PART 1

PeopleSoft Basics

1 *PeopleSoft HRMS: The Basics*

The PeopleSoft HRMS database has over 5,000 tables. Trying to navigate through this vast system takes both time and patience—and a basic understanding of where to begin. This chapter covers the basic rules of the PeopleSoft database, enabling you to obtain accurate results. Successive chapters go into each table in detail.

If you have worked with PeopleSoft before, chances are you used the online application, with menus and panels for data entry. The online system lets the user search for existing data and insert new data into the database. We will avoid panels in this book (except for references to panels as a resource), since they mask the data and make it difficult to retrieve results.

Instead, this book looks at the underlying database tables. Our goal is to retrieve accurate data from PeopleSoft by hitting the database directly using SQL. This chapter is an introduction to the PeopleSoft database and its unique requirements. It will introduce you to the database and provide the basic rules of the road.

We will look at how the database is organized, functionally and technically. We will look at the tools provided by PeopleSoft for finding the appropriate table, and review the core PeopleSoft tables. We will consider the common requirements that pertain to database queries, such as keys, table links, and required fields. All of this information is technical in nature, but essential if you want to understand how data is stored and processed in PeopleSoft.

An Introduction to the PeopleSoft HRMS Database

All PeopleSoft products attempt a balance between functional requirements (what works best for humans) and relational database requirements (what works best for computers).

This issue is especially relevant to human resources requirements. Human resources professionals must be customer-oriented, and this requires extraordinary flexibility. Casual statements such as "We have to cut 5,000 checks by tomorrow morning!" or "We just bought XYZ corporation and need to add 30,000 employees to our system!" can bring an entire human resources department to tears. PeopleSoft's role is to simplify these seemingly drastic changes in company structure. Functional flexibility, the ability to perform human resources feats in a single click, is a cornerstone of the PeopleSoft product.

On the other hand, PeopleSoft is a software product with many programmers behind it. These programmers work together, writing separate components and improving technical benchmarks, like access time and storage capabilities. The companies who purchase PeopleSoft also typically employ programmers who modify and maintain the product. The "user," then, could be a programmer or a VP of human resources. The computer and human effort required to maintain an infinitely flexible system becomes overwhelming.

To prevent a tangled web of data, PeopleSoft stores data using a database organizational concept called "data integrity." The philosophy of data integrity is simple: a database stores data efficiently by storing it once, and only once. Duplication equals inefficiency and creates a chance that data will be changed in one place and not another. The efficient storage of data assures its integrity. It is a mantra followed by database programmers, and PeopleSoft is no exception.

Data integrity keeps the database happy and allows it to grow as functionality is added to PeopleSoft. Yet it divides the human resources data we need into over 5,611 tables. With each new PeopleSoft release, data is spread thinner and thinner, to more and more tables. For version 7.5, for example, PeopleSoft took the Social Security Number and made three new tables to store it. Now PeopleSoft can store different forms of the SSN for different countries with different validations. As the functional flexibility of the program grows, so does the size of the database.

Yet for a user trying to look up information in a database, a larger database has the opposite effect. Users need to find data when and where they need it. Human time is more expensive than computer time.

The fact is that most queries against the database require at least a 7-table join—for basic information on an employee like Social Security Number, name, hire date and department name. Knowing where to find each element and how to link them together is not common knowledge, and is difficult to derive. It is critical to understand which tables are the core sources of information.

This book is a guide to a subset of about 75 tables that contain the most commonly used data in PeopleSoft HRMS. These tables are not all essential, but they are accessed often and provide the foundation for the database. By exploring these central tables, we can get a better idea of how the PeopleSoft database is organized.

TYPES OF TABLES

There are six basic types of tables that we will discuss. As an overview, let's look at each type of table. You can even perform a quick SQL statement to look at the contents of the example table names provided. Simply type (for example):

```
SELECT * FROM PS_PERSONAL_DATA
```

Each table type is explained in detail in a separate chapter, except for application tables, which are not used in reporting.

Base Table (i.e., PS_PERSONAL_DATA, PS_JOB)

A base table is the place where nearly every query starts. These tables store information about an employee and contain data about the employee. A base table stores live data that is continually changing. The table could store information about employees, their dependents, their earnings, taxes, deductions, or benefits. In short, these tables hold the real data, the non-static data. Base tables are not distinguished with a prefix or suffix; they are named according to their function. See sections 2, 3 and 4 for specific table information.

Control Table (i.e., PS_EARNINGS_TBL)

A control table contains a short list of values that classify and categorize. For example, a table that contains all of the possible earnings codes (regular, bonus, overtime, etc.) is a control table, whereas the table that contains the actual earning amounts is a base table. A table listing all department codes or state names is a control table. Each control table has a key (the code) which ties to a field on a base table. Some control tables are larger than others. Often the department and jobcode tables are thousands of rows long, and change often. Control tables are usually identified by a suffix of '_TBL'. See Chapter 3 for more information on control tables.

Control tables are also commonly known as 'lookup' or 'prompt' tables.

Views (i.e., PS_BENEFITS_VW)

Views are timesavers; they are the result set of an SQL statement. For example, the benefits view table takes fields from several tables, links them together correctly, and presents the result as a new table. Views link to original tables (base or control), so no data is duplicated or out of sync. Views are usually identified by a suffix of '_VW'. See Chapter 4 for more extensive information on views.

Reporting Tables (i.e., PS_EMPLOYEES)

In an attempt to appease those toiling away, searching for the location of basic employee data, PeopleSoft created three tables that contain the most-often-used human resources fields. These tables are similar to views, but are not dynamic. Their data is only current after a program is executed every night. Their chief benefit is performance. Instead of joining 10 tables every time you look something up, the tables are joined once at night and then used throughout the next day as a single table. PS_EMPLOYEES, PS_BEN_PER_DATA, and PS_BEN_PLAN_DATA are reporting tables. See Chapter 4 for more information on reporting tables.

Application Tables (i.e., PSTREENODE)

The PeopleSoft application stores application rules and definitions in application tables. Occasionally these tables temporarily store data in the middle of a process. With few exceptions, these tables store data that is not relevant to the organization. Most of these tables are not discussed in this book since they contain application data, not HR data. System tables often do not include an underscore after the 'PS' prefix.

The Non-Table: Sub-Records

PeopleSoft is intent on making sure no data or effort is ever duplicated. For this reason, they created sub-records. Sub-records are like a mini-table, since they are a definition of a group of fields. A useful example is the address subrecord (ADDRESS_SBR) shown in Table 1-1.

Table 1-1 Example sub-record: ADDRESS_SBR

Column number	Field name
1	Country
2	Address Line 1
3	Address Line 2
4	Address Line 3
5	Address Line 4
6	City
7	Number 1
8	Number 2
9	House Type
10	County
11	State
12	Postal Code
13	Geographical Code
14	In City Limit

This is the definition of an address in PeopleSoft. Addresses, however, are used all over the place. There is the address of the employee, the work address, the building address, the paycheck address, and so on.

The sub-record is a single definition that is used in many different records. Addresses are consistently stored, and if a change is required, such as an extra digit added to the postal code definition, that change needs to be made in only one place—that is, from an application standpoint. From the perspective of the database, the sub-record does not exist (that's why you can't find it). As far as the database is concerned, the actual fields from the sub-record reside on the table. So when PeopleSoft's definition of a table contains ADDRESS_SBR as one of the fields, there will be 14 separate fields on the database definition and no mention of ADDRESS_SBR.

As a rule, consider the sub-record as part of the table it is found on. To get the city, you can't query the ADDRESS_SBR. You must query the table the sub-record is on.

THE CORE BASE TABLES

Now that we have looked at the different table types, let's look at the major base tables in PeopleSoft. These are the core tables in the database. Almost all PeopleSoft processes touch one of these tables, and chances are you'll need to do the same in nearly every query. Since you will almost always be starting with one of these tables, learn their differences in order to choose the correct one. Each table contains every employee no matter what their status (active, inactive, on leave, etc.).

The relationship between these three tables is simple (Figure 1-1). The PS_PERSONAL_DATA table has one row per employee—it is *the* parent table. Next in line is the PS_EMPLOYMENT table, which stores one row per employee job (if employees can have more than one job at a time in your organization). Finally, the PS_JOB table stores all of the status, compensation, and position history for the employee on many rows.

- **PS_PERSONAL DATA**. Stores current and historical information on the employee's home address, SSN, and personal history such as birth date. Choose this table when you need personal (and sometimes confidential) data. Remember, everyone in the database (actives, terminated employees, and retirees) is on this table. Each person has exactly one row. See Chapter 6 for information on keys and field definitions.
- **PS_EMPLOYMENT.** Contains a general record of the employee's employment, with information such as hire date, years of service, and EEO categories. Choose this table when you're looking for a

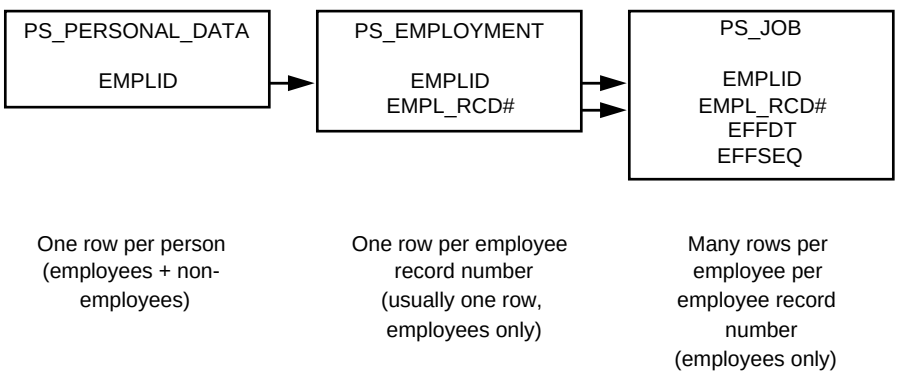


Figure 1-1 Relationships between PeopleSoft HRMS core tables

particular date. All "set-in-stone" dates, like hire date and rehire date are stored here. Just like PS_PERSONAL_DATA, everyone in the database is on this table. See Chapter 12 for information on the specific fields found on PS_EMPLOYMENT.

- **PS_JOB**. Stores current and historical information about an employee's job, such as status, compensation, promotions, and company categories. All history for this information is stored here, too. Choose this table when you need a list of employees who have a certain status. Typically users look for employees who are "A, L, P," or Active, On Leave, or On Paid Leave. Each employee will have several rows. See Chapters 7 through 10 for information on querying the PS_JOB table for different kinds of information.

Using the Application Designer To Find Table Definitions

Although data is rarely duplicated in PeopleSoft, if you look for a single field such as deduction code, thousands of references will appear. Which is the "real" field?

It depends on what you are looking for. Are you looking for a list of possible deductions, a list of deductions an employee receives, or a list of deductions that were taken from a paycheck? Each of these is in a different location, but they all share the same field name. There is no foolproof way to tell—even educated guesses are difficult.

Once you have a good idea of the piece of data you want, you must find where it is truly stored. The application designer is useful for tracking down a piece of data in PeopleSoft. You can open up tables, explore the definitions of fields, and trace the links between panels and database tables.

Querying Tables

The Application Designer can help you in your search for the perfect table. Not only does it provide an easy way to view table definitions, it also lets you quickly scan through the table names and cross-reference the occurrences of a field.

Before looking at how the Application Designer can assist, there are critical terminology changes to be aware of. First, in the Application Designer, database tables are called "records." This can be confusing, since to a database, a record is a row on a table. When PeopleSoft talks about records in the Application Designer, however, they are speaking of tables. In PeopleSoft, a record = a table.

Second, tables in PeopleSoft are referred to without the 'PS_' prefix. This prefix is masked from the user of Application Designer. Type it in, and you will find nothing.

Let's say Karen is looking for tax data. She isn't sure what tables tax data is stored on. Where to start? First, she opens the Application Designer, and goes to "Open." She picks "Records," places her cursor in the Name field, and types "TAX" (Figure 1-2).

A list of 28 tables is returned:

TAXFORM_BOX	Taxform Box Definition
TAXFORM_BX_LANG	Related Lang for TAXFORM_BOX
TAXFORM_DED	Taxform Deductions
TAXFORM_ERN	Taxform Earnings
TAXFORM_FORM	Tax Form Form Parameters
etc.	

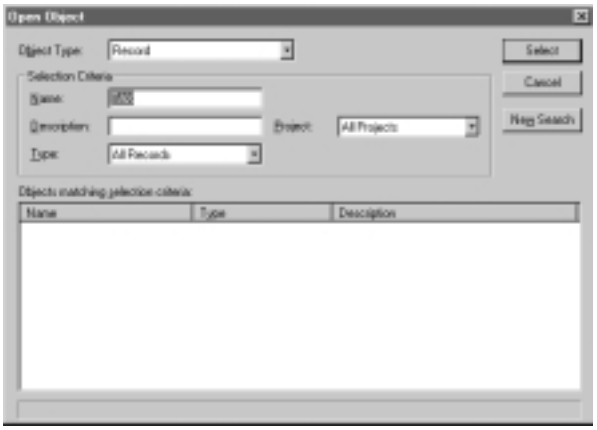


Figure 1-2 Finding a table using the Application Designer.

Essentially, PeopleSoft takes what Karen has typed and appends a wildcard to it. Application Designer sends the database a query that looks for all tables with the name "TAX%" where the "%" is the wildcard.

You can use this wildcard to your advantage. There are, for example, several tables in the database that have the phrase "TAX" in the middle of their table name. So STATE_TAX_DATA won't appear in Karen's original search. She now types "%TAX" and gets a dazzling array—127 tables.

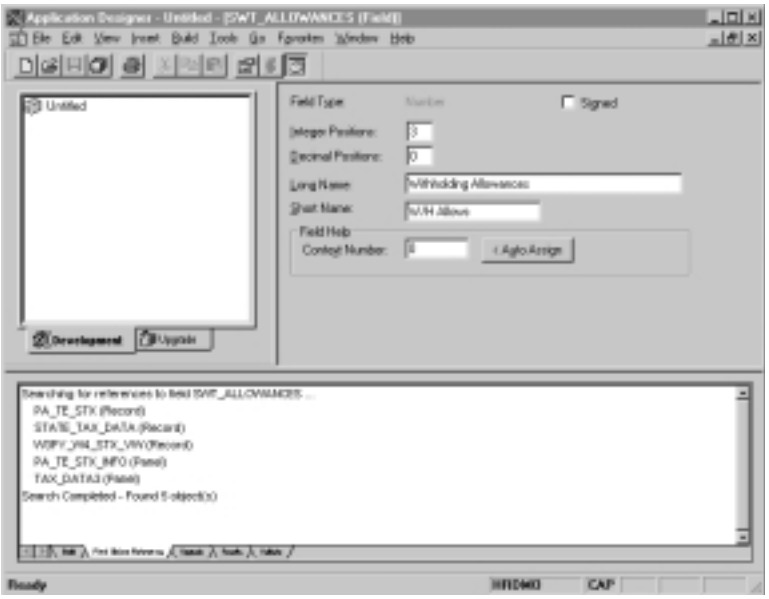
Cross-Referencing Fields

Another method that can quickly locate the data you need is to use the Application Designer's cross-referencing tool. For example, let's say Karen is looking for the number of withholding allowances an employee has in a particular state. She can search for the actual field name. First, she goes to "Open" and picks "Field." She types "%ALLOWANCE" and the following is returned:

ALLOWANCE_AMT	Allowance Amount
ALLOWANCE_DESCR	Allowance Description
ALLOWANCE_FLAG	Allowance Flag
FWT_ALLOWANCES	FWT Allowances
LWT_ALLOWANCES	Local Withholding Allowances
SWT_ALLOWANCES	Withholding Allowances
SWT_ALLOWANCE_MSG1	SWT Allowance Message 1
SWT_ALLOWANCE_MSG2	SWT Allowance Message 2

Opening SWT_ALLOWANCES brings up the right field. Now, to find where this field is located, Karen right-clicks the definition and chooses "Find Object References." Voilá—a list of locations appears in Figure 1-3.

The list that is returned contains tables (records), panels, and other elements in PeopleSoft. Karen can then double-click on these items to open them and continue to cross-reference.

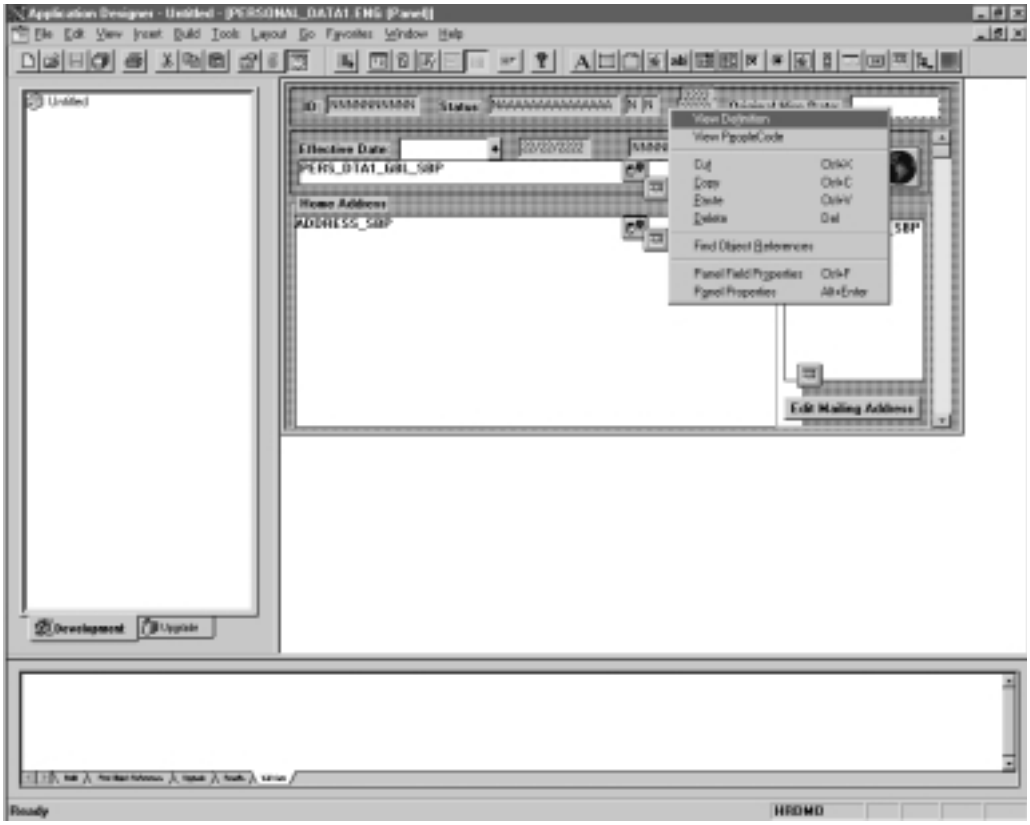


◗ **Figure 1-3** Cross-referencing results

Cross-Referencing Panels and Tables

PeopleSoft's suggested technique for finding tables is to explore the structure of a functional process. To find the location of a field, you should look at the panel that contains the field and work your way back to the underlying table. For example, we can see the original hire date field on the Go → Administer Workforce → Administer Workforce (U.S.) → Use → Personal Data panel. Let's find out where it is actually stored:

1. Obtain the panel name. Pull down the View menu and choose Panel Name. A check will appear next to Panel Name and the name will appear at the bottom of the panel. The name of the panel is PERSONAL_DATA1.
2. In the Application Designer, pull down the File menu and select Open. Choose "Panel" and type in PERSONAL_DATA1.
3. The panel is listed. Select it to open it. Then find the original hire date and right click on it. Choose View Definition. (Figure 1-4).
4. The table definition appears. The field is on PS_PERSONAL_DATA.



◗ **Figure 1-4** Cross-referencing a field in the application designer.

This method is accurate and straightforward. There are times, however, that the cross-referencing gets crazy, since panels can contain sub-panels which contain records with sub-records, and so on. You could be cross-referencing for quite some time, but eventually you will arrive at an answer.

One drawback of this method is that you must know which panel the field is on. If you have not used the online application, it could be more difficult to find the panel than to find the corresponding table. Additionally, you could choose the wrong panel. Some panels summarize many tables, and connecting these tables to the base tables can require looking at delivered PeopleSoft programs (SQR, COBOL, PeopleCode). In brief, the cross-referencing method does not provide a complete picture of what is going on and what the possible tables are. A combination of different researching methods is best.

Looking for Comments

One useful resource is the comments in the Application Designer that explain each table. A one or two-sentence description of a table can really help. Unfortunately, not all of the tables have comments. And sometimes the comments aren't very descriptive.

To view the comment in the Application Designer, open the table and choose Object Properties from the File menu. The description is under the General tab. You can see if the table is a table or a view of other tables by looking at the Type tab.

Searching the Code

Another technique for figuring out what a field or table is used for is to look at how PeopleSoft uses it in places outside of the Application Designer. There are hundreds of reporting programs that PeopleSoft delivers, written in Structured Query Reporting (SQR) language. A quick search of these programs often reveals not only the meaning of a field, but also the appropriate way to retrieve its contents.

To search through SQR, first use Windows to view the directory where delivered SQR is stored. This is typically a folder called 'sqr' (for example, g:\psoft\sqr, if your drive is 'g' and PeopleSoft is in the 'psoft' folder). Once in this folder, perform a find file (control-F, or choose 'Find…Files or Folders' from the 'Start' menu and type in the SQR directory manually). In the Find dialog box, choose the 'Advanced' tab. In 'Containing text:', type the name of the field or table you are searching for. All of the SQR programs that use this field will be located for you (Figure 1-5).

While the Application Designer contains a great deal of information, there are other techniques to consider when searching for data. It is important to consider the type of data you are searching for. Data can also be found in other areas, such as SQR or COBOL programs.

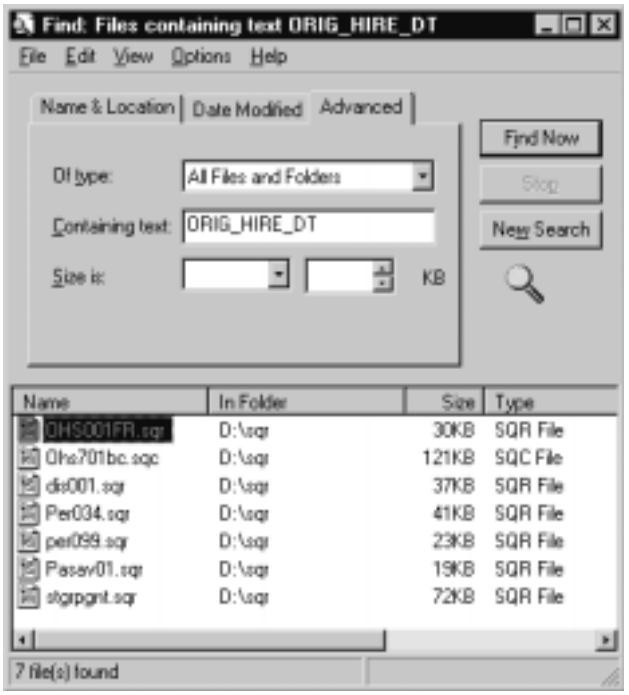


Figure 1-5 Searching through SQR code for a data element.

PEOPLESOFT PROCESSES

Many of the techniques described here are technical research methods. Using the PeopleSoft delivered tools enables us to find tables and fields, but each requires some knowledge of what we are looking for. What if we have no idea what the field is, or where the panel is?

Most of this book is dedicated to assisting with this issue—describing a functional process and providing the table and field names associated with it. For example, suppose you need a report of the flexible spending account (FSA) deductions for employees. It is difficult to determine what the field or table names would be. Where are deductions recorded? Are FSA deductions found in the benefits or payroll section of the application?

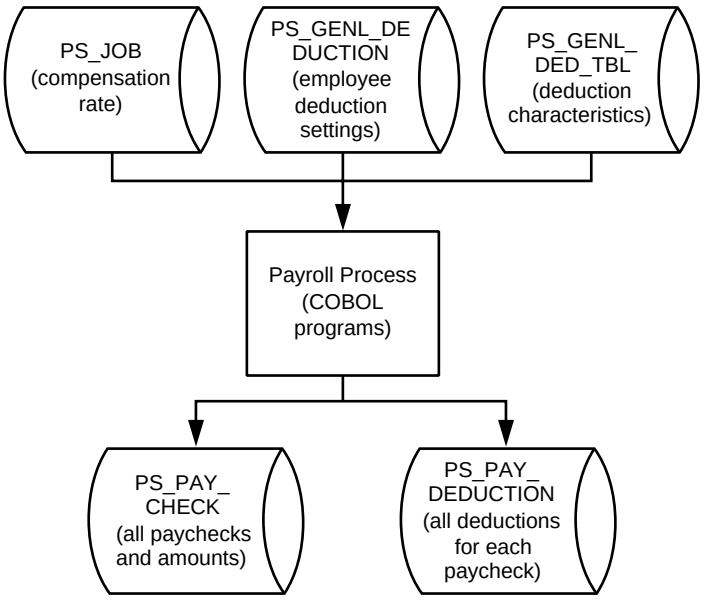
This book will provide the answers to questions like this, but it is important to understand that PeopleSoft operates with distinct functional processes. There is a payroll process, a benefits administration process, and several processes for hiring and maintaining the status and other data of employees. Corresponding to

these processes, there are base benefit tables and benefit administration tables. There are earnings tables and payroll tables.

It is important to distinguish between the tables that come before a functional process, the tables used during the process, and the tables that are populated with the results of the process. When looking for the right table, knowing what step of the process you're looking for makes a huge difference.

The two major processes in PeopleSoft are Payroll and Benefits Administration. Depending on the implementation, these processes are executed on a regular schedule. Each process is basically a series of COBOL programs that takes data from certain tables, applies calculations based on data in other tables, and writes the result to output tables. Payroll, for example, looks at the earnings, deductions, garnishments, and taxes defined for a specific employee. Depending on the rules for each of these elements, dollars are added and subtracted from the employee's gross pay. The result is more than just a check—the other elements are output as well. The deductions actually taken, the garnishments applied and the taxes paid could all fluctuate based on the gross pay and the rule definitions. This output is stored in output tables beginning with the 'PAY' prefix (see Figure 1-6).

Hence, to find deductions taken from a paycheck, you would look at PS_PAY_DEDUCTION. To find the deductions that *should* be taken from a pay-



◗ **Figure 1-6** Example tables used in the payroll process. The output tables have a 'PAY' prefix, and the control table PS_GEN_DED_TBL has a 'TBL' suffix.

check (i.e., before the process), you'd look at PS_GENL_DEDUCTION. And to find the definition of the deduction, the control table PS_GEN_DED_TBL is the appropriate place to search.

Always consider the process, not just the name of the field, when you are looking for data.

DISSECTING A PEOPLESOFT TABLE

The best way to get a good look at a PeopleSoft table is to open up the definition in PeopleSoft's Application Designer.

Identifying the Elements: A Look at a Table

Once we find the table we're looking for, there are several ways to look at it. Most people find it quicker and more convenient to print out the table. PeopleSoft prints the table with the record name, field names, definitions, formats, descriptions, and other technical information. This information will help in the construction of your query. Looking at Figure 1-7, we can see the result of an Application Designer printout.

Figure 1-7 Printout of a PeopleSoft table

> To get a simple printout, turn off the PeopleCode that by default appears on printouts of tables. You can find the check box to turn off PeopleCode by looking at Page Setup under the File menu in the Application Designer.

Along the top are the column names: Field Name, Type, Length, Format, Long Name, Short Name. These are self-explanatory. Let's look carefully at some of the remaining elements of this printout, which are critical for understanding how the table functions.

Keys

Identifying the keys in a table is a critical step to retrieving accurate data. Keys are the unique identifiers of a row in the table. Sometimes there is one key, such as employee ID, but more often there are several. When there is more than one key, the values in a certain key could be duplicated, but there are never two rows with the same values in every key. In other words, the *combination* of the key fields must be unique. In the STATE_TAX_DATA table, the following situation is likely:

```
EMPLID   COMPANY   EFFDT         STATE
-------  --------  ------------  ------
8121     CCB       01-JAN-1991   CA
8121     CCB       01-JAN-1991   PA
8200     PST       01-OCT-1993   CT
8200     PST       01-JAN-1991   CA
```

Each row is unique based on these four keys. You will *never* find the following when all four fields are keys:

```
EMPLID   COMPANY   EFFDT         STATE
-------  --------  ------------  ------
8121     CCB       01-JAN-1991   CA
8121     CCB       01-JAN-1991   CA
```

When selecting data, *always consider every key*. To retrieve the appropriate row, each key must be limited to exactly the values you are looking for. For example, to find the allowances for employee ID 8121 in California, you will need to look at four different fields (Table 1-2).

Table 1-2 Considering keys (finding allowances for employee 8121 in CA)

Key	Criteria
Employee ID	This is '8121' and can be hard-coded into the query. This value never changes.
Company	This is the company the employee belongs to. Since it could change at any time, the current company should be obtained from another table. The core table that stores company is JOB.
Effective date	Choose the most recent effective date (more information on effective dates can be found in Chapter 2) to get current information.
State	This is 'CA' and can be hard-coded into the query. This value never changes.

If you neglect one of the keys, such as effective date, you will probably get several rows back (one for each effective date). A key like 'company' is ignored more often—after all, it is rare that an employee works for two companies. Be careful: PeopleSoft allows people to work for more than one company. Also, consider what happens if an employee changes companies. There may be two records with the same effective date, for the same state, for the same employee—one with the old company and one with the new.

Often a table has many keys, but when you look at the data in it, only one or two keys are used. The other key fields all have the same value. Don't be fooled—there are two good reasons to still include those keys in your criteria. First, just because the data looks like this now doesn't mean it isn't going to change. If the ability to change is there, it will probably happen. Second, database engines (i.e., Oracle, DB2, Sybase) usually return data faster when all of the keys are specified in the criteria. The database does not know what the data looks like—it assumes that all of the keys are necessary. As soon as you leave a key out of the criteria, the database may ignore an index it has created and take longer to retrieve the data.

Another tip: Be sure to distinguish among types of keys. The first letter of the "Key" column in the Application Designer is the important one. It is either "K" for key or "A" for alternate key. Keep in mind that an alternate key is not a key. The database does not recognize this as a key—*an alternate key does not distinguish a row.* An alternate key is used by the PeopleSoft application for looking up information. An example is the PS_PERSONAL_DATA table, which has a key of employee ID. The alternate key is name. Although in the PeopleSoft application you can look people up by name, you may retrieve more than one person with the same name. Hence it is not a key; it is just an alternative way to look up rows. It is best not to use it for a direct database query, especially if you want to retrieve only one distinct row.

Required Fields

The column labeled "Req" tells you whether PeopleSoft requires a field. A required field, from an application perspective, must be filled in. If you are typing information into the state tax data panel (which updates the PS_STATE_TAX_DATA table when you click "Save"), the SWT_MAR_STATUS field will be required.

This is not the case from the database perspective. When inserting data into PS_STATE_TAX_DATA using a SQL statement, you can leave SWT_MAR_STATUS blank. The chance of this occurring is usually slim depending on the company and the environment.

At best, the required flag tells you whether to expect blanks in the data returned. If it isn't required, there is a good chance some fields will be blank. If it is required, there is a slim chance that the field will be blank.

Translation Values

Many fields have only a few possible values. These values are stored in a common table, the XLATTABLE (see Chapter 3).

The table printout provides a nice way to look at the possible values for certain fields, since they are selected straight from the XLATTABLE for the printout. The SPECIAL_SWT_STATUS field, for example, can only have three possibilities: E, G, or N. Hard-coding the criteria of 'E' for exempt into your query would return employees who are exempt from state taxes.

Never assume that translate values are set to the PeopleSoft standards. Sometimes companies customize translate values to resemble the values they are familiar with. If someone changes exempt to be an 'X,' it is an important change. Always print out the table before constructing a query to check the translate values.

Prompt Tables

The prompt table column lists the tables that provide values for a certain field. Often this is a control table, such as PS_COMPANY_TBL. In PeopleSoft, when typing state tax data into a panel, you can prompt for a list of possible companies from the PS_COMPANY_TBL. Although this is a convenience built for the application, it also indicates a link in the database.

Notice, for instance, that the SWT_MAR_STATUS field does not have translate values. Instead, the prompt table PS_SWT_MARSTAT_TBL is shown. The PS_SWT_MARSTAT_TBL contains a list of possible state marital status values. For the description of a value, it would be necessary to link to this control table.

PEOPLESOFT HRMS BASICS: A REVIEW

The first step in any query is to find where the data is located. Since the PeopleSoft HRMS database is so large, the tools described in this chapter are useful for taking this first step at a smaller scale. Research tools like cross-referencing, looking for comments, and thinking about the functional process allow us to link the online PeopleSoft panel functions to database tables and fields. The basic tables we discussed, PS_PERSONAL_DATA, PS_JOB, and PS_EMPLOYMENT, provide a starting point for understanding the structure of the database. And the knowledge of the elements of a table, such as how keys and required fields affect the storage of data, help us as we start linking tables together, searching for data and connecting PeopleSoft online with the PeopleSoft database. These guidelines pertain to all users—those who are looking at the online panels, those who are setting up the database for the first time, and those who are extracting data for reports.