PeopleSoft Query Overview

PeopleSoft Query (Query) is an end-user reporting tool that allows users to extract information from SFS easily and efficiently. Queries can be as simple or as complex as needed; and can be one-time queries or saved queries used repeatedly. This manual walks through the steps of creating a query using the web query tool, and also discusses some advanced query topics that aid in creating powerful queries.

To use Query efficiently, you need a basic understanding of the data (that is, in what record is the data stored) the query will be based upon, and relationships to other records within the database. It is fairly simple to create a query, but it takes time and understanding to create a query and bring back accurate data.

Query has several different ways to do the same task, this manual shows one way. You will likely find different ways to do many items described in this manual. Use the method that works best for you.

Before getting started with Query, remember the following:

- **Use a single mouse click** when working with Query. There is no need to double click on fields or tabs.
- **Do not use the back button** on the browser when navigating through SFS.
- If the system is processing, **there is not a way for you to stop the processing**. Pressing the stop button on the browser or clicking on another tab will likely cause the system to not respond. You will lose any unsaved data.
- PeopleSoft uses the terms “record” and “table” interchangeably. In this manual, the terms will mean the same thing, which is a two-dimensional arrangement of rows and columns that holds data.
• Queries can be either ‘Public’ or ‘Private’. Public queries are available for any user to review, run or alter. Private queries can only be accessed by the owner.

<table>
<thead>
<tr>
<th>Process Frequency</th>
<th>Used on an ad-hoc basis.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dependencies</td>
<td>Some knowledge of SFS records and data.</td>
</tr>
<tr>
<td>Assumptions</td>
<td>N/A</td>
</tr>
<tr>
<td>Responsible Parties</td>
<td>N/A</td>
</tr>
<tr>
<td>Alternate Scenarios</td>
<td>N/A</td>
</tr>
</tbody>
</table>

### Process Details - Joins

**Navigation:** Reporting Tools > Query > Query Manager

Query Manager

Enter any information you have and click Search. Leave fields blank for a list of all values.

Find an Existing Query | Create New Query

*Search By* Query Name begins with

Search Advanced Search

I. Creating a Query with Multiple Records

Query Manager allows you to build queries with multiple-record joins. Joins retrieve data from more than one record. PeopleSoft Query links the records, based on common fields, and links the rows on the two records by common fields in shared columns.

Joined records must each include **at least one** field in both records that contain similar data. For example, if you want to join Vendor record and Voucher record, they both must contain a common field, such as a Vendor ID field, or a “High level” key field. (symbolized using a Key) to serve as a field on which the data can be joined. Records can be joined on multiple (similar) fields so long as the fields have the ability to match information. Typically (but not always) records are joined on key fields. Key fields are essentially the field(s) that make the record unique, and are shown in Query with a symbol next to them. If you have a query with multiple records, and do not add joins or create incorrect joins, the results will return wrong information may result in a cartesian
join and the query could get stuck in a loop and never finish processing.....this is bad and the System Administrators will not like you.

Using joins, you define relationships among fields when you query the records. The procedure for joining records differs depending on how the records being joined are related to each other. Query Manager uses three types of joins: record hierarchy (parent-child relationships), related record (predefined relationships), and any record (relationships the user defines).

II. Creating Hierarchy Joins

A hierarchy join is similar to a parent/child relationship. An example of this is the JRNL_HEADER and JRNL_LN. This type of join is the easiest to create in Query and PeopleSoft completes the join automatically.

1. To create a Hierarchy Join, select the base record for your query (parent)
2. Click the Hierarchy Join link.
3. Records that have a parent/child relationship will appear.
4. **Select** the child record to join to the parent record....**select** JRNL_LN record. The join is reflected on . This action automatically creates the necessary joins needed between the two records.

<table>
<thead>
<tr>
<th>Chosen Records</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Alias</strong></td>
</tr>
<tr>
<td>A</td>
</tr>
<tr>
<td>B</td>
</tr>
</tbody>
</table>

5. You can **select** fields from either record to be included in the results, the same as a query with only one record.

6. **Click** next to the record name to minimize the record, OR

7. **Click** to view the fields from that record.

8. Once you have selected fields, click **“View SQL”** to see the automatically created join.

III. **Creating Related Record Joins**

In a Related Record join, you can automatically join two records based on a relationship that is predefined in the record design.

1. In the example below, the LEDGER record is used in the query. Notice there are several hyperlinks on the right hand side that say “Join...” **Clicking** on one of these hyperlinks will join the record automatically to the record you choose. Click on the link for Account.....
2. When you click on the link to add the record, you will be taken to the “Select join type” screen.

Select join type

Join Type

- Join to filter and get additional fields (Standard Join)
- Join to get additional fields only (Left outer join)

OK  Cancel

3. **Choose a join type**

4. **Click OK**. (For information on Left Outer Joins, see Section V.)

5. Once you have joined the appropriate record(s), you can add fields to your query or view the fields from multiple records in your query.

6. **Navigate** between the records the same way as the Hierarchy Joins above, by clicking the next to the record name on.
IV. Creating Any Record Joins

Query Manager allows the creation of a join between any two records. Query will attempt to automatically join the records based on the keys on each record, which may or may not be what you want. It is very important to understand the record structure to ensure the links created are appropriate. Additional joins may need to be added that Query does not create. Planning the query becomes more important as query complexity increases.

In this example, two AP records are used in the query - VENDOR_LOC and VENDOR_PAY.

1. Click Records
2. Enter the Record Name ‘VENDOR_LOC’ in the begins with field
3. Click Search

Search Results

<table>
<thead>
<tr>
<th>Record</th>
<th>Personalize</th>
<th>Find</th>
<th>View 20</th>
<th></th>
<th>First</th>
<th>1 of 1</th>
<th>Last</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recname</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VENDOR_LOC - Supplier Location</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Join Record | Show Fields
| Join Record | Show Fields
4. **Click** the *Add Record* hyperlink for the *Recname VENDOR_LOC*.

5. You will see the *VENDOR_LOC* record added to the **Query**.

6. **To select the second record, click** the **Records** tab.

7. **Enter** the *Record Name ‘VENDOR_PAY’* in the *begins with* field.

8. **Click** the **Search** button.

9. **Click** the *Join Record* hyperlink for the *Recname VENDOR_PAY*.

Select join type and then record to join with *VENDOR_PAY - Supplier Payment Defaults*.

He
10. **Select** a Join Type
   - A Standard Join will return data only if data is present that matches your criteria in, **both** records.
   - A Left Outer Join will return all data from the first record and additional data where the criteria are met on the second record. It will not filter the results to where the conditions are met on both records. (For more information on Left Outer Joins, see [Section V](#)).

In this example, we will use a Standard Join, because we want data from the records where data is present in both records.

11. **Click** on the appropriate hyperlink in the *Join Record* section to add the second record to the query. In this example, there is only one additional record, VENDOR_LOC. PeopleSoft will create criteria based on the similar keys between each record.

![Auto Join Criteria](image)

12. **Click** [Add Criteria](#) to add these criteria to the query.

![Chosen Records](image)

13. You will see the VENDOR_PAY record added to the query.
14. If additional criteria need to be entered, navigate to Criteria and add appropriate criteria, with the Expression 1 Type and Expression 2 Type for both being ‘Field’. This would look like the following:

TIP: If you are unsure of the record structures and how the records are related, run the query with only one record at a time and review the information to determine proper relationships.

V. Left Outer Joins

Query Manager enables you to easily create a Left Outer Join. The query will then bring back all rows of the first (left) record in the result set, even if there are no matches in the joining record.

1. Select the Left Outer Join option for the Type of Join
Select join type and then record to join with VENDOR_PAY - Supplier Payment Defaults.

Join Type

- Join to filter and get additional fields (Standard Join)
- Join to get additional fields only (Left outer join)

Join Record

When you select Left Outer Join as your Join Type, join to the most recently added record in the query, or you will receive an error message. If you are creating a query with more than 2 records, be sure to add the “left” record just before the “right” record.

Here is an example, following are two sample records and how the different joins would return data:

- **Standard Join** - Rows are returned only where there is data in both records. (Result set contains 6 rows)

<table>
<thead>
<tr>
<th>Cust #</th>
<th>Customer First</th>
<th>Customer Last</th>
<th>Email</th>
<th>Order ID</th>
<th>Item</th>
<th>Item#</th>
<th>Quantity</th>
<th>Shipped</th>
<th>Total Order Amt</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Frank</td>
<td>Smith</td>
<td><a href="mailto:frankie@email.com">frankie@email.com</a></td>
<td>1 Sponge</td>
<td>29990574</td>
<td>4</td>
<td></td>
<td></td>
<td>$25.25</td>
</tr>
<tr>
<td>3</td>
<td>Jennifer</td>
<td>Johnson</td>
<td><a href="mailto:jjgirl@email.com">jjgirl@email.com</a></td>
<td>6 Napkins</td>
<td>20777744</td>
<td>100</td>
<td></td>
<td></td>
<td>$4.55</td>
</tr>
<tr>
<td>4</td>
<td>Robert</td>
<td>Green</td>
<td><a href="mailto:begreen@email.mail">begreen@email.mail</a></td>
<td>4 Paper Plates</td>
<td>29179246</td>
<td>100</td>
<td></td>
<td></td>
<td>$11.56</td>
</tr>
<tr>
<td>6</td>
<td>Timothy</td>
<td>Dominick</td>
<td><a href="mailto:tdominick@emailaddress.com">tdominick@emailaddress.com</a></td>
<td>3 Dish Towel</td>
<td>20742749</td>
<td>100</td>
<td></td>
<td></td>
<td>$4.75</td>
</tr>
<tr>
<td>6</td>
<td>Timothy</td>
<td>Dominick</td>
<td><a href="mailto:tdominick@emailaddress.com">tdominick@emailaddress.com</a></td>
<td>3 Dish Towel</td>
<td>20742749</td>
<td>100</td>
<td></td>
<td></td>
<td>$4.75</td>
</tr>
</tbody>
</table>

- **Left Outer Join** - All rows are returned from the first (left) record, and any data from the second (right) record is also presented. (Result set contains 8 rows)

<table>
<thead>
<tr>
<th>Cust #</th>
<th>Customer First</th>
<th>Customer Last</th>
<th>Email</th>
<th>Order ID</th>
<th>Item</th>
<th>Item#</th>
<th>Quantity</th>
<th>Shipped</th>
<th>Total Order Amt</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Penny</td>
<td>Nickels</td>
<td><a href="mailto:pnickels@email.com">pnickels@email.com</a></td>
<td>1 Sponge</td>
<td>29990574</td>
<td>4</td>
<td></td>
<td></td>
<td>$25.25</td>
</tr>
<tr>
<td>2</td>
<td>Frank</td>
<td>Smith</td>
<td><a href="mailto:frankie@email.com">frankie@email.com</a></td>
<td>1 Sponge</td>
<td>29990574</td>
<td>4</td>
<td></td>
<td></td>
<td>$25.25</td>
</tr>
<tr>
<td>3</td>
<td>Jennifer</td>
<td>Johnson</td>
<td><a href="mailto:jjgirl@email.com">jjgirl@email.com</a></td>
<td>4 Paper Plates</td>
<td>29179246</td>
<td>25</td>
<td></td>
<td></td>
<td>$11.50</td>
</tr>
<tr>
<td>4</td>
<td>Robert</td>
<td>Green</td>
<td><a href="mailto:begreen@email.mail">begreen@email.mail</a></td>
<td>6 Napkins</td>
<td>20777744</td>
<td>100</td>
<td></td>
<td></td>
<td>$4.55</td>
</tr>
<tr>
<td>5</td>
<td>Suzy</td>
<td>Worthington</td>
<td><a href="mailto:suzyro@emaillady.com">suzyro@emaillady.com</a></td>
<td>2 Scraper</td>
<td>29779224</td>
<td>1</td>
<td></td>
<td></td>
<td>$5.22</td>
</tr>
<tr>
<td>6</td>
<td>Timothy</td>
<td>Dominick</td>
<td><a href="mailto:tdominick@emailaddress.com">tdominick@emailaddress.com</a></td>
<td>2 Scraper</td>
<td>29779224</td>
<td>1</td>
<td></td>
<td></td>
<td>$5.22</td>
</tr>
<tr>
<td>6</td>
<td>Timothy</td>
<td>Dominick</td>
<td><a href="mailto:tdominick@emailaddress.com">tdominick@emailaddress.com</a></td>
<td>3 Dish Towel</td>
<td>20742749</td>
<td>2</td>
<td></td>
<td></td>
<td>$4.75</td>
</tr>
</tbody>
</table>

**VI. Using Having Criteria**

SQL does not support the use of aggregate functions in WHERE clauses. When an aggregate function is applied to a field (i.e. sum, count, etc.) you cannot add a typical limit to it. To limit rows based on the results of an aggregate function, Query Manager creates HAVING criteria.
An example of using a HAVING criteria would be to create a list of departments that have greater than $5,000 posted to a particular account during a fiscal year.

When you click the Add Criteria icon from the tab for an aggregate field, new criteria is added to tab instead of tab. The selection criteria using the tab in the same way you add selection criteria using tab.

1. To add Having criteria, go to tab.

2. On the field you would like to aggregate, click Edit.
### Aggregate Actions

<table>
<thead>
<tr>
<th>Aggregate</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>No action</td>
</tr>
<tr>
<td>Sum</td>
<td>Numeric total of values</td>
</tr>
<tr>
<td>Count</td>
<td>Number of rows returned in criteria</td>
</tr>
<tr>
<td>Min</td>
<td>Smallest numeric value</td>
</tr>
<tr>
<td>Max</td>
<td>Largest numeric value</td>
</tr>
<tr>
<td>Average</td>
<td>Mean/Average numeric value</td>
</tr>
</tbody>
</table>

3. **Choose** an Aggregate function to perform on your field.

4. **Click** `OK`.

5. You will now see the aggregate function appear on the tab in the **Agg** column.

6. **Click** `Having`.

7. **Click** `Add Having Criteria`.
8. In the Expression 1 box, use the icon to select the field for *Record Alias.Fieldname*.

9. Any fields that have aggregate functions will be available to choose from. Click the hyperlink to select your field.
10. **Select** the appropriate *Condition Type* from the dropdown menu.

11. In the *Expression 2* box, enter your *Constant* in the Define Constant section in the same way you would define any other criteria. (In this instance, we are looking for any total payment amount greater than $5,000.00.) Your screen should look something like this:

12. **Click**

13. You will now see your Having Criteria on **Having Criteria**. Having criteria cannot be seen on **Criteria**.

14. The limit setup is the same as a traditional limit; however, the SQL written behind the scenes is different.

15. **Click** **View SQL**
VII. Using Wildcard Characters to Find Information

PeopleSoft applications support three wildcard characters to help you search for data in character fields. You can use these wildcard characters to find the exact information that you need.

NOTE: Wildcard characters only work with the ‘begins with’ and ‘contains’ operators.

The supported standard wildcard characters are:

<table>
<thead>
<tr>
<th>Wildcard</th>
<th>Search Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>% (percent symbol)</td>
<td>Match one or more characters.</td>
</tr>
<tr>
<td>_ (underscore)</td>
<td>Match any single character.</td>
</tr>
<tr>
<td>\ (backslash)</td>
<td>Escape character; do not treat the next character as a wildcard.</td>
</tr>
</tbody>
</table>

- % matches any string of zero or more characters. For example, C% matches any string starting with C, including C alone.
- _ matches any single character. Such as, _ones matches any five-character string ending with ones, such as Jones or Cones.
- To use one of the wild-card characters as a literal character, meaning to include a % in your string, precede the % character with a \, as \%.

The following will demonstrate an example, if you wanted to find the SFS Centrally maintained queries, you could search for ‘SFS%' in the begins with search box for Search By ‘Query Name’.. However, the system returns a list of all queries that begin with SFS which includes queries that are NOT maintained centrally, as you can see below:
Understanding that the centrally maintained query inventory list uses a naming convention of ‘SFS_module two letter designation_query name’ then you could reduce the list to those that are centrally maintained by using ALL three wildcard options.

1. Search by ‘Query Name’ begins with ‘SFS_%’
2. You can also find these special queries by changing the Search by to ‘Folder Name’ begins with ‘SFS%Inventory’
Search by Folder Name:

VIII. Internet Explorer Settings for Query Download to Excel

Depending on the settings on your computer, when you run a query to Excel, it may or may not open in the Internet Explorer browser window. If Excel does open in the browser window, Excel functionality is not available. This setting is defined in Internet Explorer.

To adjust the settings:

1. **Open** an Internet Explorer Window.
2. **Choose** “Tools” from the menu in the upper right hand corner.
3. Click Internet Options.

4. In the Internet Options screen, click Security.

5. Click Custom level...
6. **Scroll** down to the “Downloads” heading.

7. Under “Automatic prompting for file downloads”, **choose** the “Enable” radio button.
8. Click Yes
9. Click OK
10. Click **Apply**

11. Click **OK**.

12. Excel will now open in a new window when opened.

**Revision History**

<table>
<thead>
<tr>
<th>Author</th>
<th>Version</th>
<th>Date</th>
<th>Description of Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>Andrea Vredberg</td>
<td>1.0</td>
<td>06/24/13</td>
<td>Initial Draft from 8.9 version</td>
</tr>
<tr>
<td>Stacy VanWormer</td>
<td>2.0</td>
<td>07/26/13</td>
<td>Revision</td>
</tr>
<tr>
<td>Susan Kincanon</td>
<td>2.1</td>
<td>09/5/2013</td>
<td>Review, edit, format, add sections VI and VII for team review</td>
</tr>
<tr>
<td>Mike Niebanck</td>
<td>3.0</td>
<td>12/7/2013</td>
<td>Update to v9.2</td>
</tr>
</tbody>
</table>