



PeopleTools Tables: The Application Repository in the Database

By David Kurtz, Go-Faster Consultancy Ltd

Since their takeover of PeopleSoft, Oracle has announced project Fusion, an initiative for a new generation of Oracle Applications building on the best of existing Oracle and PeopleSoft products. This is the first in a series of articles in which I will examine some aspects of PeopleTools, the proprietary technology developed by PeopleSoft, and speculate about Fusion.

One of the fundamental design aspects of PeopleSoft technology is that the online part of the application, that PeopleSoft called their Pure Internet Architecture (PIA), and much of the batch programming, is stored as metadata in the database alongside the application data. The traditional diagram of a PeopleSoft database comprises three parts.

- The database catalogue describes all the objects in the database. The structure of the database's catalogue varies from platform to platform, but the concept is common to all. In Oracle, the catalogue is exposed through the underlying fixed objects (tab\$, col\$, etc.) and is visible in a more user-friendly format through views (DBA_TABLES, DBA_TAB_COLUMNS, etc.)
- The PeopleTools tables contain most of the definition of the PeopleSoft application as either metadata or PeopleCode (PeopleSoft's proprietary 3GL). This includes some tables that also describe the application data, so PeopleSoft maintains its own data dictionary.

- The application tables contain the users' data. They are in the same database schema as the PeopleTools tables.

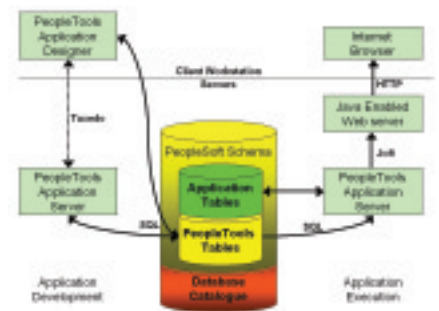
PeopleSoft's decision to store their application in the database has a number of implications.

Much of the SQL submitted by the application to the database is dynamically generated from the metadata in the PeopleTools tables. Thus developers spent less time coding SQL themselves for the online application. Although it can then be difficult to work out from whence a particular SQL statement is coming, and impossible to tune it by adding hints.

PeopleSoft provide their proprietary development tools to customers. Some companies have also licensed PeopleTools to develop entire applications themselves. Over the years, the Application Designer has matured into a complete development environment. It is used to apply patches, customise and debug the delivered application, and control migration of application elements between environments. This permits vendor patches and customisations to be progressed from development, via test, to production environments, in a controlled fashion.

When an object in a PeopleSoft application is saved in the Application Designer, rows are written to a variety of PeopleTools tables in the database. When a PeopleSoft application executes, it is effectively interpreting the content of the PeopleTools tables. The application must

connect to the database anyway, so the database is a convenient and secure location to store this information. Rather like recursive SQL issued by the database to interrogate its catalogue, at runtime PeopleTools applications will query the PeopleTools tables. There is considerable overhead in retrieving the application from the PeopleTools tables every time, so PeopleSoft programs maintain a local cache of PeopleTools objects in both memory and flat files.



The PeopleSoft architecture is usually distributed across a number of physical servers. The application server layer can be scaled horizontally by configuring additional application servers on additional physical servers. There is mostly no need to install any PeopleSoft application code on these machines. It will simply be retrieved from the database and build its own cache.

A lot of batch processing is performed with PeopleSoft's Application Engine. This also uses source code stored in the PeopleTools tables. However, many PeopleSoft application modules still make extensive use of COBOL programs. Most

of the reporting is done with Crystal and SQR. These all require the files to be installed on the batch servers.

Shortly after the takeover, Oracle started to talk publicly about aspects of the PeopleSoft technology that they admired. One of these was that PeopleSoft developers are several times more productive than Oracle application developers. They were also complimentary about the ability of the Application Designer to control the upgrade process.

I expect to see Fusion with an integrated development workbench that will bear more than a passing resemblance to Application Designer, probably with a code repository somewhere in a database. PeopleCode will be replaced, almost certainly with Java.

The Application Designer is also responsible for defining data structures in PeopleSoft applications and for building database objects, so some of the PeopleTools tables that describe data structures will correspond closely to the Oracle database catalogue views. This means that in a PeopleSoft database we are dealing with two data dictionaries rather than one: the database's catalogue defines all of the objects that exist in the database, and the PeopleSoft dictionary defines all the objects that should exist in the PeopleSoft schema, and that can therefore be referenced by the application. Except when building DDL scripts, PeopleTools never interrogates the database's catalogue, but instead relies upon its own. The importance of keeping the dictionaries synchronised cannot be overstated. Any discrepancy will cause PeopleSoft to generate invalid SQL.

Platform Agnosticism

From its very beginning, PeopleSoft applications were designed to run on various database platforms and various operating systems. In fact platform

independence has been an overriding principle in PeopleSoft. The majority of installations are on Oracle, but there are also many companies using Microsoft SQL Server and IBM's DB2, and a few still use Sybase and Informix. Therefore, as far as possible, PeopleSoft delivers the same SQL code to all database platforms. Where SQL is dynamically generated from the metadata in the PeopleTools tables it conforms to a lowest common denominator code set that will function on all supported databases. Where SQL is coded directly within the application, it will also be coded in a platform generic manner. Only occasionally, and when forced by performance considerations, does PeopleSoft deliver platform specific, but functionally equivalent, variations.

As a direct result of this platform-generic approach, a PeopleSoft database does not contain certain elements that are implemented differently on different platforms. It does not include referential constraints; instead these are maintained by the application. PeopleSoft does not use Oracle sequences; instead it uses ordinary tables to maintain sequence numbers. Consequently, the application sometimes must hold locks in order to serialise processing. Ordinarily this would lead to scalability problems. However, in the PIA this is very rarely a problem because data is not locked while the user enters data into a page. Instead, at save time the application re-queries the data in order to check that it has not changed in the meantime, at which point it does lock the data. It then performs the update, and any other save-time processing, and then commits without waiting for further user response. Only if the data has changed is an error raised. This 'optimistic' locking strategy minimises the time during which a lock is held.

Application auditing is usually performed by PeopleTools. Column changes to be are also held as meta-data in the PeopleTools

tables. Although when PeopleSoft is implemented on Oracle, there is an option to build database triggers to perform auditing. In the latest release of PeopleTools, triggers are also used to maintain control data to facilitate synchronisation of subsets of data to PDAs. With these two exceptions there are no database procedures or other stored-code executed by the database server. Instead PeopleSoft uses PeopleCode executed by either the PIA or Application Engine.

Nor is the PeopleSoft Application Designer capable of building partitioned, index-organised or global temporary tables or function based indexes. Tables have unique indexes rather than primary key or unique constraints. The Application Designer cannot maintain function-based indexes.

There is no support for maintenance of cost-based optimiser statistics anywhere within PeopleTools. The DBAs are very much on their own, although some of the batch applications will analyse working storage tables.

A platform-generic application may be easier and cheaper for a vendor to develop and maintain, but it virtually guarantees sub-optimal response on the customers' chosen platform, whichever one they choose. Oracle has hinted that Fusion will operate database platforms other than their own RDBMS, although there is not clear statement to that effect.

About the Author

David Kurtz is a performance specialist working Enterprise PeopleSoft applications and Oracle (www.go-faster.co.uk). He is the author of 'PeopleSoft for the Oracle DBA', published by Apress (www.psftdba.com). He is also the chair of UKOUG's UNIX SIG.