PS QUERY HELPFUL HINTS
FOR HRMS & PS HR TABLES COMMONLY USED FOR REPORTING

A record may consist of multiple tables. A single panel does not necessarily imply that the data on the panel is stored in a single table. Because PeopleSoft is a relational database, data is stored in multiple tables.

NOTE: Depending on your individual profile, you may not have access to all of these tables or records:

TABLES: The following are the main tables used to query HR related data:

Human Resources Information/ Personal Data:
PERSONAL_DATA, PERS_NID

Employment Data:
EMPLOYMENT-- general employment data, such as various service dates, etc.

DEPT_TBL --This is one of foundation tables in HR. You will rarely query against it directly; more likely you will join to it to get the description or short description. Department codes in PeopleSoft are different from what was used in HRS/Legacy, and department data is effective dated. For information and mapping purposes, the corresponding legacy/HRS department code is found in the Short Descr field of the Department table if the department existed in legacy before conversion.

JOBCODE_TBL --This is also a foundation table. This table holds job codes, descriptions and related data such as FLSA classification, pay plan, grade and structure information. Details on pay plan and structures are kept in other tables. The job codes that existed in legacy/HRS are the same, except those that have been added since conversion.

Job Data:
JOB-- Job (roughly equates to assignment records in legacy). Contains job records along with other data. Effective dated someone's job history will be stored in this table. Remember, Empl Records refer to an employee's job record(s). The existence of multiple Empl Records means that they have had or are currently holding multiple jobs Empl Records start with 0, and their benefits are tied to Empl Record #0. In most cases, Empl Record 0 implies their 'benefits-eligible' record.

Note: We did not convert historical records. If the employee had an active job record/balance at some time in 2001, they were converted to PeopleSoft.

Benefits Data:
BEN_PROG_PARTIC – Stores the Benefit Program to which the employee is assigned.
LEAVE_PLAN – Stores enrollment in leave plans.
HEALTH_BENEFIT – Stores enrollment in health insurance plans
SAVINGS_PLAN -- Stores enrollment in TDA, ORP and Deferred Compensation
LIFE_ADD_BEN -- Stores enrollment in life insurance plans.
RTRMNT_PLAN – Stores TRS enrollment.
DISABILITY_BEN – Stores enrollment in disability plans.

Tax Data:
FED_TAX_DATA – Contains federal tax data, including nonresident alien data

STATE_TAX_DATA – Although there is no state withholding in Texas, this record is necessary to maintain state of residence and unemployment jurisdiction.

Payroll Data:
Payroll processing check data (records used by the payroll process to store check history)

- PAY_CALENDAR
- PAY_CHECK
- PAY_LINE
- PAY_CHECK
- PAY_EARNINGS
- PAY_OTH_EARNS
- PAY_DEDUCTION
- PAY_TAX

Payroll funding distribution data (records that tie funding to payroll actuals):

- PAY_ERN_DIST
- PAY_DED_DIST
- PAY_TAX_DIST

Other employee level payroll data:

- GENL_DEDUCTION
- GENL_DED_CD
- ADDL_PAY_EFFDT
- ADDL_PAY_DATA
- ADDL_PAY_ERNCD
- DIRECT_DEPOSIT
- DIR_DEP_DISTRIB

Position Data:
POSITION_DATA—identifies position data for each business unit.

Tenure Data:
EG_TENURE Data

HELPFUL HINTS:

Run Timing:
After initially building a query, the first time it is run will be slower than on subsequent attempts. The reason is the

1) The query may be saved, but is not compiled until the first time it is ran
2) Behind the scenes, Oracle maps out an execution plan and saves it in memory, so it will usually run faster on subsequent attempts.

Joins:
Think about these carefully. If at all possible, join on related records. Related Records are records/tables that have their relationships/joins established on the database side, rather than defined in the query itself. At run time, joins on related records generally outperform those made in a query. When pulling in additional records for a multiple table query, PS will usually detect, and then default-in a join condition for you--they usually work. Keep in mind that the fewer tables that need to be joined, the better.

Common tables:

Use the Employees table whenever possible. Queries using this table will generally have better performance for several reasons. This tables has name, Empl ID, empl records, job title, job code, dept title, dept code, location code, comp rate, comp frequency, position number, standard hours, etc., data for active employees. Two fields that it does not have are Empl Class and FTE. This table is not a transactional table, meaning that records are not actively written to it throughout the day. Rather, it is refreshed nightly on process scheduler. There may be a one-day lag time for new data to hit the table, but for most reporting this is acceptable. Other important tables are JOB, PERSONAL_DATA, POSITION_DATA, EMPLOYMENT, EG_TENURE_DATA

Duplicate records:
In order to eliminate duplicates, you may have to try several methods. You may have to join on more than one field. To eliminate perceived duplicates for the same Empl ID, pulling Empl Rec ID = 0 in the criteria grid usually works. The existence of many duplicates in the query record set may mean that you inadvertently performed a Cartesian join. Cartesian joins are not good and must be avoided. Cartesian joins mean that every column in each table will try to join with every column in all other tables. Avoid using the DISTCINCT property setting in PS Query. Except in the case of a very small record set, the DISTINCT property will dramatically decrease the performance of the query. Note: At times, the existence of future dated Job rows working in conjunction with the query security records cause duplicate rows to be returned. In this case, the only way to get rid of the duplicates may be using the Distinct feature.

Benefits Eligibility:
A common criterion to filter by is Benefits Eligibility. A value of "Reg" on the Reg Temp field of the JOB record, means "eligible for full benefits". Not that they will receive full benefits, but that they are eligible. A value of "Temp" means that the employee is eligible for something other than full benefits. insurance-eligible employees are in the system as "Temp".

To query benefits-eligible employees pull the following fields into the criteria: Reg/Temp = 'R' and Empl Rec = 0 (zero).

Views:
Views are saved queries that can be referenced as record sets in other queries. They end with a suffix of in *_VW as the naming convention. Views are for various data selection and restriction purposes. End users and most technical staff do not the ability to create views. However, if a view is available as a record source (i.e., you can see it in the PS Query grid), then you can use it as a record source in a query. However, because they are basically a saved query, there is a slight performance hit when joining to one.

Intermediate Reporting Levels:
College/Division (intermediate reporting levels)--it's more difficult to query out these levels. Find out the PeopleSoft Department ID of your reporting level, and use the "in-tree" operator. This allows you to query 'down' the tree, to pull everyone in your college/division/school. NOTE: Some campuses may restrict your ability to view data outside of your immediate department or college; see HR for details.

General hints:
Build your queries incrementally. If dealing with multiple criteria, pull them in one at a time, and re-run the query after each criterion is added. Set temporary criteria of a single department, rather than the whole campus. This allows you to control the output and identify whether the query is working correctly before you run it against your entire target population.

- *Value constants*: When selecting constants for a criteria grid, and you don't know the values for that field, click the "F-4" field, and it will give you a list of valid values.
- *Columns*: Only pull the columns into the query that you need. Not all tables are populated in PeopleSoft, and not all fields in all records/tables are populated. Contact the HR Project Staff or your HR Department if you have a question the mapping and legacy data conversion for a particular field.
- *Sorts*. If it's a query with a large result set, avoid sorts at run-time within the query grid itself. Sort in Excel. Adding a sort at run-time in PS Query can have a substantial negative impact on processing speed.

- *Functions*: Functions slow down queries. Use functions sparingly, and try to use only the aggregate functions that are built into PS Query. Using Oracle functions passed-through PS Query invalidates the index. The performance loss may be negligible, but your query does take a performance hit.
- *PS Query and security*: PS Query passes through row-level security, which restricts whose data you can see. If you don't normally have access to view data through the panels, you will be subject to the same restrictions in PS Query.
- *Shaded columns*: Need to get rid of those annoying shaded columns?
    - After logging-in to PeopleSoft, but before running a query to Excel, select "Edit-Preferences-Configuration" from the main menu.
    - Select the "NVision" tab. Under the field for 'space between columns', key in "0.0".
    - Click the "Apply" button, and then "Ok".
    - Note: if you completely log-out of PeopleSoft, then you will have to set these configuration settings when you log-in again, as these are default settings pulled from the database server. When logging-in, it takes only a couple of extra steps, but when you are pulling the query results into Excel, it can save you a significant amount of time.
- *The "in-tree" operator*: Where possible, avoid the "in-tree" operator when querying through the security tree. Three aspects of security--both user and class profiles, along with the Dept Security tree, affect your query. If you use it, expect a performance hit. If you can query by department only, and not a reporting level such as by college or division, you will probably obtain a better response from your query.
- *Table ordering*: The order in which you bring tables or records into the PS Query grid is important. PS Query and Oracle have build-in optimizers that make the code more efficient, but in most cases it helps the query run better if tables that are expected to return the fewest rows should be added last in the FROM clause. The SQL statements that PS Query builds are evaluated from the bottom up. However, it will not correct a bad (or invalid) join.

?? *Multiple criteria values*: E.g., you need a query that pulls more than a single department, but not all of them. When querying for multiple values, there are several ways to do it: the "in-list" operator, multiple "OR" statements, using a UNION query, using the “like” operator, or using a sub-query.

- The "In-list" operator—recommended
- Multiple OR clauses
- The UNION operator (usually fastest, but not recommended because of increased coding complexity.
- The “like” operator may be appropriate if a large number of values are targeted, such as DEPTID like ‘C%’ will pull all Clearlake departments.
- A Subquery is usually the slowest. The subqueries have to be evaluated first, and must be compiled before the parent query runs.

?? *Calculated fields*. Avoid performing calculations on key fields. If possible, avoid pulling in too many key fields, especially in a smaller query that will pull a limited number of records. If it's a translate field, you can set the description from within the field grid in PS Query. Don't do a join with the Translate table. Instead, within the field grid of PS Query, where the output columns are defined, double-click on the XLAT column. You will get a dialog box, which will give you the option of pulling the description for the XLAT field, or the value itself. If you can, pull the value itself.

?? *Expressions*. Use expressions only when necessary. Expressions are not indexed and will take longer to process.