Advanced Query Training

This training will cover the following advanced query topics:

- Any Join
- Expressions
- Subquery
- Union
- Outer Join

Any Join

In many cases, you want to retrieve data from more than one record or specify criteria in your query from a second record. In these cases, you need to link (JOIN) at least two records in one query. First, we will review predefined joins.

Tables and Views

A record listed in your list of records may be either a table or a view. A table physically stores data. A view is a logical representation of data and may consist of data from multiple tables depending on how the record was defined in Application Designer. Additionally, views may already have criteria associated with them. Therefore, it may be easier for users to create a query from a view rather than a table. If an appropriate view is not provided and requires data form multiple tables, the Query user must know which tables the data is stored, and how to join those tables.

A simple solution is for the Query user to submit a request to developers to create a view for them. Then within Query, the user will only have to access one record (a view) for the report and not have to worry about accessing multiple tables and defining additional criteria.

Joins

A join enables the user to retrieve data from two or more records or specify criteria from more than one record. Whenever a join is defined, the records involved are linked based on common fields.

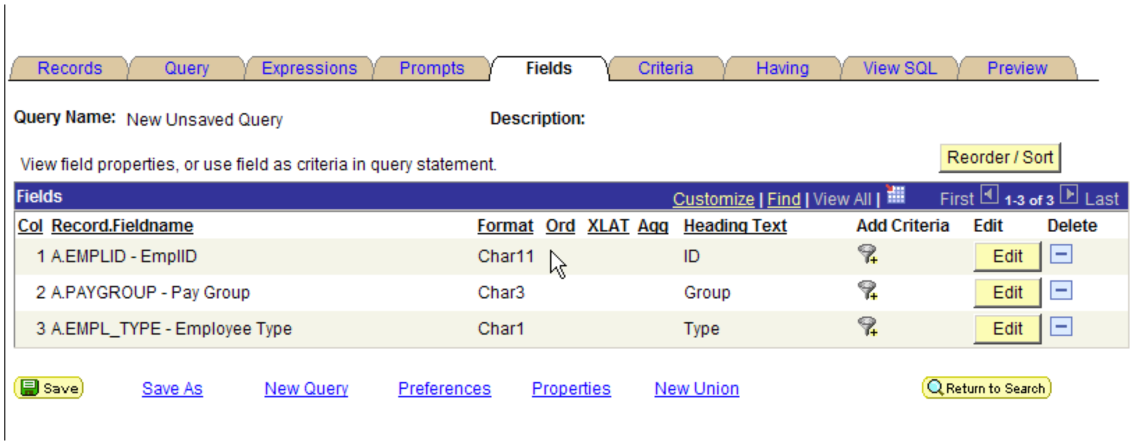
In Query, predefined joins can be generated as a ***Record Hierarchy*** join or a ***Related Record*** join. Since these types of joins are predefined, you do not have to add any criteria to link the records manually.

Record Hierarchy- Record Hierarchy joins use records that are parents or children of each other. The hierarchical relationship is defined by the *Parent Record* in the Application Designer.

Related Record- Related record joins use records from non-hierarchical records that are related by common fields. For example, description tables for common codes are related records. This relationship is determined by the Prompt Table edit defined for a field in the Application Designer.

Create a new query using predefined joins to retrieve specific information

- Create a New Query with the JOB record
- Select the three fields for output: ***EMPLID, PAYGROUP, EMPL_TYPE***



Query Name: New Unsaved Query Description:

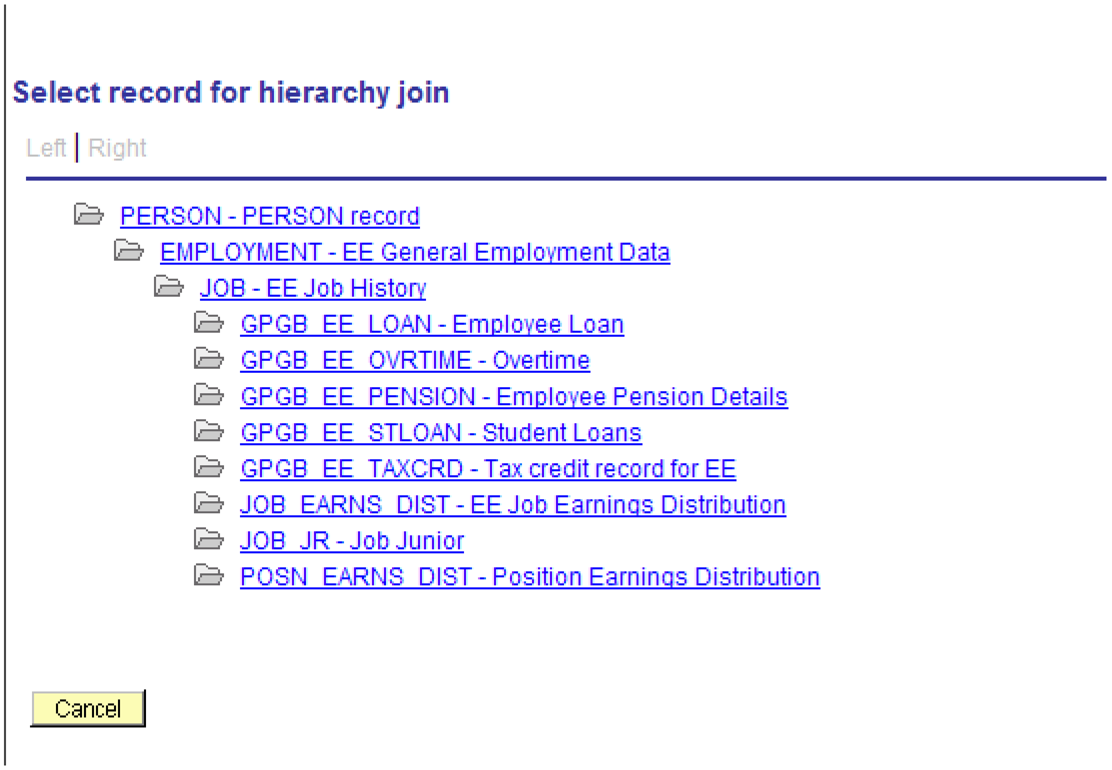
View field properties, or use field as criteria in query statement.

Fields Customize | Find | View All | First 1-3 of 3 Last

Col	Record.Fieldname	Format	Ord	XLAT	Agg	Heading Text	Add Criteria	Edit	Delete
1	A.EMPLID - EmplID	Char11				ID		Edit	
2	A.PAYGROUP - Pay Group	Char3				Group		Edit	
3	A.EMPL_TYPE - Employee Type	Char1				Type		Edit	

Record Hierarchy Join

To join records that share a common high-level key, select the **Query** tab and click the Hierarchy Join hyperlink. You'll see all the records that have a parent/child relationship to your selected record as shown in the following:


Select record for hierarchy join

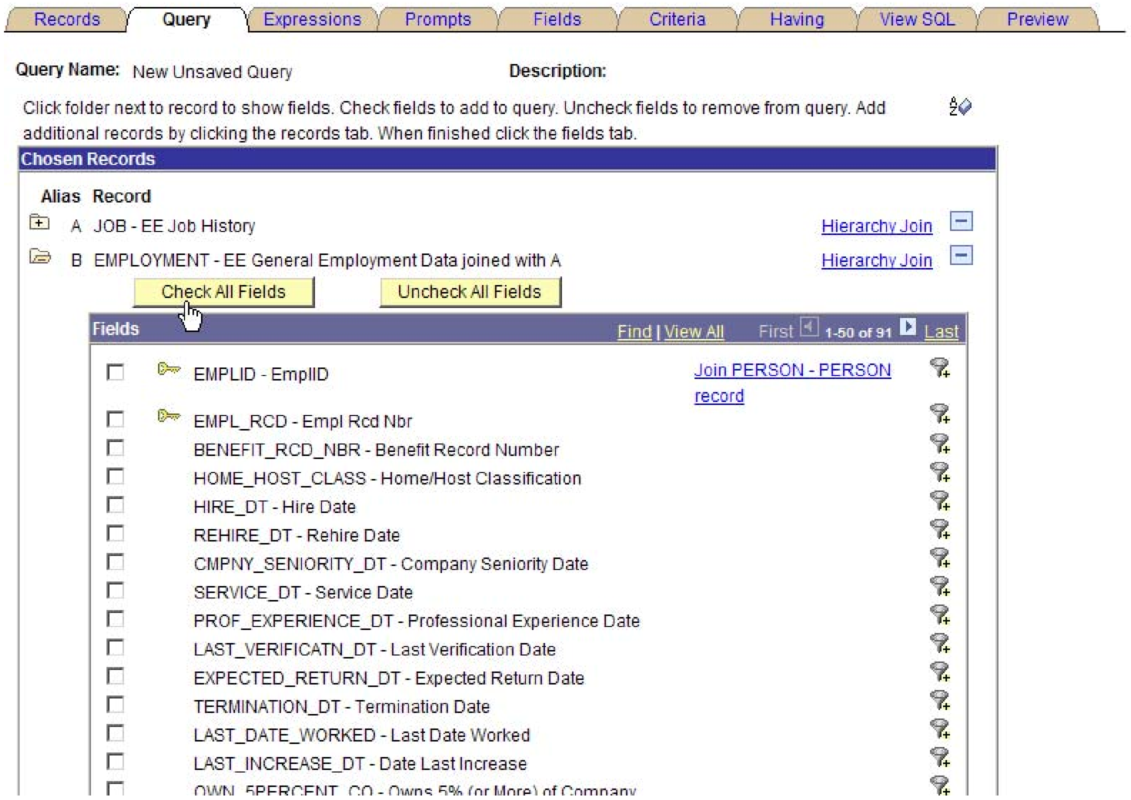
Left | Right

- PERSON - PERSON record
  - EMPLOYMENT - EE General Employment Data
    - JOB - EE Job History
      - GPGB_EE_LOAN - Employee Loan
      - GPGB_EE_OVRTIME - Overtime
      - GPGB_EE_PENSION - Employee Pension Details
      - GPGB_EE_STLOAN - Student Loans
      - GPGB_EE_TAXCRD - Tax credit record for EE
      - JOB_EARNS_DIST - EE Job Earnings Distribution
      - JOB_JR - Job Junior
      - POSN_EARNS_DIST - Position Earnings Distribution

- Select the EMPLOYMENT table from the list

The newly joined record and its fields are displayed below the first record. Notice that each record that is added to your query is assigned an incremental letter that represents an alias of the record.

The second record denotes that it was joined with the first record. In this example, EMPLOYMENT was joined with JOB.



Query Name: New Unsaved Query Description:

Click folder next to record to show fields. Check fields to add to query. Uncheck fields to remove from query. Add additional records by clicking the records tab. When finished click the fields tab.

Chosen Records

Alias	Record	
A	JOB - EE Job History	Hierarchy Join
B	EMPLOYMENT - EE General Employment Data joined with A	Hierarchy Join

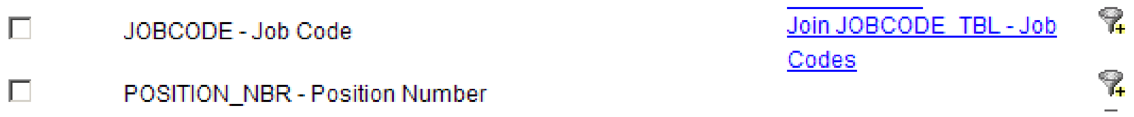
☐	EMPLID - EmplID	Join PERSON - PERSON record
☐	EMPL_RCD - Empl Rcd Nbr	
☐	BENEFIT_RCD_NBR - Benefit Record Number	
☐	HOME_HOST_CLASS - Home/Host Classification	
☐	HIRE_DT - Hire Date	
☐	REHIRE_DT - Rehire Date	
☐	CMPNY_SENIORITY_DT - Company Seniority Date	
☐	SERVICE_DT - Service Date	
☐	PROF_EXPERIENCE_DT - Professional Experience Date	
☐	LAST_VERIFICATN_DT - Last Verification Date	
☐	EXPECTED_RETURN_DT - Expected Return Date	
☐	TERMINATION_DT - Termination Date	
☐	LAST_DATE_WORKED - Last Date Worked	
☐	LAST_INCREASE_DT - Date Last Increase	
☐	OWN_5PERCENT_CO - Owns 5% (or More) of Company	

- From the EMPLOYMENT table, select HIRE_DT

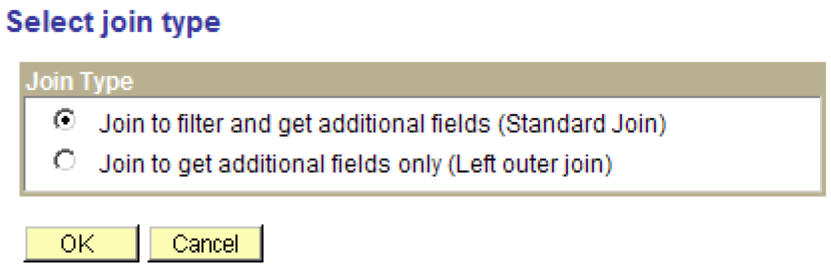
Related Record Join

The related records are specific to a field in one of the records previously selected. If a field has a related record, you will see it listed as a hyperlink across from the field name.

- Select the Query tab and from the JOB table locate the JOBCODE field
- Click the Join JOBCODE_TBL- Job Codes hyperlink



☐	JOBCODE - Job Code	Join JOBCODE_TBL - Job Codes
☐	POSITION_NBR - Position Number	

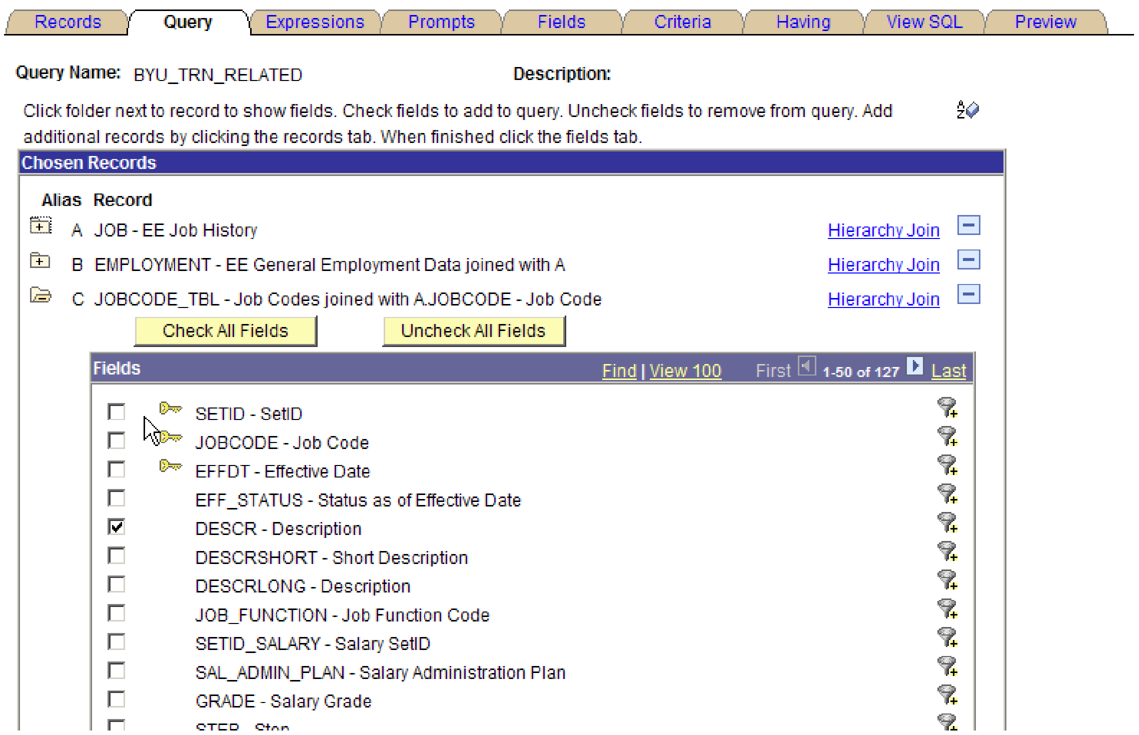


Select join type

Join Type

[✓ Join to filter and get additional fields (Standard Join)]
Join to get additional fields only (Left outer join)

- Click **OK**


Records | Query | Expressions | Prompts | Fields | Criteria | Having | View SQL | Preview

Query Name: BYU_TRN_RELATED   Description:

Click folder next to record to show fields. Check fields to add to query. Uncheck fields to remove from query. Add additional records by clicking the records tab. When finished click the fields tab.

Chosen Records

Alias	Record	
A	JOB - EE Job History	Hierarchy Join
B	EMPLOYMENT - EE General Employment Data joined with A	Hierarchy Join
C	JOBCODE_TBL - Job Codes joined with A.JOBCODE - Job Code	Hierarchy Join

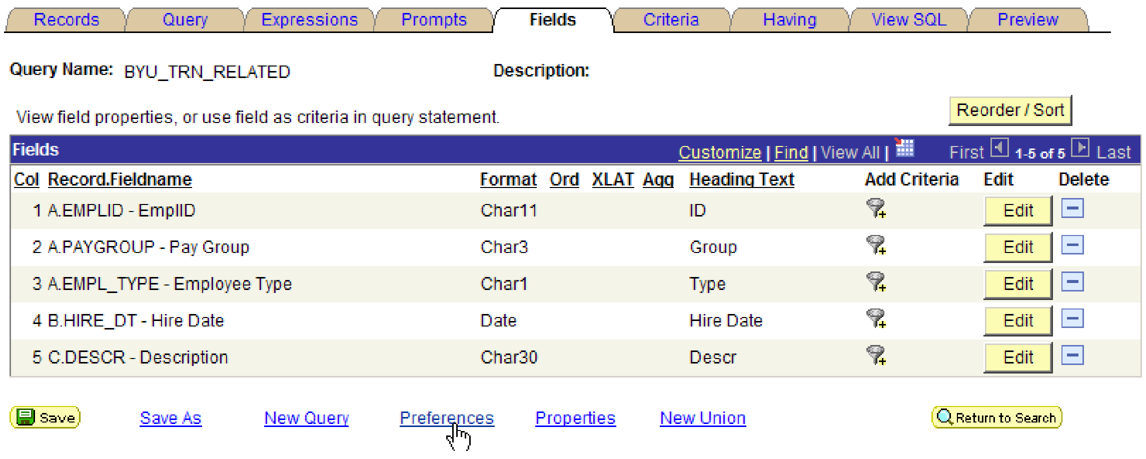
Check All Fields   Uncheck All Fields

Fields   Find | View 100   First 1-50 of 127 Last

☐	SETID - SetID
☐	JOBCODE - Job Code
☐	EFFDT - Effective Date
☐	EFF_STATUS - Status as of Effective Date
✓	DESCR - Description
☐	DESCRSHORT - Short Description
☐	DESCRLONG - Description
☐	JOB_FUNCTION - Job Function Code
☐	SETID_SALARY - Salary SetID
☐	SAL_ADMIN_PLAN - Salary Administration Plan
☐	GRADE - Salary Grade
☐	STEP - Step

Your newly joined record appears below the others in the **Query** tab, under *Chosen Records*. Notice that the description for the Record Hierarchy join includes "*joined with A*" and for the related record join the description includes "*joined with A.JOBCODE – Job Code*".

- Select the **DESCR** field from JOBCODE
- Select the **Fields** tab


Records | Query | Expressions | Prompts | Fields | Criteria | Having | View SQL | Preview

Query Name: BYU_TRN_RELATED   Description:

View field properties, or use field as criteria in query statement.   Reorder / Sort

Fields   Customize | Find | View All | First 1-5 of 5 Last

Col	Record.Fieldname	Format	Ord	XLAT	Agg	Heading Text	Add Criteria	Edit	Delete
1	A.EMPLID - EmplID	Char11				ID		Edit	
2	A.PAYGROUP - Pay Group	Char3				Group		Edit	
3	A.EMPL_TYPE - Employee Type	Char1				Type		Edit	
4	B.HIRE_DT - Hire Date	Date				Hire Date		Edit	
5	C.DESCR - Description	Char30				Descr		Edit	

Save   Save As   New Query   Preferences   Properties   New Union   Return to Search

- Let's preview the query.

You can create queries based on multiple tables even when the table you're joining is not in the parent hierarchy or related record hierarchy. You manually link the tables to retrieve the most correct output.

Enabling the Auto Join Feature

You can join any record in your dictionary tree to your currently selected record(s). PeopleSoft Query can determine the conditions of the Any Join with the Auto Join option.

Let's say you want to retrieve some information about students and the companies/customers they work for. For this example, you need to pull data from both the Student_Data table and the Customer table.

```
Student_Data        Customer
Student_ID (Key)    Customer_ID (Key)
Name                Descr
Customer_ID
```

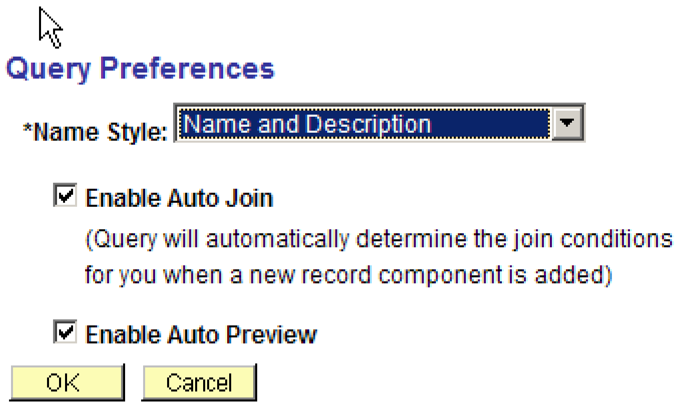
After doing an Any Join and selecting the Student_ID and Name fields from the Student_Data record and the Descr field from the Customer record, the SQL would read as follows:

```
SELECT A.Student_ID, A.Name, B.Descr FROM Student_Data A,
Customer B WHERE A.Customer_ID = B.Customer_ID
```

The WHERE clause is criteria that the Any Join would create. This criteria would be found under the Criteria tab. With the Auto Join option enabled, criteria is automatically added linking the common keys (Customer_ID) between two records.

Auto Join is automatically enabled; you can turn it off using the Preferences link in Query Manager. If the Auto Join is not enabled, you will need to create the necessary join criteria.

Click the Preferences hyperlink.


Query Preferences

*Name Style: Name and Description

[✓] Enable Auto Join
(Query will automatically determine the join conditions for you when a new record component is added)

[✓] Enable Auto Preview

Auto Join performs the join on all matching key fields (excluding EFFDT and EFFSEQ).

Cartesian Joins

When two or more records are joined, if the proper join criteria are not established (usually based on mutual KEY FIELDS) in the Query, a Cartesian Join may occur. A Cartesian Join takes the data from the first row of table A and joins it with every row of data from table B, then repeats for every row of data in table A. This does not provide the desired result and needlessly ties up the database server. It could even cause your query to time out.

The ability to create Any Joins can be disabled by user in the permission lists that are tied to their roles.

Expressions

Expressions are a way of adding fields to a query, in addition to the fields of the selected record(s). These fields can be used for outputs as well as for defining criteria.

Expressions allow the creation of calculations in a Query. Calculations are rarely stored in a relational database. Calculations are typically processed when a query is run.

To create a user-defined calculation (expression) in a query, you need to know the SQL specific syntax for the expression.

Reasons to use expressions:
- As columns in the query output
- As comparison values in selection criteria
- To create outer joins
- To translate coded values
- To use SQL commands

Creating Expressions in Queries

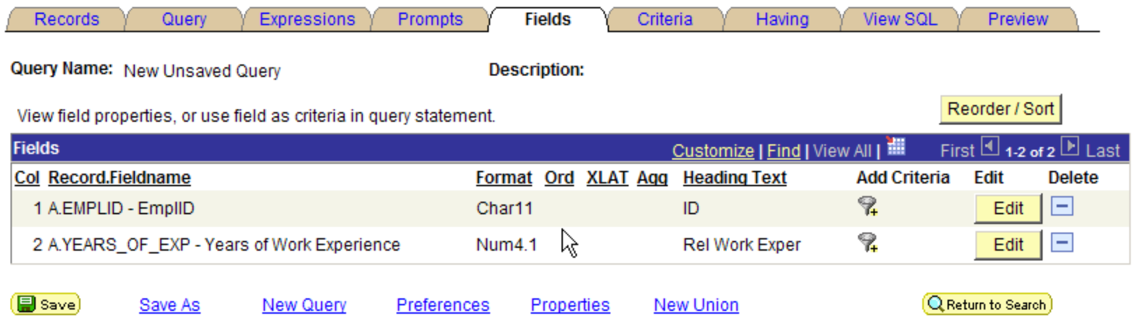
Expressions are calculations the Query Manager performs on behalf of a query. They are used to calculate a value that the database doesn't provide. They can be used in two ways: as columns in the query output or in its criteria.

Using Expressions as Columns in Query Output

An expression can be treated as a field in the query. When selected for output, you can change its column heading or sort by it.

Create a query that displays a list of employees and adds years to their years of related work experience.

- Create a new query using the PERS_DATA_EFFDT record
- Select these fields: **EMPLID, YEARS_OF_EXP**



Query Name: New Unsaved Query   Description:

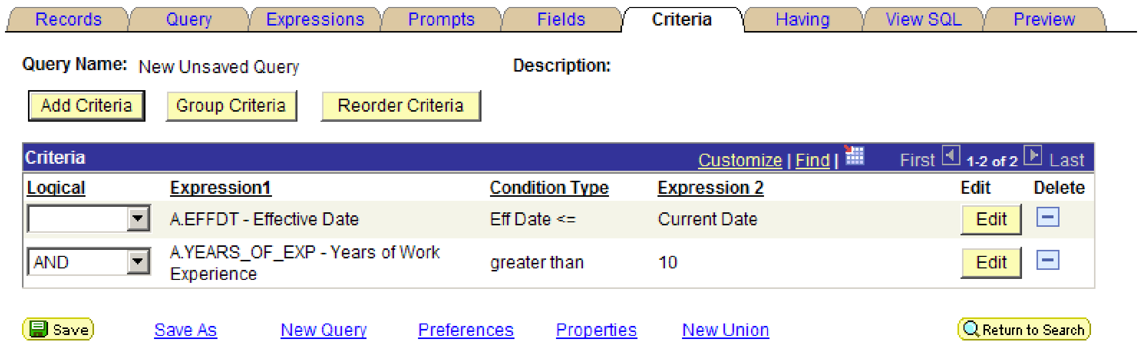
View field properties, or use field as criteria in query statement.   Reorder / Sort

Fields   Customize | Find | View All | First 1-2 of 2 Last

Col	Record.Fieldname	Format	Ord	XLAT	Agg	Heading Text	Add Criteria	Edit	Delete
1	A.EMPLID - EmplID	Char11				ID		Edit	
2	A.YEARS_OF_EXP - Years of Work Experience	Num4.1				Rel Work Exper		Edit	

Save   Save As   New Query   Preferences   Properties   New Union   Return to Search

Next, narrow the criteria by removing employees that have less than 10 years experience by adding the following criteria:



Query Name: New Unsaved Query   Description:

Add Criteria   Group Criteria   Reorder Criteria

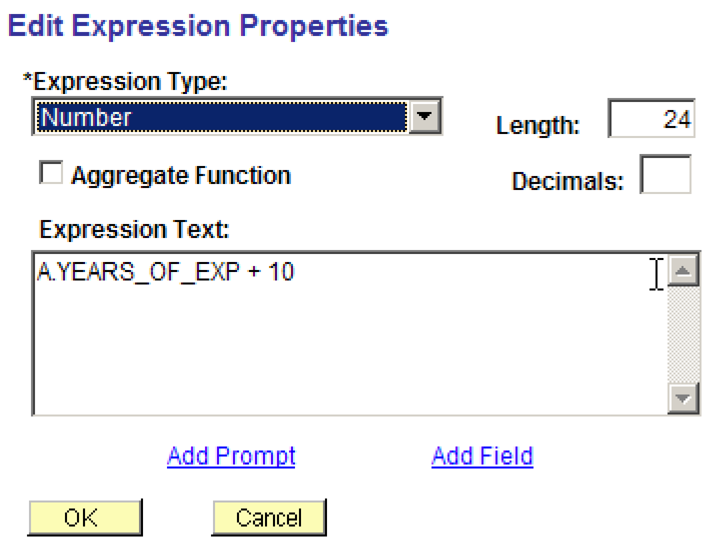
Criteria   Customize | Find | First 1-2 of 2 Last

Logical	Expression1	Condition Type	Expression 2	Edit	Delete
	A.EFFDT - Effective Date	Eff Date <=	Current Date	Edit	
AND	A.YEARS_OF_EXP - Years of Work Experience	greater than	10	Edit	

Save   Save As   New Query   Preferences   Properties   New Union   Return to Search

Next, create a simple calculation that will add 10 to the reported work experience.

Select the Expression tab and click the Add Expression button



Edit Expression Properties

*Expression Type: Number

Length: 24

Aggregate Function

Decimals:

Expression Text:

```
A.YEARS_OF_EXP + 10
```

Expression Type- Select the data type of the value that this expression returns.

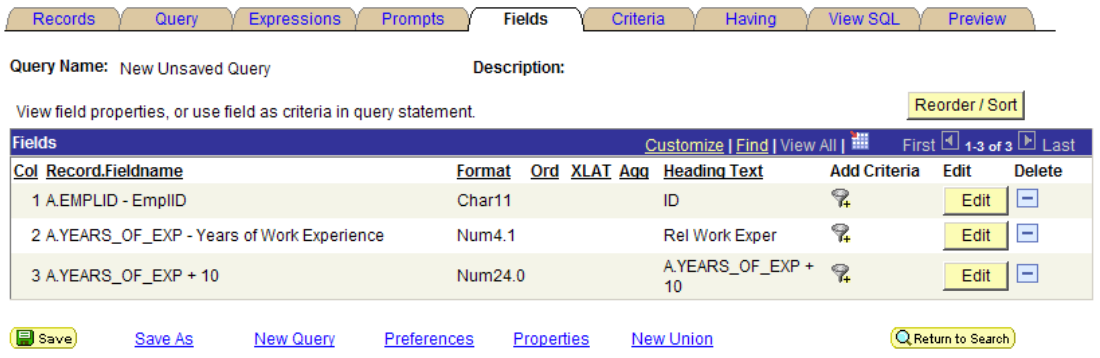
Length- Enter the maximum length of the string. If you selected *Number* or *Signed Number* as the expression type, enter the number of digits in the Integer box and the number of digits to the right of the decimal in the Decimals box. Make sure the Integer box is big enough, it will truncate the number if the size is not large enough.

Aggregate Function- Select to create an aggregate function such as Sum, Avg, or Count.

Add Prompt- Click to add a prompt as part of your expression

- Enter the expression shown above in the Expression Text box.

To display the result of the calculation (expression) in the query's output, click the Use as Field hyperlink. Select the field tab and compare with the following:



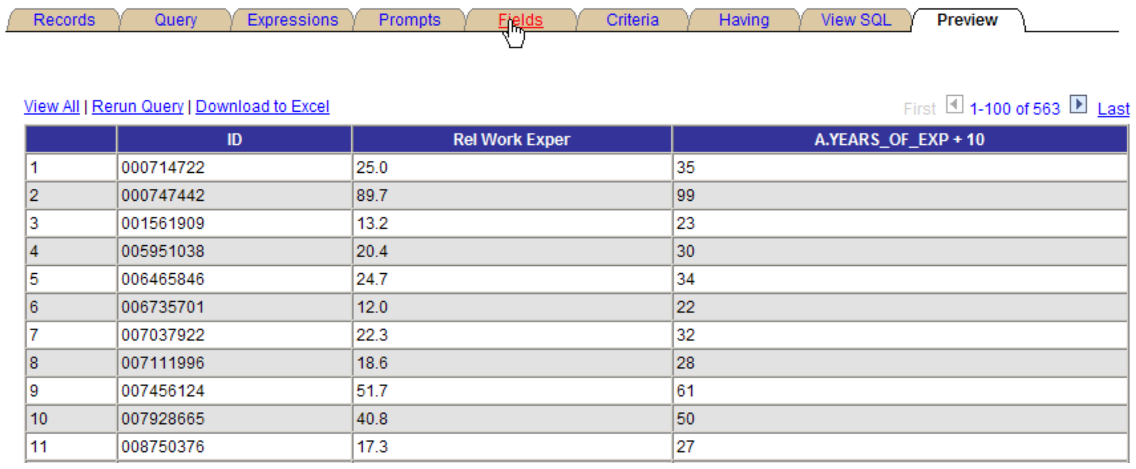
Query Name: New Unsaved Query Description:

View field properties, or use field as criteria in query statement.

Fields — Customize | Find | View All | First 1-3 of 3 Last

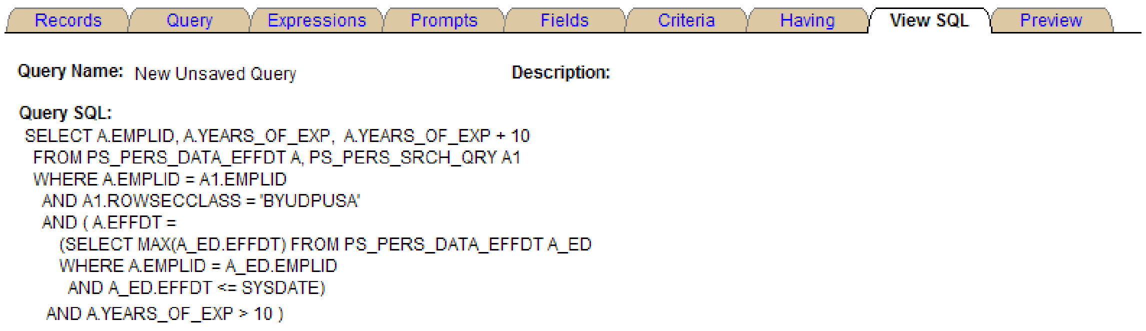
Col	Record.Fieldname	Format	Ord	XLAT	Agg	Heading Text	Add Criteria	Edit	Delete
1	A.EMPLID - EmplID	Char11				ID		Edit	
2	A.YEARS_OF_EXP - Years of Work Experience	Num4.1				Rel Work Exper		Edit	
3	A.YEARS_OF_EXP + 10	Num24.0				A.YEARS_OF_EXP + 10		Edit	

Let's preview the query.



	ID	Rel Work Exper	A.YEARS_OF_EXP + 10
1	000714722	25.0	35
2	000747442	89.7	99
3	001561909	13.2	23
4	005951038	20.4	30
5	006465846	24.7	34
6	006735701	12.0	22
7	007037922	22.3	32
8	007111996	18.6	28
9	007456124	51.7	61
10	007928665	40.8	50
11	008750376	17.3	27

The SQL should be as follows:



Query Name: New Unsaved Query Description:

Query SQL:

```
SELECT A.EMPLID, A.YEARS_OF_EXP,  A.YEARS_OF_EXP + 10
  FROM PS_PERS_DATA_EFFDT A, PS_PERS_SRCH_QRY A1
  WHERE A.EMPLID = A1.EMPLID
    AND A1.ROWSECCLASS = 'BYUDPUSA'
    AND ( A.EFFDT =
        (SELECT MAX(A_ED.EFFDT) FROM PS_PERS_DATA_EFFDT A_ED
        WHERE A.EMPLID = A_ED.EMPLID
          AND A_ED.EFFDT <= SYSDATE)
     AND A.YEARS_OF_EXP > 10 )
```

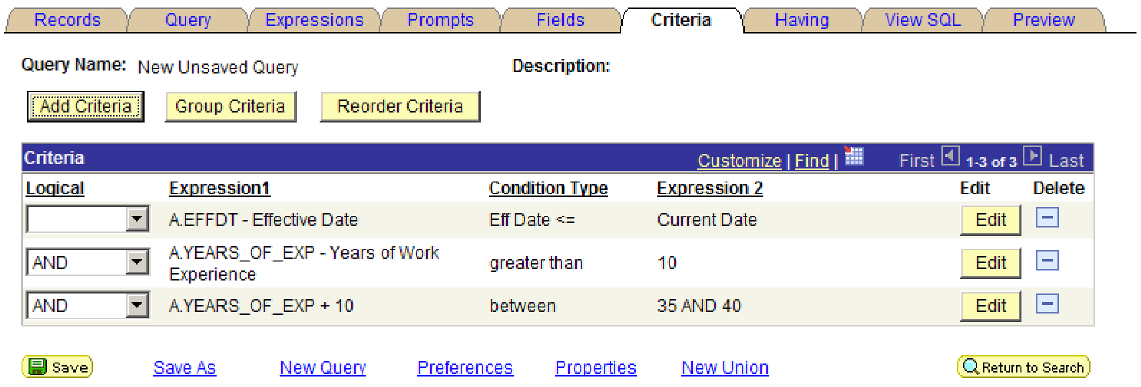
Maintaining Expressions

You maintain expressions on the Expressions page. You can:

- View all expressions
- Modify expressions by clicking the appropriate Edit button
- Delete expressions using the appropriate Delete button

Using Expressions in Criteria

In the query you just created, suppose you want to show only those employees who will have between 35 and 40 years in 10 years. To do so, add your expression to a row of criteria as follows:


Records | Query | Expressions | Prompts | Fields | Criteria | Having | View SQL | Preview

Query Name: New Unsaved Query Description:

Add Criteria | Group Criteria | Reorder Criteria

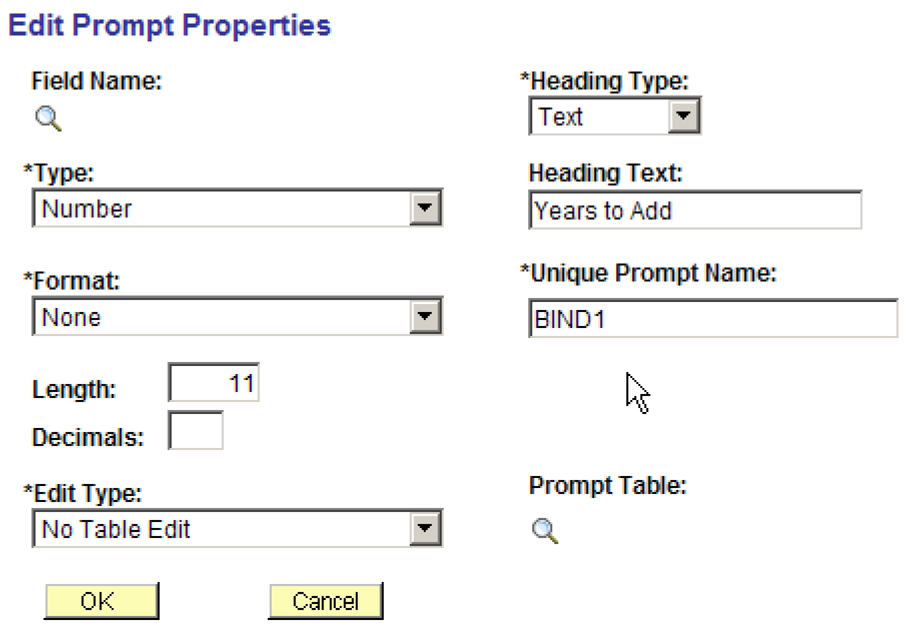
Criteria Customize | Find | First 1-3 of 3 Last

Logical	Expression1	Condition Type	Expression 2	Edit	Delete
	A.EFFDT - Effective Date	Eff Date <=	Current Date	Edit	⊟
AND	A.YEARS_OF_EXP - Years of Work Experience	greater than	10	Edit	⊟
AND	A.YEARS_OF_EXP + 10	between	35 AND 40	Edit	⊟

Save | Save As | New Query | Preferences | Properties | New Union | Return to Search

Using Prompts in Expressions

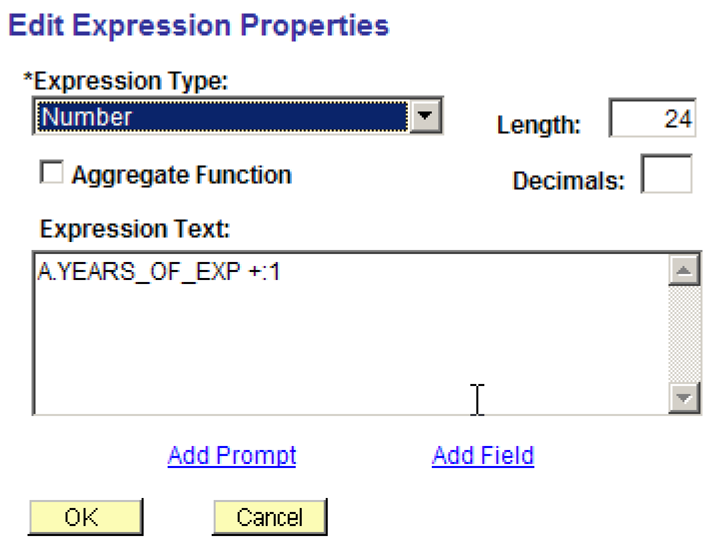
What if you want to project out 15, 20, or 30 years? First create a prompt and then edit your expression to include the new runtime prompt.


Edit Prompt Properties

Field Name:	
*Heading Type:	Text
*Type:	Number
Heading Text:	Years to Add
*Format:	None
*Unique Prompt Name:	BIND1
Length:	11
Decimals:	
*Edit Type:	No Table Edit
Prompt Table:	

OK | Cancel

Edit the expression to include the runtime prompt as follows:



Edit Expression Properties

*Expression Type: Number

Length: 24

Aggregate Function [ ]

Decimals:

Expression Text:

```
A.YEARS_OF_EXP +:1
```

Creating Expressions Using Literals

Literals are text placeholders. They are useful for combining text from two columns in a query.

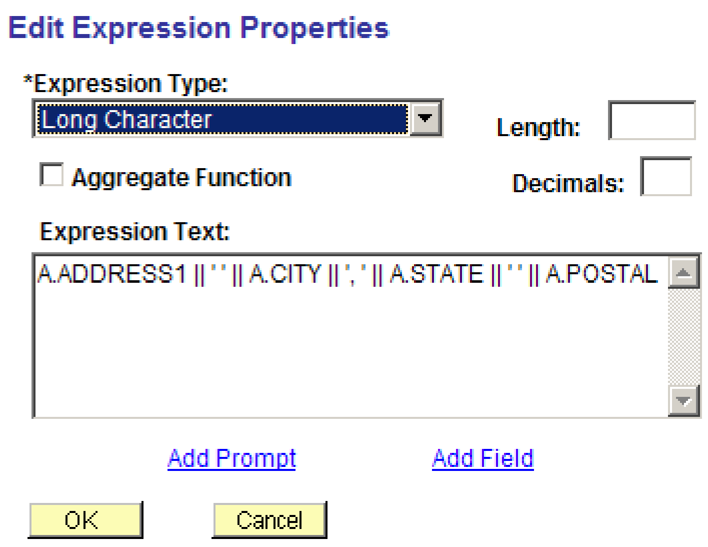
For instance, rather than have a separate column for an employees address, city, state and zip code, you can use an expression to combine them into one field.

Create a query using PERSON_ADDRESS

- Select the following fields: **EMPLID, ADDRESS1, CITY, STATE, POSTAL**
- Add the criteria that the **STATE** field must equal WA (to narrow the results)

The Oracle concatenation operator is || . Text literals are always surrounded by single quotes (‘).

Next, select the Expression tab and click on the Add Expression button. Complete the following expression:



Edit Expression Properties

*Expression Type: Long Character

Length:

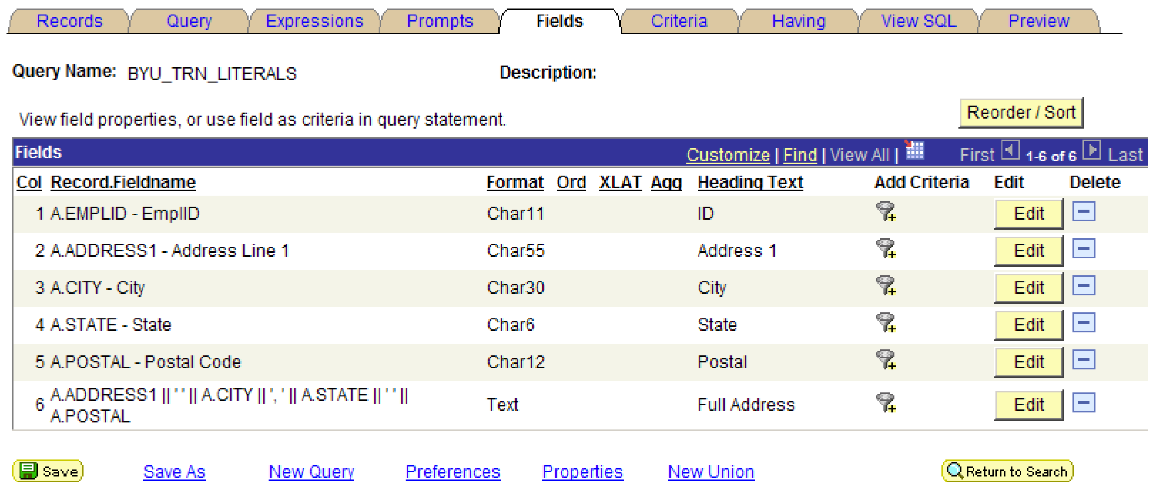
Decimals:

☐ Aggregate Function

Expression Text:

```
A.ADDRESS1 || ' ' || A.CITY || ', ' || A.STATE || ' ' || A.POSTAL
```

- Click the **OK** button and click on the Use as Field hyperlink. Select the Fields tab and then click on the Edit button. Change the heading text to “Full Address”



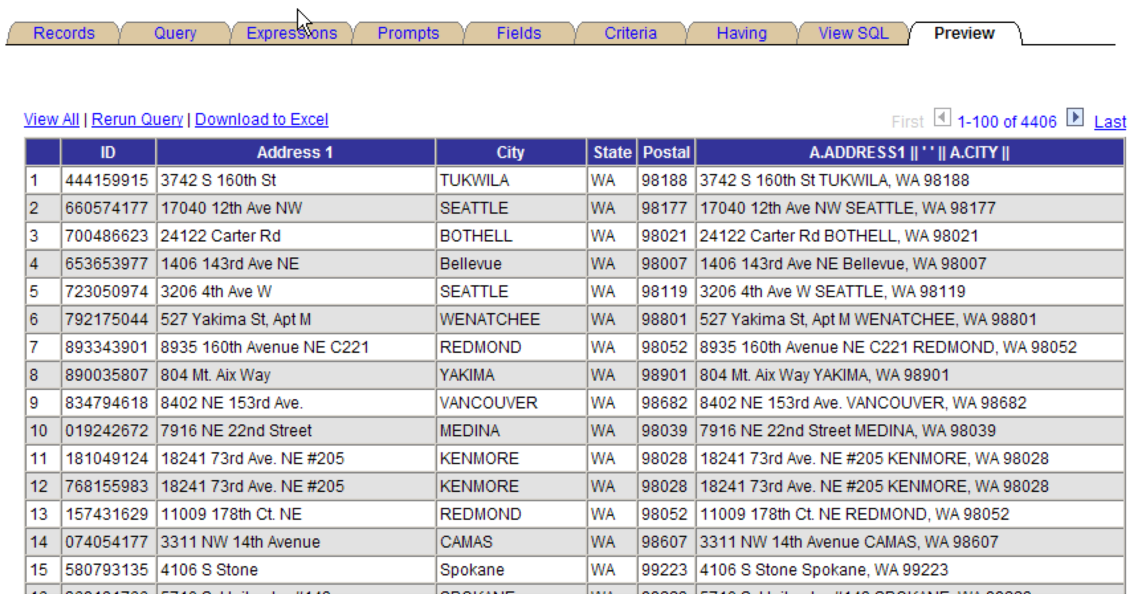
Query Name: BYU_TRN_LITERALS   Description:

View field properties, or use field as criteria in query statement.

Fields

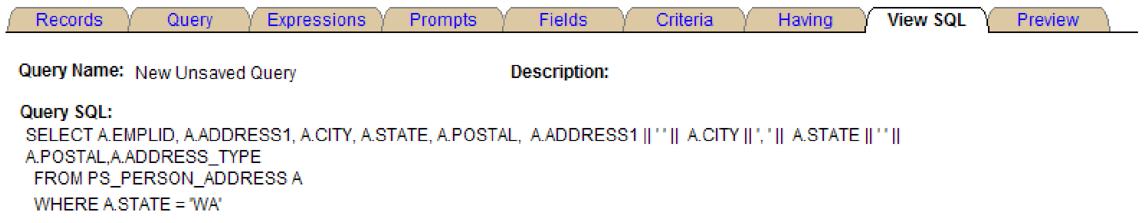
Col	Record.Fieldname	Format	Ord	XLAT	Agg	Heading Text	Add Criteria	Edit	Delete
1	A.EMPLID - EmplID	Char11				ID		Edit	
2	A.ADDRESS1 - Address Line 1	Char55				Address 1		Edit	
3	A.CITY - City	Char30				City		Edit	
4	A.STATE - State	Char6				State		Edit	
5	A.POSTAL - Postal Code	Char12				Postal		Edit	
6	A.ADDRESS1 || ' ' || A.CITY || ', ' || A.STATE || ' ' || A.POSTAL	Text				Full Address		Edit	

Here is a Preview of the Query:



	ID	Address 1	City	State	Postal	A.ADDRESS1 || ' ' || A.CITY ||
1	444159915	3742 S 160th St	TUKWILA	WA	98188	3742 S 160th St TUKWILA, WA 98188
2	660574177	17040 12th Ave NW	SEATTLE	WA	98177	17040 12th Ave NW SEATTLE, WA 98177
3	700486623	24122 Carter Rd	BOTHELL	WA	98021	24122 Carter Rd BOTHELL, WA 98021
4	653653977	1406 143rd Ave NE	Bellevue	WA	98007	1406 143rd Ave NE Bellevue, WA 98007
5	723050974	3206 4th Ave W	SEATTLE	WA	98119	3206 4th Ave W SEATTLE, WA 98119
6	792175044	527 Yakima St, Apt M	WENATCHEE	WA	98801	527 Yakima St, Apt M WENATCHEE, WA 98801
7	893343901	8935 160th Avenue NE C221	REDMOND	WA	98052	8935 160th Avenue NE C221 REDMOND, WA 98052
8	890035807	804 Mt. Aix Way	YAKIMA	WA	98901	804 Mt. Aix Way YAKIMA, WA 98901
9	834794618	8402 NE 153rd Ave.	VANCOUVER	WA	98682	8402 NE 153rd Ave. VANCOUVER, WA 98682
10	019242672	7916 NE 22nd Street	MEDINA	WA	98039	7916 NE 22nd Street MEDINA, WA 98039
11	181049124	18241 73rd Ave. NE #205	KENMORE	WA	98028	18241 73rd Ave. NE #205 KENMORE, WA 98028
12	768155983	18241 73rd Ave. NE #205	KENMORE	WA	98028	18241 73rd Ave. NE #205 KENMORE, WA 98028
13	157431629	11009 178th Ct. NE	REDMOND	WA	98052	11009 178th Ct. NE REDMOND, WA 98052
14	074054177	3311 NW 14th Avenue	CAMAS	WA	98607	3311 NW 14th Avenue CAMAS, WA 98607
15	580793135	4106 S Stone	Spokane	WA	99223	4106 S Stone Spokane, WA 99223
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

The SQL should be as follows:



Query Name: New Unsaved Query **Description:**

Query SQL:

```
 SELECT A.EMPLID, A.ADDRESS1, A.CITY, A.STATE, A.POSTAL,  A.ADDRESS1 || ' ' ||  A.CITY || ', ' ||  A.STATE || ' ' ||
 A.POSTAL,A.ADDRESS_TYPE
  FROM PS_PERSON_ADDRESS A
  WHERE A.STATE = 'WA'
```

Subquery

You use subqueries to check for information in another query and return that result set to use in the parent (original) query. You can also check for a value in another query and use it in the parent query.

Using Subqueries

A subquery is a query within a query. You use subqueries to compare a value for a field in the original query to the results of a second query. Within the WHERE clause (Criteria tab) of a query, you reference another query. Subqueries can:

- Produce a single value or a list of values
- Produce a single value that the query uses for comparison
- Return a value of true or false

The *Condition Type* that you specify in your criteria determines what the subquery returns to the query.

Although a subquery can only retrieve one data field from a single record, the subquery can contain a join. You can use this feature to specify criteria based on two records.

To set up a subquery, access the Criteria tab and specify subquery for Expression 2 Type, then click the Define/Edit Subquery hyperlink.

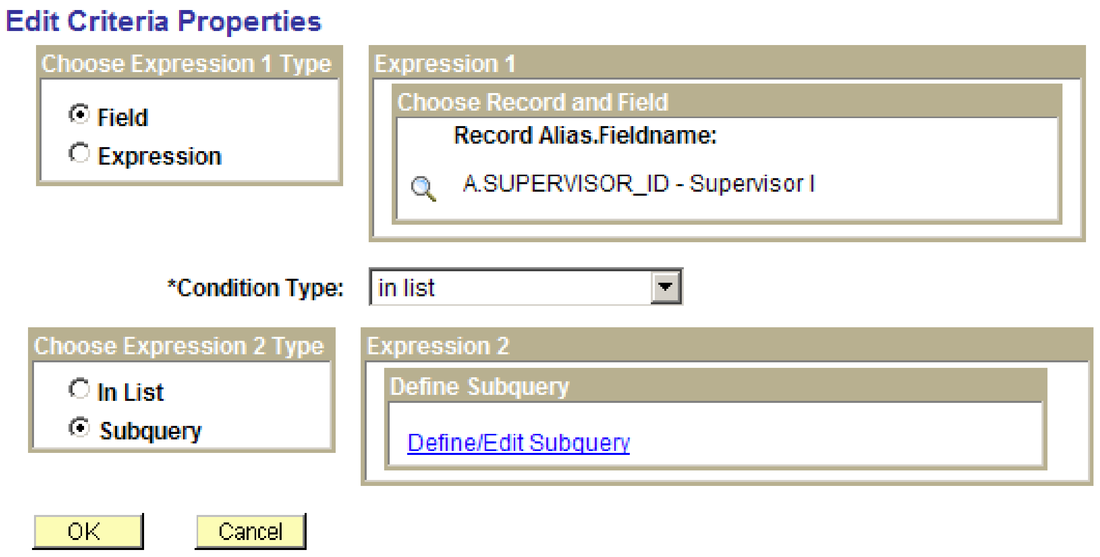
You are taken to the Records tab to select a record for the subquery definition.

Creating Subqueries

Create a query to identify employees where their department head has over 20 years related experience.

Create a new query based on the EMPLOYEES record.

- Select the follow field: **EMPLID**
- Add Criteria for the **SUPERVISOR_ID** field with the condition type of ***in list*** and the Expression Type of Subquery. Select the Define/Edit Subquery link


Edit Criteria Properties

Choose Expression 1 Type: [✓ Field] Expression

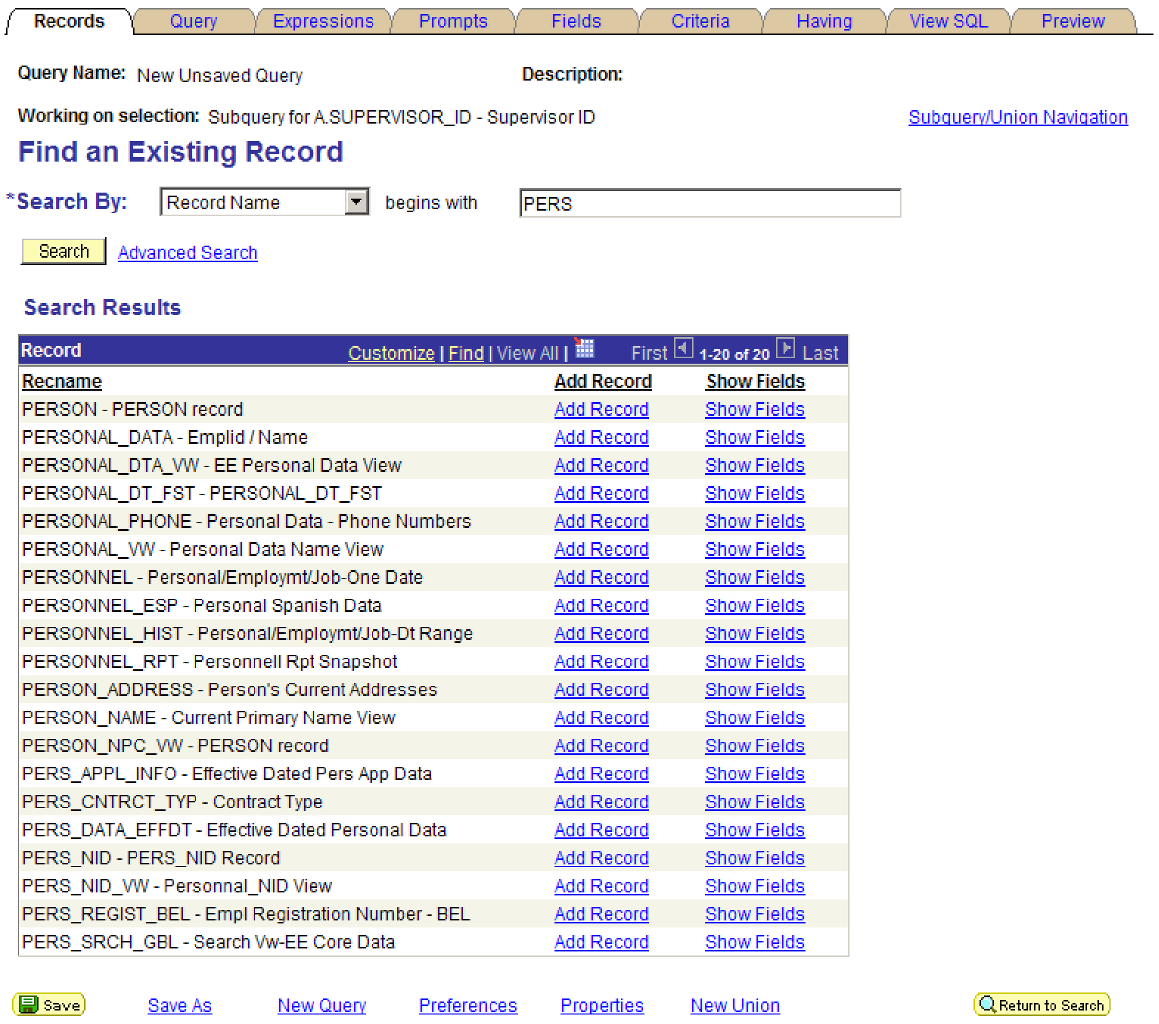
Expression 1 – Choose Record and Field – Record Alias.Fieldname: A.SUPERVISOR_ID - Supervisor I

*Condition Type: in list

Choose Expression 2 Type: In List [✓ Subquery]

Expression 2 – Define Subquery – Define/Edit Subquery

- Next, you will be taken to the Record tab where you will identify the record for the subquery. Select the Add Record link for PERS_DATA_EFFDT.


Records | Query | Expressions | Prompts | Fields | Criteria | Having | View SQL | Preview

Query Name: New Unsaved Query   Description:

Working on selection: Subquery for A.SUPERVISOR_ID - Supervisor ID   Subquery/Union Navigation

Find an Existing Record

*Search By: Record Name begins with PERS

Search Advanced Search

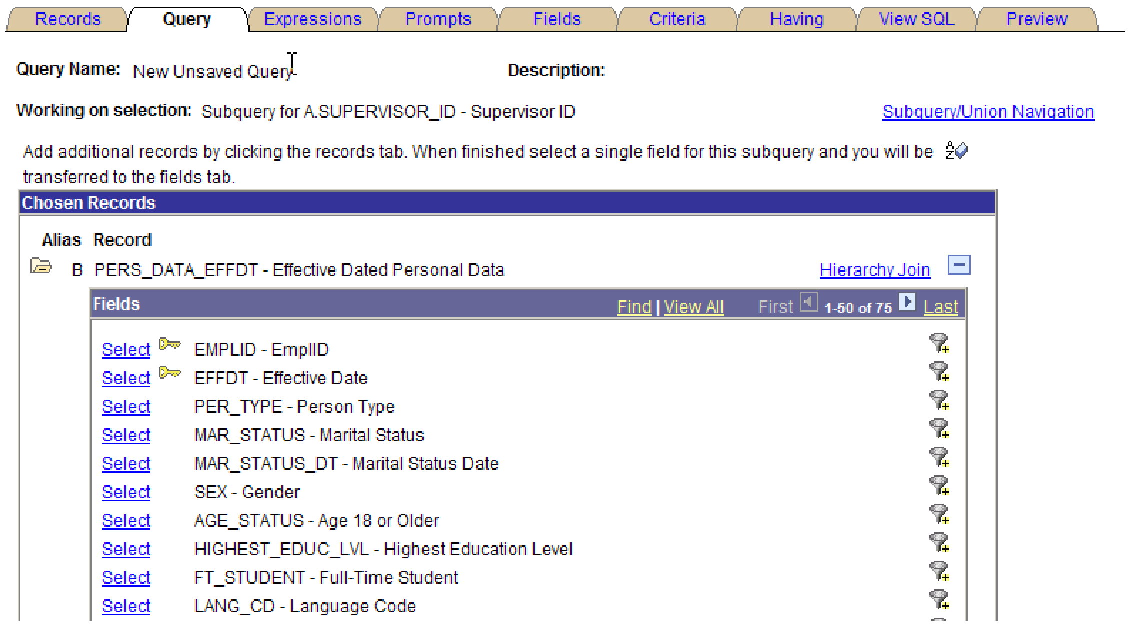
Search Results

Record   Customize | Find | View All | First 1-20 of 20 Last

Recname	Add Record	Show Fields
PERSON - PERSON record	Add Record	Show Fields
PERSONAL_DATA - Emplid / Name	Add Record	Show Fields
PERSONAL_DTA_VW - EE Personal Data View	Add Record	Show Fields
PERSONAL_DT_FST - PERSONAL_DT_FST	Add Record	Show Fields
PERSONAL_PHONE - Personal Data - Phone Numbers	Add Record	Show Fields
PERSONAL_VW - Personal Data Name View	Add Record	Show Fields
PERSONNEL - Personal/Employmt/Job-One Date	Add Record	Show Fields
PERSONNEL_ESP - Personal Spanish Data	Add Record	Show Fields
PERSONNEL_HIST - Personal/Employmt/Job-Dt Range	Add Record	Show Fields
PERSONNEL_RPT - Personnell Rpt Snapshot	Add Record	Show Fields
PERSON_ADDRESS - Person's Current Addresses	Add Record	Show Fields
PERSON_NAME - Current Primary Name View	Add Record	Show Fields
PERSON_NPC_VW - PERSON record	Add Record	Show Fields
PERS_APPL_INFO - Effective Dated Pers App Data	Add Record	Show Fields
PERS_CNTRCT_TYP - Contract Type	Add Record	Show Fields
PERS_DATA_EFFDT - Effective Dated Personal Data	Add Record	Show Fields
PERS_NID - PERS_NID Record	Add Record	Show Fields
PERS_NID_VW - Personnal_NID View	Add Record	Show Fields
PERS_REGIST_BEL - Empl Registration Number - BEL	Add Record	Show Fields
PERS_SRCH_GBL - Search Vw-EE Core Data	Add Record	Show Fields

Save Save As New Query Preferences Properties New Union Return to Search

- Next, you will be taken to the Query tab where you will identify the values which will be added to the list in the Parent Query. Click on the Select hyperlink for the EMPLID field. *Only one field should be selected as output for the subquery*.



Query Name: New Unsaved Query Description:

Working on selection: Subquery for A.SUPERVISOR_ID - Supervisor ID Subquery/Union Navigation

Add additional records by clicking the records tab. When finished select a single field for this subquery and you will be transferred to the fields tab.

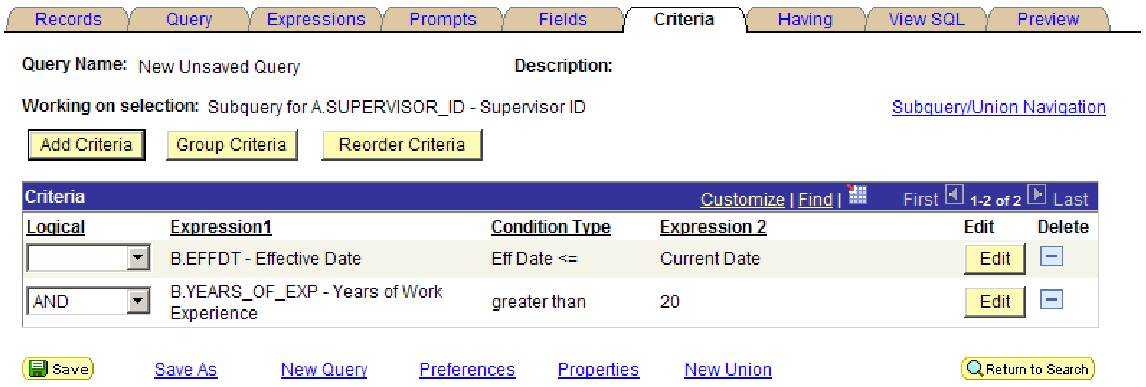
Chosen Records

Alias	Record	
B	PERS_DATA_EFFDT - Effective Dated Personal Data	Hierarchy Join

Fields — Find \| View All First 1-50 of 75 Last

Select	EMPLID - EmplID
Select	EFFDT - Effective Date
Select	PER_TYPE - Person Type
Select	MAR_STATUS - Marital Status
Select	MAR_STATUS_DT - Marital Status Date
Select	SEX - Gender
Select	AGE_STATUS - Age 18 or Older
Select	HIGHEST_EDUC_LVL - Highest Education Level
Select	FT_STUDENT - Full-Time Student
Select	LANG_CD - Language Code

- Next you will create the necessary criteria for the subquery. The subquery should return a list of EMPLID's of managers that have more than 20 years of experience. From the Criteria tab click on the Add Criteria button and add the following criteria:



Query Name: New Unsaved Query Description:

Working on selection: Subquery for A.SUPERVISOR_ID - Supervisor ID Subquery/Union Navigation

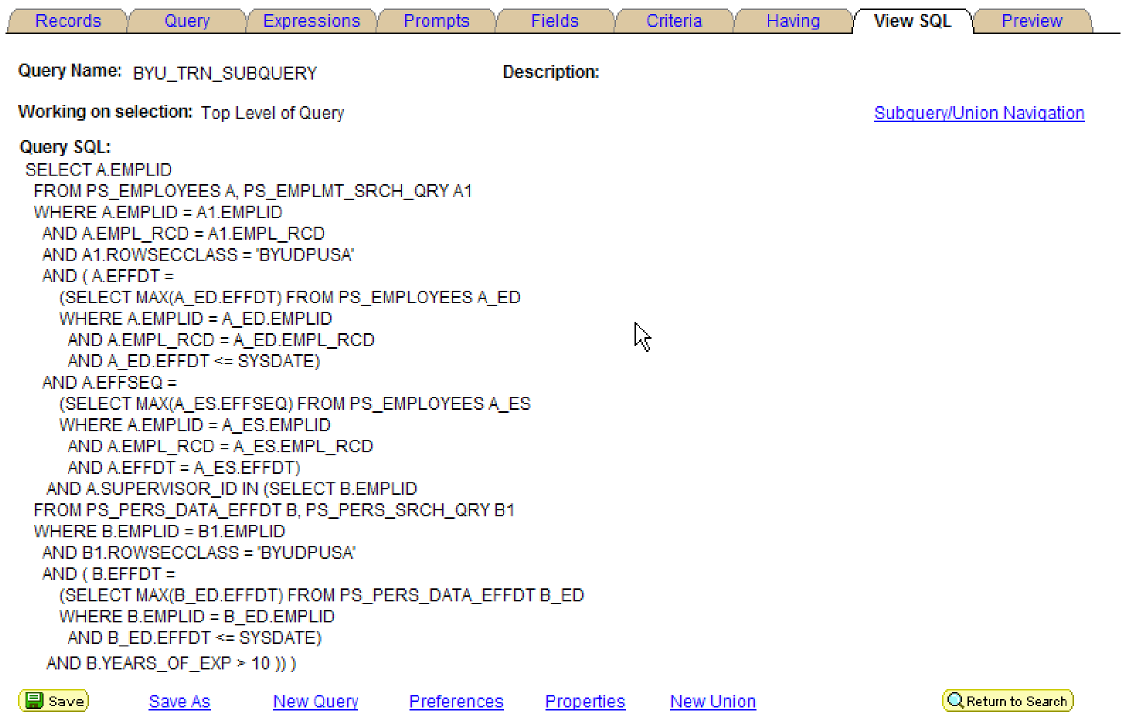
Criteria — Customize \| Find \| First 1-2 of 2 Last

Logical	Expression1	Condition Type	Expression 2	Edit	Delete
	B.EFFDT - Effective Date	Eff Date <=	Current Date	Edit	
AND	B.YEARS_OF_EXP - Years of Work Experience	greater than	20	Edit	

Important! When the data being retrieved by the subquery is dependant upon data retrieved by the parent query, the subquery must be linked to the parent query. The subquery cannot be run independently.

Notice the Subquery/Union Navigation hyperlink; this is how you navigate between the parent query and the subquery.

The SQL should be as follows:



Query Name: BYU_TRN_SUBQUERY Description:

Working on selection: Top Level of Query Subquery/Union Navigation

Query SQL:

```
SELECT A.EMPLID
  FROM PS_EMPLOYEES A, PS_EMPLMT_SRCH_QRY A1
  WHERE A.EMPLID = A1.EMPLID
    AND A.EMPL_RCD = A1.EMPL_RCD
    AND A1.ROWSECCLASS = 'BYUDPUSA'
    AND ( A.EFFDT =
        (SELECT MAX(A_ED.EFFDT) FROM PS_EMPLOYEES A_ED
        WHERE A.EMPLID = A_ED.EMPLID
          AND A.EMPL_RCD = A_ED.EMPL_RCD
          AND A_ED.EFFDT <= SYSDATE)
    AND A.EFFSEQ =
        (SELECT MAX(A_ES.EFFSEQ) FROM PS_EMPLOYEES A_ES
        WHERE A.EMPLID = A_ES.EMPLID
          AND A.EMPL_RCD = A_ES.EMPL_RCD
          AND A.EFFDT = A_ES.EFFDT)
     AND A.SUPERVISOR_ID IN (SELECT B.EMPLID
  FROM PS_PERS_DATA_EFFDT B, PS_PERS_SRCH_QRY B1
  WHERE B.EMPLID = B1.EMPLID
    AND B1.ROWSECCLASS = 'BYUDPUSA'
    AND ( B.EFFDT =
        (SELECT MAX(B_ED.EFFDT) FROM PS_PERS_DATA_EFFDT B_ED
        WHERE B.EMPLID = B_ED.EMPLID
          AND B_ED.EFFDT <= SYSDATE)
     AND B.YEARS_OF_EXP > 10 )) )
```

Union

When a join is used in a query, the resulting output is from that data which the selected records have in common. Unions allow a query to define multiple SELECT statements and execute them at the same time and to consolidate the results into one output. Unions can be used to combine records that have no fields in common and to retrieve similar data from unrelated records in one query.

A Union is two (or more) separate select statements that are brought together in the same query. There are three rules you must follow when using a Union- both select statements must have:

- Same number of fields for selected outputs
- Same field types
- Same field order

When using a union, create the first SELECT statement then click on the New Union hyperlink at the bottom of the Query Manager pages. This action allows the creation of an additional SELECT statement.

Keep in mind the following points when using Unions:

- The sorting and headings are established in the first select statement
- You cannot retrieve the long or short translate description in a Union

- Unions are automatically Distinct
- You can have more than two SELECT statements; security options can limit the number of Unions a user can create within one query
- Expressions and prompts appear at the bottom, once
- It is difficult to tell at first which prompts or expressions you are looking at
- Each SELECT statement can have its own criteria

As an example, let's say that you want a query that returns the codes and descriptions of all Deduction Codes as well as all Job Codes. This isn't very practical, but it serves as a good example of the use of Union. The only way to get this query is to use two separate SELECT statements and join them with a Union.

Create a new query using the JOBCODE_TBL record

- Select the following fields: **JOBCODE, DESCR**

Previewing this query shows us that there are 2639 job codes currently in the system.

- Select the Fields tab and click on the New Union hyperlink.
- Next you will be taken to the Record tab where you will identify the record for the next SELECT statement. Search for the DEDUCTION_TBL record, and click on the Add Record hyperlink.
- Select **DEDCD** and **DESCR** in that order

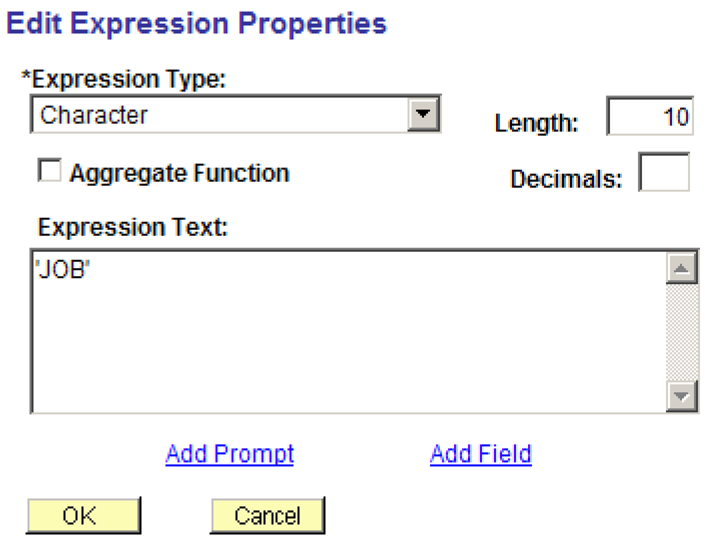
Now when we preview the query we get 2749 rows back, which now include all Deduction Codes and Job Codes. The problem, however, is now we can't tell apart the two. We will now add a literal to our query that specifies what data each row contains.

Using Literals as Placeholders

Literals can be used as placeholders, or pieces of text. Earlier in this training literals were used in an expression to concatenate the pieces of an address into one field. Literals are also useful in creating complex Unions. When a Union is created, it is required that each SELECT statement have the same number of fields (The fields don't have to be the same). This is where applying literals as placeholders comes in handy.

To help in our previous query, we will create a third row of data that will be populated with JOB if the row contains a job code and with DEDUCTION if the row contains a deduction code.

- Select the Expressions tab and note that you are **Working on selection:** Top Level of Query
- Click the Add Expression button and create an expression with an expression type of Character and a length of 10. Enter 'JOB' in the Expression Text box. The result should be as follows:



Edit Expression Properties

*Expression Type: Character

Length: 10

☐ Aggregate Function

Decimals:

Expression Text:

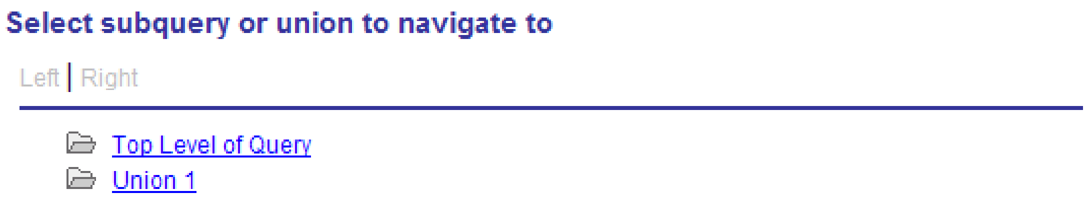
```
'JOB'
```

- Click **OK** and then click the Use as Field hyperlink
- Select the Fields tab and change the Heading Text to the new field to **Source**

The sorting and headings are established in the first SELECT statement (A Table). These headings appear in the query after the Union is created.

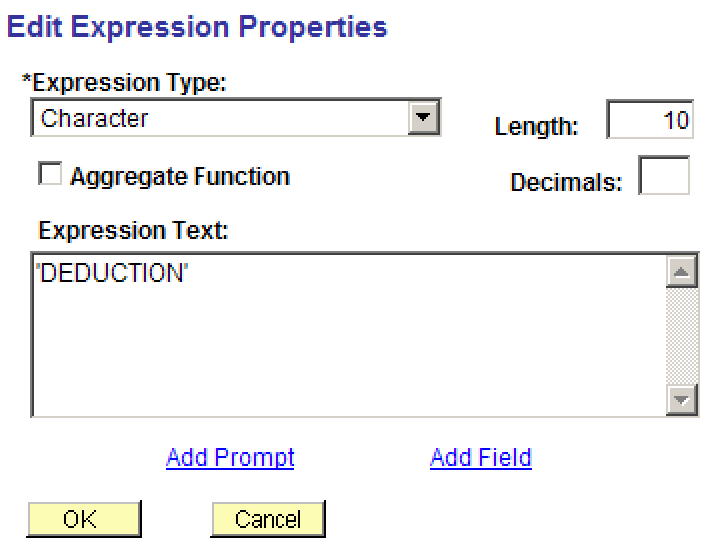
Now create a literal for the second statement:

- Select the Query tab and click on the Subquery/Union Navigation hyperlink



Select subquery or union to navigate to

- Click on the Union 1 hyperlink
- Select the Expressions tab and note that you are **Working on selection:** Union 1
- Click on the Add Expression button and create an expression with an Expression Type of Character and Length of 10. Enter 'DEDUCTION' into the Expression Text box. The result should be as follows:


Edit Expression Properties

*Expression Type: Character

Length: 10

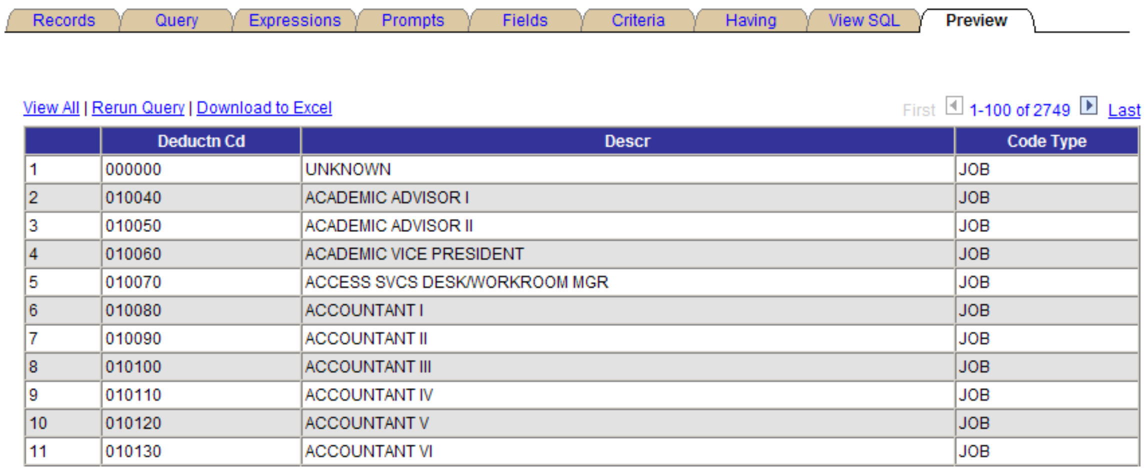
Aggregate Function

Decimals:

Expression Text:

```
'DEDUCTION'
```

- Click **OK** and then click the Use as Field hyperlink
- Preview the query, notice the new column, the results should be as follows:



	Deductn Cd	Descr	Code Type
1	000000	UNKNOWN	JOB
2	010040	ACADEMIC ADVISOR I	JOB
3	010050	ACADEMIC ADVISOR II	JOB
4	010060	ACADEMIC VICE PRESIDENT	JOB
5	010070	ACCESS SVCS DESK/WORKROOM MGR	JOB
6	010080	ACCOUNTANT I	JOB
7	010090	ACCOUNTANT II	JOB
8	010100	ACCOUNTANT III	JOB
9	010110	ACCOUNTANT IV	JOB
10	010120	ACCOUNTANT V	JOB
11	010130	ACCOUNTANT VI	JOB

Viewing Union SQL

Let’s take a look at some of features discussed earlier:

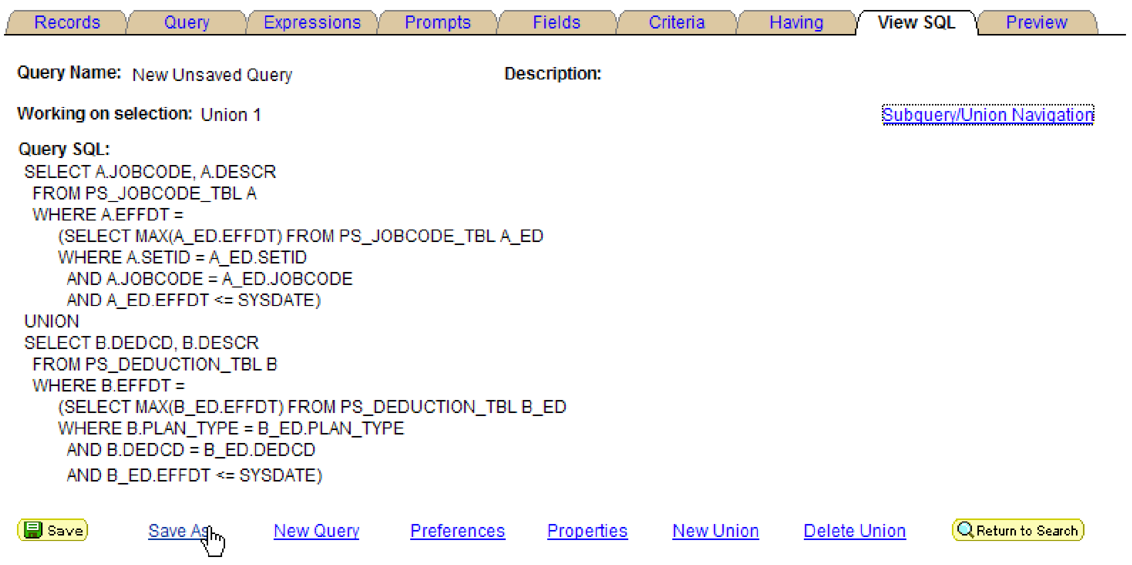
- Unions preserve unique rows
  - If both SELECT statements retrieve the same row, that row only appears once in the final query output. Unions remove duplicates based on all fields selected.

- Translate values (long or short description) cannot be displayed in a Union query
- You cannot order the results of **each** SELECT statement in the query.

   - To order the output, specify the order of the fields in the first SELECT statement

- Column heading names for the output come from the heading definitions specified in the first SELECT statement

The SQL should be as follows:



Query Name: New Unsaved Query **Description:**

Working on selection: Union 1

Query SQL:

```
SELECT A.JOBCODE, A.DESCR
  FROM PS_JOBCODE_TBL A
  WHERE A.EFFDT =
      (SELECT MAX(A_ED.EFFDT) FROM PS_JOBCODE_TBL A_ED
      WHERE A.SETID = A_ED.SETID
        AND A.JOBCODE = A_ED.JOBCODE
        AND A_ED.EFFDT <= SYSDATE)
UNION
SELECT B.DEDCD, B.DESCR
  FROM PS_DEDUCTION_TBL B
  WHERE B.EFFDT =
      (SELECT MAX(B_ED.EFFDT) FROM PS_DEDUCTION_TBL B_ED
      WHERE B.PLAN_TYPE = B_ED.PLAN_TYPE
        AND B.DEDCD = B_ED.DEDCD
        AND B_ED.EFFDT <= SYSDATE)
```

Outer Join

When a join is used in a query, the resulting output is from the data which the joined records have in common. With an Outer Join you can join two records and force a row to be returned even when there isn't a match between records.

Using Outer Joins

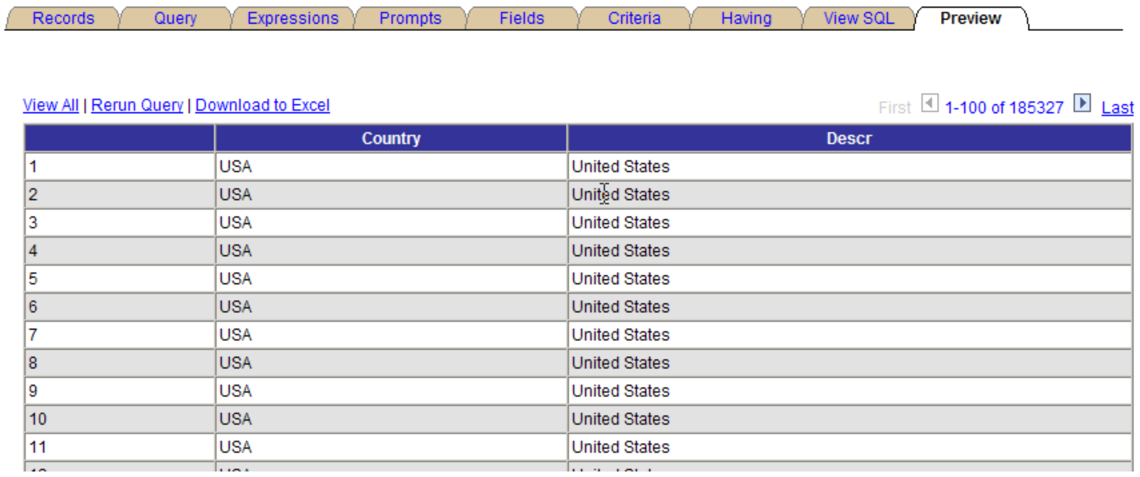
In an Outer Join, all rows of data are included from the master table (first record added to the query). Matching rows from the subordinate tables are also included. You must be aware of you database and the syntax it recognizes to perform an outer join. You will need to use different syntax according to the different database platforms. This material is presented according to Oracle recognized syntax.

Outer Joins combine concepts from Record Hierarchy joins and Subqueries. Remember that a record hierarchy join returns rows where the fields are in common from different table (Where A.Field1 equals B.Field1), and a subquery can return rows that don't exist in a secondary table.

An example of an Outer Join: We want a table that lists all countries and the employees from those countries. Countries without citizens that have worked at BYU should also be included in our output.

Create a New Query using the COUNTRIES_TBL record

- Select the **COUNTRY** and **DESCR** fields
- Join the table with the PERSON_ADDRESS table
- When preview, the query returns the following results (185.327 results returned):


Records
Query
Expressions
Prompts
Fields
Criteria
Having
View SQL
Preview
View All | Rerun Query | Download to Excel
First 1-100 of 185327 Last

	Country	Descr
1	USA	United States
2	USA	United States
3	USA	United States
4	USA	United States
5	USA	United States
6	USA	United States
7	USA	United States
8	USA	United States
9	USA	United States
10	USA	United States
11	USA	United States

We will see that not all countries are listed in this query when we add the Outer Join. Only those countries that have employees are included in the output. When the two records are joined, only the information they have in common is included in the output. An outer join will also let us include information that the records do not have in common.

Creating Outer Joins

Since we used Auto Join to join these tables, the join criteria can be modified. To perform an outer join, **you must use Auto Join** so that the join criteria can be changed. ***Modification to the Join Criteria must follow the recognized syntax based on the database being used***.

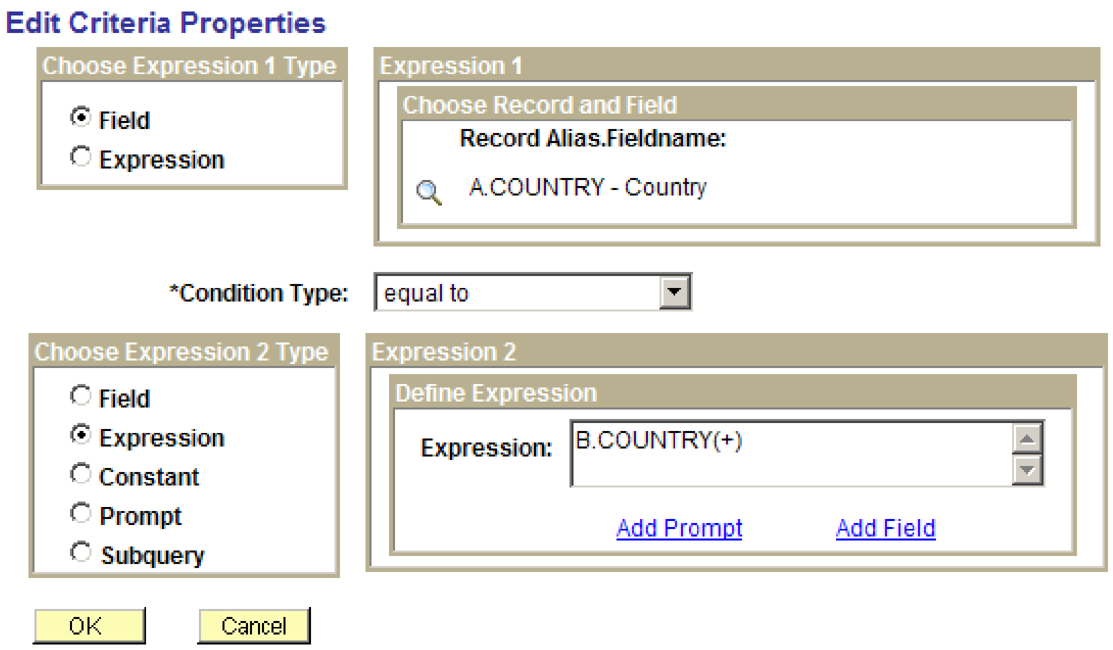
Oracle Syntax

Edit the Join Criteria and modify Expression 2 for each row of join criteria. Change the Expression 2 Type to an Expression. If you are accessing an Oracle database create an Expression with the original field name followed by a plus sign. The SQL syntax for the Join Criteria should read as follows: A.FIELD1 = B.FIELD1(+). *The recognized syntax for the Oracle database is an open parenthesis, a plus sign and a closed parenthesis [(+)] to the right of the field* ***from the subordinate record***.

Here are the steps to transform the original query into an Outer Join:

- Select the Criteria tab, edit the first row of Join Criteria (COUNTRY)
- Change the *Choose Expression 2 Type* to **Expression**
- Enter the following in the Expression box: B.COUNTRY(+)

The following is how the *Criteria Properties* should appear:


Edit Criteria Properties

Choose Expression 1 Type: [✓ Field] Expression

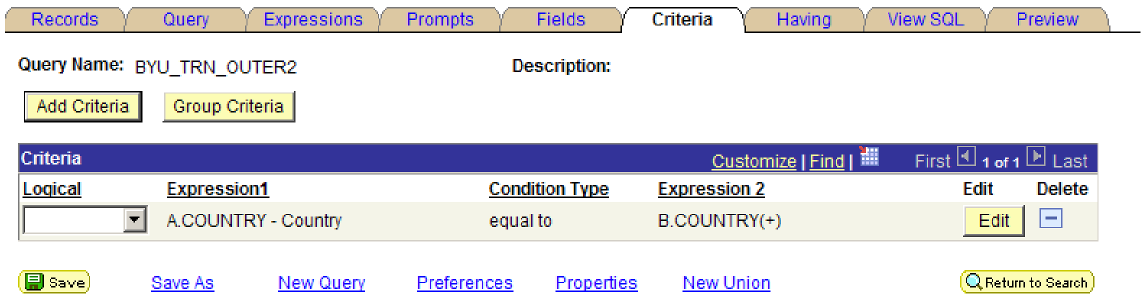
Expression 1 – Choose Record and Field – Record Alias.Fieldname: A.COUNTRY - Country

*Condition Type: equal to

Choose Expression 2 Type: Field [✓ Expression] Constant Prompt Subquery

Expression 2 – Define Expression – Expression: B.COUNTRY(+)

The following is how the Criteria Page should appear:

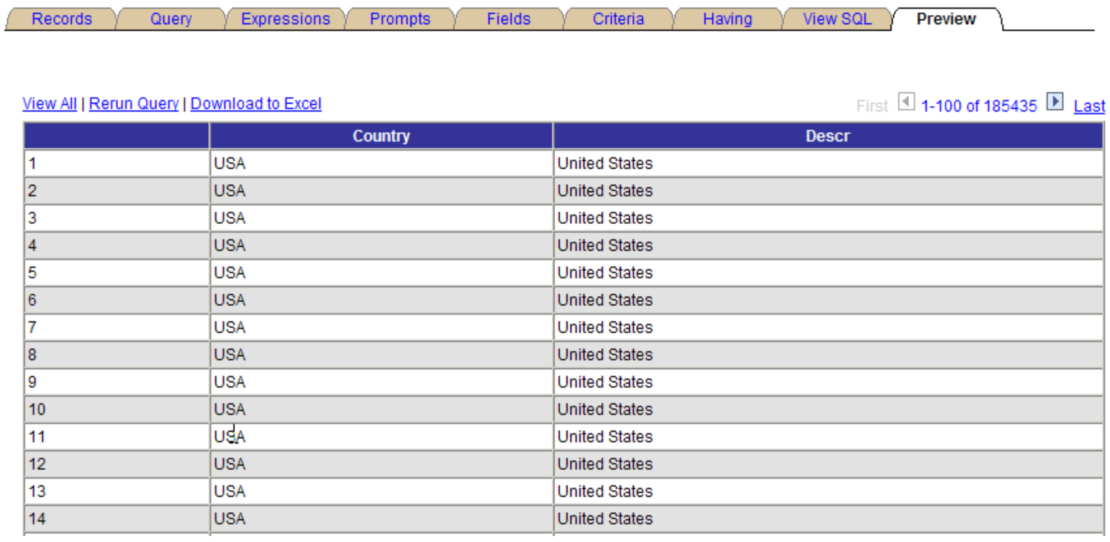


Query Name: BYU_TRN_OUTER2 Description:

Criteria

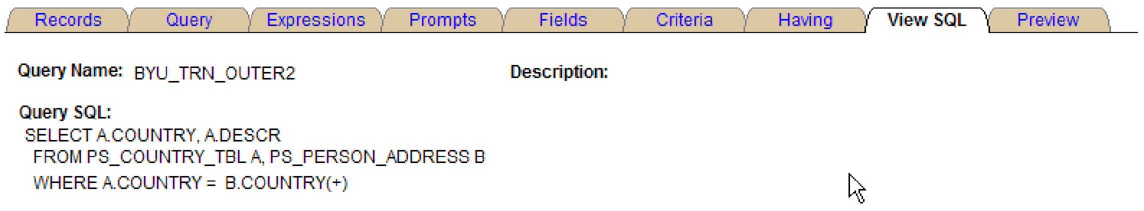
Logical	Expression1	Condition Type	Expression 2	Edit	Delete
	A.COUNTRY - Country	equal to	B.COUNTRY(+)	Edit	⊟

Previewing the query now gives us more results, which are the country rows without any employees (185,435 results returned compared to 185,327).


Records
Query
Expressions
Prompts
Fields
Criteria
Having
View SQL
Preview
View All | Rerun Query | Download to Excel
First 1-100 of 185435 Last

	Country	Descr
1	USA	United States
2	USA	United States
3	USA	United States
4	USA	United States
5	USA	United States
6	USA	United States
7	USA	United States
8	USA	United States
9	USA	United States
10	USA	United States
11	USA	United States
12	USA	United States
13	USA	United States
14	USA	United States

The SQL should look like the following:



Query Name: BYU_TRN_OUTER2 Description:

Query SQL:

```
SELECT A.COUNTRY, A.DESCR
  FROM PS_COUNTRY_TBL A, PS_PERSON_ADDRESS B
  WHERE A.COUNTRY =  B.COUNTRY(+)
```

Using a Union and a Subquery as a *Workaround* for an Outer Join

If you do not have security or rights to use an Outer Join, or if you are unaware of the recognized syntax for the database you are accessing, you cannot create an Outer Join. However, you can achieve the effect of an Outer Join by using a subquery and a union. The first select statement could retrieve countries that don’t have any employees at BYU (subquery) and the second select could retrieve countries that do have employees at BYU (Join). Create a Union for these two select statements and achieve the same results as an outer join. It is best to avoid using Outer Joins or the workaround unless absolutely necessary as they are very taxing on the database server.