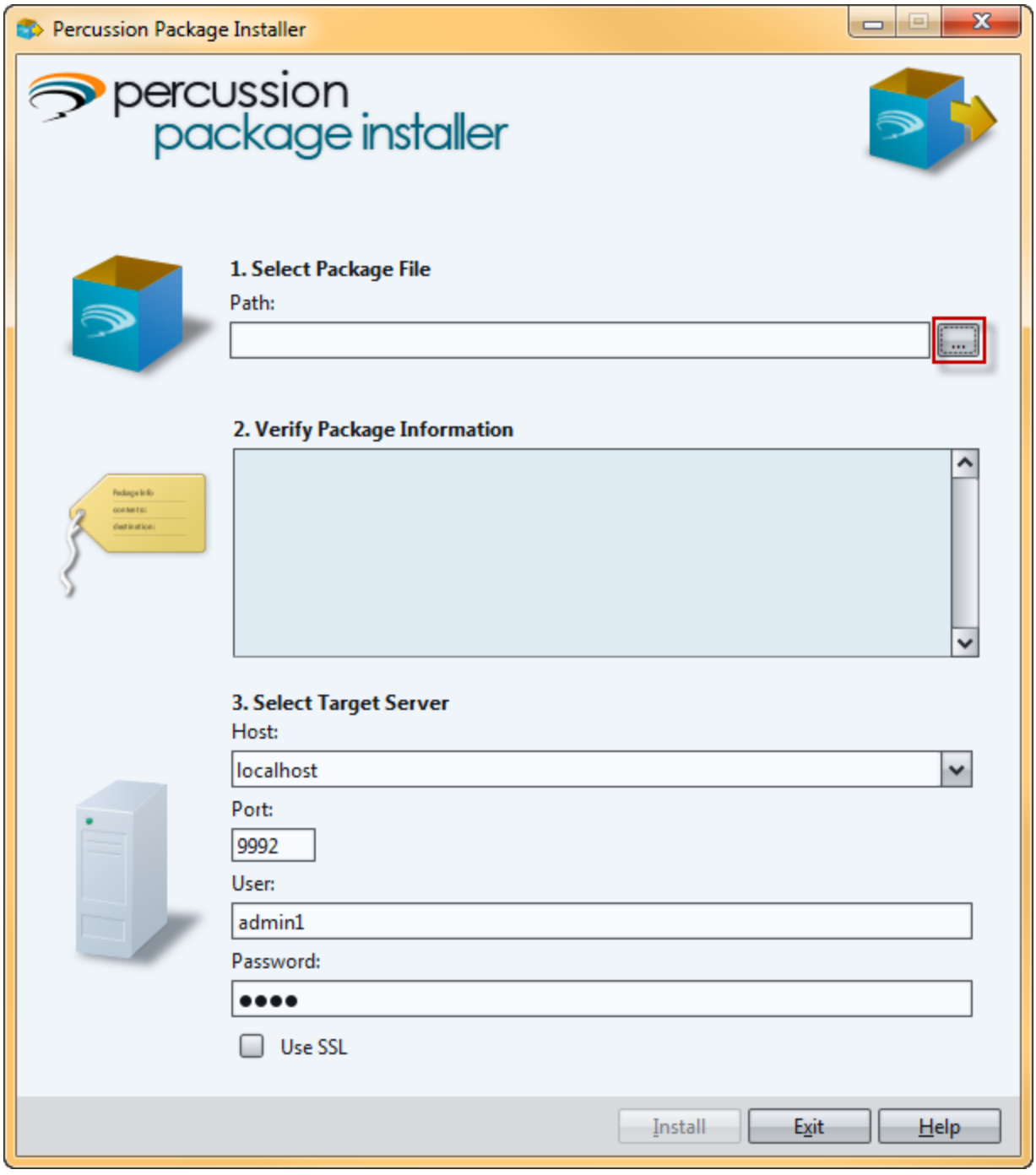


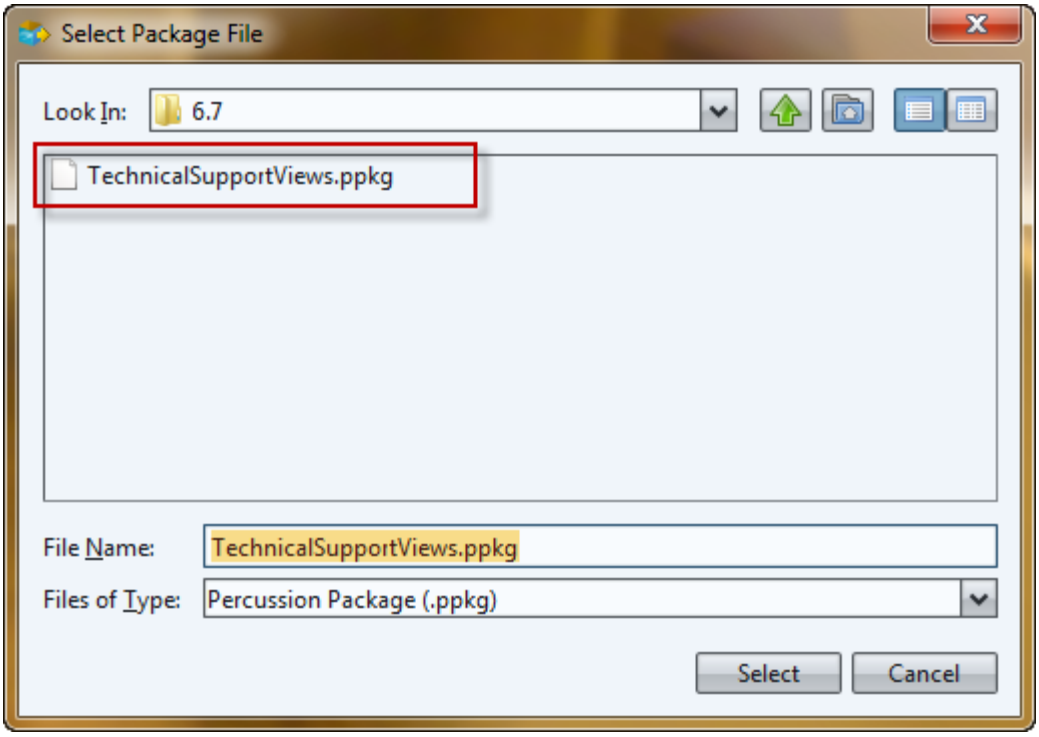
NOTE: You can shift-click all the communities at once.

Detailed Package Installation Instructions

TechnicalSupportViews.ppkg

1. Select the Package File



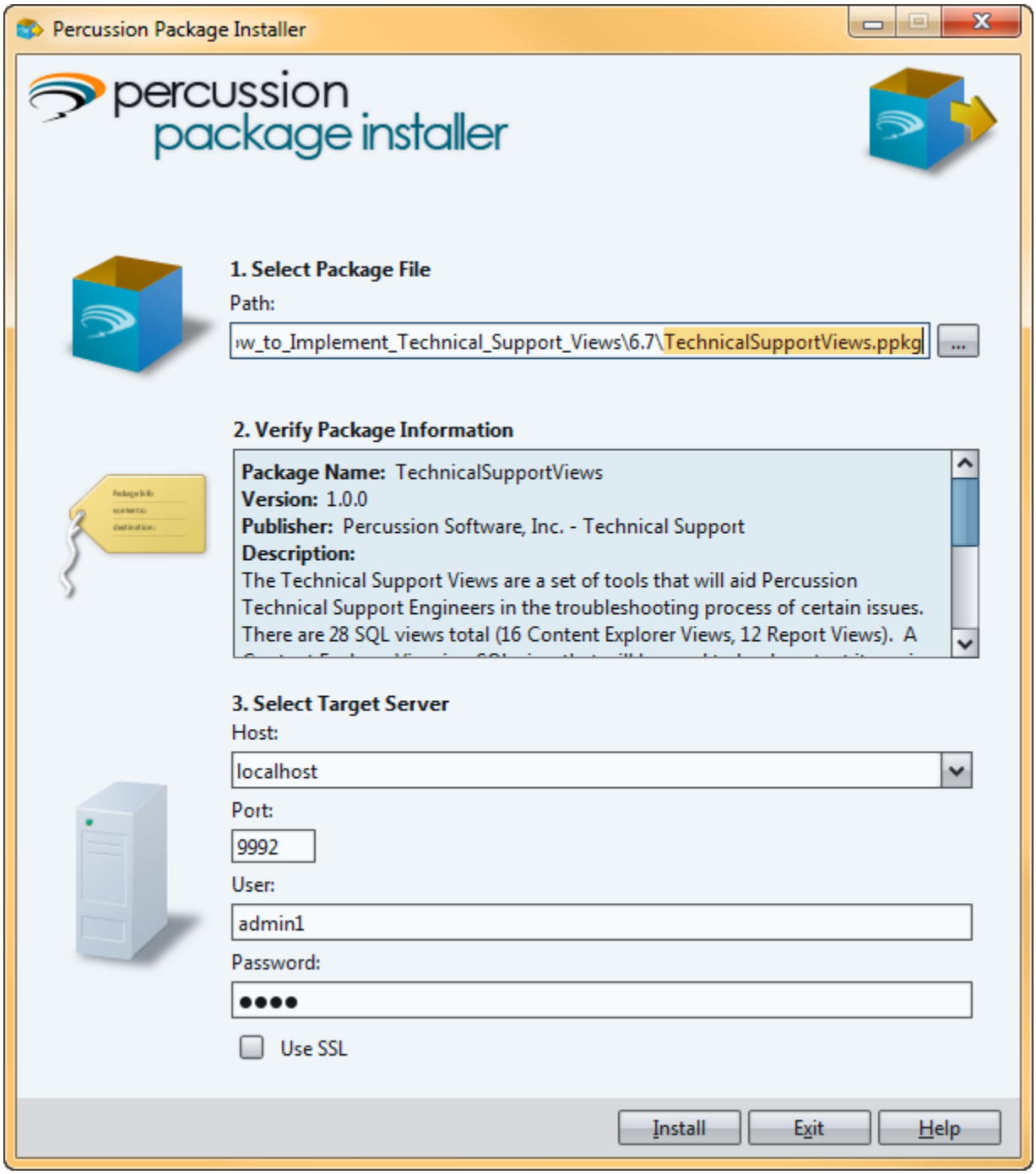


2. Verify the Package Information

Package Name: TechnicalSupportViews
Version: 1.0.0
Publisher: Percussion Software, Inc. - Technical Support
Description:
The Technical Support Views are a set of tools that will aid Percussion Technical Support Engineers in the troubleshooting process of certain issues. There are 28 SQL views total (16 Content Explorer Views, 12 Report Views). A Content Explorer View is a SQL view that will be used to load content items in the content explorer using an accompanying Rhythmyx View. A Report View is a SQL view that is mainly for the purposes of running against the database periodically to aggregate statistics to see at a high level if the customer has any issues in the database.

NOTE: All Content Explorer Views will be filtered by community when viewed in the Content Explorer. You will have to switch to each and every community to see if these views return results.

3. Select your Target Server

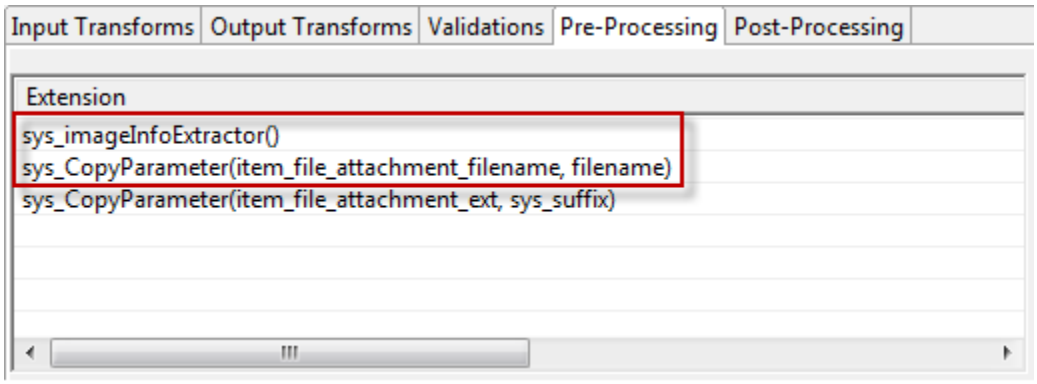


4. Click “Install”.

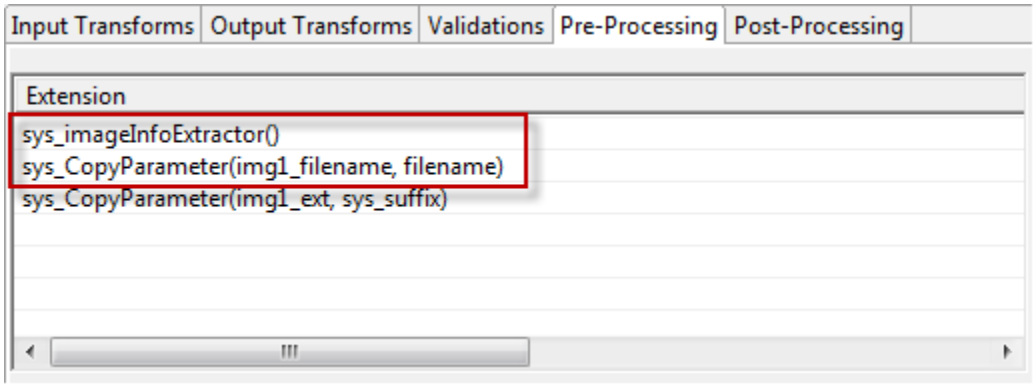
Troubleshooting Techniques

1) **TS_INVALID_CONTENTTYPEID** - These items must be deleted directly in the database. This is very tricky and dangerous. If you don't know exactly what you're doing, ask someone who does.
2) **TS_INVALID_WORKFLOWAPPID** - This is very similar to the above scenario. You can attempt to manually update the CONTENTSTATUS record so that the item has a valid WORKFLOWAPPID, or you can delete the item and any reference to that contentid throughout the entire database, just like the above scenario.
3) **TS_MULTIPLE_PARENT_NAVONS** - You will have to perform an impact analysis on these Navons to see both of its parent navons and determine which one is real, and which one is not. Remove the Navon from the submenu slot of the incorrect parent Navon.
4) **TS_NON_PUB_LANDING_PAGE** - You can simply transition these landing pages back to the public state, or archive the Public Navon depending on the customers preferences.
5) **TS_ORPHANED_DEP_ITEMS** - This is a subset view of TS_ORPHANED_ITEMS. These items may or may not be problematic. If you know that these items should be in a particular folder, you can move them.
6) **TS_ORPHANED_DEP_NAVONS** - This is a subset view of TS_ORPHANED_NAVONS and TS_ORPHANED_DEP_ITEMS. These Navons are ticking timebombs. They should be deleted, period.
7) **TS_ORPHANED_DEP_PUB_ITEMS** - This is a subset view of TS_ORPHANED_PUB_ITEMS and TS_ORPHANED_DEP_ITEMS. These items will more than likely cause publishing errors unless your location scheme doesn't use the $sys.pub_path. These items should either be archived, or deleted. On a PRODUCTION environment, they should be archived first for some period of time. And if after that time they have not caused a problem they can be safely deleted.
8) **TS_ORPHANED_FOLDERS** - This is a subset view of TS_ORPHANED_ITEMS. This view may or may not return the correct results in the content explorer. You should run the SQL version of this view just to be safe. If it returns results in the database, but not in the Content Explorer, you may have to attempt to fix these folders manually.
9) **TS_ORPHANED_ITEMS** - This is the superset view for all orphaned item views. If this view has a lot of items, you should use the other subset orphaned items views to narrow down the scope of the problematic items (e.g. TS_ORPHANED_DEP_ITEMS, TS_ORPHANED_DEP_NAVONS, TS_ORPHANED_DEP_PUB_ITEMS, TS_ORPHANED_FOLDERS, TS_ORPHANED_NAVONS, TS_ORPHANED_PUB_ITEMS).
10) **TS_ORPHANED_NAVONS** - This is a subset view of TS_ORPHANED_ITEMS. These Navons are potentially ticking timebombs. They should be deleted, period.
11) **TS_ORPHANED_PUB_ITEMS** - This is a subset view of TS_ORPHANED_ITEMS. These items will more than likely cause publishing errors unless your location scheme doesn't use the $sys.pub_path. These items should either be archived, or deleted. On a PRODUCTION environment, they should be archived first for some period of time. And if after that time they have not caused a problem they can be safely deleted.
12) **TS_PUB_NAVONS_NO_LANDING_PAGE** - These Navons should either be transitioned to archive, or a proper landing page should be added to the submenu slot.

13) **TS_RXPUBDOCS_NO_FILENAME** - These items successfully published out, however the filename used ended in a slash, which could cause all sorts of publishing issues. These items should be examined thoroughly along with their location scheme to determine what is causing the problem.
14) **TS_RXPUBDOCS_NO_FILENAME_ORPH** - This is a subset view of TS_RXPUBDOCS_NO_FILENAME. These items are orphan items that successfully published out, however the filename used ended in a slash, which could cause all sorts of publishing issues. These items should be examined thoroughly along with their location scheme to determine what is causing the problem.
15) **TS_SHAREDBINARY_NO_FILENAME** - These items are either files or images that do not have an filename for some reason. Probably the sys_imageInfoExtrator() pre-processing Java exit is missing or improperly configured, or the sys_CopyParameter() pre-processing Java exit is missing or improperly configured.



16) **TS_SHAREDIMAGE_NO_FILENAME** - These items are images that do not have an filename for some reason. Probably the sys_imageInfoExtrator() pre-processing Java exit is missing or improperly configured, or the sys_CopyParameter() pre-processing Java exit is missing or improperly configured.

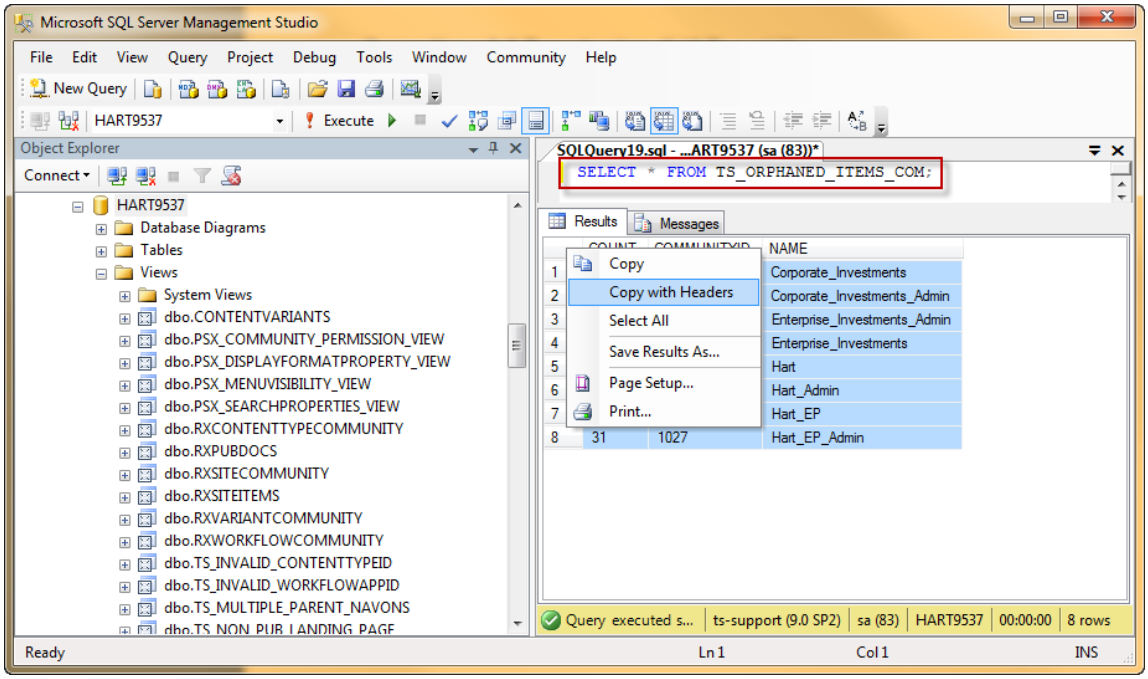


Generating a full Report using all 12 Report Views

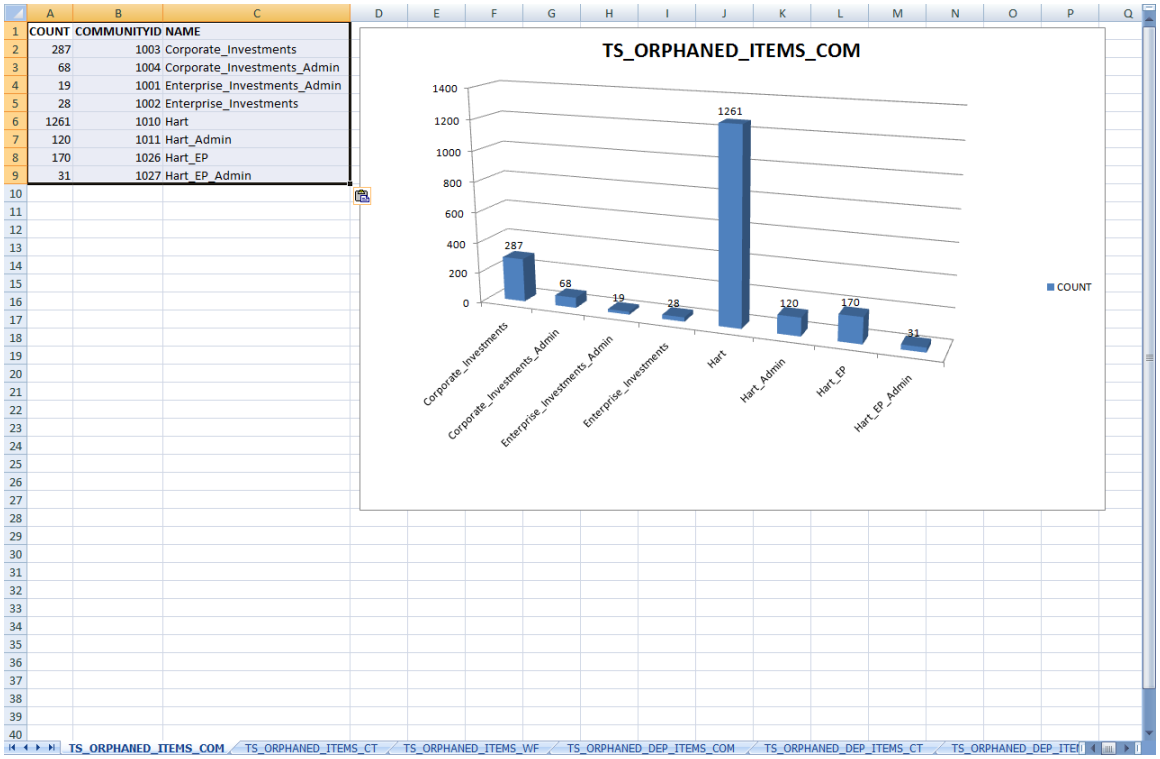
1. Run the following SQL statements to generate result sets for each of the 12 Report Views.

```
SELECT * FROM TS_ORPHANED_ITEMS_COM;
SELECT * FROM TS_ORPHANED_ITEMS_CT;
SELECT * FROM TS_ORPHANED_ITEMS_WF;
SELECT * FROM TS_ORPHANED_DEP_ITEMS_COM;
SELECT * FROM TS_ORPHANED_DEP_ITEMS_CT;
SELECT * FROM TS_ORPHANED_DEP_ITEMS_WF;
SELECT * FROM TS_ORPHANED_PUB_ITEMS_COM;
SELECT * FROM TS_ORPHANED_PUB_ITEMS_CT;
SELECT * FROM TS_ORPHANED_PUB_ITEMS_WF;
SELECT * FROM TS_ORPHANED_DEP_PUB_ITEMS_COM;
SELECT * FROM TS_ORPHANED_DEP_PUB_ITEMS_CT;
SELECT * FROM TS_ORPHANED_DEP_PUB_ITEMS_WF;
```

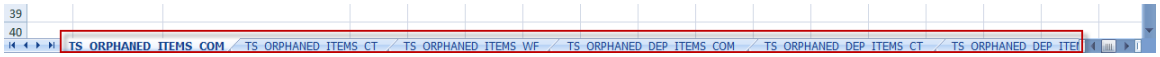
2. Copy/Paste the individual result sets into the supplied “TechnicalSupportViewsReport.xltx”



NOTE: In this example I’m using MS SQL 2005 Management Studio.



NOTE: Each Report View exists as a separate Worksheet tab at the bottom.



3. Adjust the graphs as necessary to match your specific Report View result sets.

Graphical Representation of Orphaned Items Views