

**Pinnacle Learning
Manager
Developer's Tool Kit**

Version 4.0

Last Updated September 30, 1999

Information in this document is subject to change without notice. Companies, names, and data used in examples are fictitious unless otherwise noted. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Pinnacle Software Corporation.

© Copyright 1996 - 1999 Pinnacle Software Corporation. All rights reserved.

The Pinnacle Multimedia Logo, Pinnacle Learning Manager, The Learning Processor, The Learning Tutor, The Score Reporter and their respective Logos are trademarks of Pinnacle Software Corporation, Draper, Utah.

All other product names and logos are trademarks and registered trademarks of their respective companies.

Printed in USA

Contents

1. PLM Developer's Kit Overview.....	1
1.1 Custom Reports	1
1.2 Communication with Computer Based Training (CBT) Courses	1
1.3 Automatic Course Add	1
1.4 Importing Existing Data	1
2. Custom Reports	3
2.1 Database Description.....	3
2.1.1 Database Data Types.....	3
2.1.2 Database Table Definitions.....	4
2.2 Adding Custom Reports to PLM.....	24
3. Communication with Computer Based Training (CBT) Courses	27
3.1 Introduction	27
3.2 Conventions	27
3.3 Concept Glossary.....	27
3.4 An Overview of PLM.....	28
3.5 PLM Employee-Course Data Tracking.....	28
3.5.1 Student Training Course Status	29
3.6 An Overview of the PLM APIs	30
3.6.1 Using Command-Line Parameters	30
3.6.3 Using DDE.....	30
3.7 An Overview of Program to Program Communication.....	31
3.8 PLM Support for UNC Redirection	31
3.9 Program Command-Line Communication	32
3.9.1 Summary	32
3.9.2 Replaceable Parameters.....	32
3.10 DDE Communication.....	34
3.10.1 Summary	34
3.10.2 DDE Conversation.....	34
3.10.3 DDE Request Items.....	34
3.10.4 DDE Poke Items.....	37
4. Communication with PLM Online	39
4.1 Introduction	39
4.2 Windows Program Communication with PLM Online.....	39
4.3 Web/HTML Communication with PLM Online.....	39
4.3.1 Passing Information from PLM Online to a Web/HTML course.....	39
4.3.2 Returning Information from a Web/HTML course to PLM Online.....	39
4.4 HTTP Communication with PLM Online.....	40
5. Automatic Course Add	41
5.1 Introduction	41
5.2 Concepts	41
5.2.1 Import Methods	41
5.2.2 PLM Import File Format.....	41
5.3 Sample PLL and PLC files	46
5.3.1 PLL File with 2 PLC File Items	46

5.3.2	PLC File with All Course Items.....	46
5.3.3	PLC File with Minimum Course Items	47
5.3.4	PLC File with Offerings	48
5.3.5	Blank PLC File with All Items	48
5.4	Troubleshooting	50
5.4.1	PLM Import Messages.....	50
6.	Importing Existing Data	53
6.1	Overview	53
6.1.1	Changes for Version 4.0	53
6.2	DLL Functions.....	53
6.2.1	GetDLLVersion.....	54
6.2.2	GetCapabilities.....	54
6.2.3	Initialize	54
6.2.4	Import People (GetPerson)	55
6.2.5	Import Groups (GetGroup, GetGroupMember, GetGroupManager)	55
6.2.6	Import Courses (GetCourse, GetOffering, GetUnit)	55
6.2.7	Import Scores (GetScores, GetUnitScores)	55
6.2.8	Import Authorizations (GetAuthorization)	56
6.2.9	Import Registrations (GetRegistration).....	56
6.2.10	Import Curricula (GetCurriculum, GetCurriculumMember)	56
6.2.11	Cleanup.....	56
6.3	Installation DLL	56
6.4	Import DLLs Shipped with PLM.....	57
6.4.1	Novell NetWare (NWIMPORT.DLL).....	57
6.4.2	ASCII Comma-delimited (ASC_IMP.DLL)	57
6.4.3	Courseware Imports	57
6.5	Import.h	57

1. PLM Developer's Kit Overview

The PLM Developer's Kit is designed to allow other programs to communicate and integrate with the Pinnacle Learning Manager. There are four sections to the Developer's Kit. Each section describes a communication or integration method. An overview of each of these sections is given below.

This document is intended to document version 4.0 of PLM. However features that are different from previous versions are noted where possible.

1.1 Custom Reports

The Custom Reports section provides information about PLM so you can build your own custom reports using Crystal Reports. The first part of the section describes the database tables, the fields within each table, and the relationships between the tables. With this information you know the data that is available for custom reports and how to access it. The second part of the section describes how to set up a report so it can display Select buttons like the built-in reports. The Select button allows the user of the report to select specific courses, people, and so forth to be included in the report.

1.2 Communication with Computer Based Training (CBT) Courses

The Communication with Computer Based Training (CBT) Courses section provides information on how a CBT can get information from PLM (usually about the student who is taking the course) and how a CBT can send information to PLM (usually about the student's progress or completion). There are two integration methods described:

- The first is used to integrate with existing courses that remain unchanged.
- The second is used to integrate with new or updated courses and is easier for the user to set up.

1.3 Automatic Course Add

The Automatic Course Add section provides information on how to automatically add a course to PLM's database. This can be very useful for CBT courses and seminar courses. The installation program for CBT courses can write a file to disk that contains all of the course information. PLM will then read this file and add the course using the information in the file. Providers of seminar courses can place the information about their courses and dates and times of the offerings in a file on disk. PLM can read the file from disk and add the courses and offerings. The file can also be placed on an Internet site where the user can download it. Again, PLM can read the downloaded file and add the courses and offerings.

1.4 Importing Existing Data

The Importing from Existing Data section provides information on how to program an additional database import. Import capabilities are shipped with PLM (click the **Tools** menu, then **Import Data** in PLM). This section explains how to create an import that will communicate existing information to PLM. (Often the existing information is in a database or other file.) This interface to PLM allows for information about new courses, curricula, people, groups, scores, and course registration and authorization to be imported from other sources.

2. Custom Reports

Custom reports can be created by using Crystal Reports (use version 7.0 for PLM 4.x, version 5.0 for PLM 3.x, and version 4.5 for PLM 2.x). Crystal Reports is a widely used and readily available product for quickly building reports. The built-in reports for PLM were developed using Crystal Reports. You can build a custom report from scratch or you can copy one of the built-in reports, give it a new name, and use it as a starting point for a custom report.

Crystal Reports has extensive exporting and also e-mail capabilities. Special reports can be built that take advantage of these capabilities. A few of the built-in reports are for export only. You can also create export only reports.

To take full advantage of custom reports, you will need to know what data is available to be included in your reports. The database structure described in this section will provide the information you need to include in your reports.

After you have created the Crystal Report (.RPT) file, you can add it to the Reports tab in the administrator module. (To add the .RPT file, enter PLM Administrator, select the Reports tab and click the **Add** button.) When you add a report, you can also select the filters that will allow users to limit the data included in the report.

2.1 Database Description

This document summarizes and describes the PLM database. The fields, data types, and indexes of each table are described in enough detail that a software developer will understand how to use the database. For SQL databases, domains, triggers, procedures and views are also described. All Paradox tables have a password. The general Paradox table password for table access is "Bookworm" and is case sensitive.

A PLM database is a set of required files describing the people, courses, and the relationships between the people and courses within an organization. A PLM database, including database tables and indexes, can be broken into three categories:

- 1) Database Tables: Files kept in a central location (usually a network server).
- 2) Local Database Tables: Files created on the computer running PLM Student.
- 3) Temporary Database Tables: Files created on the computer running PLM Student each time PLM Student is executed.

2.1.1 Database Data Types

The following is a list of the database data types. The list provides translation from Paradox, SQL, and Delphi data types:

Paradox	Size and Notes
Binary(B)	Binary data (in Paradox stored in .MB) used for image
ShortInt(S)	16 bit signed integer
Integer(+)	32 bit signed integer, auto-increment sequence starts at 1
Integer(I)	32 bit signed integer
Char(A)	Paradox: 1-255 bytes
TimeStamp(@)	Paradox: 1 Jan. 9999 BC to 31 Dec. 9999 AD
Date(D)	same as above
Time(T)	24 hour clock, with hours, minutes, and seconds
Memo(M)	Strings that are too long to store in an alpha field
Money(\$)	Paradox: \$9,999,999,999,999.99 to -\$9,999,999,999,999.99

2.1.2 Database Table Definitions

2.1.2.1 AUTHORIZ (Student Course Authorization Support Table)

This table is not used for PLM Version 3.0 and later. The AUTHORIZ table duplicates data from other database tables and is used to provide a view of the database for speed of table lookup. It can be rebuilt using the PLM Utility program. There is a record in this table for every person in every group who is authorized for any course. For example: a single student might be listed in this table 10 times for a single course (that is, 9 group authorizations and 1 personal authorization).

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
course_id	Integer(I)	no		no referential check: cat_item(id)
user_id	Integer(I)	no		no referential check: person(id)
group_id	Integer(I)	no	0	no referential check: igroup(id)
Reqd	ShortInt(S)	no		{0, 1} (0 = No, 1 = Yes)

Indexes

Primary Key

Required

Authorized

Field(s)

course_id, user_id, group_id

reqd

user_id, course_id, reqd

Dependent Tables

None

2.1.2.2 ARC_LOG (Log Archive Table)

This table contains the archived data from the Log table.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	no		
person	Char(A) 80	no		
Time	TimeStamp(@)	no		
Type	Integer(I)	no		
Record	Char(A) 20	no		
Data	Memo(M)	no		

Indexes

Primary Key

Dependent Tables

None

Field(s)

id

2.1.2.3 CAT_HIST (Catalog Item History Table)

The CAT_HIST table contains a record for each catalog item that has been completed by at least one person. If a catalog item is deleted from PLM before any person has completed it, it will not be listed in the Catalog Item History table. (Note: the Training Record table uses this information to provide information about each course that a particular person has completed.) Without the Catalog Item History table, each record in the Training Record table would have to contain information about the course that was completed. With the Catalog Item History table, the information about a completed course only has to be stored one time.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	no		
access_count	Integer(l)	no	0	
user_count	Integer(l)	no	0	
Completion_count	Integer(l)	no	0	
total_time	Integer(l)	no	0	
mall_number	Char(A) 24	no		no referential check: course(mall_number)
Title	Char(A) 100	no		
Description	Char(A) 255	no		
Subject	Char(A) 64	no		no referential check: ci_subjt(name)
Ceu	Char(A) 5	yes		
co_cat_num	Char(A) 20	yes		

Indexes

Primary Key

HistMallNumber
HistCourseTitle
HistCourseSubject

Field(s)

id
mall_number
title
subject

Dependent Tables

TRAINREC

2.1.2.4 CAT_ITEM (Catalog Item Table)

A catalog item is a unit of instruction that can be listed as a single entity within PLM. The only catalog items defined within PLM are courses. Since courses are the only types of catalog items currently defined, there is an exact 1:1 relationship between the Catalog Item table and the Course table. The course id is really just a link into the Catalog Item table. Catalog Item is maintained by PLM and is not directly edited by the user.

Note: The default value and check constraint on the "type" field is not a part of the Paradox database. The only value that is currently used is "1" for course.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
id	Integer(+)	no		
type	ShortInt(S)	no	1	{1}, 1 = Course

Indexes

Primary Key

AUTHORIZ
CAT_KEYS
COROFF
COURSE
UNIT
UDFC

Field(s)

id

2.1.2.5 CAT_KEYS (Catalog Item Keywords Table)

The CAT_KEYS table maintains a list of associated catalog items and keywords. A user edits this table by adding and removing items in a Keyword Include/Exclude list for each course.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Item	Integer(l)	no		Cat_item(id)
Keyword	Integer(l)	no		Keyword(id)

Indexes

Primary Key

Keyword

Field(s)

item, keyword

keyword

Foreign Keys

item

keyword

Table(Field)

cat_item(id)

keyword(id)

Dependent Tables

None

2.1.2.6 CGROUP (Course Group Table)

The CGROUP table maintains a list of course groups in PLM. The user can add, modify and delete records from this table.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	no		
Name	Char(A) 20	no		
Description	Char(A) 255	yes		

Indexes

Primary Key

GroupName

Field(s)

id

name

Initial Insert Values

{(1, All, All Courses)}

Dependent Tables

CGROUPC

2.1.2.7 CGROUPC (Course Group Course Join Table)

The CGROUPC table maintains an association between a course group and the courses that are part of that group. Users add and delete records to and from this table using an Include/Exclude list in PLM.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Cgroup	Integer(l)	no		cgroup(id)
Course	Integer(l)	no		course(id)

Indexes

Primary Key

Course

Field(s)

cgroup, course

course

Foreign Keys	Table(Field)
cgroup	cgroup(id)
course	course(id)

Dependent Tables
None

2.1.2.8 CI_SUBJT (Catalog Item Subject Table)

The CI_SUBJT table maintains a list of subjects. Each course must have an associated subject from this table.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	no		
Name	Char(A) 64	no		

Indexes	Field(s)
<i>Primary Key</i>	id
SubjectName	name

Dependent Tables
COURSE
CAT_HIST (Historical Copy)

2.1.2.9 CMP_ATHR (Course Completion Authorization Table)

The CMP_ATHR table lists the values currently allowed for who can mark a course as completed. It is used for lookup purposes by the course table and is not directly edited by the user. If the value is set to 1, then the administrator must manually enter the progress and completion information. If set to 2, the course (in the case of CBT) or the administrator may enter the progress information. In this second case, the CBT would set the completion or progress information automatically. If the value is set to 3, then the student, course, or administrator may all set completion and progress information for the course.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	no		
Type	Char(A) 20	no		{{(1, Administrator Only), (2, Course), (3, Student), (4, Electronic Signature)}

Indexes	Field(s)
<i>Primary Key</i>	id

Dependent Tables
COURSE

2.1.2.10 CMP_STAT (Course Completion Status Table)

The CMP_STAT table maintains a list of the allowed completion status settings. The Training Record table uses one of these values when a training record is created to indicate the completion status for the training record. Internally, PLM uses the value saved in the Training Record table to determine on which tab in the Student module the course should be listed. For PLM version 2.x, if the course is completed with 1, 3, or 4, then the course is listed on the Completed tab; if the course is completed with 2, then it is still listed on the Registered tab. For PLM version 3.x, completed courses are always listed on the Completed tab. The Incomplete status, which indicates that

the student started the course but then removed his registration before completing the course, is also include in PLM version 3.x. The In Progress status is new with PLM version 4.x. The user does not edit the CMP_STAT table but may choose a value from it when setting course completion information in PLM.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	no		
Status	Char(A) 12	no		{{(1, Completed), (2, Failed), (3, Passed), (4, Waived), (5, Incomplete), (6, In Progress}}

Indexes

Primary Key

Field(s)

id

Dependent Tables

TRAINREC

2.1.2.11 COMPANY (Company Table)

The COMPANY table lists information about the company using PLM and the license information for a particular PLM database. Additionally, the invalid_login_id field stores information about what to do when a user enters an invalid user id.

The UDF fields in this table store the field names for the UDFC and UDFP tables.

Access: Data entry - all fields, except: TOTAL_LICENSES and EXP_DATE are READ ONLY

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Name	Char(A) 48	no		
co_address	Char(A) 255	no		
Invalid_Login_Id	ShortInt(S)	No	0	
total_licenses	Integer(I)	yes		Value >= -1 (unlimited)
next_mall_number	Char(A) 24	yes		
crs_reg_msg	Char(A) 255	yes		
allow_course_ls	Char(A) 1	no	T	{T, F}
exp_date	Date(D)	yes		
udf1	Char(A) 20	yes		
udf2	Char(A) 20	yes		
udf3	Char(A) 20	yes		
udf4	Char(A) 20	yes		
udf5	Char(A) 20	yes		
udf6	Char(A) 20	yes		
udf7	Char(A) 20	yes		
udf8	Char(A) 20	yes		
udf9	Char(A) 20	yes		
udf10	Char(A) 20	yes		
udf11	Char(A) 20	yes		
udf12	Char(A) 20	yes		

udf13	Char(A) 20	yes		
udf14	Char(A) 20	yes		
udf15	Char(A) 20	yes		
StudentRefresh	Integer(I)	yes	300	Value >= 0
AdminRefresh	Integer(I)	yes	30	Value >= 0
LicenseNumber	Char(A) 12	no		

Indexes **Field(s)**

Primary Key name

Dependent Tables

None

2.1.2.12 COROFF (Course Offerings Table)

Each record in the COROFF table describes the necessary information about a particular offering of a course. An offering is a specific date, time, and place that a course will be taught, administered, or presented. The user can add, modify, and delete records in this table using the Course tab in PLM Administrator. Additionally, the user can selectively delete expired offerings using the **Clean Up Offerings** menu item on the **Tools** menu.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	no		
Course	Integer(I)	no		cat_item(id)
enroll_status	Integer(I)	no	4 (open)	{1..4}, no referential check: enrlistat(id)
close_reg	Date(D)	yes		
open_reg	Date(D)	yes		enforced by user interface
enroll_cap	ShortInt	no	-1 (unlimited)	
start_date	Date(D)	yes		enforced by user interface
end_date	Date(D)	yes		
start_time	Time(T)	yes		
end_time	Time(T)	yes		
Location	Char(A) 255	yes		
reg_instructions	Char(A) 255	yes		

Indexes **Field(s)**

Primary Key id

EnrollStatus enroll_status

StartDate start_date

CourseThenDate course, start_date

(All foreign keys)

Foreign Keys **Table(Field)**

course cat_item(id)

Dependent Tables

STU_SCHD

2.1.2.13 COURSE (Course Table)

Each record in the COURSE table describes a course. Each course is tied to a particular Catalog Item that provides the course's ID. A course may be either Self-Study or Scheduled. Once the course type has been set, it cannot be changed. The user can add, modify, and delete records in the Course table from within PLM.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(I)	no		cat_item(id)
Instruction_method	Integer(I)	no		{1...10}, no referential check: instrmthd(id)
enroll_type	Integer(I)	no	2 (Internal Registration)	{1, 2}, no referential check: enrlytype(id)
Type	Integer(I)	no	1 (Self Study)	{1, 2}, no referential check: crs_type(id)
reg_instructions	Char(A) 255	yes		
Company_has	Char(A) 1	no	F	{T, F}
Duration	Char(A) 24	no	"1 Hour"	
Percent_pass	ShortInt(S)	yes		Value between 0 and 100
Ceu	Char(A) 5	yes		
Learning_style	Integer(I)	yes		
Command_line	Char(A) 255	yes		
Working_directory	Char(A) 255	yes		
dll_path	Char(A) 255	yes		
dll_command_line	Char(A) 127	yes		DOS restriction
Subject	Integer(I)	no		ci_subjt(id)
Included_material	Char(A) 255	yes		
active_item	Char(A) 1	no	T	{T, F}
Inactive_reason	Char(A) 100	yes		
Provider	Char(A) 150	yes		
General_knowledge	Char(A) 255	yes		
mall_number	Char(A) 24	no		PII + 7 for internal use
Title	Char(A) 100	no		
Description	Char(A) 255	no		
recert_period	ShortInt(S)	yes		Value >= 0, Months
recert_notify	ShortInt(S)	yes		Value >= 0, Days
rate_type	Integer(I)	no	6 (user)	{1...7}, no referential check: lic_rate(id)
Rate	Currency(\$)	no	0.00	Value >= 0.00
Available	ShortInt(S)	no	-1 (unlimited)	Value >= -1
Grace	ShortInt(S)	no	0	Value >= 0
grace_period	ShortInt(S)	no	0	Value >= 0, Days
comp_authorization	Integer(I)	no	2 (course)	{1...3}, no referential check: cmp_atrh(id)
total_activities	ShortInt(S)	yes		Value >= 0
final_test	Char(A) 1	yes	F	{T, F}
co_cat_num	Char(A) 20	yes		
allow_course_ls	Char(A) 1	no	F	{T, F}
Reviewcmdln	Char(A) 255	yes		
Canreview	Char(A) 1	no	T	{T, F}

Indexes

Primary Key

MallNumber

CourseTitle

Field(s)

id

mall_number, title

title

CompanyCatNo co_cat_num

Dependent Tables

CGROUPC

CAT_HIST (Historical Copy)

SG_AUTH

2.1.2.14 CRS_TYPE (Course Type Table)

The CRS_TYPE table maintains a list of the defined course types. Currently, there are only two course types defined within PLM, Self-Study and Scheduled. A Self-Study course has a single offering. Each student who registers for a Self-Study course is registered for that single offering. A Scheduled course has one or more offerings. (See section 2.1.2.12 - *Course Offerings table* for more information about offerings.) Each course must have an associated value from the Course Type table. (See section 2.1.2.13 - *Course table* for more details about courses.) The user does not edit this table directly but chooses a value from the table for each course.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	no		
Type	Char(A) 20	no		{(1, Self Study), (2, Scheduled)}

Indexes

Primary Key

Field(s)

id

Dependent Tables

None

2.1.2.15 ENRLSTAT (Course Enrollment Status Table)

The ENRLSTAT table maintains a list of the possible values for course offering enrollment status. Each course offering has a status associated with it that describes whether or not students can currently register for the offering. The user does not edit this table but chooses a value from the table for each course offering in the system.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	no		
State	Char(A) 12	no		{(1, Canceled), (2, Closed), (3, On Hold), (4, Open)}

Indexes

Primary Key

Field(s)

id

Dependent Tables

None

2.1.2.16 ENRLTYPE (Course Enrollment Type Table)

The ENRLTYPE table lists the allowed values for the enroll_type field in the Course table. This table provides a hook for one of the open architecture features in PLM. If a course listed in PLM is not offered directly by the company (such as a time management seminar or college course), the student will need to register for the course outside of PLM. In this case, the user should set enroll_type to External Registration.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	no		
Type	Char(A) 28	no		{{(1, Internal Registration), (2, External Registration)}

Indexes **Field(s)**
Primary Key id

Dependent Tables
None

2.1.2.17 ID_PWD (Login ID and Password Join Table)

The ID_PWD table maintains an association between a login ID and a password. It is used to enforce unique passwords.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
LoginID	Integer(I)	no		Log_Hist(id)
PasswordID	Integer(I)	no		Pwd_Hist(id)

Indexes **Field(s)**
Primary Key LoginID, PasswordID
PasswordID PasswordID

Foreign Keys **Table(Field)**
LoginID log_hist(id)
PasswordID pwd_hist(id)

Dependent Tables
None

2.1.2.18 IGROUP (Instruction Group Table)

The IGROUP table maintains a list of all the people groups defined within PLM.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	no		
Title	Char(A) 60	no		
Description	Char(A) 255	yes		

Indexes **Field(s)**
Primary Key id
GroupTitle title

Initial Insert Values
{(1, Everyone, All Students)}

Dependent Tables
STU_GRP
MNG_GRP
AUTHORIZ

2.1.2.19 IMPORT (Import DLL Table)

The IMPORT table maintains a list of the import DLLs and locations. Users can add, modify, and delete items in this table from within PLM. For versions up through PLM 3.x, there was an additional item (2, Pinnacle Learning Processor, lpimport.dll); the items with ID of 4 or more are new to PLM 4.0.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	no		
Source	Char(A) 100	no		
dll_path	Char(A) 80	no		

<u>Indexes</u>	<u>Field(s)</u>
Primary Key	id
SourceName	source

Initial Insert Values

{(1, Novell NetWare, nwimport.dll),
 (3, ASCII Comma Delimited, asc_imp.dll),
 (4, Student Training Record, tr_imp.dll),
 (5, CBT Course Import, cc_imp.dll),
 (6, WinTracs-PCM Import, wt_imp.dll),
 (7, Tarragon Course Import, tc_imp.dll)}

Dependent Tables

None

2.1.2.20 INSTMTHD (Catalog Item Instruction Method Table)

The INSTMTHD table maintains a list of allowed instruction methods for a course. This table is not editable within PLM but is used by the course table to provide the instruction method for the course.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	no		
Type	Char(A) 18	no		{(1, Audio), (2, Broadcast), (3, Class), (4, Computer (CBT)), (5, Conference), (6, Printed Material), (7, Seminar), (8, Task), (9, Test), (10, Video)}

<u>Indexes</u>	<u>Field(s)</u>
Primary Key	id

Dependent Tables

None

2.1.2.21 KEYWORD (Keyword Table)

The KEYWORD table maintains a list of course keywords. Users can add keywords in PLM, but can't modify or delete keywords.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	no		
Name	Char(A) 32	no		

Indexes **Field(s)**

Primary Key id
 KeywordName name

Dependent Tables

CAT_KEYS

2.1.2.22 LEARNING (Learning Style Table)

The LEARNING table maintains a list of learning styles. Users can add items to the table from within PLM but can't delete items from the table from within PLM. Each course in the course table and each person in the person table can have a learning style assigned.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	no		
Name	Char(A) 30	no		

Indexes **Field(s)**

Primary Key id
 LSName name

Dependent Tables

None

2.1.2.23 LIC_RATE (Usage License Rate Type Table)

The LIC_RATE table maintains a list of the allowed types of licensing. Each license type can be considered a unit. There is also a unit cost associated with the license type. Using this unit cost, future versions of PLM will handle billing and licensing of training courses. Each course may have a license type.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	no		
Rate	Char(A) 18	no		{(1, Computer), (2, Concurrent User), (3, Corporation), (4, Hour), (5, Site), (6, User), (7, View)}

Indexes **Field(s)**
Primary Key id

Dependent Tables
None

2.1.2.24 LOG (Event Login Table)

The LOG table maintains all logged events.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
id	Integer(+)	no		
person	Char(A) 80	yes		
time	TimeStamp(@)	yes		
type	Integer(I)	yes		
record	Char(A) 20	yes		
data	Memo(M)	yes		

Indexes **Field(s)**
Primary Key id

Dependent Tables
None

2.1.2.25 LOGIN (Login Table)

The LOGIN table maintains a list of who can currently log into PLM. This table stores the person's name, login id, and password from the person table. This table is edited in conjunction with the Person table through the Person tab in PLM Administrator. The main purpose of this table is for security and speed during the login process.

Data size for "login_name" has been set to match the login name size for Novell network so that the same user login name can be used. The field's password_exp_date, account_locked, and account_lock_expiration are new to PLM 4.x.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Login_name	Char(A) 47	no		
Person	Integer(I)	no		Person(id)
Passwd	Char(A) 30	no		
Password_exp_date	Date(D)	yes		
Account_locked	Char(A) 1	yes		
Account_lock_expiration	TimeStamp(@)	yes		

Indexes **Field(s)**
Primary Key login_name
LoginName login_name (secondary index is not case sensitive, while primary key is case sensitive)
person person

Initial Insert Values
{(Admin, 1, Admin)}

Dependent Tables
None

2.1.2.26 LOG_HIST (Login ID History Table)

The LOG_HIST table maintains a history of all login IDs.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	no		
Login_name	Char(A) 47	no		

Indexes

Primary Key

LoginID

Field(s)

id

login_name

Dependent Tables

ID_PWD

2.1.2.27 MNG_GRP (Manager Assigned Group Table)

The MNG_GRP table maintains a list of managers for each group. Each record in the table identifies a group and a person.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Person	Integer(I)	no		
Igroup	Integer(I)	no		

Indexes

Primary Key

Igroup

Field(s)

id

igroup

Foreign Keys

igroup

person

Table(Field)

Igroup(id)

Person(id)

Dependent Tables

None

2.1.2.28 PERSON (Person Table)

The PERSON table stores information about each person in PLM. Users can add, modify, and delete records in this table from within PLM.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
id	Integer(+)	no		
e_mail	Char(A) 80	yes		
id_number	Char(A) 11	no		
first_name	Char(A) 20	no		
middle_name	Char(A) 20	yes		
last_name	Char(A) 30	no		
active_status	Char(A) 1	no	T	{T, D, F}

learning_style	Integer(I)	yes		
security_level	Integer(I)	no	2 (Student)	{1, 2}, no referential check: Seccode(id)

Indexes	Field(s)
<i>Primary Key</i>	id
Name	last_name, first_name, middle_name
IdNumber	id_number
SecurityLevel	security_level

Initial Insert Values
 {(1, NULL, 00000000000, Administrator, NULL, Administrator, T, NULL, 1)}

Dependent Tables
 LOGIN
 STU_GRP
 STU_SCHD
 TRAINREC
 MNG_GRP
 AUTHORIZ (for PLM 2.x only)
 UDFP

2.1.2.29 PWD_HIST (Password History Table)

The PWD_HIST table maintains a history of all passwords.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
id	Integer(+)	no		
password	Char(A) 15	no		

Indexes	Field(s)
<i>Primary Key</i>	id
Passwd	password

Dependent Tables
 ID_PWD

2.1.2.30 PROGRESS (Progress Table)

The PROGRESS table maintains a list of the allowed progress settings for a course. This table is not editable from within PLM but is used by the Student Course Schedule table to provide a list of textual descriptions for a student's progress in a course.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	no		
Gauge	Char(A) 24	no		{{1, Registration Pending}, (2, Not Started), (3, In Progress), (4, Completed, Test Pending), (5, Completed, Not Passed), (6, Student Complete)}

Indexes	Field(s)
<i>Primary Key</i>	id

Dependent Tables

None

2.1.2.31 REPORTS (Reports Table)

The REPORTS table maintains a list of the reports currently defined in PLM.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	no		
Rname	Char(A) 50	no		
Rfile	Char(A) 255	no		
Creator	Char(A) 80	no		
Rdate	Date(D)	no		
Description	Char(A) 255	yes		
user_defined	Char(A) 1	no	T	{T, F}
export_only	Char(A) 1	no	F	{T, F}
select_code	Char(A) 6	yes		

Indexes **Field(s)**
Primary Key id
ReportName rname

Initial Insert Values
{(All reports shipped with PLM)}

Dependent Tables
None

2.1.2.32 RUN_MENU (Menu Run Program Table)

The RUN_MENU table maintains a list of all the run menu items defined within PLM, both Student and Administrator. PLM builds the Run menu dynamically using this table. Each item in the table applies to the Administrator OR Student module but not to both. The user can add, modify, and delete records in this table from within PLM.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
id	Integer(+)	no		
security_level	Integer(I)	no	2 (Student)	{1, 2}, no referential check: Seccode(id)
menu_name	Char(A) 15	no		
command_line	Char(A) 255	yes		
working_directory	Char(A) 255	yes		

Indexes **Field(s)**
Primary Key id
MenuSecurityLevel security_level
MenuName menu_name

Dependent Tables
None

2.1.2.33 SECCODE (Security Code Table)

The SECCODE table maintains a list of security levels defined within the system. Currently, there are only two different levels of security. The Run_Menu table uses these codes to determine which items appear in PLM Administrator and which items appear in PLM Student. Each person also has a security level that determines if he can log into PLM Student and/or PLM Administrator.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
id	Integer(+)	no		
code	Char(A) 16	no		{{(1, Administrator), (2, Student)}

Indexes **Field(s)**

Primary Key id

Dependent Tables

None

2.1.2.34 SG_AUTH (Authorization Table)

The SG_AUTH table maintains a list of all authorizations in the system. The table stores information about the authorization, including the authorization type (Required / Elective) and the entity type (that is, was the course authorized for a group or just a single person). Note: the Authorization Table is not directly dependent on the Course Table. (This relationship is explained in section 2.1.2.4 - *the Catalog Item table.*)

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
entity_id	Integer(I)	no		Igroup(id) or person(id)
course	Integer(I)	no		Cat_item(id)
entity_type	ShortInt(S)	no		{0, 1}. 0 = person, 1 = instruction group
authorized_by	Char(A) 80	no		
complete_by	Data(D)	yes		
required_by	Char(A) 80	yes		
why	Char(A) 255	yes		
is_required	Char(A) 1	no		

Indexes **Field(s)**

Primary Key entity_id, course, entity_type

AuthorizedBy authorized_by

RequiredBy complete_by

(All foreign keys)

Foreign Keys **Table(Field)**

course course(id)

Dependent Tables

None

2.1.2.35 STU_GRP (Student Assigned Group Table)

The STU_GRP table maintains a list of which students are in which groups. Each record in the table identifies a group and a person.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Student	Integer(I)	no		Person(id)
Igroup	Integer(I)	no		Igroup(id)

Indexes
Primary Key
 IGroup

Field(s)
 student, igroup
 igroup

Foreign Keys
 student
 igroup

Table(Field)
 person(id)
 igroup(id)

Dependent Tables
 None

2.1.2.36 STU_SCHD (Student Course Schedule Table)

When a student is registered (or registers) for a course, a record is created in the Student Course Schedule table. This table is then updated with information about the student's current progress in the course. When a student completes a course, the information about the student's performance in the course is placed into the Training Record table, and the record from the Student Course Schedule table is deleted. The connect_option field is new with PLM 4.x.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Course_offering	Integer(I)	no		coroff(id)
Student	Integer(I)	no		person(id)
Progress	Integer(I)	no	2 (Not Started)	{1..6}, no referential check: progress(id)
Total_time	ShortInt(S)	no	0	Value >= 0, Minute
Access_count	ShortInt(S)	no	0	Value >= 0
Activities_completed	ShortInt(S)	yes		
Date_started	Date(D)	yes		
Last_access	Date(D)	yes		
Connect_option	Char(A) 1	yes		For downloaded CBT Systems courses, this contains a "P"; otherwise it is null.

Indexes
Primary Key
 progress
 Person

Field(s)
 course_offering, student
 progress
 student

Foreign Keys
 course_offering
 student

Table(Field)
 coroff(id)
 person(id)

Dependent Tables
 None

2.1.2.37 System (Configuration Table)

The System table stores customizable user settings.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Section	Char(A) 20	no		
Item	Char(A) 20	no		
Person	Integer(I)	no	0	
Value	Char(A) 255	yes		

Indexes **Field(s)**
Primary Key section, item, person
SectionPerson section, person
Person person

Dependent Tables
None

2.1.2.38 TRAINREC (Training Record Table)

Each record in the TRAINREC table is a complete training record. This table saves information about how the student did in the course, when the student completed the course, and when the student is required to recertify. (Recertification is based on the course and is stored initially in the Course table.) Note: if the recertification time is changed for the course, it is not automatically changed in the TRAINREC table, although the Administrator module provides the option of automatically updating existing recertification periods. Therefore, people who complete the course before the course's recertification period is changed will have a different recertification period than people who complete the course after the recertification period is changed.

Another note about this table is that course information is based on the Catalog Item History table, not the Course table. (See section 2.1.2.3 - the CAT_HIST table description for more details.)

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	no		
hist_catalog_item	Integer(I)	no		cat_hist(id)
Person	Integer(I)	no		person(id)
Status	Integer(I)	no		{1..5}, no referential check: cmp_stat(id)
total_questions	ShortInt(S)	yes		Value >= 0
number_correct	ShortInt(S)	yes		Value >= 0
percentage_score	ShortInt(S)	yes		Value between 0 and 100
date_completed	Date(D)	no	Today	
recorded_by	Char(A) 180	no		
recertification_period	ShortInt(S)	yes		Month
Total_time	ShortInt(S)	yes		Value >= 0, Minute
access_count	ShortInt(S)	yes		Value >= 0
number_answered	ShortInt(S)	yes		Value >= 0
test_time	ShortInt(S)	yes		Value >= 0
date_deadline	Date(D)	yes		

Indexes **Field(s)**
Primary Key Id
status status

CompletionDate date_completed
 CatHistPerson hist_catalog_item, person
 (All foreign keys)

Foreign Keys **Table(Field)**
 hist_catalog_item cat_hist(id)
 person person(id)

Dependent Tables
 TrainUnt

2.1.2.39 TRAINUNT (Training Unit Table)

Each record in the TRAINUNT table is a complete unit training record. This table saves information about how the student did in the course unit and when the student completed the unit.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(+)	No		
trainrec_id	Integer(I)	No		trainrec(id)
unit_id	Integer(I)	Yes		unit(id)
Status	Integer(I)	No		{1...6}, no referential check: cmp_stat(id)
total_questions	ShortInt(S)	Yes		Value >= 0
number_correct	ShortInt(S)	Yes		Value >= 0
percentage_score	ShortInt(S)	Yes		Value between 0 and 100
date_completed	Date(D)	No	Today	

Indexes **Field(s)**
Primary Key Id
 Trainrec_id trainrec_id
Foreign Keys **Table(Field)**
 trainrec_id trainrec(id)

Dependent Tables
 None

2.1.2.40 UNIT (Unit Table)

Each record in the UNIT table is a unit record for a course.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
id	Integer(+)	No		
course_id	Integer(I)	No		
ordinal	ShortInt(S)	No		
title	Char(A) 32	No		
has_test	Char(A) 1	Yes		

Indexes **Field(s)**
Primary Key Id
 Course_id&Ordinal course_id, ordinal
 Course_id course_id

Foreign Keys

course_id

Table(Field)

Cat_Item(id)

Dependent Tables

None

2.1.2.41 UDFC (User Defined Fields for Courses Table)

The user not only adds data to the UDFC table but also modifies the structure of the table. Each time the user adds a UDF (using the PLM Utility program), he gives the UDF a name, data type, and (if applicable) a size. When a UDF for a course is created, a column is added to the User Defined Fields for Courses (UDFC) table. When a UDF is deleted, the corresponding column is removed from the UDFC table. It is in the UDFC table that the actual data for the UDF is stored. The Course table does not store the UDF data for a course. (Similarly, the Person table does not store the UDF data for a person.) Users can add up to 10 columns to the UDFC table (UDF11 - UDF15).

The Company table stores the field names for each of the UDFs. PLM assumes within the code that if the value for UDF1 - UDF15 in the company table is not null that a corresponding UDF exists in either the UDFC or UDFP tables.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(l)	no		no referential check: cat_item(id)

Indexes*Primary Key***Field(s)**

id

Dependent Tables

None

2.1.2.42 UDFP (User Defined Fields for Person Table)

Other fields (named UDF1 to UDF10) are added with the type and size assigned in the PLM Utility program. See section 2.1.2.41 - *User Defined Fields for Course Table* for more information.

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Id	Integer(l)	no		no referential check: person(id)

Indexes*Primary Key***Field(s)**

id

Initial Insert Values

{(1)}, Administrator

Dependent Tables

None

2.1.2.43 VERSION (Database Version Table)

The VERSION table consists of a single record that specifies the version of the database. This table is only available after version 3.0 of PLM. To determine the version of a PLM database, attempt to read this table. If it is not present, then it is a version 2.x database. If this table is present, then the first (and only) record in the table specifies the version. For Paradox tables, this is actually not a database table, but a file structured as an .INI file with the following structure (where Version=Major*256 + Minor, so Version=1024 for version 4.0):

```
[DB Version]
Version=1024
```

Access: Data entry - all fields

Field Name	Data Type/Size	Null	Default Value.	Check Constraint/Domain
	Paradox			
Major	ShortInt(S)	no		
Minor	ShortInt(S)	no		

Indexes

Primary Key

Field(s)

Major, Minor

Initial Insert Values

{(4, 0)}

Dependent Tables

None

2.2 Adding Custom Reports to PLM

After you have created the Crystal Report file (the .RPT file), you can add it to the Reports tab in the administrator module. To add the new report to the Reports tab, enter Admin, select the Reports tab, and click the Add button. When you add a report, you can also select the filters that will allow users to limit the data included in the report. To select filters, your report must include the related tables, and you must select the proper options on the Reports Selections Dialog.

The following table lists the filters and their associated table.

Letter	Filter	Version	Table
A	Completed Course	2	CAT_HIST.DB
B	Course Group	2	CGROUP.DB
C	Subject	2	CI_SUBJT.DB
D	Completed Status	2	CMP_STAT.DB
E	Course Offering	2	COROFF.DB
F	Offering Starting Date	2	COROFF.DB
G	Course Title	2	COURSE.DB
H	Student Group	2	IGROUP.DB
I	Instruction Method	2	INSTMTHD.DB
J	Keyword	2	KEYWORD.DB
K	License Rate	2	LIC_RATE.DB
L	Student's Name	2	PERSON.DB
M	Authorization (Required or Elective)	2	SG_AUTH.DB
N	Recertification Due Date	3	TRAINREC.DB
O	People who have completed a selected course	2	TRAINREC.DB
O	Completion Deadline	3	TRAINREC.DB
P	Completion Date	3	TRAINREC.DB
Q	Enroll Status	2	ENRLSTAT.DB
1	PLC File	3	
2	Authorized, Non-registered courses	3	
3	Authorized Courses (with Completion)	3	
4	Course Completion	3	

The **Letter** column shows the letter that is included in the **select_code** field of the REPORTS table. The **Version** column shows what version of PLM first allowed this filter.

The tables are in the database directory and are protected. You must enter the password (which is at the beginning of section 2.1) to access the database. The new report must be saved in the same directory as the database. To add a new report entry in Reports tab of the PLM administrator module, follow these steps:

1. Click the **Add** button. The **Add Report** dialog box is displayed.
2. Enter a report name to help you identify the report.
3. Enter the report file name you created with *Crystal Reports 7.0* (or *5.0* for PLM 3.x or *4.5* for PLM 2.x). The report file must be in the same directory as the database for Paradox. For Client/Server databases, it must be in the Reports directory (as defined by the PLM.INI file). You may use the **Browse** button to select the report file name.
4. Enter the creator's name. The default is the current user.
5. If you want to limit the data shown when printing the new report and the report contains at least one of the above tables, you may add filters by clicking the **Select** button. This displays a list of the various filters. Each checkbox selected will cause a **Selection** button to be displayed in **Run Report** dialog box. The **Selection** buttons allow an administrator to limit information included in the report. You may select up to four filters.
6. Enter the report's description.
7. If this report can only be used for exporting data, then check the **Export Only** checkbox. This causes the **Preview** and **Print** buttons to be disabled in **Run Report** dialog.

Additional Information on Report Filters:

Most of the filters are simple to understand; however, some require special handling. Before you create a report using any of these filters, you should examine the built-in reports that use these filters until you understand how the filters work.

Selection Formula: For all of the built-in reports, except for the Export-Only, there is a formula called **@Selection** which is used as the subtitle of the report. When the report is generated, this formula is replaced with a description of how the filters were used. For example, the description may display like "All Students, Selected Courses, Courses Completed between 10/1/1997 and 12/31/1997".

Description and Title Formulas: For all of the built-in reports, except for the Export-Only, there are two formulas that allow you to display the title and description of the report. These formulas are called **@Title** and **@RDescription**. When the report is generated, these formulas are replaced with the title and description of the report from REPORTS.DB.

'O' People who have completed a selected course: This filter (available in version 2.x only) allows the user to select a single course using a combo box. When the report is run, PLM generates a temporary table called C:\RPTTEMP.DB that contains one field per record. That field is an ID value from the PERSON table; IDs are included for all people who have completed the selected course. The report must include three special formulas as follows:

Formula Name	Description
@CourseNotTaken	Title of the selected course
@MallIdNotTaken	Mall Number of the selected course
@CourseIdNotTaken	Course ID of the selected course

'N' Recertification: This filter allows the user to select a single course using a combo box. When the report is run, PLM generates a temporary table called C:\RPTTEMP.DB that contains one field per record. That field is an ID value from the PERSON table; IDs are included for all people who have completed the selected course.

'R' Required courses that are past due: This filter allows the user to specify a date by which courses should have been completed. When the report is run, PLM generates a temporary table called C:\RPTTEMP.DB

that contains fields as described below. Records are included for required authorizations that specified a completion deadline but had not been completed as of the specified date.

Field Name	Type	Description
Id	Integer(I)	Person ID – ID into the PERSON.DB table
Course	Integer(I)	Course ID – ID into the CAT_ITEM.DB table
GroupId	Integer(I)	Group ID – ID into the IGROUP.DB table. 0 for personal authorizations
Group	Char(A) 60	Group Title – Title field from the IGROUP.DB table
Authorized_by	Char(A) 80	Same as in SG_AUTH.DB
Complete_by	Date(D)	Same as in SG_AUTH.DB
Required_by	Char(A) 80	Same as in SG_AUTH.DB
Why	Char(A) 255	Same as in SG_AUTH.DB

‘1’ PLC File: This is not a filter. It specifies a special report that generates a PLC file. The built-in report that uses this select code is **plcfile1.rpt**, along with **plcfile2.rpt** and **plcfile3.rpt**. The last character of the filename (before the **.rpt** extension) specified in the REPORTS.DB table must be a **‘1’**. The other two reports must end with **‘2’** and **‘3’** respectively. Reports that use this select code should be Export-only and will default to using Text format. PLM will run all three reports in succession and then compile the three results into one output file.

Modifying Built-In Reports:

Some of the features described above are unique to Built-In reports. Normally, you cannot change the selection filters, the creation date, or the creator of Built-In reports. A special keystroke combination will allow you to change these values for Built-In reports. This special keystroke combination will also allow you to toggle whether or not the report is treated as a Built-In report. After you have displayed the Modify Report dialog (by clicking the **Modify** button from the Reports tab in the Administrator module), press Ctrl+Alt+F2, and you will be able to modify those values.

Problem solving:

You may receive an error message from the PLM Administrator module when you try to preview or print the new report.

Error message: Error: 515 Error in formula

- Make sure that you selected the correct checkboxes for the filters in **Report Selections** dialog box.
- Make sure that the tables in the report are correct.
- Check the syntax of all calculated fields.
- Check for logic errors in all calculated fields.

Error message: Report file does not exist

- Make sure that the new report file is in the database directory.
- Make sure that the file name is properly spelled in the **Modify Report** dialog box.

3. Communication with Computer Based Training (CBT) Courses

3.1 Introduction

This section gives you a complete list of PLM's data exchange features and detail descriptions of the supported functions and methods. While PLM provides management of training courses and employees training records, the APIs in this section let course developers provide the ability for data to be transferred between their courseware and PLM using simple Windows mechanisms.

Even though PLM is a Windows-based application, this API details methods of data exchange for both DOS and Windows-based courseware. This is facilitated through executable program command-line parameters, Windows DLLs, and DDE communication mechanisms. Command-Line parameters and DDE are used for the transmission of data from PLM to courses, where DLL and DDE communications are used for the transmission of data from the course to PLM.

The combination of Command-Line (data transmission from PLM to the course) and DLL communication (data transmission from the course to PLM) should be used for all DOS-based courses and existing Windows-based courses. DDE communication (two-way data transmission) should be used for all newly created Windows-based courses.

3.2 Conventions

Descriptions of PLM's APIs use the following conventions in order to make key words prominent:

PLM	Refers to Pinnacle Learning Manager for Windows.
<code>SampleCode</code>	Words in Courier font indicate code sample.
Keywords	Words in bold indicate PLM communication items, functions, or methods.
<i>Parm</i>	Words in italics indicate variable names.

3.3 Concept Glossary

Knowing and understanding the following terms and concepts will be helpful when using this manual.

API (Application Interface) - A set of definitions, functions, and methods that are made public to courseware developers. If followed, computer-based courseware and PLM will properly interact in starting and stopping the CBT and proper transmission of course and student data.

CBT (Computer-Based Training) - A training course that is presented entirely on a computer.

DLL (Dynamic Link Library) - A separate file containing functions that can be called by other Windows programs and DLLs to perform specific tasks.

DDE (Dynamic Data Exchange) - Is one of several mechanisms of inter-process communication supported by Windows. DDE is based on the standard messaging system built into Windows.

Command-Line - A character string that may contain any number of initialization or operational parameters. A Command-Line is passed to a program or DLL at startup or load time.

UNC (Universal Naming Convention) - A path-naming convention that provides anonymous connection requests to a shared networked resource or file. In most cases, the users will need to have an MS-DOS drive mapped to the shared network resource or file, but it may be a different drive letter for different users.

3.4 An Overview of PLM

PLM provides a centralized training solution for the diverse training needs of most companies. PLM is a fully networkable tool providing scheduling and tracking for computer-based, self-study, and scheduled training. For computer-based training (CBT), PLM schedules, tracks, and delivers those courses to the desktop of every employee within the company. Using PLM makes “Just-in-Time-Training” a reality.

PLM provides for the creation of a complete course catalog, listing all training courses available to employees. This includes seminars, classroom training, self-taught courses, and computer-based training. Scheduling of employees for training courses is managed individually, by department, or company wide. At the employee level, taking a computer-based course is as easy as starting any office application. PLM manages, launches, and records each employee’s progress and completion results for all types of training.

PLM is based on an open design, allowing existing computer-based courseware to be used within its system. Automatic score and progress tracking can be obtained on existing computer-based courseware through a simple retrofit using this API. The PLM API provides for seamless integration of course launching and central data collection of employee progress and test data. The Pinnacle Learning Manager™ provides a complete integrated solution for all training needs.

You can use the API described in this section to retrofit existing or create new courseware to be PLM-aware. Whether retrofitting old courses or creating new courses you can facilitate communication with PLM by either:

1. Embedding the necessary communication within the course or runtime.
or
2. Creating an external bridge program that embeds the necessary communication between the course and PLM.

Embedding has the advantage that no additional files are required by the user, and the course can bypass the extra login sequence when running under PLM since the course can auto-detect that PLM is present. The disadvantage to embedding is that it requires changes to the course or runtime engine of the course. This generally makes embedding an effective strategy for new courses, or for new versions of runtime engines, but less desirable for existing courseware, where changes to the core runtime or course are impractical or impossible.

External bridges have the advantage that no changes are required to the courseware. The disadvantage to external bridges is that the users must install the additional files required by the bridge in addition to installing the course itself. This is not a particularly difficult problem in traditional networked environments where the bridge files can reside on a shared server but presents more challenges when the courses are CD-ROM based or managed over the Web.

We provide examples of both methods of communication and will answer any questions you may have regarding your particular situation. No matter how you choose to communicate with PLM, you will find that making courseware PLM-aware is relatively easy to do.

3.5 PLM Employee-Course Data Tracking

PLM is a database application, allowing a company to centralize scheduling, distribution, and record keeping of company and corporate-wide training. PLM operates with Paradox, Oracle, Microsoft SQL Server, and other enterprise-wide relational database implementations.

Before looking at the API, you should become familiar with PLM’s employee and course database fields that computer-based courseware will need to interface with. The following is a list of the PLM employee and course database fields, data types, and their descriptions.

Data Field	Data Type	Description
Status	Integer	The status of this course/test. This is an enumerated type; definition follows.

Total Questions	Integer	Total number of final test questions asked.
Number Answered	Integer	Total number of final test questions answered.
Number Correct	Integer	Number of final test questions answered correctly.
Percentage Score	Integer	Percentage answered correctly.
Date of this Completion Entry	String	Date of this completion entry; format: mm/dd/yyyy.
Total Time in Course	Integer	Total time the employee spent on this course. This includes test time. The time is in <u>whole minutes</u> .
Total Time in Test	Integer	Total time the employee spent on a test. The time is in <u>whole minutes</u> .
Total number of Activities in Course	Integer	Total number of activities in the course. This is used for employee progress within the course.
Number of Completed Course Activities	Integer	Employee progress information listing the total number of course activities that have been completed.

3.5.1 Student Training Course Status

Another important aspect of integration of PLM to computer-based courseware is the status or progress of an employee taking a course. Each time an employee works on a computer-based course, PLM will automatically record and track their progress. Progress records also include final test results. The employee's training course status is just one of the progress indicators maintained by PLM.

Initially, when an employee registers for a computer-based course, their course status is set to "Not Started". After the course has been started, and until it is completed, passed, or the employee's registration in the course is dropped or removed, their course status is "In Progress". This includes opening and then closing the first page of a course without any other interaction. Subsequent status changes are assigned based on the employee's progress in the course and final test results.

The recognized employee training course statuses are listed below. Each status has an integer equivalent that PLM recognizes. It is this integer value that the course must use in communication with PLM.

Status	Integer Value	Description
Cancel	0	This launch of the course did not succeed.
In-progress	1	The course presentation material has been started but has not been completed. A record marked as such is equivalent to a progress bookmark.
Completed	2	The course presentation material has been completed and the course does not have a final test. The course is complete.
Completed_test_pending	3	The course presentation has been completed and a final test is required and available, but the test has not been taken.
Passed	4	The course final test has been completed and the employee has met the requirements established in the course for passing the final test.

Failed	5	The course final test has been completed and the employee has not met the requirements established in the course for passing the final test.
PLM Completion	6	[PLM 3.0 and later] The course has been completed and the final test has been taken, but the course makes no judgment as to whether the student passed or failed. PLM will compare the percentage score with the "Percent to Pass" associated with the course in PLM to determine if the student passed or failed.
Waived	14	[PLM 3.0 and later] The course has been waived.

3.6 An Overview of the PLM APIs

Normally, during the startup of most CBTs, dialog interaction between the user and the CBT is used to control access. Having the CBT PLM-aware removes the need for the course to query the user at startup for user identification. When a CBT is launched from within PLM's integrated environment, the login and password used to gain access to PLM may be passed to the CBT for its use. This information is passed using Command-Line arguments, or the CBT itself can query PLM for this information using DDE.

Conversely, at the completion of most CBT courses, progress and test scores are maintained in a file or database that is proprietary to the course. In order to maintain accurate and up to date employee records, PLM needs this employee-course information within its own database. This requires that at closure of the course, this information be transmitted to PLM's database by the computer-based course. Existing CBTs can be retrofitted to use a DLL for providing the necessary data exchange and new development of new CBTs is encouraged to use DDE as the mechanism for data exchange from the course to PLM.

To facilitate seamless CBT course launching supported by automatic employee-course information retrieval, PLM provides this API for establishing communication with DOS and Windows-based course applications. Command-Line parameters and DDE communication form the basis of this API. These communication mechanisms are briefly discussed below.

Command-Line parameters and DDE communication are used for the exchange of data from PLM to CBTs. DDE communication is used for the exchange of data from CBTs to PLM. It is intended that Command-Line communication be used for existing and DOS-based CBTs, and DDE communication will be used for all newly created Windows-based CBTs.

If you want your CBT to allow PLM to manage, launch, and record progress and test data, you must enable your courseware to communicate with PLM using this API. You will find that the communication connection between PLM and your CBT is relatively simple.

3.6.1 Using Command-Line Parameters

Command-Line parameters are for PLM to CBT communication. This method provides for run-time employee and course data to be passed from PLM to the course using simple parameter replaceable parameters. These replaceable parameters cover employee login, employee and course demographics, and file location information. This method is designed to work with both MS-DOS and Windows-based CBT applications.

3.6.3 Using DDE

The Dynamic Data Exchange (DDE) functions support several request items and three poke items. These DDE items cover employee login, employee and CBT demographics, and file directory information. For all DDE communication, PLM is the server and the CBT course is the client. Because PLM is the server for all communication with the client, PLM supports 'poking' data from the client (CBT) to the server (PLM). It is currently not proposed to use the Clipboard as a means for connection or data transfer with the server. The DDE functions are designed to work only with Windows-based CBTs.

3.7 An Overview of Program to Program Communication

Windows provides several ways in which programs can exchange data with each other. The primary mechanism for exchanging data is through the Clipboard. The Windows Clipboard is normally controlled by the user and not by a program. Anytime you use the cut and paste routines of a Windows program, you are using the Clipboard for data exchange. Since the Clipboard is under user control, it is not suitable for program to program data exchange. A source program can place data on the Clipboard and the user can change it, or even delete it, before the target program has the opportunity to retrieve it.

Direct program to program data exchange is more commonly performed using the Microsoft formalized method of Dynamic Data Exchange (DDE). DDE works on the same principle as all Windows programs: global memory allocation and message passing. Simply stated, a source application, using DDE, places data into global Windows memory and then posts a message to a target application indicating that the data is available and where in memory it is located. DDE uses the Client/Server model with the *items* of the conversation containing the data values. Normally the data flow is from the server to the client, but DDE has provisions for data being sent the other way, from the client to the server, called a *data poke*.

Another method of data exchange is program Command-Line parameters. When a CBT course is initially started, data can be passed from a source program to a target program using Command-Line parameters. In this type of data exchange, the data is parameterized and passed as a single character string to the target program. At startup, the target program can parse the parameterized string and initialize local variables based on its contents. Command-Line parameter passing is available for both MS-DOS and Windows-based programs. There are two major disadvantages of Command-Line data exchange:

- It is only one way (source program to target program).
- It only occurs once (when the target program is first started).

3.8 PLM Support for UNC Redirection

What is Universal Naming Convention (UNC) redirection? UNC is a path naming convention to provide anonymous connection requests to a network server or shared resource. In general, UNC provides the ability to use a network resource or file without formally connecting to it and having a mapped MS-DOS drive. UNC paths and filenames have the following form:

```
\\ServerName\Share-VolumeName\Directory\FileName.ext
```

The *ServerName* identifies the name of the shared computer or network server. The *Share-VolumeName* identifies the name of the shared file, resource, or the physical network volume name where the file or program resides. The *Directory* is the relative path to the file or program.

PLM supports the use of UNC paths for identifying the location of files and programs. A UNC path may be used for access to a program executable, working directory, and for Command-Line arguments. Support is limited to the following:

1. Full support to execute a network program without formally connecting to the networked server. This functionality is provided through the Microsoft Windows operating system.
2. For working directories, you must be connected to the networked server and have access to the specified directory. By definition, a working directory becomes the current directory for the executing application. This requires that you be able to “change to” the working directory. This command cannot be executed unless you are connected to the network server and have access rights to the specified directory.
3. For Command-Line parameters, PLM has the ability to convert UNC names to drive letter and pathname, since many applications do not support UNC names. This allows courses to execute from the server using different drive mappings. By default, PLM will attempt to convert UNC names in command-lines to mapped drive letters.

If you want PLM to convert a UNC path to a path using a mapped drive (the drive may or may not be mapped on the client system) you may precede any UNC names in parameters with ‘%A’. This will force a drive to be mapped dynamically if a drive is not currently mapped for the specified path. If the drive was mapped dynamically then it will be unmapped when the program that is launched terminates.

If the program you are executing supports UNC names, you can precede any UNC names in parameters with a '%' character then PLM will pass the UNC name to the course without any translation. Alternately, you can place the UNC name as the working directory and provide a relative path in the Command-Line. Once again, the program you are executing must support relative paths.

4. If you are using UNC names, and the file or program that you are trying to access is protected with a password, you will not be prompted for the password nor will the password be looked up. You must have the appropriate access rights for the file or program execution. In such cases, you must be connected to the network server prior to using the UNC name.

3.9 Program Command-Line Communication

3.9.1 Summary

Most CBT courses provide support for run time parameters to be passed during startup. These parameters provide specific user and course information that is known only at run time. Not only does PLM have the ability to start a CBT course but also it can provide run time information to the course during startup. This is accomplished by using PLM replaceable parameters in the CBT course Command-Line string.

3.9.2 Replaceable Parameters

For CBT courses, PLM stores the location and name of the course executable, the working directory, and the CBT learn and review mode command lines. When a course is started from within PLM, the CBT Command-Line is examined and all recognized replaceable parameters are replaced by their run time equivalents. To use PLM to pass to the CBT run time data on its Command-Line, you need to provide:

- the CBT the ability to recognize and use data passed
- documentation to the user stating the correct format of your CBT Command-Line based on replaceable parameters:

The CBT command-line is a character string containing two types of objects:

- Plain characters that are copied verbatim to the output stream.
- Replaceable parameters that are replaced by run time data provided by PLM.

The recognized replaceable parameters have the following form:

%type-char

The '%' character is a sentinel a single character replaceable parameter. The *type-char* is the single character replaceable parameter denoting the type of run time information to be inserted. (Note: the *type-char* is not case-sensitive.) If the *type-char* is recognized as a PLM replaceable parameter, both the '%' character and the *type-char* are replaced by the requested run-time data. In the event that PLM does not have the requested data, an empty string is used as a replacement for the *type-char*. For example, suppose that when an employee was entered into the PLM system, their middle name was omitted. The replaceable parameter for a person's middle name is %Y. Since the requested data is not present in the PLM database, the '%Y' replaceable parameter is replaced by an empty string.

In order to include a '%' as a valid character and still have the run-time data inserted in your CBT command-line, follow the first '%' with a second '%'. For example, you need the Employee Login ID passed to your CBT course and you use the '%' character as the parameter announcer in your CBT. The actual string that you need is "%12345". To accomplish this, you would use '%%I' as your replaceable parameter in the CBT command-line. The first two '%' characters would be replaced with a single '%' character, and the '%I' would be replaced by the employee login ID, e.g.: %12345.

Another example of using replaceable parameters in the CBT command-line is as follows:

Command-Line: -mCourse -n%L -i%P

At run-time the Command-Line is changed and passed to the course as follows:

-mCourse -n123456789 -icherries

If a UNC name appears in a command line, it will be replaced with a drive mapping. This includes all UNC's; not only those that specify the executable program, but also those that may appear as parameters. A UNC is identified by a series of characters that begins with a double backslash (“\\”) and which can be converted to a valid mapped drive. For example, if a command line is the following:

```
\\server1\vol1\test\cbt\course1.exe -m\\server1\vol1\test\cbt\module.cbt
```

and a drive is mapped as follows:

```
P: = \\server1\vol1\test
```

then the command line would be converted to:

```
P:\cbt\course1.exe -mP:\cbt\module.cbt
```

If you want a UNC name to be passed onto a CBT without being converted to a drive mapping, then precede it with a %.

The recognized type-chars, character size, replacement PLM data, and their descriptions are defined below:

Type-Char	Char Size	Data Replacement	Description
I	11	Employee ID Number	Identification number of the employee. This could be a social security number or any number to identify the person. This parameter may be used to allow an employee access to the course. This is <u>not</u> a required field for PLM.
L	47	Employee PLM Login ID	Employee login ID used to log into PLM. This parameter may be used to allow an employee access to the course.
P	15	Employee PLM Login Password	Employee password used to log into PLM. This parameter may be used to allow an employee access to a course. It is normally used in conjunction with the Employee ID Number or Employee Login ID parameter.
X	20	Employee First Name	First name of the employee. This parameter is used to supply information to the course for run-time registration of the employee into the course.
Y	20	Employee Middle Name	Middle Name of the employee. This parameter is used to supply information to the course for run-time registration of the employee into the course. This is <u>not</u> a required field in PLM.
Z	30	Employee Last Name	Last name of the employee. This parameter is used to supply information to the course for run-time registration of the employee into the course.
M	24	Course PLM Mall Number	Universally unique mall number for the course. This number is generated and maintained by PLM. This parameter may be used to identify the course.
S	8	Employee Learning Style	Learning style of the employee. This parameter is used to instruct the course to present its material and tests using this learning style. This is <u>not</u> a required field in PLM.
C	1	CD-ROM Drive	The first logically mapped MS-DOS drive letter that is for a CD-ROM drive. If the local PC has a CD-ROM, then it will always return the local drive letter over a network CD-ROM Drive.
F	80	CBT Command Line	This is the complete MS-DOS command line that launched the CBT. This includes the path to the executable, executable file, and CBT

			Command-Line.
D	80	CBT Working Directory	CBT working directory. If the working directory uses UNC, then the MS-DOS drive letter is used (see UNC support).
U	255	URL of PLM Online	This is the complete URL of PLM Online.
A		Dynamic Drive Map	%A is not actually a replaceable parameter like the others listed above. Rather, it precedes a UNC path to indicate that the UNC path is to be converted to a path using a mapped drive, and that the drive letter is to be dynamically mapped if no mapped drive exists for the specified path.

3.10 DDE Communication

3.10.1 Summary

For all DDE communication, PLM is the server and the CBT course is the client. Because PLM is the server for all DDE communication, PLM supports 'poking' data from the client to the server. It is currently not proposed to use the Clipboard as a means for communication initialization or data transfer between the server and client.

All data formats are CF_TEXT. The DDE service and topic connection strings are as follows.

Service:	null
Topic:	StudentCourseInfo

3.10.2 DDE Conversation

The PLM Server follows Windows standard DDE conversations. Conversations begin when the WM_DDE_INITIATE message and WM_DDE_ACK (positive and negative) are posted for data requests and poked data.

For poked data, to allow the CBT client to know if the data was formatted correctly, PLM has supplied a "PokeState" item. After the client has poked the data to the server, the client can then request the value of the PokeState item. If the first character of the PokeState item value is '1' then the poked data was formatted correctly. If it is '0', then there was an error in the format of the data, in which case the received data text is appended to the 0, preceded by a dash, '-', e.g., 0-(1|2|3). The primary use of the PokeState item is during debugging of the CBT course DDE communications and not as a run-time check.

3.10.3 DDE Request Items

The return string value from DDE request items includes trailing line-feed and carriage return characters (ASCII character numbers 13 and 10). These characters have to be stripped by the client if they are not needed. All return formats are CF_TEXT.

This section describes the DDE request items. The version number specifies what version of PLM first supported the item. (All items supported in version 2 are supported in version 3 and later versions.) To determine the version, use the Version request item. If it is an empty string, then the version for PLM is 2.x.

Item Name	Version	Required?	Description
StudentNumber	2	Yes	Identification number of the student. This could be a social security number or any number to identify the person. This parameter may be used to allow an employee access to the CBT course.

StudentId	2	Yes	Student login ID used to log into PLM. This parameter may be used to allow a student access to the CBT course.
StudentPassword	2	Yes	Student password used to gain access to PLM. This parameter may be used to allow a student access to the CBT course. It is normally used in conjunction with the StudentID or Login ID parameter. The value returned for PLM 4.0 is a string containing the Student login ID (which typically may be used as a password for logging into CBT courses). For PLM versions prior to 4.0, it is a string containing the <u>non-encrypted</u> Student's PLM Login Password.
FirstName	2	Yes	First name of the employee. This parameter is used to supply information to the course for run-time registration of the employee into the course.
MiddleName	2	No	Middle Name of the employee. This parameter is used to supply information to the course for run-time registration of the employee into the course. This is <u>not</u> a required field in PLM.
LastName	2	Yes	Last name of the employee. This parameter is used to supply information to the course for run-time registration of the employee into the course.
StudentStyle	2	No	Learning style of the employee. This parameter is used to instruct the course to present its material and tests using this learning style.
MallNumber	2	Yes	Universally unique mall number for the course. This number is generated and maintained by PLM. This parameter may be used to identify the course.
CommandLine	2	Yes	The CBT executable and command-line. This is the exact call, executable path, name, and command-line used to start the CBT. Here the client can parse the return value and obtain the exact call components that launched the CBT.
WorkingDirectory	2	No	CBT working directory. If the working directory uses UNC, then the MS-DOS drive letter is used (see UNC support).
CourseTitle	3	Yes	Course title from the PLM database.
CourseId	3	No	Course Id from the PLM database.
Description	3	Yes	Description of the course.
CEU	3	No	Number of CEUs (Continuing Education Units) for the course.
PercentPass	3	No	Minimum percentage to pass the course.
TotalActivities	3	No	Total number of activities in the course.
CompletedActivities	3	No	Number of activities completed thus far.
TotalTime	3	No	Total time (in minutes) spent in the course thus far.

CourseRecordNumber	3	Yes	Unique number (1-4,294,967,295) associated with the course.
StudentRecordNumber	3	Yes	Unique number (1-4,294,967,295) associated with the student.
Version	3	Yes	Version (<i>major.minor</i>) of PLM. This item may be used to determine what information PLM is capable of communicating to the course.
DateStarted	3	No	Date mm/dd/yyyy when the course was started
LastAccessed	3	No	Date mm/dd/yyyy when the course was last accessed.
AccessCount	3	No	Number of times the course has been accessed.
UDFP1 - UDFP10	3	No	Each of the ten user-defined fields for person records. The string returned consists of three items of information, separated by vertical bar characters (' '): Label Type Value Where Type is any of the following values: 1 = Text 2 = Integer 3 = Date (mm/dd/yyyy) 4 = Currency An example would be: 'Hire Date 3 04/30/1995'
UDFC1 - UDFC5	3	No	Each of the five user-defined fields for course records. See the description of UDFP1-10 above for return values.
UnitSupport	4	Yes	"Enabled" or "Disabled"
CourseUnitInfo	4	No	A set of strings describing the units of the course. Each unit is described by a string, and each string is separated by a Carriage Return/Line Feed pair. Each string consists of three items of information, separated by vertical bar characters (' '): Unit # HasTest? Title Where HasTest? is T (for True) or F for False). An example would be: '1 F Introduction' '2 T Installing PLM' '3 T Using PLM' '4 T Final Test'
ValidateManager	4	No	Reserved for Pinnacle Multimedia's use.
StudentEncryptPassword	4	No	Reserved for Pinnacle Multimedia's use.

3.10.4 DDE Poke Items

This section describes the DDE poke items. All return formats are CF_TEXT. The general syntax for ‘poking’ data from the client to the server is as follows: (Note: the open and close parentheses are required. If you are not poking any data, as in a null string, this is indicated by placing the vertical bar separators adjacent to each other, ‘||’, e.g.: (1|2|3||5|6)).

(<poke parameter>|<poke parameter>|<poke parameter>....)

Parameters:

Parameter	Required?	Description
Sstatus	Yes	The status of the student course-test information. This is the character representation of the integer values defined in the Student Training Course Status section.
Tquest	No	The total number of questions on the test.
Tansrd	No	The total number of questions answered on the test.
Tcorrect	No	The total number of correct answers on the test.
Tpercent	No	The test score percentage. If this is left blank, the value will be calculated based on TQuest and TCorrect.
Sdate	No	The date of this poke. The date format must be in mm/dd/yyyy. If not supplied, today’s date will be used by the PLM system.
Stime	No	The total time that the employee was in this session. The time is in whole minutes. If 0 or blank, then PLM will calculate the session time.
Ttime	No	The total time that the employee spent in the test. The time is in whole minutes.
Cacts	No	The total number of activities the employee has completed in this course.
Sacts	No	The total number of activities the course has. If this value is supplied, then this value will be changed immediately in the Course table and will affect all students currently and subsequently registered for the course.
SstuStyle	No	The employee learning style as determined by this course.

3.10.4.1 UploadSession

The UploadSession topic item provides progress information on both the course and final test in one call. The course progress information may optionally be followed by unit progress information. For units, the **Ordinal** or Unit Number is required to identify the unit. The **Title** is optional, but when the title is supplied for units that are not yet in the PLM database, the title is used to create the unit in the database. Unit information may be supplied for any number of units; the groups of information enclosed in square brackets should be separated by white space (CR/LF or a space).

Syntax:

(Sstatus|Tquest|Tansrd|Tcorrect|Tpercent|Sdate|Stime|Ttime|Cacts|Sacts|SstuStyle)
[Ordinal|Sstatus|Tquest|Tcorrect|Tpercent|Sdate|Title]

3.10.4.2 UploadTest

Although the UploadTest topic item is supported for PLM 4.0, its use is strongly discouraged. Pinnacle Multimedia makes no assurances that it will be supported in future versions of PLM.

The UploadTest topic item provides progress information on the course final test only.

Syntax:

(SStatus|TQuest|TAnsr|TCorrect|TPercent|SDate|TTime)

3.10.4.3 UploadTask

Although the UploadTask topic item is supported for PLM 4.0, its use is strongly discouraged. Pinnacle Multimedia makes no assurances that it will be supported in future versions of PLM.

The UploadTask topic item provides progress information on the course only.

Syntax:

(SStatus|SDate|STime|CActs|SActs)

4. Communication with PLM Online

4.1 Introduction

PLM Online can communicate with Windows-based and Web/HTML-based courses. Windows-based courses that communicate with PLM also communicate with PLM Online without modification through the same methods described in Section 3 (Command Line, DDE and Progress DLL).

Web/HTML-based courses can receive information from PLM Online through the Universal Resource Locator (URL) that launches the course. Web/HTML-based courses can return progress information to PLM using a URL.

4.2 Windows Program Communication with PLM Online

Windows-based programs communicate with PLM Online and PLM in the same way. However, there is an important consideration to remember when using PLM Online with Windows programs. The Windows program must exist in a place such that the relative path from each client machine is the same. For example:

- The Windows program is on a CD-ROM that each user puts into his CD-ROM drive.
- The Windows program is in the same directory on the C: drive for each user.
- The Windows program is on a server on the same LAN as the user.

4.3 Web/HTML Communication with PLM Online

4.3.1 Passing Information from PLM Online to a Web/HTML course

PLM Online passes information to Web/HTML courses through the command line. For a Web/HTML course, the command line is the URL to the course. The URL may contain any of the replaceable parameters described in section 3.9.2. For a course to pass back progress information to PLM, the command line must contain a MINIMUM of the following parameters:

- %u (the URL of PLM Online)
- %l (the Login ID of the Student)
- %m (the Mall Number of the Course)

The best way to pass this information in the URL is as part of the Query String. The Query String is everything to the right of the '?' symbol in the URL. For example:

```
http://www.mycompany.com/course.htm?url=%u&lid=%l&mall=%m
```

becomes:

```
http://www.mycompany.com/plm-cgi/plmweb.dll?  
url=http://www.mycompany.com/course.htm&lid=ADMIN&mall=A-PLMXDEMOPRODUCT-  
00001
```

PLM and PLM Online will treat all command lines as Windows application command lines unless the command line starts with URL:. For example, the URL from above would read:

```
URL: http://www.mycompany.com/course.htm?url=%u&lid=%l&mall=%m
```

4.3.2 Returning Information from a Web/HTML course to PLM Online

Web/HTML courses pass progress information to PLM Online using the HTTP communication method described in Section 4.4.

4.4 HTTP Communication with PLM Online

A course may pass progress information to PLM Online through the HTTP protocol using a Universal Resource Locator (URL). The main portion of the URL specifies the location of PLM Online. The query string portion of the URL specifies the student ID, the course Mall Number, and the progress information. For example:

```
http://www.mycompany.com/plm-cgi/plmweb.dll/UploadSession?lid=ADMIN&mall=A-  
PLMXDEMOPRODUCT-00001&ups=(1|||||||)
```

The keywords **lid=**, **mall=**, and **ups=** are required. The values **ADMIN** and **A-PLMXDEMOPRODUCT-00001** are values that PLM can pass to the course. The **http://www.mycompany.com/plm-cgi/plmweb.dll** value is also passed to the course from PLM. The **(1|||||||)** value is the course progress information in the same format used by the UploadSession DDE item described in Section 3.10.4.1; the course progress information may optionally include unit progress information enclosed in square brackets.

If the URL is accessed from within the same browser window that PLM Online is using, the user will see a message box in the browser window declaring that progress information has been received. If the URL is accessed outside a browser, the user will not be notified of any change but will see any changes reflected the next time he enters PLM Online.

5. Automatic Course Add

5.1 Introduction

The PLM Automatic Course Install API provides a mechanism to communicate course information to PLM. Setup programs can use the API to install course information into PLM. Course information files can also be distributed on a diskette or made available on an Internet site for download. PLM users can use the files to easily import course information.

There are two different ways to import courses into PLM: PLM Import Files and Database Import. This document describes the PLM Import Files, how to create them, and how to use the PLM Import Files to import course information into PLM.

5.2 Concepts

5.2.1 Import Methods

PLM provides three methods to retrieve the information stored in PLM Import Files. To use any of the three methods, switch to the Courses tab in the PLM Administrator, and then click the **Add From** button.

5.2.1.1 Run program to install a course on the system, then add the course to PLM

The user is presented with a browse dialog that allows him to choose a setup or install program. After the setup or install is complete, the user is returned to PLM and presented with a list of installed courses. The user then chooses which of the installed courses he wishes to add to PLM.

5.2.1.2 Add a previously installed course to PLM

The user is presented with a list of installed courses. He chooses which of the installed courses to add to PLM.

5.2.1.3 Add a course to PLM from a disk file

The user has a PLM Course List File containing information about schedules, lectures, courses, and so forth. This file may be on diskette, downloaded from the Internet, and so forth. When the user selects this option, a browse dialog is displayed. The user is prompted to select the PLM Course List File (*.PLL) containing the course information to add. The user is then presented with a list of courses contained in the PLM Course List File and may choose which of these courses he wishes to install into PLM.

In all three scenarios PLM Import Files are used to provide course information to PLM. In all cases the files have the same format. The difference is in how and when the PLM Import Files are created or updated.

5.2.2 PLM Import File Format

There are two files used during the import process, a course list file and a course information file. The course list file (PLL) contains a list of course information files. The course information file (PLC) contains the actual information about one or more courses.

5.2.2.1 PLL File

The PLL Course List File uses an INI file format. This format is easy to create during installation and can also be easily created by hand. An INI file has sections (a heading in square brackets) and items that follow the section. Items have a keyword followed by “=” (an equals sign) and a value. For example:

```
[Install Files]
Service=service.PLC
```

In this example, [Install Files] is the section and Service=service.PLC is an item. Service is the keyword, and service.PLC is the value. Note: if the value is just a filename, such as service.PLC, PLM will expect to find the PLC file in the same directory as the PLL File. Each PLL File may reference one or more PLC Files. Each

PLC File may contain one or more courses. The total number of courses that can be added to PLM at one time is 40.

5.2.2.1.1 COURSEAD.PLL

The default course list file used in PLM is a file named COURSEAD.PLL in the WINDOWS directory. When the user selects **Add a previously installed course to PLM**, PLM checks COURSEAD.PLL for any listed course information files. PLM also checks COURSEAD.PLL when the user selects **Run program to install a course on the system, then add the course to PLM**.

Key Concept

When installing CBT courses to be used with PLM, Pinnacle recommends that the setup program for the CBT installs a course information file with the course and writes the path to the course information file into the COURSEAD.PLL file.

5.2.2.2 PLC File

Each PLL file references one or more Course Information files (PLC file). The PLC file contains the actual information that will be imported into PLM. Each PLC file may contain information about multiple courses. Remember that the total number of courses for a single PLM Import File is 40.

5.2.2.2.1 PLC File Structure

The PLC file contains a [List] section. The [List] section has items that identify the courses in that PLC file. The PLC file also contains a section with a name identical to each of the items in the [List] section. For example:

[List]
Crisp Better Business Writing=Course
Franklin Quest Time Management=Course

[Crisp Better Business Writing]

[Franklin Quest Time Management]

In an actual PLC file, the sections [Crisp Better Business Writing] and [Franklin Quest Time Management] would each have items describing course information.

Scheduled courses may also have offering sections. An offering is a specific place and time at which the course is offered. The section name for an offering is the section name for the course with the additional word **Offering** followed immediately by a number. The numbers must be sequential and start at 1. For example, the section names for the first two offerings for the Franklin Quest Time Management course would be:

[Franklin Quest Time Management Offering1]

[Franklin Quest Time Management Offering2]

In an actual PLC file, each of these sections would be followed by the items that describe the offering.

5.2.2.2.2 PLC File Items for a Course

entry	Required	valid Information	description
Mall number		characters	Pinna le Mall number is a uni ue identifier for ea course offered t roug t e Pinna le Mall

¹Possible values are:

Y = Required (No Default)

N = Not Required

D = Required (Has Default Value)

Property	Required	Valid Information	Description
title		characters	title of the course
description		characters may contain multiple lines	description of the course
subject		characters	subject of the course
InstructionMethod		- default is	Method of instruction used. This is for informational purposes only and does not affect the way P M operates Audio Broadcast class computer Blended Conferencing Printed Material Seminar Synchronous Webcast Video
Registration type		- default is	type of registration. The values are: P M Registration: When a student registers for a course in P M no additional steps are needed to complete the registration process. In P M P M Registrations are called Internal Registration Third Party Registration: In some cases such as seminars students must not only register in P M but must also register with the third party who is offering the seminar. In this case the registration instructions should tell the student to call the third party to complete the registration. If the Registration type is set to P M will allow the student if they have completed the registration instructions. In P M Third Party Registrations are called External Registration
type		- default is	course type. The values are: Self Study: May be any type of course but does not have offerings at specific times/places Scheduled: May be any type of course as long as it has offerings at specific times/places. Note: this value cannot be changed for a particular course. If your P M file contains a course with a number already used in P M and if the type of the new course is different from the existing course P M will change the type of the existing course nor will any offerings be added if the existing course is self study
Company as		True/False default is True	Specifies that the course is available for students using this copy of P M. This value might be false if the course is only available at certain company sites or if it is a demo/BL course
duration		characters default is four	This is an approximate duration used to give the student an idea of average course duration
demo		True/False default is	When set to true indicates that there is some type of

Property	Required	Valid Information	Description
		False	demo available is information is displayed on the Provider tab in the course Information screen
Active		True False default is True	If a course is not active students cannot be authorized to take the course A course might be set to False if the license is expired or if the course is a demo computer based course
Rate type		- default is	type of license purchased for course the values are: computer: the course is licensed on a per user or station basis a workstation can have an unlimited number of users concurrent user: only a set number of users may access the course at the same time corporation: unlimited users at unlimited sites may access the course hour: Billing for course use is on an hourly basis site: unlimited users at one site may access the course user: Billing is based on the number of users who access the course time: Billing for course takes place each time the course is entered or started P M version does not enforce rates or licenses however later versions of P M do
Rate		dollar Amount default is	used to determine billing rate for Rate type P M version does not enforce rates or licenses however later versions of P M do
Available		number default is -	Applies to Rate type If set to - there is an unlimited amount of uses licenses available If set to indicates the number of uses licenses purchased P M version does not enforce rates or licenses however later versions of P M do
rate		number default is	number of additional licenses user can use on all available licenses have been used P M version does not enforce grace licenses or grace period however later versions of P M do
ratePeriod		number default is	the number of days the user can use grace licenses after all available licenses have been used P M version does not enforce grace licenses or grace period however later versions of P M do
completionAut		- default is	specifies who may mark that the course is complete Administrator only Administrator or course Administrator course or student
Registration Instructions		parameters may contain multiple lines	gives students information on how to register for a course In the case of seminar courses offered by third parties this should give the student instructions on how to register with the third party
PercentagePass		number	If set to the course determines whether the student passed If set ignore then pass fail will be determined when the Administrator enters a score
		parameters may be specified format is	number of continuing education units awarded for completion of course

entry	Required	Valid Information	Description
Learning Style		Parameters May	Provides students information about their learning style a course follows For example you might have two courses that present the same information one course uses a simulation and the other uses a workbook
Command Line		Parameters May	For computer-based courses this gives the complete command line to the executable including any command line parameters Note that this value may use either conventional or Windows names
Working Dir		Parameters May	Specifies the working directory for a computer-based course Note that this value may use either conventional or Windows names
Path		Parameters May	Complete path and filename of a Progress
Command Line		Parameters May	Initialization string for the Progress
Image		Parameters May	Complete path and filename of an image file in BMP format to be used for the course
IncludedMaterial		Parameters May may contain multiple lines	Information about what materials are included in the course or are needed for the course
InactiveReason		Parameters May may contain multiple lines	This information is displayed in P M if course is inactive
Provider		Parameters May may contain multiple lines	Company name address phone and so forth
Prerequisites		Parameters May may contain multiple lines	Description of prerequisites or general knowledge needed to understand the material in the course
RevertPeriod		Integer	Number of months after completion that course must be taken again
RevertNotify		Integer	Number of days before the Revertification Period ends to notify student to retake course
TotalActivities		Integer	Number of total activities chapters or sections in course
FinalTest		Boolean	Specifies if course is a final test
Keywords		Parameters May	Keywords to be added to the keyword table the student or administrator can use keywords to find courses in P M keywords are optional there may be as many as 255 keywords per course

5.2.2.2.3 PLC File Items for an Offering

entry	Required	Valid Information	Description
StartDate		mm dd yy	Date the course offering begins
StartTime		:mm	Time the course offering begins
EndDate		mm dd yy	Date the course offering ends

entry	Required	Field Information	Description
End time		:mm	Time the course offering ends
Location		Parameter may contain multiple lines	Provides students a description of the location of the course offering such as address, room, and so forth
Enrollment status		- default is	Enrollment status. The values are: an closed non open
Enrollment cap		Maximum number default is	Maximum number of students who can register for the course offering. If set to -1 there is no limit to the number of students who can register for the course offering.
Registration Instructions		Parameters may contain multiple lines. Default is taken from course Registration-Instructions	Provides students information on how to register for a course. In the case of seminar courses offered by third parties, this should give the student instructions on how to register with the third party.
Registration open date		mm dd yy default is today	Date students can begin to register for course offering.
Registration close date		mm dd yy default is indeterminate	Date when students can no longer register for course offering.

5.2.2.2.4 Fields with Multiple Lines

Some fields in the PLM database may have multiple lines. These fields include Description, Registration Instructions, Included Materials, Inactive Reason, Provider, and Prerequisites from the Course table, as well as Location and Registration Instructions from the Offerings table. Such fields are edited within the PLM Administrator module in multi-line edit boxes. As text is typed that exceeds the width of the edit box, lines will wrap automatically at spaces. Text that wraps automatically is stored in the database as a single line. However, if the user presses Enter while editing such fields, a Carriage Return-Line Feed combination (CRLF) is inserted in the field, and the line will always break at that point.

Fields that contain CRLFs are represented in the PLC file with multiple lines. For example, if the Provider is "Pinnacle Software Corporation<CRLF>P.O. Box 1409<CRLF>Draper, Utah 8402-1409" then the PLC file would contain the following lines:

```

Provider=Pinnacle Software Corporation
Provider2=P.O. Box 1409
Provider3=Draper, Utah 8402-1409

```

5.3 Sample PLL and PLC files

5.3.1 PLL File with 2 PLC File Items

```

[Install Files]
PPE=ppe.plc
Harassment=c:\pinnacle\harass.plc

```

5.3.2 PLC File with All Course Items

```

[List]
PPE=Course

```

[PPE]
MallNumber=PINN00000000000000000000
Title=Personal Protective Equipment
Description=This course describes basics types of personal protective equipment and their uses.
Subject=Industrial Safety
InstructionMethod=5
RegistrationType=2
Type=1
CompanyHas=True
Duration=45 Minutes
Demo=N
Active=T
RateType=6
Rate=15.00
Available=10
Grace=1
GracePeriod=30
CompletionAuth=2
RegistrationInstructions=From the Registered Tab, click Start Course.
RegistrationInstructions2=
RegistrationInstructions3=If the course does not appear on the Registered tab, call Bob Smith (x2554).
PercentToPass=0
CEU=01.50
LearningStyle=Presentation I Drill and Practice
CommandLine=//server/pinnacle/lt.exe -m//server/pinnacle/ppe/ppe.exe
WorkingDir=//server/pinnacle
DLLPath=
DLLCommandLine=
Image=c:\ppe.bmp
IncludedMaterial=Compliance Manual
InactiveReason=
Provider=Pinnacle Software Corporation 800-738-9800
Prerequisites=None
RecertPeriod=6
RecertNotify=30
TotalActivitles=1
FinalTest=True
Keyword1=Personal Protective Equipment
Keyword2=Safety
Keyword3=
Keyword4=
Keyword5=
Keyword6=
Keyword7=
Keyword8=
Keyword9=
Keyword10=
Keyword11=
Keyword12=
Keyword13=
Keyword14=
Keyword15=

5.3.3 PLC File with Minimum Course Items

[List]
PPE=Course
Forklift=Course

[PPE]
MallNumber=PINN00000000000000000000
Title=Personal Protective Equipment

Description=This course describes basics types of personal protective equipment and their uses.

Subject=Industrial Safety

[Forklift]

MallNumber=PINN00000000000000000001

Title=Forklift Safety

Description=This course teaches forklift safety and technique.

Subject=Industrial Safety

5.3.4 PLC File with Offerings

[List]

Forklift=Course

[Forklift]

MallNumber=PINNXXX3QCXXXXXXXXXYJXO02

Title=Forklift Safety

Description=This course teaches forklift safety.

Subject=Industrial Safety

[Forklift Offering1]

StartDate=07/01/96

StartTime=08:00 AM

EndDate=07/01/96

EndTime=04:00 PM

Location=Warehouse Dock 4

EnrollmentStatus=4

EnrollmentCap=30

RegistrationInstructions=Go to the elective tab, choose course, then click Register.

RegistrationOpenDate=06/01/96

RegistrationCloseDate=06/30/96

[Forklift Offering2]

StartDate=08/01/96

StartTime=08:00 AM

EndDate=08/01/96

EndTime=04:00 PM

Location=Warehouse Dock 4

EnrollmentStatus=4

EnrollmentCap=30

RegistrationInstructions=Go to the elective tab, choose course, then click Register.

RegistrationOpenDate=07/01/96

RegistrationCloseDate=07/30/96

5.3.5 Blank PLC File with All Items

[List]

MyCourse=Course

[MyCourse]

MallNumber=

Title=

Description=

Subject=

InstructionMethod=

RegistrationType=

Type=

CompanyHas=

Duration=

Demo=

Active=

RateType=

Rate=

Available=

Grace=
GracePeriod=
CompletionAuth=
RegistrationInstructions=
PercentToPass=
CEU=
LearningStyle=
CommandLine=
WorkingDir=
DLLPath=
DLLCommandLine=
Image=
IncludedMaterial=
InactiveReason=
Provider=
Prerequisites=
RecertPeriod=
RecertNotify=
TotalActivities=
FinalTest=
Keyword1=
Keyword2=
Keyword3=
Keyword4=
Keyword5=
Keyword6=
Keyword7=
Keyword8=
Keyword9=
Keyword10=
Keyword11=
Keyword12=
Keyword13=
Keyword14=
Keyword15=

[MyCourse Offering1]

StartDate=
StartTime=
EndDate=
EndTime=
Location=
EnrollmentStatus=
EnrollmentCap=
RegistrationInstructions=
RegistrationOpenDate=
RegistrationCloseDate=

5.4 Troubleshooting

5.4.1 PLM Import Messages

Error Message	Problem	Solution
R A F	Course has been successfully added to PLM Note: there may be some warning messages such as offerings not added or bitmap not found that are reported to the user but do not stop the course from being added	A
Course contains invalid information	Not yet implemented If possible will insert default value and notify user	
Course missing required information	There are four required fields for a course they are: Mall number Title Description Subject This message also suggests if a field is missing	Enter in the PLM file and provide any required information missing for the course
R A	This message only occurs if an error has prevented the course from being added The possible errors are: Required course information missing Duplicate Mall number found Replace so Unable to add course to database	Follow the solution provided for each of the specific problems
Duplicate Mall number found Replace so	Indicates that the PLM file contains a course with a Mall number that has already been used by another course in the system The user has given the option of replacing updating the existing course This message indicates that the user chose not to replace update the existing course	If you do not want to replace the current course but still want to add the new course to the system change the Mall number for the new course in the PLM file
Duplicate Mall number found Replace es	Indicates that the PLM file contained a course with a Mall number that has already been used by another course in the system The user has given the option of replacing updating the existing course This message indicates that the user chose to replace update the existing course	A
Image file not found	The path to the bitmap file does not point to a BMP file	This is not a serious problem If you want the bitmap for the course to be displayed you have two options: Change the path to the bitmap in the PLM file

		et the bitmap for the course in P M as you would for any other course
the course as a different type	An existing course is being updated modified its information from the P file the existing course and new course have different course types	P M handles this case for you If you need to have both the self-study and self-edited versions of the same course you should change the Mail number for the course in the P file and read the course into P M
no course add information found in P file	the P file contains no P files in the Install Files section or any existing P file entries have invalid path information	If the P file is empty there is no problem If not: Put the P file and P file in the same directory and change the path to the P file to be just the name of the P file change the path reference in the P file for the P file to point to the P file
no previously installed courses found	the P file being accessed contains no Install Files section or there are no P files listed in the Install Files section displayed only if the user chose the Add previously installed course option	there are two possible solutions: Put the P file and P file in the same directory and change the path to the P file to be just the name of the P file change the path reference in the P file for the P file to point to the P file
offering added successfully	Message also includes the offering title	A
offering not added offering contains invalid information	not yet implemented	
offering not added offering missing required information	there are five required fields for an offering they are: location start date start time end date end time the message also says if a field is missing note that this error only occurs if there is an offering in the P file that is missing information It is not displayed for courses that do not list offerings in the P file	Add the appropriate information to the P file
one or more offerings were ignored because course type is self-study	two scenarios may produce this error: An existing course is being updated modified its information from the P file the existing course is self-study and the new course is a self-edited course its offerings A new course is being added that has offerings listed but its course type set default is self-study or course type is self-study	P M handles this case for you If you need both self-study and self-edited versions of the same course: If the existing course being updated or replaced is self-study change the Mail number for the course in the P file and re-add the course to P M If the course being added is not explicitly set to self-edited edit the P file and add or modify the type to read self-edited

self study offering added successfully	In the case of self- study courses a self- study offering is automatically added to the course offerings	A
unable to add course to database	some unknown problem prevented the course from being added to the database	all technical support

6. Importing Existing Data

This section describes the interface used by the Pinnacle Learning Manager (PLM) for importing data from various sources. There are seven types of imports: People, Groups (of people), Courses (with optional associated offerings), Scores, Curricula, Authorizations, and Registrations. All of the imports are available through the Database Import option on the Tools menu of the Administration Module of PLM.

6.1 Overview

Importing data from other databases or software applications into PLM is accomplished through Windows DLLs (Dynamic Link Libraries). A few import DLLs are shipped with PLM; others may be supplied by third-party developers or may be available from Pinnacle Software Corporation.

The PLM database includes a table called IMPORT.DB. Each record in this table specifies a Source (a description of where the data will be imported from) and the full path name of the DLL. For example, an IMPORT.DB table could look like the following:

Source	DLL Name
ASCII comma-delimited	C:\PINNACLE\PLM\ASC_IMP.DLL
Human Resources Database	C:\HRDB\HRDBIMP.DLL
Novell NetWare	C:\PINNACLE\PLM\NWIMPORT.DLL
Registrar	C:\REGIST\RGSTRIMP.DLL

A single DLL may contain any or all of the import types. A single import source may use the same DLL for different import types, or it may use different DLLs for different import types. If it uses different DLLs, then the Source name will need to be changed so that it is different for each entry in the table.

To automate creation of entries in the IMPORT.DB table, a DLL called PLM_INST.DLL is available. See the section titled "Installation DLL" below for more details.

6.1.1 Changes for Version 4.0

For PLM version 4.0, the DLL must be a 32-bit DLL. You can make sure this is the case by starting up the PLM Administrator module, and then going into Import Data (on the Tools menu). You will see the various import types listed. To the right of the import name is a column labeled "Version". If it is 0 then you are using a 16-bit DLL or the path to the DLL is incorrect.

6.2 DLL Functions

The following function declarations are written in Borland C. Note: most of the functions return BOOL; a function should only return TRUE if it accomplished its task and is returning valid data in the structure for which a pointer was supplied. The structures used are defined in a header file called import.h. A similar source file for Delphi is available upon request.

Each of the structures passed to the DLL functions will be initialized with default values. Only fields that are listed as required below need to be supplied. Other fields may be left at their default values. For many of the fields of type WORD (or BYTE), there is a special value NULL_WORD (or NULL_BYTE) which indicates that the field will be set to a null string (not a numeric value of 0) in the PLM database.

```
#define _EXP __far __pascal _export
```

```
WORD _EXP Initialize(HWND ParentWindow, HINSTANCE Instance, CAPABILITES_INFO *pCapabilities);
```

```
DWORD _EXP GetCapabilities();
```

```
WORD _EXP GetDLLVersion();
```

```
BOOL _EXP GetPerson(PERSON_INFO *pPerson);
```

```
BOOL _EXP GetGroup(GROUP_INFO *pGroup);
```

```

BOOL _EXP GetGroupMember(MEMBER_INFO *pMember);
BOOL _EXP GetGroupManager(MANAGER_INFO *pManager);
BOOL _EXP GetCourse(COURSE_INFO *pCourse);
BOOL _EXP GetOffering(OFFERING_INFO *pOffering);
BOOL _EXP GetUnit(UNIT_INFO *pUnit);
BOOL _EXP GetScores(SCORE_INFO *pScores);
BOOL _EXP GetUnitScores(UNIT_SCORE_INFO *pUnitScores);
BOOL _EXP GetRegistration(REG_INFO *pReg);
BOOL _EXP GetCurriculum(CURRICULUM_INFO *pCurriculum);
BOOL _EXP GetCurriculumMember(CURRICULUM_MEMBER_INFO
    *pCurriculumMember);
BOOL _EXP GetAuth(AUTH_INFO *pAuth);
void _EXP Cleanup();

```

Every valid import DLL must have three functions: GetDLLVersion, Initialize, and GetCapabilities. In order to be useful, it should also have at least one of the functions that return a BOOL type.

6.2.1 GetDLLVersion

```
WORD _EXP GetDLLVersion()
```

GetDLLVersion is a required function that simply returns a word containing the DLL version number. It is anticipated that future versions of PLM will be able to support older DLLs, but a DLL is required only to support one level of functionality. Since PLM 4.0 DLLs are 32-bit DLLs, there is no support for older 16-bit DLLs used by previous versions of PLM. This function will be called before any other functions.

6.2.2 GetCapabilities

```
DWORD _EXP GetCapabilities()
```

GetCapabilities is a required function that simply returns a double word containing bit flags (see import.h) that indicate which of the imports the DLL is capable of performing. This function will be called before Initialize.

6.2.3 Initialize

```
WORD _EXP Initialize(HWND ParentWindow, HINSTANCE Instance, CAPABILITES_INFO
    *pCapabilities)
```

The Initialize function is required for all import types. It should configure any variables needed so that a subsequent call to GetPerson, GetCourse, and so forth, will get the first Person, Course, and so forth to be imported. The HWND and HINSTANCE are supplied in case the Initialize function needs to display any dialog boxes or other Windows user interface elements. The Capabilities flags indicates which of the import types the DLL will be asked to perform. The DLL will not necessarily be asked to perform all of the imports it is capable of performing. The imports will be performed in the following order: People, Groups, Courses, Scores, Authorizations, Registrations, Curricula.

The Initialize function returns values that indicate if the initialization was successful. Constants for the return values are defined in import.h.

- IR_OK means that the operation was successful.
- IR_CANCEL means that the user canceled the operation.
- IR_SILENT should be returned if an error occurred but the DLL has already reported it to the user.
- IR_ERROR means an error occurred and PLM should put up a generic error message indicating that the initialization failed.

If anything but IR_OK is returned, PLM will not make any more DLL calls (including Cleanup), so before the DLL returns any other value, it should first perform any necessary cleanup.

6.2.4 Import People (GetPerson)

BOOL _EXP GetPerson(PERSON_INFO far *pPerson)

The LoginID field must be supplied. The LoginID and IdNumber fields must be unique. If the FirstName and LastName fields are blank, then the FullName field is parsed to find the FirstName, MiddleName, and LastName fields; if the FullName field is also blank, the LoginID will be used for both the first and last names. If the IdNumber field is blank, the LoginID will be used for the IdNumber. If the Password field is left blank, the default password will be the LoginID. PLM calls GetPerson repeatedly to get users until FALSE is returned.

People must be imported before groups (see the *Import Groups* section below).

6.2.5 Import Groups (GetGroup, GetGroupMember, GetGroupManager)

BOOL _EXP GetGroup(GROUP_INFO far *pGroup)

BOOL _EXP GetGroupMember(MEMBER_INFO far *pMember)

BOOL _EXP GetGroupManager(MANAGER_INFO far *pManager)

For GetGroup, the Name is required, but the Description may be left blank. For GetGroupMember and GetGroupManager, either the LoginID, the ID Number, or the First and Last Name is required. PLM calls GetGroup to get a Group name and description, and then it calls GetGroupMember repeatedly until FALSE is returned. GetGroupManager will then be called repeatedly until FALSE is returned. Then GetGroup will be called again, followed by repeated calls to GetGroupMember and then GetGroupManager. This process is repeated until GetGroup returns FALSE.

Only group members who match a person already in PLM will be included in imported groups. Thus an import of People should always be done before an import of Groups. It is not required that a group has any members.

6.2.6 Import Courses (GetCourse, GetOffering, GetUnit)

BOOL _EXP GetCourse(COURSE_INFO *pCourse)

BOOL _EXP GetOffering(OFFERING_INFO *pOffering)

BOOL _EXP GetUnit(UNIT_INFO *pUnit)

For GetCourse, the Title is required. For GetCourseOffering, the Location is required. PLM calls GetCourse to get a Course with its associated information, and then it calls GetOffering repeatedly until FALSE is returned. It is not necessary for a course to have any offerings. In fact, if the course type is Self-Study, then the course should not have any offerings. Then GetCourse will be called again, followed by repeated calls to GetOffering to get the offerings for the course. This process is repeated until GetCourse returns FALSE.

6.2.7 Import Scores (GetScores, GetUnitScores)

BOOL _EXP GetScores(SCORE_INFO *pScores)

BOOL EXP GetUnitScores(UNITSCORE_INFO *pUnitScores)

PLM calls GetScores repeatedly to get scores until FALSE is returned. After each call to GetScores, GetUnitScores will be called repeatedly until FALSE is returned, and the GetScores will be called again.

People and courses must be imported before scores (see the appropriate sections above).

6.2.8 Import Authorizations (GetAuthorization)

BOOL _EXP GetAuthorization(REG_INFO *pAuth)

PLM calls GetAuthorization repeatedly to get authorizations until FALSE is returned.

People and courses must be imported before authorizations (see the appropriate sections above).

6.2.9 Import Registrations (GetRegistration)

BOOL _EXP GetRegistration(REG_INFO *pReg)

PLM calls GetRegistration repeatedly to get registrations until FALSE is returned.

People and courses must be imported before registrations (see the appropriate sections above).

6.2.10 Import Curricula (GetCurriculum, GetCurriculumMember)

BOOL _EXP GetCurriculum(CURRICULUM_INFO far *pCurriculum)

BOOL _EXP GetCurriculumMember(CURRICULUM_MEMBER_INFO far *pCurriculumMember)

For GetCurriculum, the Name is required, but the Description may be left blank. For GetCurriculumMember, the CourseTitle, the Course ID, or the Mall Number is required. PLM calls GetCurriculum to get a Curriculum name and description, and then it calls GetCurriculumMember repeatedly until FALSE is returned. Then GetCurriculum will be called again, followed by repeated calls to GetCurriculumMember. This process is repeated until GetCurriculum returns FALSE.

Only curriculum members that match a course already in PLM will be included in imported curricula. Thus an import of courses should always be done before an import of curricula. It is not required that a curriculum has any members.

6.2.11 Cleanup

void _EXP Cleanup()

Cleanup will be called at the end of the import. It provides the DLL with an opportunity to release memory or other resources that may have been allocated. Once Cleanup has been called, Initialize and the other import functions may be called again.

6.3 Installation DLL

A DLL called plm_inst.dll is shipped with PLM that is used to update the IMPORT.DB table with new or modified entries. When an application installs an import DLL, it should use plm_inst.dll to create a new entry for itself in the IMPORT.DB table.

The interface is as follows:

BOOL _EXP Initialize;
BOOL _EXP AddImportDLL(char *Source, char *DllPath);
void _EXP Cleanup;

If Initialize or AddImportDLL is unable to perform its function, it will return FALSE. Source can be up to 100 characters; DllPath can be up to 80 characters. If the Source matches an existing entry in IMPORT.DB, then the DllPath of the existing entry will be modified to match the new value. Cleanup should always be called if Initialize was called.

6.4 Import DLLs Shipped with PLM

Several import DLLs are shipped with PLM: Novell NetWare (NWIMPORT.DLL), ASCII (ASC_IMP.DLL), and various courseware imports.

6.4.1 Novell NetWare (NWIMPORT.DLL)

Capabilities: People and Groups

People are imported from the NetWare bindery. The Login ID is imported for each user. The ID Number field is set to the object ID for the User object in the bindery. The FullName field is set to the value of the Identification property for the user.

If the person doing the import has Supervisor rights, then the bindery's Login_control property is inspected to determine the value of the Account Disabled Flag. If a user's account is disabled, then that user will not be imported. If the person doing the import does not have Supervisor rights, then PLM will not have rights to inspect the Account Disabled Flag and thus all users will be imported, whether or not their accounts have been disabled.

6.4.2 ASCII Comma-delimited (ASC_IMP.DLL)

Capabilities: People, Groups, Courses, Scores, Authorizations, Registrations, and Curricula

The ASCII comma-delimited import provides a means for importing data from any source that can produce a standard comma-delimited ASCII file. Such a file consists of one record per line, with each line ending in a carriage return-line feed combination. Within each line, the fields are separated by commas. Each field may be enclosed in double quotes (""). (Any field that contains a comma must be enclosed in double quotes.)

ASC_IMP.DLL supports all seven imports and uses all of the fields in the order they appear in the _INFO structures (see IMPORT.H for the fields and possible values). The People, Authorization, and Registration import files are simple flat files, but the Groups, Courses, Scores, and Curriculum imports are more complicated because the database tables for these imports have a master-detail relationship.

Each record in the file for importing groups should contain the fields for the group (GROUP_INFO) followed by the fields for the member (MEMBER_INFO). The first record containing the group should contain the Title and Description followed by the information describing the first member of that group. Subsequent records defining members for that group need only include the Title (the Description field may be blank) followed by the member information, since the Group information in these subsequent records is used only to connect the member to the group. (It is not used to create the group.) Records for group managers then follow the records for group members, with similar structure. Other master-detail imports work much like the Groups import.

See the PLM Administrator help file or the PLM Administrator Reference Manual for more details.

6.4.3 Courseware Imports

Capabilities: Courses and (sometimes) Curricula

A variety of courseware imports are available for certain types of courseware. Courseware must first be installed before the course information can be imported into PLM. See the PLM Administrator help file or the PLM Administrator Reference Manual for more details.

6.5 Import.h

The C header file IMPORT.H defines the structures, constants, and function prototypes that are required for creating a PLM import DLL.