SQL Joins

**SELECT Statement with a JOIN: JOINS** are used when it is necessary to combine fields from two or more tables in one row of the query results. The tables must contain at least one common field through which the join can be generated. Designators (or aliases) must be assigned in order to identify each selected field name with its corresponding table. Joins can be executed using either point-and-click or “free form” query building methods.

**Sample JOIN QUERY:**

**Scenario:** A Department Director wishes to see a report that shows how many current Twin City Campus students having non-resident, non-citizen status transferred to the University with recognized undergraduate level credits. The records are to show the University from which the student transferred and should be sorted by gender and campus code.

```
SELECa.TRANS_CRS, a.RES_CIT_STA, a.SEX_CD, b.TRANSCR_CD,
b.PREVIOUS_COLLEGB
FROM SYSADM.PS_DWSA_RRDB_STUDENT a, SYSADM.PS_DWSA_RRDB_TRANSFER b
WHERE b.FILE_NO = a.FILE_NO AND (b.TRANSCR_CD = 'G' AND a.TRANS_CRS > 0)
ORDER BY b.PREVIOUS_COLLEGB, a.SEX_CD
```

**WHERE clause:** indicates the common fields on which the two tables are joined.

**Designator (alias):** identifies a field with the table from which it comes. Designators precede a field name and are followed by a period (no spaces) for all clauses except the FROM clause. In the FROM clause, designators follow the table name separated by one space.
SQL Unions

**SELECT Statements with a UNION:** **UNIONS** are used when it is necessary to combine two or more SELECT statements within the same query. Two queries can have their results grouped together simply by typing the two queries with the word UNION between them. The execution of a UNION displays both query results in one table. UNIONS can be executed using the "free form" method of query building.

In order to form a UNION, the following conditions must be met:

1) Both SELECT clauses must request the same number of fields (or columns)
2) Fields in both queries must mirror each other exactly in terms of order listed, data type, and length. For example, both obj and rev are listed as the fourth field and are four character alphanumeric fields.

Five fields have been selected in each clause. All fields appear in the same request order with relation to data type and length. See *note below.*

```
SELECT DISTINCT a.obj, a.objrev_name30, b.org, b.tot_encumb, b.tot_exp
FROM dwfs_coa_sobjrev a, dwfs_sum_obj_last_clsd_pd b
WHERE a.obj = b.obj AND b.area = '766' AND b.org = '1001'
UNION
SELECT DISTINCT a.rev, a.objrev_name30, b.org, b.unrec_rev, b.tot_rev
FROM dwfs_coa_sobjrev a, dwfs_sum_rev_last_clsd_pd b
WHERE a.rev = b.rev AND b.area = '766' AND b.org = '1001'
ORDER BY b.org, a.obj
```

**ORDER BY clause:** sorts rows of data (or records) returned from smallest to largest in number or alphabetically. If used, ORDER BY must reference info from the first select clause and is always the last clause to appear in a select statement.

**Results:** The column headings are determined by the field names in the first select statement.

<table>
<thead>
<tr>
<th>obj</th>
<th>objrev_name30</th>
<th>org</th>
<th>tot_encumb</th>
<th>tot_exp</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>will include</td>
<td>will include</td>
</tr>
<tr>
<td><em>will include</em></td>
<td><em>will include</em></td>
<td></td>
<td>unrec_rev</td>
<td>tot_rev</td>
</tr>
<tr>
<td>rev</td>
<td>information</td>
<td></td>
<td>information</td>
<td>information</td>
</tr>
</tbody>
</table>

Fields requested in the second select statement will not appear as column headings. However, this field information will be included in the query results.

*Note:* Obj and rev are both Type A (alphanumeric) fields of four characters. Tot_encumb and unrec_rev are both Type N (numeric) fields of 11.2 characters (eleven characters with the last two after a decimal).