Services to Support the Upgrade for the Unified Export Strategy (UES) System

UES Upgrade Requirements Document

Oct. 3, 2007

CONTENTS

1 INTRODUCTION

The Foreign Agricultural Service (FAS) of the U.S. Department of Agriculture (USDA) works to improve foreign market access for U.S. products. The FAS operates programs designed to build new markets and improve the competitive position of U.S. agriculture in the global marketplace. FAS bears the primary responsibility for USDA's overseas activities—market development, international trade agreements and negotiations, and the collection and analysis of statistics and market information. FAS also administers USDA's export credit guarantee and food aid programs and helps increase income and food availability in developing nations by mobilizing expertise for agriculturally led economic growth.

The data and information in the Unified Export Strategy (UES) system is vital to all FAS program areas. During the 10 years since its original deployment, the UES system has expanded in scope, functionality, and user base. It is the primary source of industry input used by FAS staff, by Senior Executives in all of the Agency's program areas, by the Administrator, and ultimately by the Under Secretary and Secretary in making key decisions for prioritizing and allocating funds and staff resources for the programs and initiatives.

In recent years, as advances in technology have overtaken the current system, the number and severity of UES shortcomings have grown. As a result, the USDA/FAS is in urgent need of upgrading the UES system to improve operational efficiency, planning and coordination, analysis of effectiveness, and performance measurement. FAS is, therefore, embarking on a program to upgrade the UES system, with the twin goals of maximizing improvements in system performance and utility to stakeholders while minimizing disruption to current users.

1.1 Background

As the first step in the upgrade program, the SAIC team carried out an extensive assessment of UES stakeholders' needs from December, 2004 through March, 2005. The results were reported in the *UES Stakeholder Needs Assessment Report*. The report summarized the needs of the principal UES stakeholder groups that are users of the current system, OTP staff, Cooperators and Program Participants, PPS, and Overseas Posts. Other stakeholder groups, such as FAS Senior Management, other FAS staff, and IT staff contributed to the definition of the needs of potential additional users of the upgraded system and to the identification of technical and other constraints on development of the upgraded system.

The business requirements in Appendix C of the *UES Stakeholder Needs Assessment Report* were the starting point for definition of the detailed technical requirements presented in this Requirements Document. A UES Requirements Work Group (RWG), with representatives from each of the stakeholder groups, was formed to provide guidance to the SAIC team during the requirements definition process. Additionally, a focused work group was formed for each of the major stakeholder groups that will be "user communities," i.e., Cooperators/program participants, Marketing staff, PPS, and FAS Managers, to provide feedback and clarification on specific topics.

Analysis of the current UES applications and databases, along with weekly discussions with the work groups, drove development of the conceptual data model included in this document that depicts the major groups of information that must be included in the upgraded database. Data requirements, defining these groups of information, identifying permitted values for individual items, and specifying the relationships among the groups of information, were developed from the diagram.

Additional weekly meetings with the work groups and further analysis of the use and operation of the existing UES applications led to development of the functional, user interface, and system interface requirements. These specify where the information in the upgraded database will come from (e.g., another system, entry by a user, or calculation by the system); who will enter or edit it and under what circumstances; and the rules for how the system should validate or calculate it. These requirements also specify reporting requirements for the upgraded system, i.e., the circumstances and methods for extracting information from the upgraded database, for searching, displaying, downloading, and formatting reports.

With the initiation of a second contracting effort to complete the upgrade, the requirements documents were revised to include minor clarifications and changes resulting from the FAS reorganization and stakeholder groups were tasked with reviewing and suggesting updates to the requirements. The recommended changes were incorporated in a Sept. 2007 CCB meeting.

1.2 Purpose

The principal purpose of this Requirements Document is to provide a foundation for the future work products resulting from the construction phase of the upgraded UES system, for example, design specifications, test procedures, or deployment plans. This document specifies the technical requirements the upgraded system must meet, describes the upgraded system's operation, and identifies the constraints that will affect its design.

The Final version of this Requirements Document will reflect the concurrence of FAS with the detailed technical requirements and constraints, thereby defining the Requirements Baseline for the start of the construction phase. Following establishment of that initial baseline, changes and additions will be subject to a formal change control process involving FAS and the upgraded UES system construction team.

1.3 Scope

The scope of this Requirements Document encompasses an upgraded UES system, including a database upgrades and the software applications necessary to provide user and system interfaces to meet the needs described in the *Final UES Stakeholder Needs Assessment Report*. The upgraded system specified by this Requirements Document will fulfill FAS needs for:

- automated support of strategic export market planning
- submission and review of applications for funding from the programs named in the requirements
- financial tracking of program funding from the application stage through reimbursement
- generation and maintenance of annual agreement "documentation of record"

- automated information sharing among user groups
- seamless integration of upgraded system with legacy data

1.4 References

- UES Upgrade Planning Phase Project Plan, Final (January 28, 2005)
- UES Stakeholder Needs Assessment Report, Final (April 25, 2005)
- USDA Web Style Guide, Version 1.1 (December 10, 2004)
- NIST Special Publication 800-64 Rev. 1, Security Considerations in the Information System Development Life Cycle (June, 2004)
- NIST Special Publication 800-37, Guide for the Security Certification and Accreditation of Federal Information Systems (May, 2004)
- Foreign Agricultural Service: An Analysis of Current Agency Functions and Information Requirements in an Information Architecture Framework (June 25, 2005)
- Logic Model for PART in FAS, memo and Excel workbook (March 27, 2005)
- UES Draft Requirements Document (SAIC, August 30, 2006)
- Appendix B Conceptual Data Entity Model (SAIC, August 2006)
- Appendix C Requirements (SAIC, August 2006)

1.5 Document Overview

This Requirements documentation consists of four major documents. The contents of each section and appendix are summarized below.

1. The Draft Requirement Document – this document. The high level overview of the UES upgraded system is included.

2. The Detailed Requirement Document – the document includes the numbered requirements related to the new system.

3. The UES Sample User Interface Screens – the document includes sample screenshots for the required functionality.

2 SECTION 1, INTRODUCTION TO UPGRADED SYSTEM

This section provides descriptions of the upgraded system's operation, leading to specification of the detailed technical requirements contained in the accompanying Detailed Requirements Document. Figure 1 depicts the analytical process and identifies the figures and tables in this document containing the interim results.

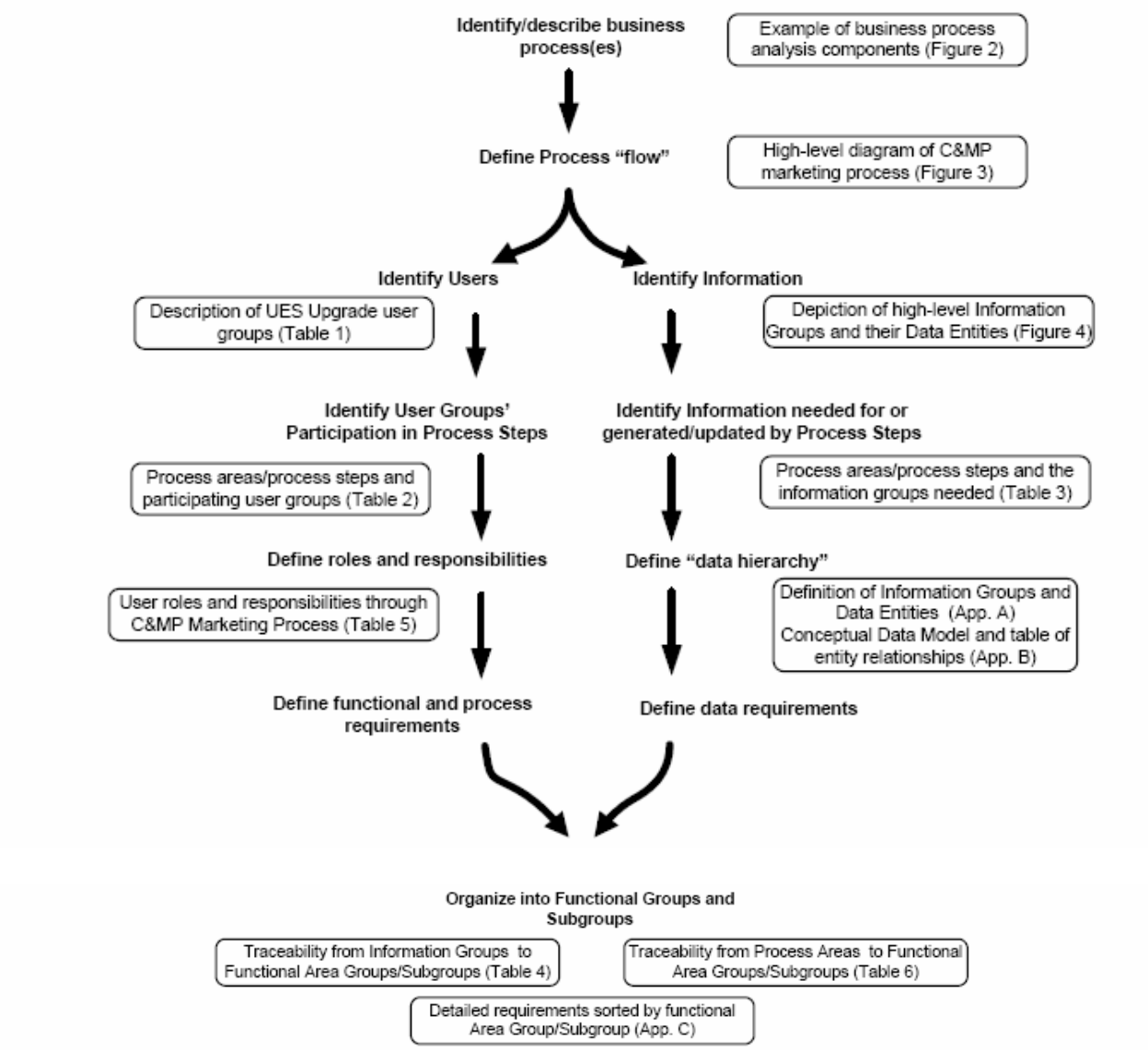


Figure 1. Requirements Definition Process

2.1 Objectives

The upgraded UES system is expected to address the critical issues identified during the stakeholder needs assessment analysis by providing an integrated user interface, allowing easier input of strategic and tactical planning and financial information into a re-engineered relational database. ~~System-to-system integration with the USDA's Grants Interface Module (GIM) is expected to bring FAS into line with USDA's and the federal government-wide e-grants initiative~~. The incorporation of a robust, modern, flexible reporting capability is expected to provide users across FAS, whether in OTP or not, with the ability to easily find and extract export marketing information to meet their information needs.

In summary, the upgraded system reflected in the high-level and detailed requirements contained in this document is expected to provide the following benefits to its user community:

- Streamline the application submission process.

- Enable a flexible, powerful reporting capability to extract useful information from UES and allow users to combine it with information from other FAS information sources.
- Facilitate improved information sharing among cooperators/program participants, marketing specialists, and staff at overseas posts.
- Streamline financial functions such as budget allocation, claims processing, and financial status reporting without sacrificing accounting accuracy and reliability.
- Provide collaborative tools to enable strategic and tactical planning by cooperators/programs participants, marketing specialists, and staff at overseas posts.

2.2 System Operation

Analysis of the existing UES applications, additional interviews with UES stakeholders and weekly meetings with the UES Upgrade Requirements Work Groups (Steering Work Group, Marketing Work Group, Cooperator Work Group, PPS Work Group, Manager Work Group and Post [FSO/FSN] Work Group) produced descriptions of the business process support by UES containing the basic information displayed in Figure 2. Specifics of the process components; the steps involved; the information needed, generated, or updated by each process step; and the participants in each step are described in the subsection that follows.

Figure 2. Business Process Components

2.2.1 Process Flow

The upgraded UES system shall support the overall process illustrated in Figure 3; i.e., the execution of the Market Access Program (MAP) and the Foreign Market Development (FMD) program, as well as several other, smaller programs aimed at improving foreign market access for U.S. agricultural products such as the Emerging Markets Program (EMP), Technical Assistance for Specialty Crops (TASC), Quality Samples Program (QSP), and the Cochran Fellowship Program.

This process, a main focus of the Office of Trade Programs (OTP) organization, comprises on-going strategic and tactical planning and two core sub-processes, the Application sub-process and the Agreement sub-process. The on-going strategic and tactical planning are collaborative efforts undertaken by cooperators and program participants, OTP marketing specialists, and the FSOs and FSNs at overseas posts.

Figure 3. OTP Marketing Program Process

The Application process area consists of a series of sequential steps, performed once (and often more than once) per year for the MAP and FMD programs and on an as-needed basis for the other programs:

- Prepare—includes preparation of the Federal Register announcements by OTP, as well as preparation of program funding applications, proposals or the "UES Plan," by cooperators and program participants.
- Submit—includes submission of the funding request and necessary supporting information on the part of the cooperators and program participants in addition to receipt and sufficiency checks on the part of PPS.
- Review/Evaluate—includes review of the applications by OTP Marketing Specialists, EMP, QSP, or TASC program staff, and overseas Post personnel.
- Allocate/approve funding—includes development of the annual funding recommendations by the commodity divisions, analysis and application of the annual funding allocation formula by PPS, and final decisions on the approved budgets for each cooperator and program participant.

The Award process area is the transition from the Application process area to the Agreement Implementation process area and includes preparation of the Agreement or Amendment,

Approval Letter, attachments for each new or amended agreement by PPS and the OTP divisions, and signature by the OTP Deputy Administrator. Transmittal of the Approval Letter to the recipient and receipt of the recipient-signed Agreement or Amendment signals authorization for the cooperator or program participant to proceed with the approved plans documented by the Approval Letter attachment.

The Agreement Implementation process area consists of a number of parallel process steps continuing through the year:

- Administration—includes processing and recording: claims for both advances and reimbursements, maintaining overall financial status, and performing compliance reviews, involving cooperators and program participants, PPS, Grants Management Branch and marketing specialists.
- Cooperation—includes sharing planning information, schedules, calendars, and other marketing support information among cooperators and program participants, marketing specialists, and overseas post personnel.
- Global Marketing—includes executing the agreements by cooperators and program participants and performing planned activities.
- Reporting Results and Progress—includes reporting by cooperators and program participants of actual results against performance measures and goals, longer-term progress in the Country Progress Report format, and other reports, such as trip reports, research reports, periodic progress reports, EMP or TASC Final Reports, or Contribution Reports.

The Agreement Implementation process area leads to updates of both strategic and tactical plans and notifications of those changes, "closing the loop," and informing preparation of the next application for funding and its accompanying planning details. All users, i.e., cooperators and program participants, marketing specialists, PPS, FSOs and FSNs, and other FAS management and staff, shall have the ability to find, extract, and report information entered or generated by these processes.

2.2.2 User Groups

The principal user communities of the upgraded UES applications shall remain the same communities as the current UES applications. Streamlined functions, powerful reporting capabilities, and wide-ranging information sharing features should expand the size of each of the user groups described in Table 1.

User Group	Description
Cooperators and Program Participants	Staff of industry groups, state-regional organizations, and other entities who participate in the USDA programs administered using the UES.
PPS and Grants Management	Management and staff of OTP's Program Policy Staff (PPS), and OAO's Contracts and Agreements Division (Grants Management

	Branch) responsible for program agreement administration and financial tracking and reporting. Grants Management responsible for expense claims and advances and preparing agreements.
OTP Marketing Staff	Marketing Specialists and other staff and management of OTP's commodity divisions
Overseas Post Staff	Foreign Service Officers (FSO) and Foreign Service Nationals (FSN) at Agricultural Trade Offices (ATO) and other overseas Posts.
Other FAS (Managers and Staff)	Management and staff in all FAS organizations: Administrator's Office (including Compliance Review Team), Commodity & Marketing Programs (OTP) (other than PPS and OTP Marketing Staff), Foreign Agricultural Affairs (FAA), International Cooperation & Development (ICD), and International Trade Policy (ITP).

Table 1. User Groups

Table 2 identifies the user groups participating in each of the process areas discussed in Table 1.

Process Area/ Process Step	**Participants**	**PPS**	**Grants Mgmt Branch**	**OTP Marketing Staff**	**Overseas Posts (FSOs & FSNs**	**Other FAS Managers and Staff**
Strategic Planning	X			X	X	
Tactical Planning	X			X	X	
Application						
Prepare	X	X				
Submit	X	X				

Review				X	X	
Allocate		X		X		
Award			X	X		
Agreement Implementation			X			
Claims	X		X			
Financial Tracking		X				
Compliance Review						X
Cooperation	X			X	X	
Global Marketing	X				X	
Results & Progress	X				X	
Notifications	X			X	X	
Finding, Extracting & Reporting	X		X	X	X	X

Table 2. Process Areas and Participating User Groups

2.2.3 Information Groups

The information needed by each of the process areas discussed in Table 2 is represented by large Information Groups, depicted in Figure 4. Each Information Group is a group of Data Entities related primarily by the process areas or steps where the information is used, generated, or updated. Each of the Information Groups and Data Entities in the figure below is defined in Appendix A, Glossary, Terms, and Acronyms.

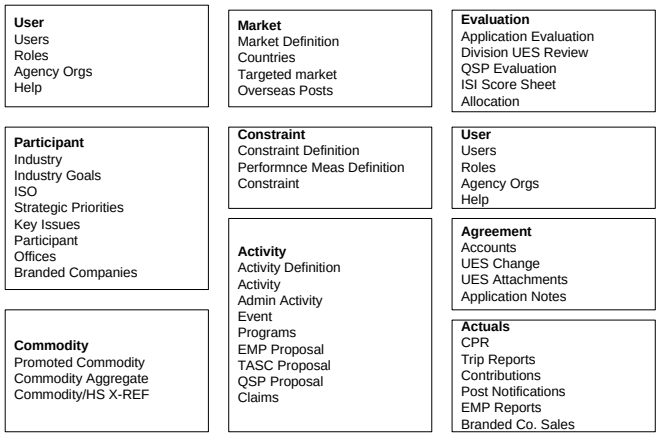


Figure 4. Information Groups and Their Data Entities

Table 3 identifies the Information Group(s) related to each Process Area, linking process and data at the highest level. An "X" in an intersection indicates that data falling into that Information Group is used by, created by, or updated by performance of one or more of the process steps in that Process Area. Which specific data (part of which specific Data Entity and Data Element within the Information Group) are specified in the detailed data-type requirements.

Process Area/ *Process Step*	Information Group								
	Participant	Commodity	Market	Condition	Practice	Evaluation	Agreement	Actuals	Users
Strategic Planning	X	X	X	X				X	
Tactical Planning				X	X				
Application									
Prepare		X	X	X	X			X	
Submit	X								X
Review		X	X	X	X	X		X	
Allocate						X			
Award							X		X
Agreement Implementation									
Claims					X		X	X	
Financial Tracking					X		X	X	
Compliance Review							X	X	
Cooperation	X	X	X	X	X			X	
Global Marketing	X	X	X	X	X			X	
Results & Progress								X	
Notifications		X	X	X	X		X		
Finding, Extracting & Reporting	X	X	X	X	X	X	X	X	X

Table 3. Process Areas and Related Information Groups
("Practice" info group is Activity, "Condition" info group is Constraint)

The hierarchical breakdown of data from Information Groups to Data Entities and thence to Data Elements and individual data type requirements is presented in Table 4. The Functional Area Group/Subgroups shown in the table are defined in Section 3, System Requirements. Table 4 provides a "cross-reference" from the information needs related to process steps discussed in this section and the detailed requirements listed in the Detailed Requirements document.

2.3 User Roles and Responsibilities

Analysis of the process areas to be supported by the upgraded UES system and identification of the user groups that must participate in those processes (depicted in Table 2) led to definition of the requirements specifying the necessary user roles and capabilities. In the upgraded UES system, each user's ability to "enter and edit" or "find and view" specific types of information shall be controlled through the user access features of the system. Beyond the basic requirements for assignment of user names and passwords, the upgraded system shall require that users be assigned specific roles and given authorization to perform specific functions.

2.3.1 Roles

Users accessing the upgraded UES system shall be assigned one or more roles. Each role shall have a range of functions available to users assigned that role. A primary responsibility of users in the System Administration role shall be to indicate which role or roles are assigned to a user and which function or functions that user is authorized to perform. The roles were defined by analysis of the UES stakeholder groups and discussions by the UES Requirements Work Groups of preferences for access to data and functions by these groups.

The roles identified for UES Upgrade system users shall be:

- **Program Participant**—Staff of industry groups and state-regional organizations who participate in the USDA programs administered using the UES.
- **OTP—**Marketing specialists, analysts, team leaders, and managers in the Office of Trade Program commodity divisions (Processed and High Value Products Branch; Livestock and Seafood Products Branch; Field Crops and Forest Products Branch; Horticultural Crops Branch)
- **PPS**—Management and staff of OTP's Program Policy Staff (PPS), the organization that administers program agreements and performs financial tracking for the programs; responsible for OTP financial reporting. May also include members of the OAO **Grants Management Branch** with responsibility for approving claims.
- **Posts**—Foreign Service Officers (FSO) and Foreign Service Nationals (FSN) located at Agricultural Trade Offices (ATO) and other overseas posts worldwide.
- **FAS**—Other managers and staff across the FAS organization, such as Foreign Agricultural Affairs, Export Credits, International Trade Policy, or International Cooperation and Development.
- **System Administration**—Personnel involved in keeping the operational data and capabilities of the system up to date, for example, managing user accounts or updating reference tables.

2.3.2 Responsibilities and Functions

Tables 5 and 6 present the results of the process analysis that led to the definition of user roles and responsibilities and detailed functional and process requirements. By breaking down the User Group participation in Process Areas presented in Table 2, the responsibilities of each role in each process step are defined in Table 5.

Process Area	Process Step	Responsibility	Role
Strategic Planning			
	Initial strategic planning		
		Identify commodities & targeted markets	Program Participant
		Identify goals	Program Participant
		Assess markets	Program Participant
			Posts
		Identify constraints and opportunities and performance measures	Program Participant
		Evaluate and comment on commodities, markets, constraints and performance measures	OTP
			Posts
	Update strategic planning		
		Revise commodities & targeted markets	Program Participant
		Update goals	Program Participant
		Reassess markets	Program Participant
			Posts
		Revise constraints and opportunities and performance measures	Program Participant
		Re-evaluate and comment on commodities, markets, constraints and performance measures	OTP
			Posts
Tactical Planning			
	Initial tactical planning for a specific year		
		Plan activities/budgets/schedules	Program Participant
		Define performance measures	Program Participant
		Evaluate and comment on activities, budgets, and schedules	Posts
			OTP
		Plan EMP project/budget/schedule	Program Participant
		Plan TASC project/budget/schedule	Program Participant
		Plan QSP	Program Participant

		project/budget/schedule	
	Update tactical plans during execution		
		Modify activities	Program Participant
		Revise budgets	Program Participant
		Update performance measures	Program Participant
		Re-evaluate and comment on activities, budgets, and schedules	OTP
			Posts
		Revise plans for completion of EMP project	Program Participant
		Generate initial budget allocation for each submitted application, using allocation formula	PPS
		Revise budget allocation for each submitted application, using [refined] Allocation formula	PPS
		Generate final budget allocation for all submitted applications, using Allocation Formula and summarize in the Decision memo	PPS
		Revise planned budgets to match allocations	Program Participant
		Approve revised planned budgets	OTP
Award			
	Prepare award packages		
		Prepare Agreements and Amendments	Grants Management Branch
		Prepare Approval Letters and attachments	Grants Management Branch
			OTP
		Review and forward award packages	OTP
			PPS
	Track status of award packages		
		Track signature and distribution of award packages and receipt of bilaterally completed Agreements and Amendments	PPS
Agreement Implementation			
	Notifications		
		Describe background/purpose for modifications to activities/schedules/budgets	Program Participant
		Review and comment on background/purpose for modifications to activities/schedules/budgets	OTP
	Claims		
		Submit claims for advance funding	Program Participant

		Approve claims for advance funding	Grants Management Branch
		Submit Claims for reimbursements	Program Participant
		Approve Claims for reimbursements	Grants Management Branch
		Transmit approved claims to CCC	Grants Management Branch
	Financial Tracking		
		Identify appropriated program funding for each year	PPS
		Report on contributions	Program Participant
		Report on participant allocations, obligations, advances, reimbursements, and contributions	PPS
		Track status of advances and reimbursements by CCC	PPS
	Cooperation		
		Share research results and other regulatory, policy, or export marketing-relevant information	Program Participant
			OTP
			Posts
			FAS
		Share market assessment information	Program Participant
			OTP
			Posts
			FAS
		Share calendar information	Program Participant
			OTP
			Posts
			FAS
		Discuss issues relating to a commodity/market	Program Participant
			OTP
			Posts
			FAS
	Global Marketing		
		Plan overseas trips	Program Participant
		Notify posts of planned trips	Program Participant
		Acknowledge trip notifications and advise	Posts
		Report on overseas trips	Program Participant
		Execute activities, specific events	Program Participant
		Undertake marketing activities	Program Participant
			Posts
	Results & Progress		
		Report progress against strategic goals	Program Participant
		Report progress against performance measures	Program Participant
		Report locations/dates of events	Program Participant

		Comment on progress	OTP
			Posts
		Report periodic project status for EMP and TASC projects	Program Participant
		Report completion of EMP, TASC, and QSP projects	Program Participant
Notifications			
		Describe background/purpose for modifications to commodities, markets, constraints, performance measures, activities, schedules, budgets	Program Participant
		Review and comment on background/purpose for modifications to commodities, markets, constraints, performance measures, activities, schedules, budgets	OTP
Finding, Extracting & Reporting			
	Reporting		
		Reporting on strategic, tactical, planning, results, financial, and administrative information from UES and other data sources	Program Participant
			OTP
			PPS, Grants Management Branch
			Posts
			FAS

Table 5. Responsibilities of Each Role throughout UES Process

These responsibilities were further decomposed into functional capabilities and detailed functional and process type requirements. Further information may be found in the accompanying USE Cases document.

2.3.3 Data Access

User Access to information in the upgraded UES system shall be controlled by a combination of factors:

- The user's assigned role(s).
- The user's authorized functions.
- The preferences of the "owner" of that information.

Users assigned Program Participant, OTP, PPS, and Post roles each shall be authorized to create or edit certain types of information and read other types. Users assigned the FAS role shall be authorized to have only Read access to all information.

Performance of each function requires access to specific types of information. For example, a user assigned the Program Participant role and authorized to submit reimbursement claims needs Read and Write access to reimbursement claim information.

The system shall consider the individual user who has entered an item of data (i.e., created the record) as the "owner" of that data. For example, a user assigned the Program Participant role, authorized to "file trip reports," would be considered as the "owner" of a trip report they have entered. That user shall be able to grant read access to their trip report to other users who would not have access based on their assigned roles and authorized functions. Table 7 summarizes the high-level rules for access to information by individual users assigned to the various roles.

As an example, a OTP user authorized to perform marketing specialist functions, such as "manage program participant's agreement," would have Read access to the strategic and tactical planning, financial, and progress information of all program participants associated with that user's "agency organization" (in this case, commodity division) and Write access to such information as comments on participant's plans or results or division recommendations for program funding.

It is envisioned that the System Administration role could be assigned to a specific user along with the Program Participant role; e.g., allowing that participant organization's staff member to maintain the organization's administrative information and manage their user accounts.

User Role	Default Data Access
Program Participant	• May read/write their own UES (strategic plan and associated items) • May read others' UESs (if access is universally granted to other participants by the owning participant • May read/write their own claims • May not access others' claims • May read non-participant comments on their own UES • May read non participant comments on others' UESs (dependent upon whether the relevant plan is accessible) • May not access Evaluation information • May not access Allocation information • May grant or limit read access to their UES by other participants
OTP	• May read participants' UESs • May read participants' claims • May read/write non-participant comments on UESs (write/edit their own comments; read all others) • May read/write Evaluation information • May read Allocation information
PPS	• May read participants' UESs • May read/write participants' claims • May read/write non-participant comments on UESs (write/edit their own comments;

	read all others) • May read/write Evaluation information • May read/write Allocation information
CAD Grants Management Branch	• May read/write participants'claims
Posts	• May read participants' UESs • May read participants' claims • May read/write non-participant comments on UESs (write/edit their own comments; read all others) • May read Evaluation information • May read Allocation information • May read/write Overseas Post information (write access for Posts with which the user is associated)
FAS	• May read Participants' UESs • May read participants' claims • May read non-participant comments on UESs • May read Evaluation information • May read Allocation information
System Administrator	• May read/write/delete all UES data (plans, associated items, evaluations, claims, etc…) • May read/write/delete all system data, such as reference tables • May read/write/delete user account information
All	• May read Overseas Post information • May read all GBI and Research Activities and any associated Attachments (independent of Participants' granting or limiting of access to other participants)

Table 7. Data Access Rules by User Role

2.3.4 Table 7B: Statements of Intent (further clarification on roles)

#	Subject	verb phrase	Object	Conditional phrase(s)	Requirement #
01	Participant users	may read/write	their own UES (strategic plan and associated items)		06.02.62.204
02	Participant users	may read	others' UESs	if access is universally granted to other participants by the owning participant	06.02.62.206

#	Subject	verb phrase	Object	Conditional phrase(s)	Requirement #
03	Participant users	may read/write	their own claims		06.02.62.114
04	Participant users	may not access	others' claims		06.02.62.115
05	Participant users	may read	Non-Participant comments on their own UES		06.02.62.116
06	Participant users	may read	Non-Participant comments on others' UESs	dependent upon whether the relevant plan is accessible	06.02.62.116
07	Participant users	may not access	Evaluation information		06.02.62.216
08	Participant users	may not access	Allocation information		06.02.62.216
09	Participant users	may grant or limit	read access to their UES by other participants		06.02.62.112

10	OTP users	may read	Participants' UESs		06.02.62.111
11	OTP users	may read	Participants' claims		06.02.62.111
12	OTP users	may read/write	Non-Participant comments on UESs	(write/edit their own comments; read all others)	06.02.62.116
13	OTP users	may read/write	Evaluation information		06.02.62.304
14	OTP users	may read	Allocation information		06.02.62.306
20	PPS users	may read	Participants' UESs		06.02.62.111
21	PPS users	may read/write	Participants' claims		06.02.62.504
22	PPS users	may read/write	Non-Participant comments on UESs	(write/edit their own comments; read all others)	06.02.62.116
21	PPS users, CAD Grants Mgmt Branch	may read/write	Participants' claims		06.02.62.504

#	Subject	verb phrase	Object	Conditional phrase(s)	Requirement #
23	PPS users	may read/write	Evaluation information		06.02.62.508
24	PPS users	may read/write	Allocation information		06.02.62.508

30	POST users	may read	Participants' UESs		06.02.62.111
31	POST users	may read	Participants' claims		06.02.62.111
32	POST users	may read/write	Non-Participant comments on UESs	(write/edit their own comments; read all others)	06.02.62.116
33	POST users	may read	Evaluation information		06.02.62.408
34	POST users	may read	Allocation information		06.02.62.408
35	POST users	may read/write	Overseas Post information	(write access for Posts with which the user is associated)	06.02.62.406

40	FAS users	may read	Participants' UESs		06.02.62.111
41	FAS users	may read	Participants' claims		06.02.62.111
42	FAS users	may read	Non-Participant comments on UESs		06.02.62.116
43	FAS users	may read	Evaluation information		06.02.62.604
44	FAS users	may read	Allocation information		06.02.62.604

50	System Admin users	may read/write/delete	all UES data (plans, associated items, evaluations, claims, etc…)		06.02.62.700
51	System Admin users	may read/write/delete	all system data, such as reference tables.		06.02.62.710
52	System Admin users	may read/write/delete	User account information.		06.02.62.708

#	Subject	verb phrase	Object	Conditional phrase(s)	Requirement #
60	All	may read	Overseas Posts information		06.02.62.800
61	All	may read	all GBI and Research Activities and any associated Attachments.	independent of Participants' granting or limiting of access to other Participants	06.02.62.113

Table 8: Permissions Intent

2.4 User Interfaces

The upgraded UES system shall provide a graphical user interface that complies with the USDA web application style guide and the requirements of Section 508 (related to accessibility standards). The user interface shall provide a common interface, "look, and feel" for users in all roles when logging in and using the system. Section 6.1 of the *Final UES Stakeholder Needs Assessment Report* describes the concept of operations of the upgraded system and its user interface application.

The user interface application shall provide to each user, based upon the user's role and authorized functional capabilities:

- Status information concerning documents or information the user is authorized to read, enter, or update.
- Notification information concerning documents or information that has recently changed status or that requires action by the user.
- Means to access the user's authorized functional capabilities.
- Access to appropriate shared information features (calendars, discussion forums).

The upgraded UES system shall provide "active" notifications or alerts to users of upcoming events, required actions, and upcoming or overdue deadlines. These notifications and alerts may take the form of e-mail messages, "pop-up" windows or other active elements to be defined during design of the system. Some examples include notifying a post when a participant has submitted a Post Notification of an upcoming trip, notifying a participant when a Post Notification has been acknowledged, notifying a marketing specialist when a participant's funding applications have been received, notifying participants when comments have been entered by posts or marketing specialists concerning their strategic plans, or notifying a marketing specialist when the Approval Letter for a participant has been generated and is ready for review and forwarding for signature.

The accompanying UES Sample User Interface Screens document provides guidance on the user interface design.

2.5 Types of Uses

The primary uses of the upgraded UES system shall be:

- Preparing and updating export marketing strategic planning, i.e., industry, commodity, market, and condition information.
- Preparing and updating export marketing tactical planning, i.e., practices, events, schedule, and budget information.
- Preparing and submitting applications for funding from FAS-managed programs (principally, MAP, FMD, EMP, TASC, and QSP).
- Reviewing and evaluating applications and preparing recommendations for program funding.
- Allocating annual program funding to selected applicants and issuing agreements and amendments obligating that funding.
- Managing and tracking program funding, including requests for advances and reimbursements and entering program announcements, allocations and allocation adjustments in the system.
- Reporting progress against export marketing goals and specific performance measures.
- Sharing strategy, planning, scheduling, and results information.
- Examining and analyzing export market goals, plans, and results in the context of FAS programs and initiatives.

The table below outlines by user group the main functions the software is expected to perform. The table also differentiates functions of the upgraded system that are new to the legacy system from those that currently exist in the legacy system by showing an asterisk by items with a data counterpart in the legacy system.

Elements of the Plan (Participant Owned Data) (* items have a counterpart in the legacy system)		
Element	Description	User Action
Participant Info*	The participant information and industry information, including the Industry Goals	Fill out the Participant Profile information. Information will include profile, offices, contacts, bank information, affiliated organizations, personnel, industry info, and industry goals. "Goal Metrics" tab can be completed by cutting and pasting from a spreadsheet.
Commodity*	The promoted commodities described as the commodity aggregates for which funds are being requested in this application	Use the screen to add, delete, or change the products that will be promoted per the plan. For each promoted Commodity fill out basic information, domestic information, international information, data source and the metrics tab. "Metrics" tab can be completed by cutting and pasting from a spreadsheet.
Market Definitions*	The areas where activities will take place per the plan. Market definitions are made from groups of countries that form a region or from a single country.	List all the Market Definitions for which funding for activities are being requested per the plan. A participant can define their own Market Definitions as necessary or select from FAS Standard. Single countries will need to have a Market Definition if they are used in the Plan.
Targeted Market*	A Market Definition/Commodity Association	Make an association between every Market Definition and Commodity association that will be addressed in the Plan. Enter plan data associated with the Targeted Market, including basic information, the market assessment, the long term strategy the past performance text and the export goals metrics. "Export Metrics" tab can be completed by cutting and pasting from a spreadsheet.
Constraint Definitions	A reusable title and description of a Constraint to trade	Enter the high level constraint information for each kind of constraint the plan will address

Constraints*	A Constraint Definition combined with a Targeted Market (components: Market Definition/Commodity/Constraint Definition)	Enter the specific constraints that will be addressed in the plan; this consists of associating Constraint Definitions with specific Targeted Markets. Follow up assessment "Evaluation and Findings" data needs to be entered after implementation of the plan.
Performance Measures Definitions	A reusable title and description of a performance measure that will be associated with a high level Constraint Definition	Assign high level performance measures definitions to the constraint definitions where they will apply
Performance Measures Specific*	The performance measures for a specific Constraint	Assign a Performance Measure Definition to a constraint, enter baseline and goal values. After completion of activites, return and enter actual values
Activity Definition	A reusable title, type and description that can be used in defining one or more activities	Fill out high level descriptions for each type of activity that will be carried out in the plan
Activity*	An activity is the reimbursable unit of the plan made by an association between an Activity Definition and a Constraint	Indicate the funding stream, the request amount and a unique activity code identifier that will be used to claim reimbursements. Fill in other activity information as necessary. EMP, QSP and TASC activities need to be associated with a Proposal.
Events	Additional activity information that includes location and date. Branded activities will include branded sub budgets with associated Branded Company in the Event information.	Event information can be associated with any activity. After a "Branded" type activity has been approved the branded child activity information (with child activity code identifier, child activity sub-budget and associated Company will be entered as events.
Branded Sub-Budgets (see Events)*		
Administrative Costs	Admin activities	Indicate the funding stream, the request amount and a unique activity code identifier that will be used to claim reimbursements. Fill in other admin activity information, such as incumbent data, as necessary.
Accounts*	Program related information	Enter the proposed contributions. If applying for MAP funds enter the plan start date and the plan end date.
Contingent Liability*	FMD application information on unfunded liabilities	Use this screen to enter all contingent obligations which would be due if your overseas offices were to close on the last day of the marketing year (FMD applicants only)
Worldwide Personnel*	FMD application information on personnel	Use this screen to enter the number of U.S. citizens employed by your organization overseas, the number of those U.S. citizens tallied previously whose salaries are paid in whole or in part with FMD funds, and the total dollar amount of overseas U.S. citizen salaries and allowances paid with FMD funds (FMD applicants only)
EMP Proposal*	A request for EMP funds	First enter the Commodity, Targeted Market and Constraint data. Then enter the proposal information. Enter the related activity information.
QSP Proposal	A request for QSP funds	First enter the Commodity, Targeted Market and Constraint data. Then enter the proposal information. Enter the related activity information.
TASC Proposal*	A request for TASC funds	First enter the Commodity, Targeted Market and Constraint data. Then enter the proposal information. Enter the related activity information.
Submit the Plan*	Submission implies that the strategic plan is complete	Change the plan status to submitted
Promised Contributions*	Percentage or dollar amount that the participant will contribute to the marketing effort	Enter the promised contribution amount for each program – follow up with the actual contributions at the end of the cycle
See an historical Plan*	Data from prior years	Change the year context and see data from a previous year. Compare the 'living document' version of the plan with a snapshot saved at submission time.
Initiate a Plan for a new year*	Create a new plan using historical data	Create an annual submission of the UES strategic and tactical

		Plan
Supplements to the Plan		
Help*	On-line information on the UES system	From any screen click the !HELP link.
Announcements	Announcements of interest to program participants or FAS	Read system announcements.
Alerts	System generated messages	Cause alert to be seen by another user in the system (example submitting a Post notification will alert Post)
Branded Company Information*	List of companies with their information, including DUNN number	Enter branded company data
Change	A record of a change to the plan, either a Notification or a Footnote	Use this feature to document changes to the plan
Attachments	Any document that will supplement the plan	Upload the file
Strategic Executive Summary	Strategic Executive Summary	Enter Strategic Executive Summary information, including key issues and strategic priorities
Trip Reports	Each Activity can be associated with one or more trip reports.	Make the association between an activity and a trip report.
Forums	A system enabled collaborative forum for sharing information	Enter a new discussion thread or comment on an existing thread on the Participant or System Forum
Calendar	A system enabled calendar that users can contribute to	Enter date and location data
Post Notification	A notification to a Post regarding travel information associated with an activity in the Post's area of responsibility	Enter the post notification and Submit
CPR	Country Progress report information	Enter CPR information for each Constraint
Application Notes*	Notes for participant use only regarding the Plan	Enter a note
Reimbursements		
Expense Claims*	A request for reimbursement	Create an Expense Claim with associated line items that include the activity, cost category, country and amount information.
Advance Requests*	A request for an advance payment	
Actuals		
EMP Report	A report on the success of an EMP activity	Enter the quarterly progress or the final report data at the activity level. Approval of the EMP Final Report will release the final 15% of approved funding.
Performance Measures actuals*	A report back on actual numbers per specific performance measure/targeted market to compare with baseline and goals set at the time of application	Return to specific performance measures from previous application and enter 'actual' numbers. Set 'Met' indicator on met performance measures, where appropriate
Contributions*	The actual contributions reported as a follow-up to the plan	Enter participant and Industry contribution information for each program for which it is required (MAP, FMD)
Reports		
Ad-Hoc reports	Ad-hoc reporting on UES data	Build the report as needed. Search the plan by country. Search the plan by commodity. Save report in excel. See print preview of report. Easy generation of contribution reports, funding reports, performance measure reports, financial summary reports, claim history reports, Branded company reports, activity summary reports, evaluation results reports, CPR reports
Pre-defined reports*	Sufficiency Check Report Branded Company report Program Ceiling Report Allocation Balance Carryover Report UES Application Report	Select the desired report, enter any necessary parameters

	Activity Plan Approval Report Contribution Detail Report CPR	
Elements of the UES Plan (PPS/Grants Mgmt Owned Data) ("elements have a counterpart in the legacy system)		
Element	Description	User Action
Accounts*	Agreement Numbers, Allocation and account data	Maintain account data associated with each program participant; includes entering appropriation information, allocations and adjustments to both.
Pay Claims*	An approval of an expense claim or an advance request	Approve the Claim, print out the supporting SF1166 (Grants Management Branch, not PPS)
Formula	A system process to determine allocations based on elements of the current plan and previous plans	Run "formula backsheet" stored procedures
Forums	A system enabled collaborative forum for sharing information	Enter a new discussion thread or comment on an existing thread on Forum
Calendar	A system enabled calendar that users can contribute to	Enter date and location data
Reports		
Ad-Hoc reports	Ad-hoc reporting on UES data	Build the report as needed. Accurately report on the plan contents for current and previous years. Save report in excel. See print preview of report. Easy generation of contribution reports, funding reports, performance measure reports, financial summary reports, claim history reports, Branded company reports, activity summary reports, evaluation results reports, CPR reports
Pre-defined reports*	Sufficiency Check Report Branded Company report Part Report Program Ceiling Report Allocation Balance CCC Reconciliation Carryover Report UES Application Report Activity Plan Approval Report Country Summary Report Contribution Detail Report CPR	Select the desired report, enter any necessary parameters
Elements of the UES Plan (OPT Owned Data or View) (* elements have a counterpart in the legacy system)		
Element	Description	User Action
Change Response	A response to change data entered by participant	Enter response date and change status
The Strategic Plan*	The Participants application for program funds	View the plan
Targeted Market Division Assessment	The division's comments on a Participant's Targeted Market	Enter the assessment/Comments
Constraint Division Assessment	The division's comments on a Participant's Constraint	Enter relevant comments
Evaluation Elements	Division UES Review, GBI Factor, ISI Factor, CPR Factor, QSP Evaluation	Enter evaluation information
Forums	A system enabled collaborative forum for sharing information	Enter a new discussion thread or comment on an existing thread on Forum
Calendar	A system enabled calendar that users can contribute to	Enter date and location data
Reports		
Ad-Hoc reports	Ad-hoc reporting on UES data	Build the report as needed. Save report in excel.

		See print preview of report. Easy generation of contribution reports, funding reports, performance measure reports, financial summary reports, claim history reports, Branded company reports, activity summary reports, evaluation results reports, CPR reports.
Pre-defined reports*	Sufficiency Check Report Branded Company report Part Report Program Ceiling Report Allocation Balance Carryover Report UES Application Report Activity Plan Approval Report Country Summary Report Contribution Detail Report CPR	Select the desired report, enter any necessary parameters
Elements of the UES Plan (Post Owned Data or view) (* elements have a counterpart in the legacy system)		
Element	Description	User Action
The Strategic Plan*	The Participants application for program funds	View the plan
Targeted Market Post Assessment	The Post's comments on a Participant's Targeted Market	Enter the assessment/Comments for Targeted Markets within the Post's area of responsibility
Constraint Post Assessment	The Post's comments on a Participant's Constraint	Enter relevant comments for constraints occurring within the Post's area of responsibility
Forums	A system enabled collaborative forum for sharing information	Enter a new discussion thread or comment on an existing thread on Forum
Calendar	A system enabled calendar that users can contribute to	Enter date and location data
Reply to Post Notification	A reply to a notification to a Post regarding travel information associated with an activity in the Post's area of responsibility	Change the status of the Post Notification to "Post Acknowledgement" or "Post Cautions"
Reports		
Ad-Hoc reports	Ad-hoc reporting on UES data	Build the report as needed. Save report in excel. See print preview of report.
Pre-defined reports*	Branded Company report Part Report Program Ceiling Report Allocation Balance Carryover Report UES Application Report Activity Plan Approval Report Country Summary Report Contribution Detail Report CPR	Select the desired report, enter any necessary parameters
System Administrator (* elements have a counterpart in the legacy system)		
Element	Description	User Action
The Strategic Plan*	The Participants application for program funds	View the plan
User information*	List of UES system users and their roles	Add or edit user information for UES system, control access to Business Objects reporting tool
Reference tables*	Static data used by the system	Maintain the reference tables
Program information*	Annual data associated with a program (deadline, appropriation number, etc.)	Maintain the annual data

(* elements have a counterpart in the legacy system)

Table 9: User Actions in the system by User Group

2.6 States and/or Modes

The UES Upgrade system shall provide the following operating modes:

- Normal operating mode—shall allow entry and update of all information, interaction with other data sources, reporting, and collaborative sharing of information, while connected to the internet. The following is deferred until a future upgrade when specific requirements have been developed: Shall also allow upload and synchronization of updates performed off-line.
- Training mode—shall allow exercise of all system features by all user roles, using a database specifically identified for training use, simulating both normal operating mode.
- Administrative mode—shall allow maintenance of reference tables, backup and archiving of system data, and other system troubleshooting/database administration functions while connected to the FAS intranet and/or the Internet.
- Deferred until a future upgrade when specific requirements have been developed: Off-line operating mode—shall allow entry, update, and reporting of strategic and tactical planning information (i.e., commodity, market, condition, and practice information) and administrative information while not connected to the internet, for future upload.

2.7 System Administration

The UES Upgrade system shall provide the following functions to users in the System Administration role:

- Maintain reference tables.
- Generate and maintain archive data.
- The generation and restoration of backup data will be handled by the FAS infrastructure processes that backup databases and web servers.
- Generate “audit trail” reports of entry and update of specified information.
- Manage participant (organization) administrative information.
- Manage branded company administrative information.
- Manage user administrative information.
- Maintain database and application.

3 SYSTEM REQUIREMENTS

The Conceptual Data Model for the upgraded UES system is shown in section 4.

This document is accompanied by the detailed listing of all the technical requirements for the UES Upgrade. Specification of each data element and its definition and permitted values make up the bulk of the detailed data-type requirements. Additional data-type requirements specify the relationships among data entities. Functional-type requirements specify how the data is created in the system, i.e., via data entry, system calculation, or system-to-system interface, and how it is extracted, with process type requirements documenting the business rules governing those functions.

The requirements, regardless of type, are organized into functional area groups, Export Strategy, Agreement Administration, Financial Tracking, Actuals/Results, Reporting, User Interface, and System Administration. They are further organized into subgroups under each of those categories as shown in Table 8. The unique Requirement Number for each requirement indicates the functional area group and subgroup, then further logical breakdown within that. For example, the requirement specifying the items that comprise Substantiative Industry Information is numbered as 01.02.11.200 while the requirement specifying which users may enter and update this information is numbered as 01.02.11.204 and the requirements defining each of the Substantiative Industry Information items are numbered as 01.02.11.220 through 01.02.11.228.

Requirement Functional Area Group	Requirement Functional Area Subgroup	Req. Number "Family"
Export Strategy	Industry	01.02
	Strategic Executive Summary	01.04
	Commodity	01.06
	Market	01.08
	Constraints/Opportunities	01.10
	Activities	01.12
	Collaborative Information Sharing	01.14
Agreement Administration	Application	02.02
	Application Review and Evaluation	02.04
	Agreement Award and Amendment	02.06
	Agreement Modifications	02.08
Financial Tracking	Requesting Funding	03.02
	Budget Allocation	03.04
	Claims (Advances/Reimbursements)	03.06
	Contributions	03.10
	Financial Status	03.10
	Accounting	03.12
Actuals/Results	Country Progress Report (CPR)	04.02
	Trip Report	04.08
	EMP Reports	04.10
	Branded Company Sales	04.10
Reporting	Reporting – General	05.02
	Pre-Defined Reports	05.04
	Ad-Hoc Reporting	05.06
	External Data Sources	05.08
User Interface	Roles	06.02
	Data Entry	06.04

	Alerts	06.06
	Attachments	06.08
	Application Notes (Footnotes)	06.09
	Help	06.10
System Administration	Participant Administrative Information	07.02
	Branded Company Administrative Information	07.04
	Post Administrative Information	07.06
	User Administrative Information	07.08
	Access Control	07.10
	Database Administration	07.12
	Reference Table Maintenance	07.14
Data Migration	Data Migration	10.02

Table 10. Requirements: Functional Area Groups and Subgroups

The requirements statements listed in the accompanying detailed requirements document are also characterized in several other ways, including requirement type, verification approach, and priority. Table 9 lists the defined requirement types.

Type	Type Definition
Data	Specify data entity contents or characteristics, such as type, edit criteria, permitted values, relationship to other data, or whether it is mandatory or optional
Functional	Specify actions the system must perform or provide users the capability to perform
Process	Specify rules for performing actions, such as the order in which actions must take place or conditions necessary for the performance of an action
System Interface	Specify interfaces and interactions with anything outside the system, for example, other databases or applications, procedures, hardware
User Interface	Specify the interaction between the system and users and standards defining that interaction such as standards for Graphical User Interfaces (GUI) or Section 508
Performance	Specify the system's operation in its environment, including response timing, calculation accuracies, capacity, flexibility, expansion, maintainability, availability, and reliability
Security	Specify rules for access and modification of system data, data storage, communications, and other controls mandated by security standards and procedures

Table 11. Requirement Types

Data type requirements make up the preponderance of the requirements, defining the detailed elements that comprise the higher-level conceptual entities presented in Section 4's conceptual

data model. Data type requirements also specify the relationships among the individual data elements, e.g., a single constraint/opportunity must have at least one activity. Permitted values for a specific data element are also specified in data type requirements, for example, permitted values for Country Average Income Level are: high, upper-middle, lower-middle, and low. A number of data type requirements also specify whether a particular data element is mandatory or optional, e.g., Market Type would be mandatory to describe a Targeted Market.

Functional requirements address actions that the UES Upgrade system must be capable of, for example, enable users in specific roles to enter certain data, or specify that the system must calculate certain information. UES Upgrade functional requirements include requirements such as: the system shall enable users in the "program participant" role to enter contribution data for each Program Year or the system shall calculate, for each contribution cost category, the total across the three contribution source categories.

Process type requirements translate business processes or other business rules into technical requirements, for example, the system shall require that the user enter one, and only one, value for Promised Contribution, either Promised Contribution $ OR Promised Contribution %. For another example, if the value for Type in the associated Activity is "Branded" then the following Branded-specific Event information shall be considered mandatory: 1) Branded Company Name 2) Child Amount.

System interface type requirements specify the rules for interactions between the UES Upgrade system and other systems. ~~System interface type requirements spell out the details of the information flow between USDA's Grants Interface Module (GIM) and UES, accepting application content and receipt information from GIM and transmitting application status information back to GIM.~~

User interface type requirements are used to specify the performance, appearance, and operation of the human-machine interface for the system, encompassing requirements stating USDA web-page design standards, federal government accessibility standards (Section 508), and other usability requirements articulated by the UES user community.

Performance type requirements specify a wide range of characteristics describing acceptable performance from the Upgrade UES system. Performance type requirements place an upper limit on response time from the system to an entry or query by a user. This type of requirement also specify, for example, that funding calculations shall be calculated to decimal places and displayed as US Dollars currency. An important set of performance type requirements state the degree of expandability and flexibility of the system, for example, that the system shall capable of expansion to handle funding applications for another USDA program.

Security type requirements address many aspects of the system, including such things as defining the privileges available to users in various roles, specifying the rules for user access to data, and tracking changes to data. Security type requirements also specify that the upgraded system must remain in compliance with USDA and federal government security guidelines, for example, those necessary to successfully undergo Certification and Accreditation (C&A).

Table 10 defines the verification approach proposed for each requirement. An important criterion for a valid, well-defined requirement is that it must be possible to verify that requirement once the system has been constructed. Every requirement, therefore, has a recommended or proposed approach to its verification. **Analysis** is proposed to verify those requirements that cannot be "observed' from outside the system, for example, verifying that an allocated funding or contribution data element is numeric. **Inspection** is an appropriate approach to verifying such requirements as the system shall include a graphical user interface (GUI), i.e., those requirements for which simply observing the application will suffice, without any particular activities taking place. **Demonstration** is the preferred approach to verifying a system operation or function that does not depend upon specific data values, for example, the system allowing entry of an export goal value, validating that only a number is allowed, but placing no specific limits on that number. **Test** is the most stringent type of verification approach, recommended to verify that specific inputs result in specific outputs, for example, calculation of a total contribution value as the sum of contribution values for several sources. High-level requirements or constraints are verified by addressing the lower-level derived requirements.

Verification Approach	Verification Approach Description
Analysis	Verify via analytical examination of models or mathematical algorithms,
Inspection	Verify via examination of the database structure, data contents, or application user interface screens
Demonstration	Verify via exercise of the system, but not with specific Input and Expected Output values
Test	Verify via exercise of the system, using specified scenarios and test data.
None	Concepts, guidance, high-level statements verified at the detail level

Table 12. Requirement Verification Approaches

The final characteristic describing each requirement is the recommended priority. Each requirement listed in the accompanying detailed requirements document includes a recommended or proposed priority, described in Table 4. **Critical** requirements represent essential data, features, and functions. Failure to implement a critical requirement means the system will not meet user needs. All requirements in the approved baseline identified as critical priority must be implemented in the upgraded system. **Important** requirements define data, features, and functions that are important to the effectiveness and efficiency of the system for most uses. In these cases, the data or functionality cannot be easily provided in some other way so that lack of inclusion of an important feature will probably reduce user satisfaction. **Useful** requirements define data, features, or functions that are useful in less typical operating modes, will be used less frequently, or for which reasonably efficient workarounds can be achieved. These features can be earmarked for consideration during future enhancements to the system but their lack should not reduce user satisfaction with the upgraded system.

Priority	Priority Definition
Critical	Essential features. Failure to implement means the system will not meet user needs. All critical features must be implemented.

Important	Features important to the effectiveness and efficiency of the system for most uses. The functionality cannot be easily provided in some other way. Lack of inclusion of an important feature may affect user satisfaction.
Useful	Features that are useful in less typical operating modes, will be used less frequently, or for which reasonably efficient workarounds can be achieved.

Table 13. Requirement Priority Levels

Business Requirements

Business Requirements	
BusReq_ID	**Requirement**
1	Users shall have the ability to enter and update export marketing strategies.
2	Users shall have the ability to search for and report on long-term export marketing strategies that apply to a particular country or countries.
3	Users shall have the ability to search for and report on long-term export marketing strategies that apply to one or more particular commodities or products.
4	Users shall have the ability to associate long-range export marketing strategies to specific markets and commodities or products.
5	The system shall support the annual submission of applications for program funding.
6	Cooperator/Program Participant users shall have the ability to submit requests for funding from the following programs as part of their applications for cooperative agreements: - MAP (Market Access Program) - FMD (Foreign Market Development Program) - TASC (Technical Assistance for Specialty Crops Program) - EMP (Emerging Markets Program) - QSP (Quality Samples Program) - Cochran Fellowship Program
7	The system shall be extensible to allow support for requesting funding from additional FAS programs in the future.
8	Access to information contained in the system shall be determined by the user's role.
9	The system shall contain information categorized as: - Administrative - Financial - Strategic Planning - Annual/Tactical Planning - Progress
10	Administrative data consists of information that identifies users, their locations, and contacts. Administrative data includes cooperator or program participant identifying information necessary to administer a program agreement.
12	OTP staff users and PPS users shall have the ability to review and evaluate Cooperator's/Program Participant's requests for program funding.
13	OTP staff users shall have access to Cooperator's/Program Participant's strategic planning and annual planning information when reviewing requests for program funding.
14	OTP staff users shall have access to Cooperator's/Program Participant's performance reporting information when reviewing their request for program funding.
15	Financial data includes funding information described by the following categories: - Appropriations and Apportionment - Requests - Allocations - Obligations - Budgets - Expenditures - Claims - Contributions
16	Financial data in the "Requests" category is information concerning funding requested from a program. This includes information such as that specifying funding amounts, identifying a program, identifying a program year, or identifying the requestor (Cooperator/Program Participant or Overseas post).
17	Financial data in the "Allocations" category is information concerning funding identified to address a request for program funding. This includes information such as that specifying

Business Requirements	
BusReq_ID	**Requirement**
	funding amounts, identifying a program, identifying a program year, or identifying to whom the funding is allocated (i.e., Cooperator/Program Participant or Overseas Post) .
18	Financial data in the "Obligations" category is information concerning funding identified in an executed agreement or amendment. This includes information such as that specifying funding amounts, identifying a program, identifying a program year, or identifying a specific agreement.
19	Financial data in the "Budgets" category is information concerning funding whose planned use has been identified by the Cooperator/Program Participant or Overseas Office.
20	Financial data in the "Expenditures" category is information concerning funding identified by the Cooperator/Program Participant or Overseas Post as having been already used. This includes information about expenditures that may or may not be reimbursable under the terms of a program agreement.
21	Financial data in the "Claims" category is information concerning funding identified by a program agreement as being "reimbursable" by the government, whether an "advance" before expenditure or a "reimbursement" following expenditure. This includes information such as that specifying the funding amount, identifying the relevant program, describing the expenditure's purpose, identifying the relevant agreement, or indicating the status of a claim.
22	Financial data in the "Contributions" category is information concerning funding identified as provided by the Cooperator/Program Participant under an agreement. This includes information such as that specifying the funding amount, identifying the contribution source, describing the purpose for which the contribution was used, identifying the program and program year, or identifying the relevant agreement.
23	OTP staff users shall have the ability to review and evaluate Cooperator's/Program Participant's applications for program funding.
24	OTP staff shall have the ability to share questions and comments with Cooperators/Program Participants concerning their applications for program funding and their strategic and tactical/annual export marketing plans.
25	Overseas Posts staff shall have the ability to share with OTP staff questions and comments concerning Cooperators'/Program Participants' applications for program funding and their strategic and tactical/annual export marketing plans.
26	Cooperators/Program Participants shall have the ability to share strategic and annual planning information (developed both by themselves and by Overseas Post staff) with OTP staff and Overseas Posts staff during development and refinement of their strategic and tactical/annual export marketing plans and their annual applications for program funding.
27	Data entry user interfaces shall comply with "industry best practices" and USDA standards and guidelines.
28	The system shall provide easy generation of a "Cooperator/Program Participant UES Report," presenting the marketing strategies and annual plans for a specified year for a specified cooperator/program participant, in an easy to read, printable format. The family of reports shall be filtered, sorted, grouped, and formatted as specified by the user. The "owner" Cooperator/Program Participant shall have the ability to save a preferred format for the report for use by other system users.
29	The system shall provide an "ad hoc" reporting capability, allowing users to generate reports in user-specified format, containing user-specified information.
30	The system shall provide users with the ability to specify report contents, constraints, and formats in "plain English" terms.
31	The system shall provide the capability of generating reports displayable on web pages.
32	The system shall provide the capability of generating reports that can be downloaded

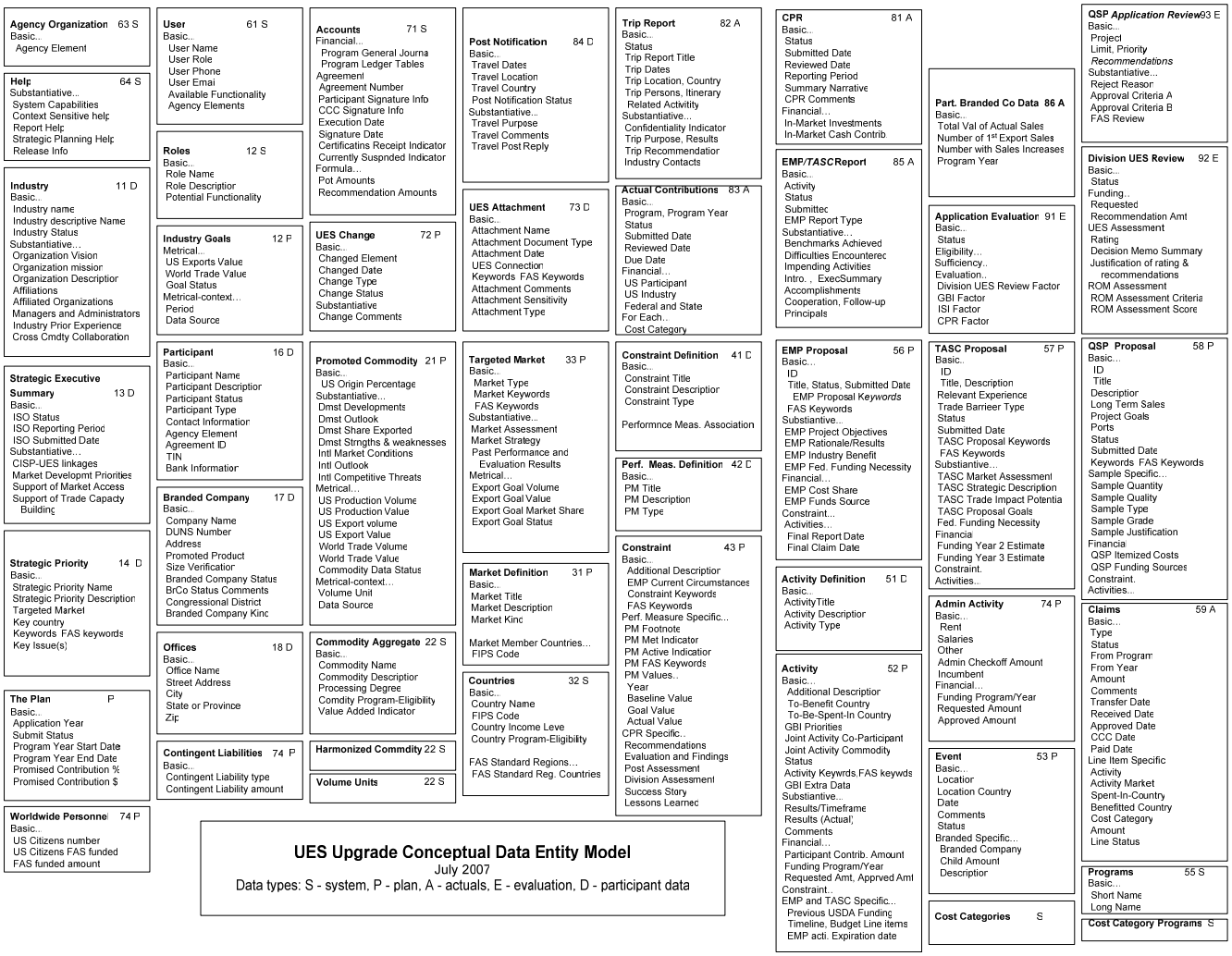
Business Requirements	
BusReq_ID	**Requirement**
	directly into Excel or Lotus 123 spreadsheet files.
33	The system shall provide the capability of generating reports in printable formats that are "previewable" on a web page (i.e., What You See Is What You Get).
34	The system shall be compatible with the defined FAS Enterprise Architecture.
35	Users shall have the ability to define a "market" as one or more countries related in some way that affects strategic export planning, e.g., geographically close, similar level of "market maturity," part of a "supply chain," or part of an international organization with common trade policies.
36	Users shall have the ability to define one or more opportunities for creation or expansion of a market. An opportunity may apply to more than one market.
37	Users shall have the ability to identify one or more constraints to creation or expansion of a market. A constraint may apply to more than one market.
38	Users shall have the ability to identify constraints that apply to more than one market.
39	Financial data in the "Appropriations and Apportionment" category is information concerning funding approved by Congress for a program. This includes information such as that specifying funding amounts, identifying a program, or identifying a program year.
40	Overseas Post users shall have the ability to review and comment on strategic and tactical information supporting Cooperators'/Program Participants' applications for program funding.
41	OTP staff users shall have the ability to approve terms for program agreements that have been successfully negotiated. The terms of the agreement include identification of the Cooperator/Program Participant, program funding amounts, period of performance, and specification of the planned use of program funding.
42	Cooperator/Program Participant users shall have the ability to submit proposed modifications to their approved strategic and tactical plans during the period of performance of their agreement.
43	Overseas Post users shall have the ability to review and comment on proposed modifications to Cooperators'/Program Participants' approved strategic and tactical plans.
44	Users shall have the ability to generate periodic progress reports.
45	Users shall have the ability to generate end-of-year progress reports. Cooperators/Program Participants reports will address their agreements; overseas posts reports will address their country's(ies') progress.
46	Users shall have the ability to enter trip report information or upload it as text or spreadsheet files.
47	Users shall have the ability to enter or upload calendar information, i.e., information concerning planned events.
48	Users shall have the ability to update calendar information, e.g., noting changes in planned events or completion of scheduled events.
49	All users shall have access to calendar information.
50	All users shall have the ability to participate in discussion forums.
51	Users shall have the ability to specify "notification" of specific events. Notification may be (at the user's discretion) active or passive, for example, displayed as an alert on a web page, accessible in a list of notifications transmitted in an e-mail.
52	PPS users shall have the ability generate reports for the "back sheets" that support the Allocation Formula for a specified year.
53	MOS users shall have the ability to view the results of "what if" analyses on the Allocation Formula, i.e., modifying one or more factors and viewing the resulting budget allocations.

Business Requirements	
BusReq_ID	**Requirement**
	REQUIREMENT DROPPED
54	Users shall have access to "on-line" Help information. Help information shall be accessible via several means, for example, "context-sensitive," indexed by functions and features, searchable by "plain English" questions.
55	"Help" information shall be provided for each type of user (role) as well as for users in general.
56	"Help" information shall include guidance for developing strategic and tactical plans.
57	"Help" information shall include instructions for developing and entering information necessary for generation of the "UES Plan" report.
58	"Help" information shall include guidance on how to submit applications for program funding using the grants.gov web site.
59	The system shall comply with USDA and FAS security requirements.
60	The system shall follow USDA guidance on integration with the e-Grants program.
61	"Help" information shall include notifications of and explanations of new releases of the system. Information should include, for example, summaries of new features or functionality, access to specialized help for new features, or "alerts" that changes are scheduled.
62	The system shall provide users access to non-CMP (OTP) sources of information. These sources may include other databases or web sites owned and managed by FAS, for example, the Food Aid Data System (FADS).
63	The system shall provide users access to non-FAS data sources. These sources may include other USDA or other federal agency databases or market research data prepared under contract to USDA, cooperators/program participants, or states.
64	User access to the system shall be controlled, e.g., password-protected.
65	User access to system functions and features shall be controlled by the user's "role." Roles shall include: Cooperator/Program Participant, Overseas Post, OTP Staff, PPS staff, Grants Management, Management, non-CMP FAS staff, IT/system administration/support.
66	The system shall provide users access to historical data. Historical data shall include data in any UES information category (administrative, financial, strategic planning, annual/tactical planning, progress) from the legacy UES database and its predecessors (USAM and MDIS).
68	Users shall have the ability to enter and update annual/tactical export marketing plans associated with longer-term marketing strategies.
69	Users shall have the ability to enter and update specific activities associated with annual/tactical marketing plans.
70	Users shall have the ability to enter and update performance measures associated with opportunities and constraints.
71	Users shall have the ability to enter information concerning performance of planned activities. This information would include any differences between the planned and actual activity, its dates, participants, or costs.
72	Users shall have the ability to enter information assessing progress against specific performance measures resulting from performance of planned activities.
73	Users shall have the ability to enter and update market assessment information. This information would include "background" or historical information concerning the market, recent economic or political developments, and conclusions on future market conditions.
74	Strategic planning data consists of information defining long-term strategies or their goals, such as commodity and product definitions, market definitions, strategy descriptions, market assessments, opportunities, constraints, or performance measures.

Business Requirements	
BusReq_ID	**Requirement**
75	Tactical or annual data consists of information concerning the translation of long-term strategy into short-term plans, usually on an annual basis, such as specific planned activities, their schedules and budgets.
76	Progress data consists of information concerning performance of planned activities, implementing tactical or annual plans, and their relation to fulfilling strategic plans.
77	Progress data includes dates, participants, contents, results, and costs of planned activities that have been performed.
78	Progress data includes information concerning performance against performance measures, for example, numbers of potential importers soliciting information on a particular product, or quantity/value of exports of a particular commodity.
79	Cooperator/Program Participant users shall have the ability to submit claims for advances or reimbursements against their approved budgets of program funding.
80	The system shall track financial status and details of each Cooperator/Program Participant for all programs across all program years.
81	Users shall have the ability to attach/upload or enter information for and generate other reports such as research reports or final Project Reports.
82	The system shall provide a set of pre-defined reports containing planning, administrative, financial, or progress information.
84	The system shall provide easy generation of Contribution Reports containing contribution-related information such as promised contributions, actual contributions, submittal date of actual contributions reporting, and reviewed date for actual contributions reporting. The reports shall show, for each participant, the total contribution for an "application year" as well as program-by-program details.
85	The system shall provide easy generation of Combined Funding Sources Reports, containing reported contribution amounts and program funding amounts, across time (actuals for past years, approved budgets for current year, proposed/requested for application year) and related administrative, commodity, and market information and keywords. Filtering, sorting and grouping of the reports shall be specified by the user, allowing filtering on participant, year, and/or funding source and grouping by market, program, or commodity.
86	The system shall provide easy generation of Performance Measure Reports containing performance measures across time (past years, current year, proposed/requested application year) and identifying administrative and condition information, filterable by participant(s), year(s), and key words. Report sorting, grouping, and formatting shall be specified by the user.
87	The system shall provide easy generation of Financial Summary Reports containing program participant financial information (any or all categories) and related administrative and practice information, filterable by participant(s), program(s), or year(s). For Program Participant users, reports shall be limited to that participant's information, sorted by program. For OTP staff users, reports shall be capable of showing multiple participants' information, sorted by participant and by program. For all users, the level of detail (individual line items, activities, or summaries) shall be specified by the user.
88	The system shall provide easy generation of Claim Status/History Reports containing claim information (financial, administrative, and status) filtered by participant(s), program(s), or year(s). Report sorting, grouping, and formatting shall be specified by the user.
89	The system shall provide easy generation of Branded Company Reports containing Branded Company administrative, status, and related activity information, across multiple years, filtered by branded company(ies) or participant(s). For Participant users, the report shall be

Business Requirements	
BusReq_ID	**Requirement**
	limited to that participant and the branded companies associated with them. For other users, the reports shall be limited to the time frame, status, type of branded company, and/or key words specified by the user. Report sorting, grouping, and formatting shall be specified by the user.
90	The system shall provide easy generation of Activity Summary Reports containing Practice, and related strategic, financial, and administrative information, filtered by a user-specified combination of program(s), year(s), funding source(s), market(s) spent in, market(s) benefited, commodity(ies), commodity aggregate(s), or HS code(s). Report sorting, grouping, and formatting shall be specified by the user.
91	The system shall provide easy generation of Application Review and Evaluation Results Reports containing results (narrative and scoring) of application reviews filtered by a user-specified combination of participants, OTP commodity branches, project types, or category of evaluation (ex., UES scoring, ISI, GBI, etc.). Report sorting, grouping, and formatting shall be specified by the user.
92	The system shall provide easy generation of Application Review and Analysis Reports containing practice, financial, administrative, and other strategic and tactical information useful as “raw materials” for application reviews by marketing specialists, filtered by commodity branch. Report sorting, grouping, and formatting (including calculation of averages, sums or other mathematical functions) shall be specified by the user.
93	The system shall provide easy generation of Country Progress Reports containing all progress information, administrative information, and financial information currently contained in CPRs, for one participant and one reporting year. Report sorting, grouping, and formatting shall be specified by the user. Cooperator/Program Participant users shall be capable of defining a user-specified format, defined by the “owner,” identified as the “preferred presentation” format, available for use by all authorized system users.
94	The system shall provide easy generation of Combined Funding and Trade/Exports Reports containing program funding and contribution financial information combined with trade and export statistics information, over several years. Report sorting, grouping, and formatting (including calculation of averages, sums, or other mathematical functions useful for trend analyses) shall be specified by the user.
95	The system will allow for a participant to create a new yearly plan by copying from the 'living document' version of a previous plan where the user can specify the year of the plan to copy from.
96	The system will save a file snapshot of the plan at submission time, and allow the users to save a snapshot of the plan at any other desired time.
97	The system will produce a CPR report that will produce for each participant/targeted market per year the reviewed data and status, in-market investments, the in-market cash contributions, a summary narrative, comments and for each related constraint the Recommendations, Evaluation and Findings, Post Assessment, Division Assessment, Success Stories and Lessons Learned and the performance measures with the performance history starting from two years back.

4 CONCEPTUAL DATA ENTITY MODEL



5 SYSTEM DESIGN CONSTRAINTS

This section provides a discussion of identified design constraints, expanding on the brief summaries in the UES Stakeholder Needs Assessment Report.

5.1 Constraints

Requirements have been defined in the accompanying Detailed Requirements Document reflecting the following constraints on the
design and/or implementation of the upgraded system:

- System Security and integration with e-Authentication.
- FAS Technical Infrastructure.
- Integration with e-Grants (Note: this can be deferred until a future release, if it becomes necessary: new grant.gov compliance rules now require only the posting of the opportunity announcement).

5.2 Development Process

Research on reasons for software development failure at The Standish Group, who studies such things, indicates that smaller time frames, with delivery of software components early and often, will increase the success rate of software development projects. Shorter time frames result in an iterative process of design, prototype, develop, test, and deploy *small elements*. This process is known as "growing" software, as opposed to the old concept of "developing" software. Growing software engages the user earlier, each component has an owner or a small set of owners, and expectations are realistically set. In addition, each software component has a clear and precise statement and set of objectives. Software components and small projects tend to be less complex. Making the projects simpler is a worthwhile endeavor because complexity causes only confusion and increased cost. With this as the preferred development method, the UES legacy system could be “grown” to meet the UES upgrade requirements. The starting place could be the porting of the legacy Sybase database to SQL server 2005. The current ‘Marketing Specialist Interface’ could have minor modifications to accept participant users, if run on an ISA server that would allow outside users to run an application through the FAS firewall. The ‘Marketing Specialist Interface’ could be upgraded piece by piece and then the new functionality added, bringing the UES into compliance with the upgrade requirements by proceeding in small stages. Stakeholders do not have to wait for the full blown Upgrade Software to see benefits.

5.3 System Security

The upgraded UES system shall be subject to the same security requirements as the current UES system. The current UES system, along with all other FAS systems, underwent the security Certification and Accreditation (C&A) process specified by the National Institute of Standards and Technology (NIST) during the 2007 Fiscal Year. USDA’s (and, thus, FAS’s) security policies are based on the NIST guidelines. Completion of the C&A process included a complete assessment of system vulnerabilities and verification and validation of the system design and implementation features that address them. The upgrade system shall be subject to the same C&A process.

From a security standpoint, the most important factors of a system upgrade are the extent of changes to: the information contained in it, information transmitted to and from it, identification and location of its users, and the system’s specific security functions and features. In the case of the planned UES upgrade, the categories of information to be contained in or transmitted to and from the upgraded system shall not be changed or expanded. The system’s information domains shall continue to include Cooperator/Program Participant and FAS “strategic” and “tactical” planning information and cooperative agreement administrative and financial tracking information. Similarly, the user groups shall remain unchanged, cooperators and industry representatives from the “public” domain and FAS staff and managers (located in Washington and at overseas posts) from the “FAS” domain.

The specific security functions and features of a system are dependent upon the system’s design and implementation so the upgraded system will address security requirements differently than the current system does. The constraint on design and implementation of the upgraded system is that it must meet the same security requirements and address the same identified vulnerabilities.

The upgraded system is not, however, constrained to meet these requirements in the same way as the current system.

The upgraded UES system shall provide access control by integration with the USDA e-Authentication system.

5.4 Technical infrastructure:

Over the last several years, FAS has pursued a schedule of upgrades to server hardware and software, telecommunications bandwidth (particularly with overseas posts), and desktop computer hardware and software (at Headquarters and overseas posts). The current FAS infrastructure offers more than adequate server storage, backup, and redundancy capacity with the latest hardware and operating systems necessary to implement the Microsoft .NET/Sybase system architecture. Recent upgrades in telecommunications quality and bandwidth for overseas posts have made the current system less susceptible to sudden "dropped connection" problems. The web services-based design philosophy generally used today also minimizes the amount of data entered per individual web page and minimizes the expected telecommunications burden. The upcoming deployment of an ISA server by FAS will eliminate the need for using web services to cross the firewall.

The upgraded UES system shall be implemented using a Sybase database and Microsoft .NET technology for web-based user interfaces, in compliance with the identified FAS technology solution.

5.4.1 Sharepoint

Microsoft Sharepoint is part of the FAS infrastructure. Where appropriate, Sharepoint should be used to meet UES Upgrade requirements. UES requirements that may be able to be met with Sharepoint include the Issues Forum collaborative discussion tool, calendar, alerts, announcements, and attachments (upload, search, retrieve), the configurable portal, etc.

5.5 eGrants

The federal eGrants initiative is one of 24 initiatives of the overall eGovernment program for improving access to government services via the Internet. The UES is considered a grant making system. Currently, agencies can comply with this initiative by posting the opportunity announcements. UES users will be submitting their request for funding through the UES online system, not by submitting a SF424 through grants.gov, which is no longer required.

5.6 Data Migration

The new system will need to migrate legacy data so the interface and reporting tools can be used to seamlessly view legacy UES data with the future UES upgrade data.
The legacy inside the firewall production server will be the main source of migrated data (sybase07.usamprod). The difficulty of the data migration task is heavily dependent on the database design used for the upgraded system. The bulk of the Plan data is the same in the new

system as it was in the old system so many tables will need little or no modification. If the design of the upgraded system is compatible with the legacy system database design, migration issues will be minimized.

The Data Migration Problem

The current UES system used two Sybase databases, one inside (sybase07.usamprod) and one outside the firewall (sybase01.usamec), with some tables now being replicated between the two. The new system will have a single SQL Server 2005 database, inside the firewall.
All the data we want to keep needs to end up in the new system. There are no requirements for archiving or retiring any data.

system data can be categorized in 4 ways, with different categories potentially requiring different migration treatments.

1. Unneeded tables – some tables in exist to support the replication process or reports that will not be part of the new system. No data migration needed. These tables will be dropped from new system.

2. Historical tables – some tables exist in the current system with historical data and no corresponding counterpart in the new system. These tables should be ported and populated as is because the legacy database will be retired after the completion of the upgrade. Tables with MDIS and USAM data in them meet these conditions, but there is no separate MDIS or USAM database. Accounting stored procedures may access data in some of these tables so the name of the table should be the same unless there is a plan to also change dependant stored procedures. No new data is being added to these tables so they can be ported at any time and no further work would be necessary.

3. Accounting tables – These tables with data should be ported and populated as is. The dependent objects (stored procedures, views, triggers, user defined types, etc) should also be migrated. The tables have both historical and current sequential data and when we cut over to the new system the sequences (transaction number, document number) should be maintained and increment smoothly in the new system. Data is being continually added to these tables so there needs to be a coordinated cut off from the old system and transition to the new system. The raiserror numbers in the sybase stored procedures will have to be migrated to numbers above 50000 in SQL Server and the messages migrated from the Sybase sysusermessages to the sql server sysmessages table.

4. UES application data tables – Some transformation of the data tables may be required to meet the requirements of upgraded system. The migration of the legacy data should be kept in mind during any table design. Table changes will also affect the dependent stored procedures, which will need to be upgraded in turn. In general, these requirements do not specify the data type so for data that exists in the legacy system that will continue to be collected in the new system, the data type should be determined from legacy data.

Other Issues

Maintenance Site – these are tables outside the firewall that handle pending requests to make changes in the plan. These have a status and it would be good to ensure that there are no open requests at the time of migration. Both Participants and Marketing Specialists are involved in closing the requests. The “maintenance site” functionality will no longer be necessary when the UES system is upgraded because participant users will have a much more seamless way to keep their data current.

Logins migration – Legacy Participant representative users will not have a login from the legacy system and they must register through eAuthentication for level 2 access. Legacy Participants have a login for the group which will not be used in the new system since the login will be by individual, through eAuthentication.

Archiving – there is no requirement to archive anything

The cutover from old system to new needs to be part of the migration plan. Because of the transactional accounting there cannot be two systems (old and new) running in parallel. The cutover time should be minimized.

5.7 Reporting

FAS will choose a reporting tool that the application will integrate with. The scope of the upgrade includes designing a reporting tool semantic layer to support ad-hoc queries. The upgraded system requires a single sign on so an ad hoc user can go to the reporting function from the upgraded UES without having to log in and so the tool will recognize the user and apply appropriate security. The pre-defined reports will also be accessed through the application without the user needing to log in.

5.8 Applicable Standards

The following standards will be applicable to the UES Upgrade system design and construction.

USDA Web Style Guidelines

SQL Server 2005 database

Asp.NET interface

APPENDIX A: Terms

Accounts -Data Entity that contains financial details for participants by program and year. One of the Data Entities in the "Agreement" Information Group.
Activity -Data Entity that contains financial, expected/actual results and supplemental descriptive information, concerning a UES Activity, including financial details
of EMP and TASC proposed projects. One of the Data Entities in the "Activity" Information Group.
Activity Definition -Data Entity that contains basic descriptive information concerning a UES Activity. One of the Data Entities in the "Activity" Information Group.
Actuals -Information Group that collectively represents the information related to progress and actual vs. planned performance, such as CPRs, reported contributions, or project progress reports.
Ad hoc -A term meaning "for the particular case at hand without consideration of wider application," generally used to refer to data reporting capabilities allowing an individual user to define a report, in contrast to pre-defined, formatted reports.
Advance -A type of UES Claim in which a participant is advanced funds after program ceilings are set and allocations are determined. Advances are analogous to loans--Advance balances must be paid down to zero before reimbursement payments may be received.
Agency Organizations -Data Entity that contains basic identification information about FAS organizational elements, such as the C&MP commodity divisions, ICD … One of the Data Entities in the "Users" Information Group.
Agreement -The legal instrument binding the U.S. government, USDA/FAS, and an organization participating in the MAP, EMP, TASC, QSP, Cochran, or other program. The agreement specifies the terms and conditions including the level of FAS program funding and the cost-share contribution requirements from the program participant. In the case of EMP, TASC, and QSP agreements, details of each project as approved by FAS will also be included. Information Group that collectively represents the information related to program participant agreements.
Allocation Formula -A series of calculations based on performance-related factors used by MOS to define the budget allocations for each applicant for the MAP and FMD programs during the annual (or additional out of cycle) application cycle.
Allocation -The amount of funding for a program awarded to a program participant for a specific program year (documented in the Agreement or Agreement Amendment).
Allocation -Data Entity that contains information concerning the determination of the allocated budgets for each applicant each year. One of the Data Entities in the "Evaluation" Information Group.
Allocation Formula -A series of calculations based on performance-related factors used by MOS to define the budget allocations for each applicant for the MAP and FMD programs during the annual (or additional out of cycle) application cycle.
Amendment -A change or modification of an Agreement that happens one or more times per year. The change nearly always is a change in allocated funding. AMP Annual Marketing Plan, outlining Overseas Posts' priorities and plans.
Announcement -Official government notice, published in the Federal Register, regarding a year's funding of a specific program, including program summary, rules, and application information. A UES accounting "event", a.k.a. Program Announcement, which sets the overall program/year obligation.
Application Evaluation -Data Entity that contains detailed information concerning FAS review and evaluation of applications, including the UES Score Sheet. One of the Data Entities in the "Evaluation" Information Group.
Application Notes -Data Entity. Application Notes is a "feature" which provides a tool for participant users which helps them track the completeness of their UES submission on a component-by-component, or section-by-section basis.
APHIS -Animal and Plant Health Inspection Service (USDA agency)

Appropriation -The amount of funding authorized for expenditure for a specific program, for a specific program year (e.g., MAP-2005, EMP-2005, or FMD-2004).
ATO -Agriculture Trade Offices
Award Amount -A dollar amount representing FAS approved program/year funds which will be received by a Project or Participant. The Award Amount sets a Participant's program-budget ceiling.
Award Notice -An FAS notification to each applicant in writing of the final disposition of its application. The FAS sends an approval letter and project agreement to each approved applicant. The approval letter and agreement will specify the terms and conditions applicable to the project, including the levels of MAP funding and cost-share contribution requirements.
BICO -Bulk, Intermediate, and Consumer
Branded Company -A small company producing a specific product(s) that are the subject of export marketing under the auspices of a MAP program participant organization.
Branded Companies Data Entity that contains detailed information concerning the branded companies, such as name, DUNS, promoted products, etc.
Briefing Book Issue -An item in the UES Assessment Score Sheet for a specific program participant's application identified as an issue that should be included in the MAP/FMD briefing books for senior USDA officials.
Budget -The amount of planned funding identified for performance of an
Activity, either at a total Activity level or at an individual Event level or at an individual Branded Company activity level or at an individual EMP or TASC proposed activity level or at an individual EMP or TASC line item (within a proposed activity) level.
Carry-over- A UES Accounting term describing a dollar amount of unspent allocated program funds.
C&A -Certification and Accreditation of a system's compliance with National Institute of Standards and Technology (NIST) security guidelines and procedures.
CCC -Commodity Credit Corporation, the USDA/Farm Service Agency organization that disburses funds to program participants as advances or reimbursements.
Ceiling -A UES accounting term describing a dollar amount available to a Participant at any given time which is less than or equal to their total allocated funds available.
CISP -Comprehensive Industry Strategic Plan, combines high-level strategies addressing both domestic and export marketing.
Claims -Data Entity that contains information concerning claims by program participants for advances or reimbursements. One of the Data Entities in the "Activity" Information Group.
Commodity -A generic, largely unprocessed agricultural good or group of goods that can be processed and resold. These include food, feed, fiber, wood, livestock or insect, and any product thereof; and fish harvested from a U.S. aquaculture farm, or harvested by a vessel as defined in title 46, United States Code, in waters that are not waters (including the territorial sea) of a foreign country. Information Group that collectively represents the information related to commodities.
Commodity/HS X-Ref -Data Entity that contains a cross-reference between the UES commodity codes and the Harmonized Code System commodity codes. One of the Data Entities in the "Commodity" Information Group. The specific commodity that a program participant wishes to increase exports of to a specific market.
Constraint -A constraint (barrier or limitation) or opportunity (favorable or advantageous) related to increasing exports of a specific commodity to a specific market, i.e., the unique combination of a Constraint Definition and a Targeted Market. Information Group that collectively represents the information related to conditions (opportunities/constraints).
Data Entity that contains additional descriptions of specific conditions as well as performance measure results and CPR-specific assessment information from posts and commodity division.
One of the Data Entities in the "Constraint" Information Group.
Constraint Definition -Data Entity that contains information defining "generic' or common constraints or EMP Proposal-specific conditions. One of the Data Entities in the "Condition" Information Group.

Cash and services that various entities invest in UES activity that are not FAS program funds.
Contributions -Data Entity that contains details of participant's contributions by program, by year (essentially the contents of the Contribution Report). One of the Data Entities in the "Actuals" Information Group.
Cooperative Agreement -The legal instrument binding the U.S. government, USDA/FAS, and an organization participating in the FMD program. The agreement specifies the terms and conditions including the level of FAS program funding and the costshare contribution requirements from the program participant.
Cooperator -A non-profit U.S. trade organization, representing producers of the commodity being exported, that has entered into a Cooperative Agreement with USDA/FAS, receiving funding via FMD.
Cooperators/ProgramParticipants -The inclusive term used for organizations that are participating in any of the programs within the scope of the UES upgrade system (MAP, FMD, EMP, TASC, QSP, and Cochran) and that enter into a Program Agreement with USDA/FAS.
COTS -Commercial Off The Shelf, a software application that is commercially available.
Countries -Data Entity that contains basic descriptive and categorical information about individual foreign countries. One of the Data Entities in the "Market" Information Group.
CPR -Country Progress Report Data Entity that contains information concerning reported progress against performance measures. One of the Data Entities in the "Actuals" Information Group that collectively represents the information related to
CSREES -Cooperative State Research, Education and Extension Service (USDA agency)
DUNS Number -Data Universal Numbering System Number, a unique nine-digit number provided by Dunn & Bradstreet to business entities.
EGGM -USDA's eGrants Grants Management Committee
eGrants -U.S. Federal government-wide program to apply the federal guidance on electronic government to the processes related to Federal Financial Assistance (i.e., grants).
EMP -Emerging Markets Program, CFDA # 10.603
EMP Proposal -Data Entity that contains details of the proposed EMP project. One of the Data Entities in the "Practice" Information Group.
Evaluation -Information Group that collectively represents the information related to review and evaluation of applications for MAP, FMD, EMP, QSP, TASC, or Cochran funding.
Event -Data Entity that contains detailed information, such as date and location, about a specific instance of a Practice. One of the Data Entities in the "Activity" Information Group. Expenditures Funds that have been spent by a program participant to accomplish a specific Event or part of a larger Practice (both facets of a UES Activity).
FAQs -Frequently Asked Questions
FAS -Foreign Agricultural Service (USDA agency)
FASTNET -FAS intranet
Final Reports -Data Entity that contains information comprising EMP Final Reports. One of the Data Entities in the "Actuals" Information Group.
FMD -Foreign Market Development Program (AKA the Cooperator Program), CFDA# 10.600
FNS -Food and Nutrition Service (USDA agency)
FS -Forest Service (USDA agency)
FSA -Farm Service Agency (USDA agency)
FSN -Foreign Service Nationals (overseas post personnel)
FSO -Foreign Service Officers (overseas post personnel)
Funding Stream –A reference to the detailed funding information, such as program, program year, funding for a Practice. One of the entities in the "Practice" Information Group.
GATEWAYS -Global Agricultural Trade Enterprise-Wide Access Systems
GBI -Global Broad-based Initiative

GBI Factor -Items designated by FAS as GBI "Core Services," sometimes referred to as Priorities. Included in the Activity Definition, one of the entities in the "Activity" Information Group that collectively represents the information related to a UES Activity.
GBI Proposal -Global Broad-based Initiative Proposal. UES Participants may submit GBI Proposals. The Activities associated with these are funded from a unique pot of money.
GIM -Grants Interface Module
Grants -U.S. government grants are legal agreements providing funding categorized by the Catalog of Federal Domestic Assistance (CFDA).
Group -A trade organization or Participant. See "Sub-group" and "Parent Group".
GUI -Graphical user interface
Harmonized Codes -The Harmonized Code System classifies transactions under the categories of approximately 8,000 different products leaving the United States. Every item is assigned a unique 10-digit code and then is placed into broader categories of 6 and 4 digit codes. This code is part of a system in which worldwide tariffs are levied by harmonized code.
Help -Data Entity that contains information whose purpose is to provide assistance to users in operating the system. One of the Data Entities in the "Users" Information Group.
Industry -Data Entity that contains identifying and descriptive information concerning industries participating in FAS programs. One of the Data Entities in the "Industry" Information Group. Information Group that collectively represents the information related to the industry participants in the UES programs.
Industry Goals -Data Entity that contains information defining an industry's export and world trade value goals and status. One of the Data entities in the "Industry" Information Group.
ISI -Industry Specific Initiatives
ISO -Industry Strategic Overview, summarizing an industry's strategic approach to export marketing.
IT Staff -FAS IT staff personnel responsible for direct support of the UES system components, as well as managers responsible for the FAS Washington, DC, and oversees hardware, software, and telecommunications infrastructure and security.
JAD-Joint Application Design, meetings with user work groups and application design team to allow collaborative, iterative definition of user interface and system functionality.
Joint Activity -A specific activity whose costs and benefits are shared with another commodity or program participant.
Key Issues -Data Entity that contains information describing issues raised in the Industry Strategic overview report. One of the Data Entities in the "Industry" Information Group.
MAP -Market Access Program, CFDA # 10.601
Market Information -Group that collectively represents the information related to commodity export markets.
Market Definition -Data Entity that contains basic descriptive information concerning a specific market. One of the Data Entities in the "Market" Information Group.
Market Type -Categorizes a targeted market as "new," "growth," or "mature."
MDIS -Market Development Information System
Modifications -Data Entity that contains information describing changes to strategic or tactical plans (known as notifications) or agreements (known as amendments). One of the Data Entities in the "Agreement" Information Group.
MOS -Marketing Operations Staff, replaced after the FAS reorganization with PPS (Program Policy Staff)
NIST -National Institute of Standards and Technology
Non-CMP sources -Data sources "owned" by FAS organizational entities, other than OTP (CMP is the former acronym for OTP), potentially searchable using the upgraded UES reporting capability.
Non-FAS sources -Data sources "owned" by other federal government or other entities, potentially searchable using the upgraded UES reporting capability.

Notification -A change to an Export Strategy in which the Program Participant notifies FAS and which may or may not require FAS approval depending upon the change.
Obligations -Funding that has been identified in a signed agreement.
Offices -Data Entity that contains participant administrative information such as local office names and addresses. One of the Data Entities in the “Industry” Information Group.
Overseas Posts -Data Entity that contains basic administrative information concerning FAS overseas posts, such as post name, telephone, or url/e-mail address. One of the Data Entities in the “Market” Information Group.
OTP –FAS Office of Trade Programs, replaced CMP after FAS reorganization.
Parent Group -The "owner" organization of one or more Sub-Groups.
PART -Program Assessment Rating Tool
Participant -An organization or entity receiving federal funding via MAP, EMP, QSP, TASC, or Cochran. Also referred to as “program participant.”
Participant-Branded Company Data -Data Entity that contains information, entered by participant users, representing their aggregated branded company sales data which is needed by the system to make some PART report calculations.
Performance Measure -A qualitative or quantitative performance indicator against which success in addressing a condition (constraint or opportunity) can be measured.
Performance Measures Definition -Data Entity that contains basic descriptive information concerning a type of performance measure. One of the Data Entities in the “Condition” Information Group.
Post -Common name for FAS overseas locations, including Agricultural Trade Offices and Offices of Agricultural Affairs.
Post Notifications -Data Entity that contains information concerning participants’ plans for an overseas visit. One of the Data Entities in the “Actuals” Information Group.
Information Group that collectively represents the information related to a UES Activity.
PPS – Program Policy Staff
Programs -Data Entity that contains basic descriptive information specifying the FAS funding source programs and specifics such as application deadline(s). One of the Data Entities in the “Activity” Information Group.
Project -A term used in EMP signifying the activity prescribed by an approved Proposal.
In the system, an EMP "Proposal" becomes a "Project" upon approval. Proposal A mechanism in the UES system by some EMP, QSP, and TASC program participants. Proposals are used by organizations that do may not submit an application (UES) for other program funding. The Proposal contains a prescribed subset of UES plan information and is associated with one or more Activities.
QSP -Quality Samples Program, CFDA # 10.605
QSP Evaluation -Data Entity that contains information concerning approval criteria and FAS evaluation results for QSP proposals. One of the Data Entities in the “Evaluation” Information Group.
QSP Proposal -Data Entity that contains the basic, financial, and sample-specific information for a QSP proposal. One of the Data Entities in the “Practice” Information Group.
RD -Rural Development
Reimbursement -Payment of funds to program participants, reimbursing some of the expenses incurred in performing a specific practice (activities) previously approved by FAS.
Requested -Amount of funding proposed by a program participant in a program application.
Research Reports -Data Entity that contains 9 potentially searchable) basic descriptive and substantive information concerning reports generated by program participants during the course of execution of a project. One of the Data Entities in the “Actuals” Information Group.
Roles -Data Entity that contains information describing possible user roles and specifying the functions available to each. One of the Data Entities in the “Users” Information Group.

ROM Assessment Score Sheet -Set of questions and evaluation factors used by commodity divisions to evaluate each participant on their Results Oriented Management (ROM) approach and performance; part of the overall evaluation of the annual UES application for program funding.
Roles -Specific "category" of system user, defined by the types of system data to which the user has access and the specific functions for which the user can be authorized. Roles for the upgraded UES include: Program Participant, CMP, MOS, Posts, FAS, and System Administration.
RWG -Requirements Work Group, representatives of UES Stakeholder Groups, formed to review the proposed requirements for the UES Upgrade. Stakeholder Individuals who have used components of the legacy UES system, may use the upgraded UES system, or have an interest in the system's use, data, or capabilities. These include people in the following groups: Cooperator/Program Participant, MOS (PPS), C&MP (OTP) Staff, FSOs and FSNs, FAS Senior Management, Other FAS Staff, and IT Staff.
Strategic Priorities -Data Entity that contains basic descriptive information concerning industry strategic objectives for export marketing.
One of the Data Entities in the "Industry" Information Group.
Sub-group -An organization to which a participant can direct a payment such as a claim reimbursement. Bank information for Sub-groups and their Parent Group association is currently maintained in the system.
Targeted Market -Data Entity containing information describing the unique combination of a Market and Commodity that is the essence of a UES market (i.e. Kiwi fruit in Canada). One of the Data Entities in the "Market" Information Group.
TASC -Technical Assistance for Specialty Crops Program, CFDA # 10.604
TASC Proposal -Data Entity that contains detailed information specific to the TASC program, describing a TASC proposal. One of the Data Entities in the "Practice" Information Group.
Trip Reports -Data Entity that contains detailed information concerning the goal(s), outcome(s), contact(s), and follow up to any travel made in support of a Practice. One of the Data Entities in the "Actuals" Information Group.
UAT -User Acceptance Testing
UES -Unified Export Strategy
UES Assessment -Set of questions and evaluation factors used by commodity divisions to evaluate each participant on their UES application, focusing on their marketing plans; part of the overall evaluation of the annual UES application for program funding.
UES Attachments -Data Entity that contains basic information concerning attached files. One of the Data Entities in the "Agreement" Information Group.
UES Score Sheet -Data Entity that contains detailed information concerning the UES Assessment, ROM Assessment, and overall assessment of each participant's UES application for program funding. One of the Data Entities in the "Evaluation" Information Group.
UES-X -Term used in the *UES Upgrade Stakeholders Needs Assessment Report* to designate the "To Be" or future, upgraded UES system.
UMI -Unified Marketing Initiative
USAEDC -U.S. Agricultural Export Development Council, umbrella organization representing the cooperator/program participant agricultural commodity trade organizations.
USAM -A system which contains the 2 basic accounting components used by FAS, including the "document based information" system used primarily by CMP and the "Accounting Subsystem" used primarily by MOS.
USDA -U.S. Department of Agriculture Information Group that collectively represents the information related to system users.
Users -Data Entity that contains basic identifying and descriptive information concerning system users, such as name, telephone, organization, assigned roles, and authorized functionality. One of the Data Entities in the "Users" Information Group.

APPENDIX B: Key Legacy tables

The legacy database holds tables that have historic data, tables that support the accounting systems, tables that support reporting, and tables that support saving the elements of the plan. Legacy tables are in Sybase 12.5 with stored procedures coded in Transact-SQL. The target upgraded database is SQL Server 2005 where the stored procedure language is Transact-SQL. Scripts to create key tables currently used by the legacy system are listed below. The table scripts are followed by scripts for the user defined types and rules that are referenced in the tables. Legacy stored procedure objects are numerous, but not listed here.

```
/*
  Table(s)
*/

PRINT 'TABLE : active_participant'
GO


create table active_participant (
participant_id typ_partid not null,
active_sw typ_swy not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_y_n_sw, 'active_participant.active_sw'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_swy, 'active_participant.active_sw'

PRINT 'TABLE : advance_payback_exception'
GO


create table advance_payback_exception (
claim_doc_yr int not null,
claim_doc_nbr int not null)
GO


PRINT 'TABLE : advance_requests'
GO


create table advance_requests (
doc_yr typ_docyr not null,
doc_nbr smallint not null,
plan_doc_yr typ_docyr not null,
plan_doc_nbr smallint not null,
budget_yr char(4) not null,
status char(1) not null,
received_date smalldatetime not null,
amount money not null,
line_comments typ_description null,
recovered_date smalldatetime null,
approved_date smalldatetime null,
to_voucher_date smalldatetime null,
advance_nbr char(5) null)
GO
```

```
PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_doc_yr, 'advance_requests.doc_yr'
execute sp_bindrule rul_doc_yr, 'advance_requests.plan_doc_yr'

GO

PRINT 'TABLE : agreement'
GO


create table agreement (
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
agreement_nbr char(10) null,
participant_signee_name char(25) null,
participant_signee_title char(25) null,
participant_signee_date smalldatetime null,
ccc_signee_name char(25) null,
ccc_signee_title char(25) null,
ccc_signee_date smalldatetime null,
execution_date smalldatetime null,
expiration_date smalldatetime null,
signature_card_date smalldatetime null,
signature_card_signee char(25) null,
certifications_received_sw typ_swn not null,
currently_suspended_sw typ_swn not null,
timestamp timestamp null,
commodity_description char(40) null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_doc_yr, 'agreement.agreement_doc_yr'
execute sp_bindrule rul_y_n_sw, 'agreement.certifications_received_sw'
execute sp_bindrule rul_y_n_sw, 'agreement.currently_suspended_sw'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_swn, 'agreement.certifications_received_sw'
execute sp_bindefault def_swn, 'agreement.currently_suspended_sw'


PRINT 'TABLE : alloc_bal_calculated'
GO


create table alloc_bal_calculated (
program_yr int not null,
part_id varchar(50) not null,
agreement_nbr varchar(7) null,
exp_date datetime null,
allocation money null,
expenses money null,
cum_alloc money null,
cum_expense money null,
balance money null,
agreement_status char(7) null,
date_of_calculation datetime not null,
program varchar(3) not null,
expired_funds money null,
cum_expired_funds money null,
yearly_bal money null)
GO


PRINT 'TABLE : alloc_bal_calculated_by_date'
GO
```

```
create table alloc_bal_calculated_by_date (
program_yr int not null,
part_id varchar(50) not null,
agreement_nbr varchar(7) null,
exp_date datetime null,
allocation money null,
expenses money null,
cum_alloc money null,
cum_expense money null,
balance money null,
agreement_status char(7) null,
date_of_calculation datetime not null,
program varchar(3) not null,
expired_funds money null,
cum_expired_funds money null,
yearly_bal money null,
simple_balance money null)
GO


PRINT 'TABLE : ApplicationCertification'
GO


create table ApplicationCertification (
application_doc_yr typ_docyr not null,
application_doc_nbr smallint not null,
CertificationSigned bit not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_doc_yr, 'ApplicationCertification.application_doc_yr'

GO

PRINT 'TABLE : branded_company_graduations'
GO


create table branded_company_graduations (
company_id smallint not null,
company_name char(58) not null,
size char(1) not null,
program_type char(3) not null,
participant_id char(8) not null,
program_yr int not null,
market_code char(2) not null,
country_name char(39) not null,
budget money not null,
expense money not null,
exmpt_expense money not null)
GO


PRINT 'TABLE : ccc_rpt_by_date'
GO


create table ccc_rpt_by_date (
user_nbr int not null,
part_id varchar(8) not null,
participant_name varchar(50) null,
end_date datetime null,
year int not null,
alloc int null,
expenses int null,
bal int null,
total_bal int null,
program varchar(3) not null)
GO
```

```
PRINT 'TABLE : ccc_rpt_correct_alloc'
GO


create table ccc_rpt_correct_alloc (
program_yr typ_progyr not null,
part_id typ_partid not null,
alloc money null,
program varchar(3) not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ccc_rpt_correct_alloc.program_yr'

GO

PRINT 'TABLE : commodity_aggregate'
GO


create table commodity_aggregate (
aggregate_comm_code typ_agg_comm not null,
aggregate_comm_desc char(40) not null,
parent_aggregate_comm_code typ_agg_comm null,
aggregate_level char(1) not null,
end_node_sw typ_swy not null,
processing_degree char(4) not null,
valid_control_sw typ_swy not null,
lats_sw typ_swy not null,
fmd_eligible_flag typ_swy not null,
mpp_eligible_flag typ_swy not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_y_n_sw, 'commodity_aggregate.end_node_sw'
execute sp_bindrule rul_y_n_sw, 'commodity_aggregate.valid_control_sw'
execute sp_bindrule rul_y_n_sw, 'commodity_aggregate.lats_sw'
execute sp_bindrule rul_y_n_sw, 'commodity_aggregate.fmd_eligible_flag'
execute sp_bindrule rul_y_n_sw, 'commodity_aggregate.mpp_eligible_flag'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'commodity_aggregate.aggregate_comm_code'
execute sp_bindefault def_agg_comm, 'commodity_aggregate.parent_aggregate_comm_code'
execute sp_bindefault def_swy, 'commodity_aggregate.end_node_sw'
execute sp_bindefault def_swy, 'commodity_aggregate.valid_control_sw'
execute sp_bindefault def_swy, 'commodity_aggregate.lats_sw'
execute sp_bindefault def_swy, 'commodity_aggregate.fmd_eligible_flag'
execute sp_bindefault def_swy, 'commodity_aggregate.mpp_eligible_flag'

PRINT 'TABLE : commodity_ineligibility_period'
GO


create table commodity_ineligibility_period (
harmonized_code typ_harmcommcd not null,
program char(3) not null,
ineligible_start_date smalldatetime not null,
ineligible_end_date smalldatetime null,
reason_for_ineligibility char(240) not null)
GO


PRINT 'BINDING Rule(s) to column'
```

```
GO

execute sp_bindrule rul_harmcommcd, 'commodity_ineligibility_period.harmonized_code'

GO

PRINT 'TABLE : contact'
GO


create table contact (
contact_id smallint not null,
last_name char(25) null,
first_name char(20) null,
salutation char(10) null,
job_title char(30) null,
participant_id typ_partid null,
organization char(50) null,
address_1 typ_addrlin null,
address_2 typ_addrlin null,
address_3 typ_addrlin null,
address_4 typ_addrlin null,
address_5 typ_addrlin null,
city typ_city null,
state_code char(2) null,
zip_1 typ_zip1 null,
zip_2 typ_zip2 null,
phone typ_phonenbr null,
fax typ_phonenbr null,
email_id char(20) null,
responsible_division typ_orgcd not null,
comment char(240) null,
contact_user_nbr typ_usernbr null)
GO


PRINT 'TABLE : contact_email'
GO


create table contact_email (
contact_id int not null,
email varchar(255) null)
GO


PRINT 'TABLE : cost_categories'
GO


create table cost_categories (
cost_category_code typ_cost_category not null,
cost_category_desc typ_description not null,
valid_for_non_admin_sw typ_swy not null,
valid_for_admin_sw typ_swn not null,
program_yr typ_progyr not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_y_n_sw, 'cost_categories.valid_for_non_admin_sw'
execute sp_bindrule rul_y_n_sw, 'cost_categories.valid_for_admin_sw'
execute sp_bindrule rul_program_yr, 'cost_categories.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_swy, 'cost_categories.valid_for_non_admin_sw'
execute sp_bindefault def_swn, 'cost_categories.valid_for_admin_sw'

PRINT 'TABLE : country'
```

```
GO


create table country (
fips_code typ_fipcd not null,
country_name char(30) not null,
fmd_eligible_flag typ_swy not null,
mpp_eligible_flag typ_swy not null,
stage_of_development char(14) not null,
reporting_post_id char(10) null,
fas_region_code char(6) null,
mdis_country_code char(2) null,
status_flag char(1) not null,
us_census_nbr smallint null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_y_n_sw, 'country.fmd_eligible_flag'
execute sp_bindrule rul_y_n_sw, 'country.mpp_eligible_flag'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_swy, 'country.fmd_eligible_flag'
execute sp_bindefault def_swy, 'country.mpp_eligible_flag'

PRINT 'TABLE : country_ineligibility_period'
GO


create table country_ineligibility_period (
fips_code typ_fipcd not null,
program char(3) not null,
ineligible_start_date smalldatetime not null,
ineligible_end_date smalldatetime null,
reason_for_ineligibility char(240) not null)
GO


PRINT 'TABLE : UES_CostCategory_Programs'
GO


create table UES_CostCategory_Programs (
cost_category_code char(8) not null,
program_id smallint not null)
GO



use master
GO

/*
  Script for Server sybase07 (Adaptive Server Enterprise/12.5.3/EBF 13330 ESD#7/P/Compaq
Tru64/OSF1 V5.0A/ase1253/1951/64-bit/FBO/Fri Mar 24 07:24:51 2006) on axposf
*/

/*  Database 'usamprod'  */
use usamprod
GO


/*
  Table(s)
*/

PRINT 'TABLE : disbur_sched_counter'
GO
```

```
create table disbur_sched_counter (
latest_disbur_sched_nbr int not null)
GO


PRINT 'TABLE : division'
GO


create table division (
division_code typ_orgcd not null,
division_title char(50) not null,
division_director_user_nbr typ_usernbr null,
commodity_division_sw typ_swn not null,
dept_dir_for_mkt_user_nbr typ_usernbr null,
fas_program_area char(8) not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_y_n_sw, 'division.commodity_division_sw'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_swn, 'division.commodity_division_sw'

PRINT 'TABLE : document'
GO


create table document (
doc_yr typ_docyr not null,
doc_nbr smallint not null,
doc_type_code char(4) null,
program char(3) null,
program_yr typ_progyr not null,
agreement_doc_yr typ_docyr null,
agreement_doc_nbr smallint null,
responsible_division typ_orgcd not null,
participant_id typ_partid null,
eip_sw typ_swn not null,
create_date smalldatetime not null,
doc_date smalldatetime null,
date_fas_received_doc smalldatetime null,
date_fas_sent_doc smalldatetime null,
last_modification_date smalldatetime not null,
last_user_nbr_to_modify typ_usernbr not null,
archived_date smalldatetime null,
originator_user_nbr typ_usernbr null,
originator_organization typ_orgcd not null,
destination_user_nbr typ_usernbr null,
destination_organization typ_orgcd null,
doc_locked_by_user_nbr typ_usernbr null,
doc_status char(1) not null,
timestamp timestamp null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_doc_yr, 'document.doc_yr'
execute sp_bindrule rul_program_yr, 'document.program_yr'
execute sp_bindrule rul_doc_yr, 'document.agreement_doc_yr'
execute sp_bindrule rul_y_n_sw, 'document.eip_sw'

GO

PRINT 'BINDING Default(s) to column'
```

```
GO

execute sp_bindefault def_swn, 'document.eip_sw'

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'document', doc_yr, doc_nbr

GO


PRINT 'TABLE : document_counter'
GO


create table document_counter (
latest_doc_no smallint not null)
GO


PRINT 'TABLE : document_type'
GO


create table document_type (
doc_type_code char(4) not null,
doc_type char(30) not null,
originator_organization_type typ_orgtyp not null,
mpp_eligible_flag typ_swy not null,
tea_eligible_flag typ_swy not null,
fmd_eligible_flag typ_swy not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_y_n_sw, 'document_type.mpp_eligible_flag'
execute sp_bindrule rul_y_n_sw, 'document_type.tea_eligible_flag'
execute sp_bindrule rul_y_n_sw, 'document_type.fmd_eligible_flag'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_swy, 'document_type.mpp_eligible_flag'
execute sp_bindefault def_swy, 'document_type.tea_eligible_flag'
execute sp_bindefault def_swy, 'document_type.fmd_eligible_flag'

PRINT 'TABLE : dsc_schedule_nbr'
GO


create table dsc_schedule_nbr (
program varchar(3) not null,
dsc_number char(10) not null,
participant_id char(10) not null,
claim_nbr char(5) not null,
advance char(1) null)
GO


PRINT 'TABLE : emp_activity_budget_yr_rec'
GO


create table emp_activity_budget_yr_rec (
program_yr typ_progyr not null,
part_id typ_partid not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_code char(5) not null,
```

```
market_id typ_mktcd not null,
year int not null,
cost_category varchar(8) not null,
description varchar(255) null,
amount int not null,
part_contrib int null,
third_party_contrib int null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule typ_progyr, 'emp_activity_budget_yr_rec.program_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'emp_activity_budget_yr_rec', program_yr, part_id, activity_program,
activity_progyr, activity_type, activity_code, market_id, year

GO


PRINT 'TABLE : emp_activity_budgets'
GO


create table emp_activity_budgets (
transaction_nbr typ_transaction_nbr not null,
transaction_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
debit_credit_flag typ_debit_credit_sw not null,
amount money not null,
user_nbr typ_usernbr not null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
organization_id typ_orgid not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_modifier char(1) not null,
activity_code char(5) not null,
activity_title char(40) not null,
market_code char(16) not null,
region_flag char(1) not null,
com_agg_code char(5) not null,
activity_market_code char(16) not null,
control_level char(1) not null,
deleted_flag char(1) not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'emp_activity_budgets.program_yr'
execute sp_bindrule rul_debit_credit_sw, 'emp_activity_budgets.debit_credit_flag'
execute sp_bindrule rul_doc_yr, 'emp_activity_budgets.agreement_doc_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'emp_activity_budgets', transaction_nbr, transaction_segment

GO


PRINT 'TABLE : emp_activity_ceilings'
```

```
GO


create table emp_activity_ceilings (
transaction_nbr typ_transaction_nbr not null,
transaction_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
debit_credit_flag typ_debit_credit_sw not null,
amount money not null,
user_nbr typ_usernbr not null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
organization_id typ_orgid not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_modifier char(1) not null,
activity_code char(5) not null,
activity_title char(40) not null,
market_code char(16) not null,
region_flag char(1) not null,
com_agg_code char(5) not null,
activity_market_code char(16) not null,
control_level char(1) not null,
deleted_flag char(1) not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'emp_activity_ceilings.program_yr'
execute sp_bindrule rul_debit_credit_sw, 'emp_activity_ceilings.debit_credit_flag'
execute sp_bindrule rul_doc_yr, 'emp_activity_ceilings.agreement_doc_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'emp_activity_ceilings', transaction_nbr, transaction_segment

GO


PRINT 'TABLE : emp_activity_expiration'
GO


create table emp_activity_expiration (
program_yr typ_progyr not null,
part_id typ_partid not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_code char(5) not null,
market_id typ_mktcd not null,
com_agg_code typ_agg_comm not null,
expiration_date datetime not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'emp_activity_expiration.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'emp_activity_expiration.com_agg_code'
```

```
PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'emp_activity_expiration', program_yr, part_id, activity_program,
activity_progyr, activity_type, activity_code, market_id

GO


PRINT 'TABLE : emp_activity_phases'
GO


create table emp_activity_phases (
program_yr typ_progyr not null,
part_id typ_partid not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_code char(5) not null,
market_id typ_mktcd not null,
com_agg_code typ_agg_comm not null,
phase_id int not null,
phase_desc varchar(200) not null,
phase_start datetime not null,
phase_end datetime not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'emp_activity_phases.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'emp_activity_phases.com_agg_code'

PRINT 'TABLE : emp_anncmt_payable'
GO


create table emp_anncmt_payable (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr not null,
appropriation_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'emp_anncmt_payable.program_yr'
execute sp_bindrule rul_approp_yr, 'emp_anncmt_payable.appropriation_yr'
execute sp_bindrule rul_debit_credit_sw, 'emp_anncmt_payable.debit_credit_flag'

GO

PRINT 'TABLE : emp_disbur_sched_counter'
GO


create table emp_disbur_sched_counter (
latest_disbur_sched_nbr int not null)
```

```
GO


PRINT 'TABLE : emp_funds'
GO


create table emp_funds (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr null,
appropriation_nbr smallint null,
debit_credit_flag typ_debit_credit_sw not null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'emp_funds.program_yr'
execute sp_bindrule rul_approp_yr, 'emp_funds.appropriation_yr'
execute sp_bindrule rul_debit_credit_sw, 'emp_funds.debit_credit_flag'

GO

PRINT 'TABLE : emp_general_journal'
GO


create table emp_general_journal (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_type typ_progtype not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr null,
appropriation_nbr smallint null,
agreement_doc_yr typ_docyr null,
agreement_doc_nbr smallint null,
organization_id typ_orgid null,
activity_code typ_activity_code null,
cost_category_code typ_cost_category null,
amount money not null,
user_nbr typ_usernbr not null,
initiating_system char(1) not null,
ceiling_type typ_ceiling_type null,
market_code typ_mktcd null,
fips_code typ_fipcd null,
aggregate_comm_code typ_agg_comm null,
branded_generic_flag typ_brand_gener_flag null,
exp_claim_number char(8) null,
exp_line_item_id smallint null,
relevant_doc_type char(15) null,
relevant_doc_yr typ_docyr null,
relevant_doc_nbr smallint null,
rev_transaction_nbr typ_transaction_nbr null,
rev_transaction_nbr_segment smallint null,
comment_sw typ_swn null,
activity_market_code typ_mktcd null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_progtype, 'emp_general_journal.program_type'
execute sp_bindrule rul_program_yr, 'emp_general_journal.program_yr'
execute sp_bindrule rul_approp_yr, 'emp_general_journal.appropriation_yr'
execute sp_bindrule rul_doc_yr, 'emp_general_journal.agreement_doc_yr'
```

```
execute sp_bindrule rul_ceiling_type, 'emp_general_journal.ceiling_type'
execute sp_bindrule rul_brand_gener_flag, 'emp_general_journal.branded_generic_flag'
execute sp_bindrule rul_doc_yr, 'emp_general_journal.relevant_doc_yr'
execute sp_bindrule rul_y_n_sw, 'emp_general_journal.comment_sw'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'emp_general_journal.aggregate_comm_code'
execute sp_bindefault def_swn, 'emp_general_journal.comment_sw'

PRINT 'TABLE : emp_prog_ceilings'
GO


create table emp_prog_ceilings (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
agreement_doc_yr typ_docyr null,
agreement_doc_nbr smallint null,
debit_credit_flag typ_debit_credit_sw not null,
organization_id typ_orgid null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'emp_prog_ceilings.program_yr'
execute sp_bindrule rul_doc_yr, 'emp_prog_ceilings.agreement_doc_yr'
execute sp_bindrule rul_debit_credit_sw, 'emp_prog_ceilings.debit_credit_flag'

GO

PRINT 'TABLE : emp_program_announcement'
GO


create table emp_program_announcement (
doc_yr typ_docyr not null,
doc_nbr smallint not null,
fed_register_annc_yr smallint null,
fed_register_annc_nbr int null,
application_deadline smalldatetime not null,
total_funding_available money not null,
appropriation_yr typ_approp_yr not null,
appropriation_nbr smallint not null,
tstamp timestamp null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_doc_yr, 'emp_program_announcement.doc_yr'
execute sp_bindrule rul_approp_yr, 'emp_program_announcement.appropriation_yr'

GO

PRINT 'TABLE : emp_proposal'
GO


create table emp_proposal (
part_id typ_partid not null,
program_yr typ_progyr not null,
market_id typ_mktcd not null,
com_agg_code typ_agg_comm not null,
```

```
title varchar(255) not null,
date datetime null,
submitted char(1) not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'emp_proposal.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'emp_proposal.com_agg_code'

PRINT 'TABLE : emp_proposal_text'
GO


create table emp_proposal_text (
part_id typ_partid not null,
program_yr typ_progyr not null,
market_id typ_mktcd not null,
com_agg_code typ_agg_comm not null,
title varchar(255) not null,
proposal_text_id int not null,
proposal_text text not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'emp_proposal_text.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'emp_proposal_text.com_agg_code'

PRINT 'TABLE : emp_proposal_text_type'
GO


create table emp_proposal_text_type (
proposal_text_id int not null,
proposal_text_desc varchar(255) not null)
GO


PRINT 'TABLE : emp_transaction_comments'
GO


create table emp_transaction_comments (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
comment char(255) not null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'emp_transaction_comments', transaction_nbr, transaction_nbr_segment

GO


PRINT 'TABLE : emp_transaction_defns'
```

```
GO


create table emp_transaction_defns (
transaction_code typ_trancd not null,
credit_table char(30) not null,
debit_table char(30) null,
trans_description char(40) not null,
user_nbr typ_usernbr not null,
definition_datetime datetime not null)
GO


PRINT 'TABLE : emp_transaction_type'
GO


create table emp_transaction_type (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_type char(1) not null)
GO


PRINT 'TABLE : expense_claim_line_items'
GO


create table expense_claim_line_items (
claim_doc_yr typ_docyr not null,
claim_doc_nbr smallint not null,
line_item_number smallint not null,
activity_code typ_activity_code not null,
ceiling_market_code typ_mktcd not null,
cost_category_code typ_cost_category not null,
amount money not null,
line_status typ_line_status null,
line_comments typ_description null,
timestamp timestamp null,
activity_market_code typ_mktcd null,
activity_program char(1) null,
activity_progyr char(2) null,
activity_type char(1) null,
activity_modifier char(1) null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_doc_yr, 'expense_claim_line_items.claim_doc_yr'
execute sp_bindrule rul_line_status, 'expense_claim_line_items.line_status'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'expense_claim_line_items', claim_doc_yr, claim_doc_nbr, line_item_number

GO


PRINT 'TABLE : expense_claim_line_items_mkt'
GO


create table expense_claim_line_items_mkt (
claim_doc_yr typ_docyr not null,
claim_doc_nbr smallint not null,
line_item_number smallint not null,
market_country typ_fipcd not null)
GO
```

```
PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_doc_yr, 'expense_claim_line_items_mkt.claim_doc_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'expense_claim_line_items_mkt', claim_doc_yr, claim_doc_nbr,
line_item_number

GO


PRINT 'TABLE : expense_claim_SAPR_offset'
GO


create table expense_claim_SAPR_offset (
claim_doc_yr typ_docyr not null,
claim_doc_nbr smallint not null,
line_item_number smallint not null,
offset char(1) not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_doc_yr, 'expense_claim_SAPR_offset.claim_doc_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'expense_claim_SAPR_offset', claim_doc_yr, claim_doc_nbr, line_item_number

GO


PRINT 'TABLE : expense_claims'
GO


create table expense_claims (
claim_doc_yr typ_docyr not null,
claim_doc_nbr smallint not null,
plan_doc_yr typ_docyr not null,
plan_doc_nbr smallint not null,
budget_yr char(4) not null,
claim_nbr char(8) null,
claim_status typ_claim_status not null,
received_date datetime not null,
approved_date datetime null,
claim_comments typ_description null,
payment_number smallint null,
ccc_date datetime null,
paid_date datetime null,
to_voucher_date smalldatetime null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_doc_yr, 'expense_claims.claim_doc_yr'
execute sp_bindrule rul_doc_yr, 'expense_claims.plan_doc_yr'
execute sp_bindrule rul_claim_status, 'expense_claims.claim_status'

GO

PRINT 'CREATING Primary Key'
```

```
GO

execute sp_primarykey 'expense_claims', claim_doc_yr, claim_doc_nbr

GO


PRINT 'TABLE : expense_claims_vxp'
GO


create table expense_claims_vxp (
doc_yr typ_docyr not null,
doc_nbr smallint not null,
subgrp_id typ_partid not null,
vxp char(9) not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_doc_yr, 'expense_claims_vxp.doc_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'expense_claims_vxp', doc_yr, doc_nbr, subgrp_id

GO



use master
GO

/*
  Script for Server sybase07 (Adaptive Server Enterprise/12.5.3/EBF 13330 ESD#7/P/Compaq
Tru64/OSF1 V5.0A/ase1253/1951/64-bit/FBO/Fri Mar 24 07:24:51 2006) on axposf
*/

/*  Database 'usamprod'  */
use usamprod
GO


/*
  Table(s)
*/

PRINT 'TABLE : fmd_activity_budgets'
GO


create table fmd_activity_budgets (
transaction_nbr typ_transaction_nbr not null,
transaction_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
debit_credit_flag typ_debit_credit_sw not null,
amount money not null,
user_nbr typ_usernbr not null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
organization_id typ_orgid not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_modifier char(1) not null,
activity_code char(5) not null,
activity_title char(40) not null,
market_code char(16) not null,
```

```
region_flag char(1) not null,
com_agg_code char(5) not null,
activity_market_code char(16) not null,
control_level char(1) not null,
deleted_flag char(1) not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'fmd_activity_budgets.program_yr'
execute sp_bindrule rul_debit_credit_sw, 'fmd_activity_budgets.debit_credit_flag'
execute sp_bindrule rul_doc_yr, 'fmd_activity_budgets.agreement_doc_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'fmd_activity_budgets', transaction_nbr, transaction_segment

GO


PRINT 'TABLE : fmd_agreement'
GO


create table fmd_agreement (
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
participant_id char(8) not null,
agreement_nbr int null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_doc_yr, 'fmd_agreement.agreement_doc_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'fmd_agreement', agreement_doc_yr, agreement_doc_nbr

GO


PRINT 'TABLE : fmd_anncmt_payable'
GO


create table fmd_anncmt_payable (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr not null,
appropriation_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'fmd_anncmt_payable.program_yr'
execute sp_bindrule rul_approp_yr, 'fmd_anncmt_payable.appropriation_yr'
```

```
execute sp_bindrule rul_debit_credit_sw, 'fmd_anncmt_payable.debit_credit_flag'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'fmd_anncmt_payable', transaction_nbr, transaction_nbr_segment

GO


PRINT 'TABLE : fmd_base_bdgt'
GO


create table fmd_base_bdgt (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_code typ_trancd not null,
transaction_date datetime not null,
program_yr typ_progyr not null,
originating_yr typ_progyr null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
organization_id typ_orgid null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'fmd_base_bdgt.program_yr'
execute sp_bindrule rul_program_yr, 'fmd_base_bdgt.originating_yr'
execute sp_bindrule rul_doc_yr, 'fmd_base_bdgt.agreement_doc_yr'
execute sp_bindrule rul_debit_credit_sw, 'fmd_base_bdgt.debit_credit_flag'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'fmd_base_bdgt', transaction_nbr, transaction_nbr_segment

GO


PRINT 'TABLE : fmd_carry_over'
GO


create table fmd_carry_over (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
organization_id typ_orgid null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'fmd_carry_over.program_yr'
execute sp_bindrule rul_doc_yr, 'fmd_carry_over.agreement_doc_yr'
execute sp_bindrule rul_debit_credit_sw, 'fmd_carry_over.debit_credit_flag'
```

```
GO

PRINT 'TABLE : fmd_comments'
GO


create table fmd_comments (
doc_yr typ_docyr not null,
doc_nbr smallint not null,
comment_desc char(255) not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_doc_yr, 'fmd_comments.doc_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'fmd_comments', doc_yr, doc_nbr

GO


PRINT 'TABLE : fmd_contingent_liab_bdgt'
GO


create table fmd_contingent_liab_bdgt (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr null,
appropriation_nbr smallint null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
organization_id typ_orgid null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'fmd_contingent_liab_bdgt.program_yr'
execute sp_bindrule rul_approp_yr, 'fmd_contingent_liab_bdgt.appropriation_yr'
execute sp_bindrule rul_doc_yr, 'fmd_contingent_liab_bdgt.agreement_doc_yr'
execute sp_bindrule rul_debit_credit_sw, 'fmd_contingent_liab_bdgt.debit_credit_flag'

GO

PRINT 'TABLE : fmd_contingent_liabilities'
GO


create table fmd_contingent_liabilities (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr null,
appropriation_nbr smallint null,
agreement_doc_yr typ_docyr null,
agreement_doc_nbr smallint null,
debit_credit_flag typ_debit_credit_sw not null,
```

```
organization_id typ_orgid null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'fmd_contingent_liabilities.program_yr'
execute sp_bindrule rul_approp_yr, 'fmd_contingent_liabilities.appropriation_yr'
execute sp_bindrule rul_doc_yr, 'fmd_contingent_liabilities.agreement_doc_yr'
execute sp_bindrule rul_debit_credit_sw, 'fmd_contingent_liabilities.debit_credit_flag'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'fmd_contingent_liabilities', transaction_nbr, transaction_nbr_segment

GO


PRINT 'TABLE : fmd_cooperator_funds'
GO


create table fmd_cooperator_funds (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr null,
appropriation_nbr smallint null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
organization_id typ_orgid null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'fmd_cooperator_funds.program_yr'
execute sp_bindrule rul_approp_yr, 'fmd_cooperator_funds.appropriation_yr'
execute sp_bindrule rul_doc_yr, 'fmd_cooperator_funds.agreement_doc_yr'
execute sp_bindrule rul_debit_credit_sw, 'fmd_cooperator_funds.debit_credit_flag'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'fmd_cooperator_funds', transaction_nbr, transaction_nbr_segment

GO


PRINT 'TABLE : fmd_disbur_sched_counter'
GO


create table fmd_disbur_sched_counter (
latest_disbur_sched_nbr int not null)
GO


PRINT 'TABLE : fmd_funds'
GO
```

```
create table fmd_funds (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
originating_yr typ_progyr not null,
appropriation_yr typ_approp_yr not null,
appropriation_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'fmd_funds.program_yr'
execute sp_bindrule rul_program_yr, 'fmd_funds.originating_yr'
execute sp_bindrule rul_approp_yr, 'fmd_funds.appropriation_yr'
execute sp_bindrule rul_debit_credit_sw, 'fmd_funds.debit_credit_flag'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'fmd_funds', transaction_nbr, transaction_nbr_segment

GO


PRINT 'TABLE : fmd_general_journal'
GO


create table fmd_general_journal (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_type typ_progtype not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr null,
appropriation_nbr smallint null,
agreement_doc_yr typ_docyr null,
agreement_doc_nbr smallint null,
organization_id typ_orgid null,
activity_code typ_activity_code null,
cost_category_code typ_cost_category null,
amount money not null,
user_nbr typ_usernbr not null,
initiating_system char(1) not null,
ceiling_type typ_ceiling_type null,
market_code typ_mktcd null,
fips_code typ_fipcd null,
aggregate_comm_code typ_agg_comm null,
branded_generic_flag typ_brand_gener_flag null,
exp_claim_number char(8) null,
exp_line_item_id smallint null,
relevant_doc_type char(15) null,
relevant_doc_yr typ_docyr null,
relevant_doc_nbr smallint null,
rev_transaction_nbr typ_transaction_nbr null,
rev_transaction_nbr_segment smallint null,
comment_sw typ_swn null,
activity_market_code typ_mktcd null)
GO


PRINT 'BINDING Rule(s) to column'
GO
```

```
execute sp_bindrule rul_progtype, 'fmd_general_journal.program_type'
execute sp_bindrule rul_program_yr, 'fmd_general_journal.program_yr'
execute sp_bindrule rul_approp_yr, 'fmd_general_journal.appropriation_yr'
execute sp_bindrule rul_doc_yr, 'fmd_general_journal.agreement_doc_yr'
execute sp_bindrule rul_ceiling_type, 'fmd_general_journal.ceiling_type'
execute sp_bindrule rul_brand_gener_flag, 'fmd_general_journal.branded_generic_flag'
execute sp_bindrule rul_doc_yr, 'fmd_general_journal.relevant_doc_yr'
execute sp_bindrule rul_y_n_sw, 'fmd_general_journal.comment_sw'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'fmd_general_journal.aggregate_comm_code'
execute sp_bindefault def_swn, 'fmd_general_journal.comment_sw'

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'fmd_general_journal', transaction_nbr, transaction_nbr_segment

GO


PRINT 'TABLE : fmd_operating_advance'
GO


create table fmd_operating_advance (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr null,
appropriation_yr typ_approp_yr null,
appropriation_nbr smallint null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
organization_id typ_orgid null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'fmd_operating_advance.program_yr'
execute sp_bindrule rul_approp_yr, 'fmd_operating_advance.appropriation_yr'
execute sp_bindrule rul_doc_yr, 'fmd_operating_advance.agreement_doc_yr'
execute sp_bindrule rul_debit_credit_sw, 'fmd_operating_advance.debit_credit_flag'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'fmd_operating_advance', transaction_nbr, transaction_nbr_segment

GO


PRINT 'TABLE : fmd_program_announcement'
GO


create table fmd_program_announcement (
doc_yr typ_docyr not null,
doc_nbr smallint not null,
fed_register_annc_yr smallint null,
fed_register_annc_nbr int null,
application_deadline smalldatetime not null,
total_funding_available money not null,
```

```
appropriation_yr typ_approp_yr not null,
appropriation_nbr smallint not null,
tstamp timestamp null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_doc_yr, 'fmd_program_announcement.doc_yr'
execute sp_bindrule rul_approp_yr, 'fmd_program_announcement.appropriation_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'fmd_program_announcement', doc_yr, doc_nbr

GO


use master
GO

/*
  Script for Server sybase07 (Adaptive Server Enterprise/12.5.3/EBF 13330 ESD#7/P/Compaq
Tru64/OSF1 V5.0A/ase1253/1951/64-bit/FBO/Fri Mar 24 07:24:51 2006) on axposf
*/

/*  Database 'usamprod'  */
use usamprod
GO


/*
  Table(s)
*/

PRINT 'TABLE : fmd_transaction_comments'
GO


create table fmd_transaction_comments (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
comment char(255) not null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'fmd_transaction_comments', transaction_nbr, transaction_nbr_segment

GO


PRINT 'TABLE : fmd_transaction_defns'
GO


create table fmd_transaction_defns (
transaction_code typ_trancd not null,
credit_table char(30) not null,
debit_table char(30) null,
trans_description char(40) not null,
user_nbr typ_usernbr not null,
definition_datetime datetime not null)
GO


PRINT 'CREATING Primary Key'
GO
```

```
execute sp_primarykey 'fmd_transaction_defns', transaction_code

GO


PRINT 'TABLE : fmd_transaction_type'
GO


create table fmd_transaction_type (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_type char(1) not null)
GO



use master
GO

/*
  Script for Server sybase07 (Adaptive Server Enterprise/12.5.3/EBF 13330 ESD#7/P/Compaq
Tru64/OSF1 V5.0A/ase1253/1951/64-bit/FBO/Fri Mar 24 07:24:51 2006) on axposf
*/

/*  Database 'usamprod'  */
use usamprod
GO


/*
  Table(s)
*/

PRINT 'TABLE : harmonized_commodity'
GO


create table harmonized_commodity (
harmonized_code typ_harmcommcd not null,
commodity_desc char(40) null,
commodity_short_desc char(16) not null,
parent_aggregate_comm_code typ_agg_comm not null,
fmd_eligible_flag typ_swy not null,
mpp_eligible_flag typ_swy not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_harmcommcd, 'harmonized_commodity.harmonized_code'
execute sp_bindrule rul_y_n_sw, 'harmonized_commodity.fmd_eligible_flag'
execute sp_bindrule rul_y_n_sw, 'harmonized_commodity.mpp_eligible_flag'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'harmonized_commodity.parent_aggregate_comm_code'
execute sp_bindefault def_swy, 'harmonized_commodity.fmd_eligible_flag'
execute sp_bindefault def_swy, 'harmonized_commodity.mpp_eligible_flag'

PRINT 'TABLE : mdis_mpp_reg_history'
GO


create table mdis_mpp_reg_history (
participant_id typ_partid not null,
history_yr char(4) not null,
program char(3) not null,
mdis_part_id char(3) null,
mdis_budget money not null,
mdis_expense money not null,
```

```
mdis_contribution money not null,
mdis_allocation money not null)
GO


PRINT 'TABLE : mos_claim_rep'
GO


create table mos_claim_rep (
user_nbr smallint not null,
participant_id char(8) not null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'mos_claim_rep', participant_id

GO


PRINT 'TABLE : mpp_activity_budgets'
GO


create table mpp_activity_budgets (
transaction_nbr typ_transaction_nbr not null,
transaction_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
debit_credit_flag typ_debit_credit_sw not null,
amount money not null,
user_nbr typ_usernbr not null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
organization_id typ_orgid not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_modifier char(1) not null,
activity_code char(5) not null,
activity_title char(40) not null,
market_code char(16) not null,
region_flag char(1) not null,
com_agg_code char(5) not null,
activity_market_code char(16) not null,
control_level char(1) not null,
deleted_flag char(1) not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'mpp_activity_budgets.program_yr'
execute sp_bindrule rul_debit_credit_sw, 'mpp_activity_budgets.debit_credit_flag'
execute sp_bindrule rul_doc_yr, 'mpp_activity_budgets.agreement_doc_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'mpp_activity_budgets', transaction_nbr, transaction_segment

GO


PRINT 'TABLE : mpp_advances_rec'
GO
```

```
create table mpp_advances_rec (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr not null,
appropriation_nbr smallint null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
organization_id typ_orgid null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'mpp_advances_rec.program_yr'
execute sp_bindrule rul_approp_yr, 'mpp_advances_rec.appropriation_yr'
execute sp_bindrule rul_doc_yr, 'mpp_advances_rec.agreement_doc_yr'
execute sp_bindrule rul_debit_credit_sw, 'mpp_advances_rec.debit_credit_flag'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'mpp_advances_rec', transaction_nbr, transaction_nbr_segment

GO


PRINT 'TABLE : mpp_anncmt_payable'
GO


create table mpp_anncmt_payable (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr not null,
appropriation_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'mpp_anncmt_payable.program_yr'
execute sp_bindrule rul_approp_yr, 'mpp_anncmt_payable.appropriation_yr'
execute sp_bindrule rul_debit_credit_sw, 'mpp_anncmt_payable.debit_credit_flag'

GO

PRINT 'TABLE : mpp_branded_generic_cap'
GO


create table mpp_branded_generic_cap (
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
program_yr typ_progyr not null,
organization_id char(8) not null,
branded_generic_flag char(7) not null,
spending_cap money not null,
effective_date datetime not null,
user_nbr smallint not null)
```

```
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_doc_yr, 'mpp_branded_generic_cap.agreement_doc_yr'
execute sp_bindrule rul_program_yr, 'mpp_branded_generic_cap.program_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'mpp_branded_generic_cap', agreement_doc_yr, agreement_doc_nbr, program_yr,
branded_generic_flag, effective_date

GO


PRINT 'TABLE : mpp_branded_sub_budgets'
GO


create table mpp_branded_sub_budgets (
participant_id char(8) not null,
program_yr typ_progyr not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_modifier char(1) not null,
activity_code char(5) not null,
modified_date datetime not null,
market_code char(2) not null,
budget money not null,
com_agg_code char(5) not null,
user_nbr smallint not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'mpp_branded_sub_budgets.program_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'mpp_branded_sub_budgets', participant_id, program_yr, activity_program,
activity_progyr, activity_type, activity_modifier, activity_code, market_code

GO


PRINT 'TABLE : mpp_carry_over'
GO


create table mpp_carry_over (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
organization_id typ_orgid null,
user_nbr typ_usernbr not null,
amount money not null)
GO
```

```
PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'mpp_carry_over.program_yr'
execute sp_bindrule rul_doc_yr, 'mpp_carry_over.agreement_doc_yr'
execute sp_bindrule rul_debit_credit_sw, 'mpp_carry_over.debit_credit_flag'

GO

PRINT 'TABLE : mpp_contingent_liab_bdgt'
GO


create table mpp_contingent_liab_bdgt (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr null,
appropriation_nbr smallint null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
organization_id typ_orgid null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'mpp_contingent_liab_bdgt.program_yr'
execute sp_bindrule rul_approp_yr, 'mpp_contingent_liab_bdgt.appropriation_yr'
execute sp_bindrule rul_doc_yr, 'mpp_contingent_liab_bdgt.agreement_doc_yr'
execute sp_bindrule rul_debit_credit_sw, 'mpp_contingent_liab_bdgt.debit_credit_flag'

GO

PRINT 'TABLE : mpp_contingent_liabilities'
GO


create table mpp_contingent_liabilities (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr null,
appropriation_nbr smallint null,
agreement_doc_yr typ_docyr null,
agreement_doc_nbr smallint null,
debit_credit_flag typ_debit_credit_sw not null,
organization_id typ_orgid null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'mpp_contingent_liabilities.program_yr'
execute sp_bindrule rul_approp_yr, 'mpp_contingent_liabilities.appropriation_yr'
execute sp_bindrule rul_doc_yr, 'mpp_contingent_liabilities.agreement_doc_yr'
execute sp_bindrule rul_debit_credit_sw, 'mpp_contingent_liabilities.debit_credit_flag'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'mpp_contingent_liabilities', transaction_nbr, transaction_nbr_segment
```

```
GO


PRINT 'TABLE : mpp_finding_rec'
GO


create table mpp_finding_rec (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr not null,
appropriation_nbr smallint not null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
organization_id typ_orgid null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'mpp_finding_rec.program_yr'
execute sp_bindrule rul_approp_yr, 'mpp_finding_rec.appropriation_yr'
execute sp_bindrule rul_doc_yr, 'mpp_finding_rec.agreement_doc_yr'
execute sp_bindrule rul_debit_credit_sw, 'mpp_finding_rec.debit_credit_flag'

GO

PRINT 'TABLE : mpp_funds'
GO


create table mpp_funds (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr not null,
appropriation_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'mpp_funds.program_yr'
execute sp_bindrule rul_approp_yr, 'mpp_funds.appropriation_yr'
execute sp_bindrule rul_debit_credit_sw, 'mpp_funds.debit_credit_flag'

GO

PRINT 'TABLE : mpp_general_journal'
GO


create table mpp_general_journal (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_type typ_progtype not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr null,
appropriation_nbr smallint null,
```

```
agreement_doc_yr typ_docyr null,
agreement_doc_nbr smallint null,
organization_id typ_orgid null,
activity_code typ_activity_code null,
cost_category_code typ_cost_category null,
amount money not null,
user_nbr typ_usernbr not null,
initiating_system char(1) not null,
ceiling_type typ_ceiling_type null,
market_code typ_mktcd null,
fips_code typ_fipcd null,
aggregate_comm_code typ_agg_comm null,
branded_generic_flag typ_brand_gener_flag null,
exp_claim_number char(8) null,
exp_line_item_id smallint null,
relevant_doc_type char(15) null,
relevant_doc_yr typ_docyr null,
relevant_doc_nbr smallint null,
rev_transaction_nbr typ_transaction_nbr null,
rev_transaction_nbr_segment smallint null,
comment_sw typ_swn null,
activity_market_code typ_mktcd null,
r2_agreement_doc_yr typ_docyr null,
r2_agreement_doc_nbr smallint null,
r2_organization_id typ_orgid null,
eip_commodity_desc char(40) null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_progtype, 'mpp_general_journal.program_type'
execute sp_bindrule rul_program_yr, 'mpp_general_journal.program_yr'
execute sp_bindrule rul_approp_yr, 'mpp_general_journal.appropriation_yr'
execute sp_bindrule rul_doc_yr, 'mpp_general_journal.agreement_doc_yr'
execute sp_bindrule rul_ceiling_type, 'mpp_general_journal.ceiling_type'
execute sp_bindrule rul_brand_gener_flag, 'mpp_general_journal.branded_generic_flag'
execute sp_bindrule rul_doc_yr, 'mpp_general_journal.relevant_doc_yr'
execute sp_bindrule rul_y_n_sw, 'mpp_general_journal.comment_sw'
execute sp_bindrule rul_doc_yr, 'mpp_general_journal.r2_agreement_doc_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'mpp_general_journal.aggregate_comm_code'
execute sp_bindefault def_swn, 'mpp_general_journal.comment_sw'

PRINT 'TABLE : mpp_partic_funds'
GO


create table mpp_partic_funds (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr null,
appropriation_nbr smallint null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
organization_id typ_orgid null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'mpp_partic_funds.program_yr'
```

```
execute sp_bindrule rul_approp_yr, 'mpp_partic_funds.appropriation_yr'
execute sp_bindrule rul_doc_yr, 'mpp_partic_funds.agreement_doc_yr'
execute sp_bindrule rul_debit_credit_sw, 'mpp_partic_funds.debit_credit_flag'

GO

PRINT 'TABLE : mpp_prog_ceilings'
GO


create table mpp_prog_ceilings (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
organization_id typ_orgid null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'mpp_prog_ceilings.program_yr'
execute sp_bindrule rul_doc_yr, 'mpp_prog_ceilings.agreement_doc_yr'
execute sp_bindrule rul_debit_credit_sw, 'mpp_prog_ceilings.debit_credit_flag'

GO

PRINT 'TABLE : mpp_program_announcement'
GO


create table mpp_program_announcement (
doc_yr typ_docyr not null,
doc_nbr smallint not null,
fed_register_annc_yr smallint null,
fed_register_annc_nbr int null,
application_deadline smalldatetime not null,
total_funding_available money null,
appropriation_yr typ_approp_yr not null,
appropriation_nbr smallint not null,
timestamp timestamp null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_doc_yr, 'mpp_program_announcement.doc_yr'
execute sp_bindrule rul_approp_yr, 'mpp_program_announcement.appropriation_yr'

GO

PRINT 'TABLE : mpp_transaction_comments'
GO


create table mpp_transaction_comments (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
comment char(255) not null)
GO


PRINT 'TABLE : mpp_transaction_type'
GO


create table mpp_transaction_type (
```

```
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_type char(1) not null)
GO


PRINT 'TABLE : mpp_treasury_pay'
GO


create table mpp_treasury_pay (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr not null,
appropriation_nbr smallint not null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
organization_id typ_orgid null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'mpp_treasury_pay.program_yr'
execute sp_bindrule rul_approp_yr, 'mpp_treasury_pay.appropriation_yr'
execute sp_bindrule rul_doc_yr, 'mpp_treasury_pay.agreement_doc_yr'
execute sp_bindrule rul_debit_credit_sw, 'mpp_treasury_pay.debit_credit_flag'

GO


use master
GO

/*
  Script for Server sybase07 (Adaptive Server Enterprise/12.5.3/EBF 13330 ESD#7/P/Compaq
Tru64/OSF1 V5.0A/ase1253/1951/64-bit/FBO/Fri Mar 24 07:24:51 2006) on axposf
*/

/*  Database 'usamprod'  */
use usamprod
GO


/*
  Table(s)
*/

PRINT 'TABLE : org_type_description'
GO


create table org_type_description (
org_type_code smallint not null,
org_type_desc varchar(50) not null)
GO


PRINT 'TABLE : part_duns'
GO


create table part_duns (
part_id typ_partid not null,
duns_number char(9) null)
GO
```

```
PRINT 'TABLE : participant'
GO


create table participant (
participant_id typ_partid not null,
participant_name char(50) not null,
participant_type char(10) null,
non_profit_sw typ_swy not null,
tax_exempt_id char(10) null,
hdqtrs_addr_1 typ_addrlin null,
hdqtrs_addr_2 typ_addrlin null,
hdqtrs_addr_3 typ_addrlin null,
hdqtrs_addr_4 typ_addrlin null,
hdqtrs_addr_5 typ_addrlin null,
hdqtrs_city typ_city null,
hdqtrs_state char(2) null,
hdqtrs_zip_1 typ_zip1 null,
hdqtrs_zip_2 typ_zip2 null,
ceo_contact_id smallint null,
accounting_contact_id smallint null,
policy_contact_id smallint null,
small_business_sw char(1) null,
bank_name char(40) null,
bank_aba_nbr char(9) null,
bank_acct_nbr char(40) null,
payable_to_info char(50) null,
payment_xfer_method char(4) null,
agreement_nbr char(6) null,
alias_id typ_partid null,
org_type_code smallint null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_y_n_sw, 'participant.non_profit_sw'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_swy, 'participant.non_profit_sw'

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'participant', participant_id

GO


PRINT 'TABLE : participant_bank_info'
GO


create table participant_bank_info (
agreement_doc_yr smallint not null,
agreement_doc_nbr smallint not null,
program_yr smallint not null,
program char(40) null,
agreement_nbr char(9) null,
bank_acct_info_1 char(40) null,
bank_acct_info_2 char(40) null,
bank_acct_info_3 char(40) null,
bank_acct_info_4 char(40) null,
bank_acct_info_5 char(40) null,
bank_acct_info_6 char(40) null,
bank_acct_info_7 char(40) null,
bank_acct_info_8 char(40) null)
GO
```

```
PRINT 'TABLE : participant_bank_program'
GO


create table participant_bank_program (
participant_id typ_partid not null,
program char(3) not null,
hdqtrs_addr_1 typ_addrlin null,
hdqtrs_addr_2 typ_addrlin null,
hdqtrs_addr_3 typ_addrlin null,
hdqtrs_addr_4 typ_addrlin null,
hdqtrs_addr_5 typ_addrlin null,
hdqtrs_city typ_city null,
hdqtrs_state char(2) null,
hdqtrs_zip_1 typ_zip1 null,
hdqtrs_zip_2 typ_zip2 null,
payment_xfer_method char(4) null,
agreement_nbr char(6) null)
GO


PRINT 'TABLE : participant_fax'
GO


create table participant_fax (
part_id typ_partid not null,
fax varchar(20) null)
GO


PRINT 'TABLE : participant_gen_info'
GO


create table participant_gen_info (
appropriation_summary_code char(7) not null,
agency_station_nbr char(4) not null,
agency_acct_code char(7) not null,
dept_or_estab varchar(30) not null,
bureau_or_office varchar(30) not null,
xmit_office_loc varchar(30) not null,
boss_mosdiv_fas char(30) not null,
boss_mosdiv_fas_title char(30) not null,
controller_ccc char(30) not null,
boss_fisdiv_ascs char(30) not null,
boss_fisdiv_ascs_title char(30) not null,
boss_cmp_fas char(30) not null,
boss_cmp_fas_title char(30) not null,
fmd_agency_acct_code char(7) null,
qsp_agency_acct_code char(7) null,
emp_agency_acct_code char(7) null)
GO


PRINT 'TABLE : participant_payment_contact'
GO


create table participant_payment_contact (
participant_id typ_partid not null,
program char(3) not null,
program_yr typ_progyr not null,
payment_contact varchar(50) not null,
contact_salutation varchar(50) not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'participant_payment_contact.program_yr'

GO
```

```
PRINT 'TABLE : participant_subgrp'
GO


create table participant_subgrp (
subgrp_id typ_partid not null,
parent_id typ_partid not null,
program char(3) not null,
subgrp_name char(50) not null,
hdqtrs_addr_1 typ_addrlin null,
hdqtrs_addr_2 typ_addrlin null,
hdqtrs_addr_3 typ_addrlin null,
hdqtrs_addr_4 typ_addrlin null,
hdqtrs_addr_5 typ_addrlin null,
hdqtrs_city typ_city null,
hdqtrs_state char(2) null,
hdqtrs_zip_1 typ_zip1 null,
hdqtrs_zip_2 typ_zip2 null,
payment_xfer_method char(4) null,
agreement_nbr char(6) null,
subgrp_alias_id typ_partid null,
tax_exempt_id char(10) null)
GO


PRINT 'TABLE : participant_subgrp_bank_info'
GO


create table participant_subgrp_bank_info (
subgrp_id typ_partid not null,
program_yr typ_progyr not null,
parent_id typ_partid not null,
program char(3) null,
bank_acct_info_1 char(40) null,
bank_acct_info_2 char(40) null,
bank_acct_info_3 char(40) null,
bank_acct_info_4 char(40) null,
bank_acct_info_5 char(40) null,
bank_acct_info_6 char(40) null,
bank_acct_info_7 char(40) null,
bank_acct_info_8 char(40) null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'participant_subgrp_bank_info.program_yr'

GO

PRINT 'TABLE : perf_meas_reporting'
GO


create table perf_meas_reporting (
reporting_id int not null,
years_out_from_baseline int not null,
goal numeric(11,1) null,
actual numeric(11,1) null)
GO


PRINT 'TABLE : perf_meas_reporting_desc'
GO


create table perf_meas_reporting_desc (
reporting_id int not null,
part_id typ_partid not null,
program_yr int not null,
market_id typ_mktcd not null,
com_agg_code typ_agg_comm not null,
constraint_no int not null,
```

```
perf_meas_desc varchar(255) not null,
perf_meas_footnote varchar(255) null,
baseline_yr int not null,
baseline_nbr numeric(11,1) null,
baseline_text varchar(255) null)
GO


PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'perf_meas_reporting_desc.com_agg_code'

PRINT 'TABLE : perf_meas_reporting_text'
GO


create table perf_meas_reporting_text (
reporting_id int not null,
years_out_from_baseline int not null,
goal varchar(255) null,
actual varchar(255) null)
GO


PRINT 'TABLE : posts'
GO


create table posts (
usam_post_id char(8) not null,
active_sw char(1) not null,
post_name char(50) not null,
post_level char(12) null)
GO



use master
GO

/*
  Script for Server sybase07 (Adaptive Server Enterprise/12.5.3/EBF 13330 ESD#7/P/Compaq
Tru64/OSF1 V5.0A/ase1253/1951/64-bit/FBO/Fri Mar 24 07:24:51 2006) on axposf
*/

/*  Database 'usamprod'  */
use usamprod
GO


/*
  Table(s)
*/

PRINT 'TABLE : qsp_activity_budgets'
GO


create table qsp_activity_budgets (
transaction_nbr typ_transaction_nbr not null,
transaction_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
debit_credit_flag typ_debit_credit_sw not null,
amount money not null,
user_nbr typ_usernbr not null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
organization_id typ_orgid not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_modifier char(1) not null,
```

```
activity_code char(5) not null,
activity_title char(40) not null,
market_code char(16) not null,
region_flag char(1) not null,
com_agg_code char(5) not null,
activity_market_code char(16) not null,
control_level char(1) not null,
deleted_flag char(1) not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'qsp_activity_budgets.program_yr'
execute sp_bindrule rul_debit_credit_sw, 'qsp_activity_budgets.debit_credit_flag'
execute sp_bindrule rul_doc_yr, 'qsp_activity_budgets.agreement_doc_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'qsp_activity_budgets', transaction_nbr, transaction_segment

GO


PRINT 'TABLE : qsp_anncmt_payable'
GO


create table qsp_anncmt_payable (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr not null,
appropriation_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'qsp_anncmt_payable.program_yr'
execute sp_bindrule rul_approp_yr, 'qsp_anncmt_payable.appropriation_yr'
execute sp_bindrule rul_debit_credit_sw, 'qsp_anncmt_payable.debit_credit_flag'

GO

PRINT 'TABLE : qsp_disbur_sched_counter'
GO


create table qsp_disbur_sched_counter (
latest_disbur_sched_nbr int not null)
GO


PRINT 'TABLE : qsp_funds'
GO


create table qsp_funds (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr not null,
```

```
appropriation_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'qsp_funds.program_yr'
execute sp_bindrule rul_approp_yr, 'qsp_funds.appropriation_yr'
execute sp_bindrule rul_debit_credit_sw, 'qsp_funds.debit_credit_flag'

GO

PRINT 'TABLE : qsp_general_journal'
GO


create table qsp_general_journal (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_type typ_progtype not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr null,
appropriation_nbr smallint null,
agreement_doc_yr typ_docyr null,
agreement_doc_nbr smallint null,
organization_id typ_orgid null,
activity_code typ_activity_code null,
cost_category_code typ_cost_category null,
amount money not null,
user_nbr typ_usernbr not null,
initiating_system char(1) not null,
ceiling_type typ_ceiling_type null,
market_code typ_mktcd null,
fips_code typ_fipcd null,
aggregate_comm_code typ_agg_comm null,
branded_generic_flag typ_brand_gener_flag null,
exp_claim_number char(8) null,
exp_line_item_id smallint null,
relevant_doc_type char(15) null,
relevant_doc_yr typ_docyr null,
relevant_doc_nbr smallint null,
rev_transaction_nbr typ_transaction_nbr null,
rev_transaction_nbr_segment smallint null,
comment_sw typ_swn null,
activity_market_code typ_mktcd null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_progtype, 'qsp_general_journal.program_type'
execute sp_bindrule rul_program_yr, 'qsp_general_journal.program_yr'
execute sp_bindrule rul_approp_yr, 'qsp_general_journal.appropriation_yr'
execute sp_bindrule rul_doc_yr, 'qsp_general_journal.agreement_doc_yr'
execute sp_bindrule rul_ceiling_type, 'qsp_general_journal.ceiling_type'
execute sp_bindrule rul_brand_gener_flag, 'qsp_general_journal.branded_generic_flag'
execute sp_bindrule rul_doc_yr, 'qsp_general_journal.relevant_doc_yr'
execute sp_bindrule rul_y_n_sw, 'qsp_general_journal.comment_sw'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'qsp_general_journal.aggregate_comm_code'
execute sp_bindefault def_swn, 'qsp_general_journal.comment_sw'

PRINT 'TABLE : qsp_partic_funds'
```

```
GO


create table qsp_partic_funds (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr null,
appropriation_nbr smallint null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
organization_id typ_orgid null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'qsp_partic_funds.program_yr'
execute sp_bindrule rul_approp_yr, 'qsp_partic_funds.appropriation_yr'
execute sp_bindrule rul_doc_yr, 'qsp_partic_funds.agreement_doc_yr'
execute sp_bindrule rul_debit_credit_sw, 'qsp_partic_funds.debit_credit_flag'

GO

PRINT 'TABLE : qsp_prog_ceilings'
GO


create table qsp_prog_ceilings (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
organization_id typ_orgid null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'qsp_prog_ceilings.program_yr'
execute sp_bindrule rul_doc_yr, 'qsp_prog_ceilings.agreement_doc_yr'
execute sp_bindrule rul_debit_credit_sw, 'qsp_prog_ceilings.debit_credit_flag'

GO

PRINT 'TABLE : qsp_program_announcement'
GO


create table qsp_program_announcement (
doc_yr typ_docyr not null,
doc_nbr smallint not null,
fed_register_annc_yr smallint null,
fed_register_annc_nbr int null,
application_deadline smalldatetime not null,
total_funding_available money not null,
appropriation_yr typ_approp_yr not null,
appropriation_nbr smallint not null,
tstamp timestamp null)
GO
```

```
PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_doc_yr, 'qsp_program_announcement.doc_yr'
execute sp_bindrule rul_approp_yr, 'qsp_program_announcement.appropriation_yr'

GO

PRINT 'TABLE : qsp_transaction_comments'
GO


create table qsp_transaction_comments (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
comment char(255) not null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'qsp_transaction_comments', transaction_nbr, transaction_nbr_segment

GO


PRINT 'TABLE : qsp_transaction_defns'
GO


create table qsp_transaction_defns (
transaction_code typ_trancd not null,
credit_table char(30) not null,
debit_table char(30) null,
trans_description char(40) not null,
user_nbr typ_usernbr not null,
definition_datetime datetime not null)
GO


PRINT 'TABLE : qsp_transaction_type'
GO


create table qsp_transaction_type (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_type char(1) not null)
GO


PRINT 'TABLE : reason_for_rejection'
GO


create table reason_for_rejection (
application_doc_yr typ_docyr not null,
application_doc_nbr smallint not null,
reason_for_rejection text not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_doc_yr, 'reason_for_rejection.application_doc_yr'

GO

PRINT 'TABLE : reimbursement_text'
GO


create table reimbursement_text (
```

```
program varchar(7) not null,
list_order char(1) not null,
line char(1) not null,
text varchar(80) not null)
GO


PRINT 'TABLE : sector'
GO


create table sector (
sector_code typ_agg_comm not null,
sector_desc char(80) not null)
GO


PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'sector.sector_code'

PRINT 'TABLE : standard_region_countries'
GO


create table standard_region_countries (
fips_code typ_fipcd not null,
region_id int not null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'standard_region_countries', fips_code

GO


PRINT 'TABLE : standard_regions'
GO


create table standard_regions (
region_id int not null,
country_name varchar(50) not null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'standard_regions', region_id

GO


PRINT 'TABLE : state'
GO


create table state (
state_code char(2) not null,
state_name char(25) not null)
GO



use master
GO

/*
  Script for Server sybase07 (Adaptive Server Enterprise/12.5.3/EBF 13330 ESD#7/P/Compaq
Tru64/OSF1 V5.0A/ase1253/1951/64-bit/FBO/Fri Mar 24 07:24:51 2006) on axposf
```

```
*/

/*  Database 'usamprod'  */
use usamprod
GO


/*
  Table(s)
*/

PRINT 'TABLE : tasc_activity_budget_yr_rec'
GO


create table tasc_activity_budget_yr_rec (
program_yr typ_progyr not null,
part_id typ_partid not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_code char(5) not null,
market_id typ_mktcd not null,
description varchar(255) not null,
amount int not null,
part_contrib int null,
third_party_contrib int null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule typ_progyr, 'tasc_activity_budget_yr_rec.program_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'tasc_activity_budget_yr_rec', program_yr, part_id, activity_program,
activity_progyr, activity_type, activity_code, market_id, description

GO


PRINT 'TABLE : tasc_notify'
GO


create table tasc_notify (
notification_type varchar(10) not null,
notification_time datetime not null)
GO


PRINT 'TABLE : tasc_proposal'
GO


create table tasc_proposal (
part_id typ_partid not null,
program_yr typ_progyr not null,
market_id typ_mktcd not null,
com_agg_code typ_agg_comm not null,
title varchar(255) not null,
est_funding_yr_2 int null,
est_funding_yr_3 int null,
exp_date datetime null,
status varchar(255) null,
trade_barrier_type int null)
GO


PRINT 'BINDING Rule(s) to column'
```

```
GO

execute sp_bindrule typ_progyr, 'tasc_proposal.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'tasc_proposal.com_agg_code'

PRINT 'TABLE : tasc_proposal_text'
GO


create table tasc_proposal_text (
part_id typ_partid not null,
program_yr typ_progyr not null,
market_id typ_mktcd not null,
com_agg_code typ_agg_comm not null,
title varchar(255) not null,
proposal_text_id int not null,
proposal_text text not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'tasc_proposal_text.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'tasc_proposal_text.com_agg_code'

PRINT 'TABLE : tasc_trade_barrier_type'
GO


create table tasc_trade_barrier_type (
type_id int not null,
type_desc varchar(255) not null)
GO


PRINT 'TABLE : tom_branded_company'
GO


create table tom_branded_company (
Company_ID smallint null,
Name_in_USAM char(58) null,
Size char(1) null,
Company_Name char(58) null,
Address_1 char(34) null,
Address_2 char(25) null,
City char(20) null,
State char(2) null,
Zip char(15) null,
Area_Code char(15) null,
Telephone char(17) null,
FAX char(17) null,
Participant char(27) null,
Promoted_Product char(57) null,
FIELD015 char(1) null,
Origin char(3) null)
GO


PRINT 'TABLE : tom_branded_dunn'
GO
```

```
create table tom_branded_dunn (
Company_ID smallint not null,
DUNN_number char(10) not null)
GO


PRINT 'TABLE : transaction_counter'
GO


create table transaction_counter (
max_transaction_nbr typ_transaction_nbr not null)
GO


PRINT 'TABLE : transaction_defns'
GO


create table transaction_defns (
transaction_code typ_trancd not null,
credit_table char(30) not null,
debit_table char(30) null,
trans_description char(40) not null,
user_nbr typ_usernbr not null,
definition_datetime datetime not null)
GO


PRINT 'TABLE : tsc_activity_budgets'
GO


create table tsc_activity_budgets (
transaction_nbr typ_transaction_nbr not null,
transaction_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
debit_credit_flag typ_debit_credit_sw not null,
amount money not null,
user_nbr typ_usernbr not null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
organization_id typ_orgid not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_modifier char(1) not null,
activity_code char(5) not null,
activity_title char(40) not null,
market_code char(16) not null,
region_flag char(1) not null,
com_agg_code char(5) not null,
activity_market_code char(16) not null,
control_level char(1) not null,
deleted_flag char(1) not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'tsc_activity_budgets.program_yr'
execute sp_bindrule rul_debit_credit_sw, 'tsc_activity_budgets.debit_credit_flag'
execute sp_bindrule rul_doc_yr, 'tsc_activity_budgets.agreement_doc_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'tsc_activity_budgets', transaction_nbr, transaction_segment
```

```
GO


PRINT 'TABLE : tsc_advances_rec'
GO


create table tsc_advances_rec (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr null,
appropriation_nbr smallint null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
organization_id typ_orgid null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'tsc_advances_rec.program_yr'
execute sp_bindrule rul_approp_yr, 'tsc_advances_rec.appropriation_yr'
execute sp_bindrule rul_doc_yr, 'tsc_advances_rec.agreement_doc_yr'
execute sp_bindrule rul_debit_credit_sw, 'tsc_advances_rec.debit_credit_flag'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'tsc_advances_rec', transaction_nbr, transaction_nbr_segment

GO


PRINT 'TABLE : tsc_anncmt_payable'
GO


create table tsc_anncmt_payable (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr not null,
appropriation_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'tsc_anncmt_payable.program_yr'
execute sp_bindrule rul_approp_yr, 'tsc_anncmt_payable.appropriation_yr'
execute sp_bindrule rul_debit_credit_sw, 'tsc_anncmt_payable.debit_credit_flag'

GO

PRINT 'TABLE : tsc_disbur_sched_counter'
GO


create table tsc_disbur_sched_counter (
latest_disbur_sched_nbr int not null)
```

```
GO


PRINT 'TABLE : tsc_final_rpt'
GO


create table tsc_final_rpt (
program_yr typ_progyr not null,
part_id typ_partid not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_code char(5) not null,
market_id typ_mktcd not null,
com_agg_code typ_agg_comm not null,
submit_date datetime not null,
fas_reviewed char(1) not null,
fas_comments char(255) null,
fas_reviewed_date datetime null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'tsc_final_rpt.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'tsc_final_rpt.com_agg_code'

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'tsc_final_rpt', program_yr, part_id, activity_program, activity_progyr,
activity_type, activity_code, market_id

GO


PRINT 'TABLE : tsc_funds'
GO


create table tsc_funds (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr not null,
appropriation_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'tsc_funds.program_yr'
execute sp_bindrule rul_approp_yr, 'tsc_funds.appropriation_yr'
execute sp_bindrule rul_debit_credit_sw, 'tsc_funds.debit_credit_flag'

GO

PRINT 'TABLE : tsc_general_journal'
GO
```

```
create table tsc_general_journal (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_type typ_progtype not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr null,
appropriation_nbr smallint null,
agreement_doc_yr typ_docyr null,
agreement_doc_nbr smallint null,
organization_id typ_orgid null,
activity_code typ_activity_code null,
cost_category_code typ_cost_category null,
amount money not null,
user_nbr typ_usernbr not null,
initiating_system char(1) not null,
ceiling_type typ_ceiling_type null,
market_code typ_mktcd null,
fips_code typ_fipcd null,
aggregate_comm_code typ_agg_comm null,
branded_generic_flag typ_brand_gener_flag null,
exp_claim_number char(8) null,
exp_line_item_id smallint null,
relevant_doc_type char(15) null,
relevant_doc_yr typ_docyr null,
relevant_doc_nbr smallint null,
rev_transaction_nbr typ_transaction_nbr null,
rev_transaction_nbr_segment smallint null,
comment_sw typ_swn null,
activity_market_code typ_mktcd null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_progtype, 'tsc_general_journal.program_type'
execute sp_bindrule rul_program_yr, 'tsc_general_journal.program_yr'
execute sp_bindrule rul_approp_yr, 'tsc_general_journal.appropriation_yr'
execute sp_bindrule rul_doc_yr, 'tsc_general_journal.agreement_doc_yr'
execute sp_bindrule rul_ceiling_type, 'tsc_general_journal.ceiling_type'
execute sp_bindrule rul_brand_gener_flag, 'tsc_general_journal.branded_generic_flag'
execute sp_bindrule rul_doc_yr, 'tsc_general_journal.relevant_doc_yr'
execute sp_bindrule rul_y_n_sw, 'tsc_general_journal.comment_sw'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'tsc_general_journal.aggregate_comm_code'
execute sp_bindefault def_swn, 'tsc_general_journal.comment_sw'

PRINT 'TABLE : tsc_partic_funds'
GO


create table tsc_partic_funds (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
appropriation_yr typ_approp_yr null,
appropriation_nbr smallint null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
organization_id typ_orgid null,
user_nbr typ_usernbr not null,
amount money not null)
GO
```

```
PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'tsc_partic_funds.program_yr'
execute sp_bindrule rul_approp_yr, 'tsc_partic_funds.appropriation_yr'
execute sp_bindrule rul_doc_yr, 'tsc_partic_funds.agreement_doc_yr'
execute sp_bindrule rul_debit_credit_sw, 'tsc_partic_funds.debit_credit_flag'

GO

PRINT 'TABLE : tsc_prog_ceilings'
GO


create table tsc_prog_ceilings (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_date datetime not null,
transaction_code typ_trancd not null,
program_yr typ_progyr not null,
agreement_doc_yr typ_docyr not null,
agreement_doc_nbr smallint not null,
debit_credit_flag typ_debit_credit_sw not null,
organization_id typ_orgid null,
user_nbr typ_usernbr not null,
amount money not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'tsc_prog_ceilings.program_yr'
execute sp_bindrule rul_doc_yr, 'tsc_prog_ceilings.agreement_doc_yr'
execute sp_bindrule rul_debit_credit_sw, 'tsc_prog_ceilings.debit_credit_flag'

GO

PRINT 'TABLE : tsc_program_announcement'
GO


create table tsc_program_announcement (
doc_yr typ_docyr not null,
doc_nbr smallint not null,
fed_register_annc_yr smallint null,
fed_register_annc_nbr int null,
application_deadline smalldatetime not null,
total_funding_available money not null,
appropriation_yr typ_approp_yr not null,
appropriation_nbr smallint not null,
tstamp timestamp null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_doc_yr, 'tsc_program_announcement.doc_yr'
execute sp_bindrule rul_approp_yr, 'tsc_program_announcement.appropriation_yr'

GO

PRINT 'TABLE : tsc_transaction_comments'
GO


create table tsc_transaction_comments (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
comment char(255) not null)
GO


PRINT 'CREATING Primary Key'
```

```
GO

execute sp_primarykey 'tsc_transaction_comments', transaction_nbr, transaction_nbr_segment

GO


PRINT 'TABLE : tsc_transaction_defns'
GO


create table tsc_transaction_defns (
transaction_code typ_trancd not null,
credit_table char(30) not null,
debit_table char(30) null,
trans_description char(40) not null,
user_nbr typ_usernbr not null,
definition_datetime datetime not null)
GO


PRINT 'TABLE : tsc_transaction_type'
GO


create table tsc_transaction_type (
transaction_nbr typ_transaction_nbr not null,
transaction_nbr_segment smallint not null,
transaction_type char(1) not null)
GO



use master
GO

/*
  Script for Server sybase07 (Adaptive Server Enterprise/12.5.3/EBF 13330 ESD#7/P/Compaq
Tru64/OSF1 V5.0A/ase1253/1951/64-bit/FBO/Fri Mar 24 07:24:51 2006) on axposf
*/

/*  Database 'usamprod'  */
use usamprod
GO


/*
  Table(s)
*/

PRINT 'TABLE : ues_activity_addit_text_type'
GO


create table ues_activity_addit_text_type (
text_id int not null,
description varchar(255) not null)
GO


PRINT 'TABLE : ues_activity_additional_text'
GO


create table ues_activity_additional_text (
program_yr typ_progyr not null,
part_id typ_partid not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_code char(5) not null,
market_id typ_mktcd not null,
com_agg_code typ_agg_comm not null,
text_id int not null,
text text null)
```

```
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ues_activity_additional_text.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'ues_activity_additional_text.com_agg_code'

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_activity_additional_text', program_yr, part_id, activity_program,
activity_progyr, activity_type, activity_code, market_id

GO


PRINT 'TABLE : ues_activity_branded_company'
GO


create table ues_activity_branded_company (
participant_id char(8) not null,
program_yr typ_progyr not null,
plan_activity_code char(10) not null,
plan_market_code char(16) not null,
activity_code char(10) not null,
market_code char(2) not null,
company_id smallint not null,
modified_date datetime not null,
user_nbr smallint not null,
budget money not null,
status char(1) not null,
activity_description char(64) not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ues_activity_branded_company.program_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_activity_branded_company', participant_id, program_yr,
plan_activity_code, plan_market_code, activity_code, market_code

GO


PRINT 'TABLE : ues_activity_budgets'
GO


create table ues_activity_budgets (
program_yr smallint not null,
part_id char(8) not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_code char(5) not null,
activity_title char(40) not null,
market_id char(16) not null,
region_flag char(1) not null,
com_agg_code char(5) not null,
```

```
requested int not null,
recommended int not null,
approved int not null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_activity_budgets', program_yr, part_id, activity_program,
activity_progyr, activity_type, activity_code, market_id

GO


PRINT 'TABLE : ues_activity_budgets_outside'
GO


create table ues_activity_budgets_outside (
program_yr smallint not null,
part_id char(8) not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_code char(5) not null,
activity_title char(40) not null,
market_id char(16) not null,
region_flag char(1) not null,
com_agg_code char(5) not null,
requested int not null,
recommended int not null,
approved int not null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_activity_budgets_outside', program_yr, part_id, activity_program,
activity_progyr, activity_type, activity_code, market_id

GO


PRINT 'TABLE : ues_activity_contrib'
GO


create table ues_activity_contrib (
program_yr typ_progyr not null,
part_id typ_partid not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_code char(5) not null,
market_id typ_mktcd not null,
com_agg_code typ_agg_comm not null,
constraint_no int null,
contribution int not null,
contrib_type int null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ues_activity_contrib.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'ues_activity_contrib.com_agg_code'
```

```
PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_activity_contrib', program_yr, part_id, activity_program,
activity_progyr, activity_type, activity_code, market_id

GO


PRINT 'TABLE : ues_activity_contrib_type'
GO


create table ues_activity_contrib_type (
contrib_type int not null,
contrib_desc varchar(255) not null)
GO


PRINT 'TABLE : ues_activity_incumbents'
GO


create table ues_activity_incumbents (
program_yr typ_progyr not null,
part_id typ_partid not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_code char(5) not null,
market_id typ_mktcd not null,
incumbent_title varchar(40) null,
incumbent_grade varchar(40) null,
incumbent_name varchar(40) not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ues_activity_incumbents.program_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_activity_incumbents', program_yr, part_id, activity_program,
activity_progyr, activity_type, activity_code, market_id

GO


PRINT 'TABLE : ues_activity_tag'
GO


create table ues_activity_tag (
program_yr typ_progyr not null,
part_id typ_partid not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_code char(5) not null,
market_id typ_mktcd not null,
com_agg_code typ_agg_comm not null,
constraint_no int null,
tag varchar(255) not null,
tag_type int null)
GO


PRINT 'BINDING Rule(s) to column'
GO
```

```
execute sp_bindrule rul_program_yr, 'ues_activity_tag.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'ues_activity_tag.com_agg_code'

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_activity_tag', program_yr, part_id, activity_program, activity_progyr,
activity_type, activity_code, market_id

GO


PRINT 'TABLE : ues_activity_tag_type'
GO


create table ues_activity_tag_type (
tag_type int not null,
tag_desc varchar(255) not null)
GO


PRINT 'TABLE : ues_activity_text'
GO


create table ues_activity_text (
program_yr typ_progyr not null,
part_id typ_partid not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_code char(5) not null,
market_id typ_mktcd not null,
com_agg_code typ_agg_comm not null,
constraint_no int null,
activity_desc text null,
expected_result text null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ues_activity_text.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'ues_activity_text.com_agg_code'

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_activity_text', program_yr, part_id, activity_program,
activity_progyr, activity_type, activity_code, market_id

GO


PRINT 'TABLE : ues_activity_types'
GO


create table ues_activity_types (
activity_type char(1) not null,
```

```
branded_generic_flag char(7) not null,
activity_type_desc char(15) not null)
GO


PRINT 'TABLE : ues_admin_activity_budgets'
GO


create table ues_admin_activity_budgets (
program_yr typ_progyr not null,
part_id typ_partid not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_code char(5) not null,
activity_title char(40) null,
market_id typ_mktcd not null,
region_flag char(1) not null,
rent int not null,
salaries int not null,
other int not null,
recommended int not null,
approved int not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ues_admin_activity_budgets.program_yr'

GO

PRINT 'TABLE : ues_analysis_of_market'
GO


create table ues_analysis_of_market (
part_id typ_partid not null,
program_yr typ_progyr not null,
com_agg_code typ_agg_comm not null,
exported_product char(200) not null,
text_type varchar(15) not null,
application_text text null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ues_analysis_of_market.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'ues_analysis_of_market.com_agg_code'

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_analysis_of_market', program_yr, part_id, com_agg_code,
exported_product, text_type

GO


PRINT 'TABLE : ues_applicant_notes'
GO


create table ues_applicant_notes (
part_id typ_partid not null,
```

```
program_yr typ_progyr not null,
application_section_id int not null,
section_completed char(1) not null,
comments varchar(255) null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ues_applicant_notes.program_yr'

GO

PRINT 'TABLE : ues_applicants_by_program_yr'
GO


create table ues_applicants_by_program_yr (
program_yr smallint not null,
part_id char(8) not null,
plan_start_month int null,
plan_end_month int null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_applicants_by_program_yr', program_yr, part_id

GO


PRINT 'TABLE : ues_application_section'
GO


create table ues_application_section (
application_section_id int not null,
section_order int not null,
application_section_desc varchar(255) null)
GO


PRINT 'TABLE : ues_application_session'
GO


create table ues_application_session (
session_id int not null,
login varchar(15) not null,
part_id typ_partid null,
program_yr typ_progyr not null,
user_type_code char(10) not null,
session_start datetime not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ues_application_session.program_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_application_session', session_id

GO


PRINT 'TABLE : ues_application_tag'
```

```
GO


create table ues_application_tag (
program_yr typ_progyr not null,
part_id typ_partid not null,
tag varchar(255) null,
text_id int not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule typ_progyr, 'ues_application_tag.program_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_application_tag', program_yr, part_id, text_id

GO


PRINT 'TABLE : ues_application_text'
GO


create table ues_application_text (
part_id typ_partid not null,
program_yr typ_progyr not null,
text_id int not null,
application_text text null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ues_application_text.program_yr'

GO

PRINT 'TABLE : ues_cmdty_promoted'
GO


create table ues_cmdty_promoted (
program_yr smallint not null,
part_id char(8) not null,
com_agg_code char(5) not null,
exported_product char(200) not null,
percent_us_org smallint not null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_cmdty_promoted', program_yr, part_id, com_agg_code

GO


PRINT 'TABLE : ues_constraint_tag'
GO


create table ues_constraint_tag (
program_yr typ_progyr not null,
part_id typ_partid not null,
com_agg_code typ_agg_comm not null,
market_id typ_mktcd not null,
```

```
constraint_no int not null,
tag_type int not null,
tag varchar(255) not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ues_constraint_tag.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'ues_constraint_tag.com_agg_code'

PRINT 'TABLE : ues_constraint_tag_type'
GO


create table ues_constraint_tag_type (
tag_type int not null,
tag_desc varchar(255) not null)
GO


PRINT 'TABLE : ues_constraint_types'
GO


create table ues_constraint_types (
constraint_type_id int not null,
constraint_type varchar(255) not null)
GO


PRINT 'TABLE : ues_constraints'
GO


create table ues_constraints (
program_yr typ_progyr not null,
part_id typ_partid not null,
com_agg_code typ_agg_comm not null,
market_id char(16) not null,
constraint_no int not null,
constraint_title varchar(80) not null,
constraint_text text null,
constraint_type varchar(40) null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ues_constraints.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'ues_constraints.com_agg_code'

PRINT 'TABLE : ues_contrib_type'
GO


create table ues_contrib_type (
type_id int not null,
description varchar(55) null)
GO
```

```
PRINT 'TABLE : ues_contributions'
GO


create table ues_contributions (
program_yr smallint not null,
program_id smallint not null,
part_id char(8) not null,
part_percent float null,
part_dollars int null,
industry_percent float null,
industry_dollars int null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_contributions', program_yr, program_id, part_id

GO


PRINT 'TABLE : ues_ec_expiration'
GO


create table ues_ec_expiration (
part_id varchar(8) not null,
application_yr int not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
expires datetime null)
GO


PRINT 'TABLE : ues_expenditure_est'
GO


create table ues_expenditure_est (
part_id typ_partid not null,
program_yr typ_progyr not null,
open_year typ_progyr not null,
expenditures money null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ues_expenditure_est.program_yr'
execute sp_bindrule rul_program_yr, 'ues_expenditure_est.open_year'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_expenditure_est', part_id, program_yr

GO


PRINT 'TABLE : ues_export_goals_by_country'
GO


create table ues_export_goals_by_country (
program_yr smallint not null,
part_id char(8) not null,
market_id char(16) not null,
region_flag char(1) not null,
com_agg_code char(5) not null,
```

```
export_yr smallint not null,
value numeric(11,0) null,
volume numeric(11,0) null,
vol_unit char(5) not null,
market_share numeric(5,1) not null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_export_goals_by_country', program_yr, part_id, market_id,
com_agg_code, export_yr

GO


PRINT 'TABLE : ues_joint_activity'
GO


create table ues_joint_activity (
program_yr typ_progyr not null,
part_id typ_partid not null,
activity_program char(1) not null,
activity_progyr char(2) not null,
activity_type char(1) not null,
activity_code char(5) not null,
market_id typ_mktcd not null,
com_agg_code typ_agg_comm not null,
constraint_no int null,
joint_part_id typ_partid not null,
joint_com_agg typ_agg_comm not null,
joint_market_id typ_mktcd not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ues_joint_activity.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'ues_joint_activity.com_agg_code'
execute sp_bindefault def_agg_comm, 'ues_joint_activity.joint_com_agg'

PRINT 'TABLE : ues_map_carryover'
GO


create table ues_map_carryover (
program_yr smallint not null,
part_id char(8) not null,
allocation int not null,
expenditures int not null,
carryover int not null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_map_carryover', program_yr, part_id

GO


PRINT 'TABLE : ues_map_plan_yr'
GO
```

```
create table ues_map_plan_yr (
program_yr typ_progyr not null,
part_id typ_partid not null,
plan_start_month smallint null,
plan_end_month smallint null,
plan_start_year int null,
plan_end_year int null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ues_map_plan_yr.program_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_map_plan_yr', program_yr, part_id

GO


PRINT 'TABLE : ues_market_assessment_text'
GO


create table ues_market_assessment_text (
program_yr typ_progyr not null,
part_id typ_partid not null,
market_id typ_mktcd not null,
com_agg_code typ_agg_comm not null,
text_id int not null,
application_text text null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ues_market_assessment_text.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'ues_market_assessment_text.com_agg_code'

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_market_assessment_text', program_yr, part_id, market_id, com_agg_code,
text_id

GO


PRINT 'TABLE : ues_market_ident_source'
GO


create table ues_market_ident_source (
program_yr typ_progyr not null,
part_id typ_partid not null,
com_agg_code typ_agg_comm not null,
source text null)
GO


PRINT 'BINDING Rule(s) to column'
GO
```

```
execute sp_bindrule rul_program_yr, 'ues_market_ident_source.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'ues_market_ident_source.com_agg_code'

PRINT 'TABLE : ues_market_identification'
GO


create table ues_market_identification (
part_id typ_partid not null,
program_yr typ_progyr not null,
com_agg_code typ_agg_comm not null,
market_id typ_mktcd not null,
curr_val_imports_from_all numeric(11,0) not null,
rate_of_import_growth numeric(5,1) not null,
curr_val_imports_from_us numeric(11,0) not null,
relative_us_mrkt_share numeric(5,2) not null,
abs_us_mrkt_share_start numeric(5,1) not null,
abs_us_mrkt_share_end numeric(5,1) not null,
goal_us_export_growth numeric(5,0) not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ues_market_identification.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault ues_market_com_ag_99531438, 'ues_market_identification.com_agg_code'

PRINT 'TABLE : ues_org_types'
GO


create table ues_org_types (
org_type_id smallint not null,
org_type char(50) not null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_org_types', org_type_id

GO


PRINT 'TABLE : ues_part_contribution'
GO


create table ues_part_contribution (
participant_id typ_partid not null,
program_yr typ_progyr not null,
program_id smallint not null,
cost_category typ_cost_category not null,
type_id int not null,
amount money null)
GO


PRINT 'BINDING Rule(s) to column'
GO
```

```
execute sp_bindrule rul_program_yr, 'ues_part_contribution.program_yr'

GO

PRINT 'TABLE : ues_participant_profile'
GO


create table ues_participant_profile (
part_id char(8) not null,
hdqtrs_addr_1 char(40) null,
hdqtrs_city char(30) null,
hdqtrs_state char(2) null,
hdqtrs_zip_1 char(5) null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_participant_profile', part_id

GO


PRINT 'TABLE : ues_post_countries'
GO


create table ues_post_countries (
activity_country typ_fipcd not null,
reporting_post_city typ_city not null,
reporting_post_country typ_fipcd not null)
GO


PRINT 'TABLE : ues_program_resources'
GO


create table ues_program_resources (
program_yr smallint not null,
program_id smallint not null,
part_id char(8) not null,
requested numeric(11,0) not null,
recommended numeric(11,0) not null,
allocated numeric(11,0) not null,
last_modified_by smallint not null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_program_resources', program_yr, program_id, part_id

GO


PRINT 'TABLE : ues_programs'
GO


create table ues_programs (
program_id smallint not null,
program_name char(30) not null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_programs', program_id

GO
```

```
PRINT 'TABLE : ues_regions'
GO


create table ues_regions (
program_yr smallint not null,
part_id char(8) not null,
region_id char(16) not null,
fips_code char(2) not null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_regions', program_yr, part_id, region_id, fips_code

GO


PRINT 'TABLE : ues_report_programs'
GO


create table ues_report_programs (
activity_program char(1) not null,
program_id smallint not null,
program_name char(30) not null)
GO


PRINT 'TABLE : ues_sf1166_definition'
GO


create table ues_sf1166_definition (
activity_program char(1) not null,
appropriation_yr char(4) not null,
program_code varchar(12) not null)
GO


PRINT 'TABLE : ues_sum_exports_by_year'
GO


create table ues_sum_exports_by_year (
program_yr smallint not null,
part_id char(8) not null,
status char(1) not null,
export_yr smallint not null,
us_exports numeric(12,0) not null,
world_exports numeric(12,0) not null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_sum_exports_by_year', program_yr, part_id, export_yr

GO


PRINT 'TABLE : ues_text_desc'
GO


create table ues_text_desc (
text_id int not null,
text_desc varchar(80) not null,
table_id_is_used_in varchar(40) not null,
related_part_of_system varchar(40) not null)
```

```
GO


PRINT 'TABLE : ues_trade_data_gen_info'
GO


create table ues_trade_data_gen_info (
program_yr smallint not null,
part_id char(8) not null,
period_id int not null,
source text null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_trade_data_gen_info', program_yr, part_id

GO


PRINT 'TABLE : ues_trade_data_periods'
GO


create table ues_trade_data_periods (
period_id int not null,
period_start_month int not null,
period_start_day int not null,
period_end_month int not null,
period_end_day int not null)
GO


PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_trade_data_periods', period_id

GO


PRINT 'TABLE : ues_unfunded_liabilities'
GO


create table ues_unfunded_liabilities (
part_id typ_partid not null,
program_yr typ_progyr not null,
program_id int not null,
unfunded_liabilities_id int not null,
amount int not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ues_unfunded_liabilities.program_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_unfunded_liabilities', program_yr, part_id, program_id,
unfunded_liabilities_id

GO


PRINT 'TABLE : ues_unfunded_liabilities_type'
GO
```

```
create table ues_unfunded_liabilities_type (
unfunded_liabilities_id int not null,
unfunded_liabilities_desc varchar(50) not null)
GO


PRINT 'TABLE : ues_user'
GO


create table ues_user (
user_nbr typ_usernbr not null,
login varchar(40) not null,
first_name varchar(40) null,
last_name varchar(40) null,
user_organization typ_orgcd not null,
user_type_code varchar(10) not null,
user_job_title varchar(20) null,
active_flag typ_swy not null,
lan_id varchar(50) null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_y_n_sw, 'ues_user.active_flag'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_swy, 'ues_user.active_flag'

PRINT 'TABLE : ues_value_added_product'
GO


create table ues_value_added_product (
program_yr typ_progyr not null,
part_id typ_partid not null,
com_agg_code typ_agg_comm not null,
value_added_swy typ_swy not null,
product_desc varchar(255) not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule typ_progyr, 'ues_value_added_product.program_yr'
execute sp_bindrule rul_y_n_sw, 'ues_value_added_product.value_added_swy'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'ues_value_added_product.com_agg_code'
execute sp_bindefault def_swy, 'ues_value_added_product.value_added_swy'

PRINT 'TABLE : ues_world_production'
GO


create table ues_world_production (
program_yr typ_progyr not null,
part_id typ_partid not null,
com_agg_code typ_agg_comm not null,
production_yr int not null,
us_production_vol numeric(12,0) null,
us_production_val numeric(12,0) not null,
```

```
us_exports_to_wld_vol numeric(12,0) null,
us_exports_to_wld_val numeric(12,0) not null,
world_trade_vol numeric(12,0) null,
world_trade_val numeric(12,0) not null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ues_world_production.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'ues_world_production.com_agg_code'

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_world_production', program_yr, part_id, com_agg_code, production_yr

GO


PRINT 'TABLE : ues_world_production_source'
GO


create table ues_world_production_source (
program_yr typ_progyr not null,
part_id typ_partid not null,
com_agg_code typ_agg_comm not null,
volume_unit_code typ_vol_um null,
source text null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'ues_world_production_source.program_yr'

GO

PRINT 'BINDING Default(s) to column'
GO

execute sp_bindefault def_agg_comm, 'ues_world_production_source.com_agg_code'

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_world_production_source', program_yr, part_id, com_agg_code

GO


PRINT 'TABLE : ues_worldwide_personnel'
GO


create table ues_worldwide_personnel (
part_id typ_partid not null,
program_yr typ_progyr not null,
program_id int not null,
worldwide_personnel_id int not null,
amount int not null)
GO


PRINT 'BINDING Rule(s) to column'
GO
```

```
execute sp_bindrule rul_program_yr, 'ues_worldwide_personnel.program_yr'

GO

PRINT 'CREATING Primary Key'
GO

execute sp_primarykey 'ues_worldwide_personnel', program_yr, part_id, program_id,
worldwide_personnel_id

GO


PRINT 'TABLE : ues_worldwide_personnel_type'
GO


create table ues_worldwide_personnel_type (
worldwide_personnel_id int not null,
worldwide_personnel_desc varchar(255) not null)
GO


PRINT 'TABLE : usam_alloc_history'
GO


create table usam_alloc_history (
part_id typ_partid not null,
program_yr typ_progyr not null,
allocation int not null,
adjustment int null,
adjustment2 int null)
GO


PRINT 'BINDING Rule(s) to column'
GO

execute sp_bindrule rul_program_yr, 'usam_alloc_history.program_yr'

GO

PRINT 'TABLE : volume_units'
GO


create table volume_units (
volume_unit_code typ_vol_um not null,
volume_unit_desc char(20) null)
GO




/* USER DEFINED TYPES FOLLOW */

use master
GO

/*
  UserDefinedType(s)
*/

PRINT 'DATA TYPE : typ_activity_code'
GO



execute sp_addtype typ_activity_code, 'nchar(5)', nonull

GO
```

```
PRINT 'DATA TYPE : typ_activity_type'
GO


execute sp_addtype typ_activity_type, 'nchar(15)', nonull

GO


PRINT 'DATA TYPE : typ_addrlin'
GO


execute sp_addtype typ_addrlin, 'nchar(40)', nonull

GO


PRINT 'DATA TYPE : typ_agg_comm'
GO


execute sp_addtype typ_agg_comm, 'nchar(5)', nonull

GO

execute sp_bindefault def_agg_comm, 'typ_agg_comm'

GO


PRINT 'DATA TYPE : typ_approp_yr'
GO


execute sp_addtype typ_approp_yr, 'smallint', nonull

GO

execute sp_bindrule rul_approp_yr, 'typ_approp_yr'

GO


PRINT 'DATA TYPE : typ_brand_gener_flag'
GO


execute sp_addtype typ_brand_gener_flag, 'nchar(7)', nonull

GO

execute sp_bindrule rul_brand_gener_flag, 'typ_brand_gener_flag'

GO


PRINT 'DATA TYPE : typ_ceiling_type'
GO


execute sp_addtype typ_ceiling_type, 'nchar(15)', nonull

GO

execute sp_bindrule rul_ceiling_type, 'typ_ceiling_type'

GO
```

```
PRINT 'DATA TYPE : typ_city'
GO


execute sp_addtype typ_city, 'nchar(30)', nonull

GO


PRINT 'DATA TYPE : typ_claim_status'
GO


execute sp_addtype typ_claim_status, 'nchar(1)', nonull

GO

execute sp_bindrule rul_claim_status, 'typ_claim_status'

GO


PRINT 'DATA TYPE : typ_cntrl_lvl'
GO


execute sp_addtype typ_cntrl_lvl, 'nchar(10)', nonull

GO


PRINT 'DATA TYPE : typ_comment'
GO


execute sp_addtype typ_comment, 'nchar(240)', nonull

GO


PRINT 'DATA TYPE : typ_control_level'
GO


execute sp_addtype typ_control_level, 'nchar(1)', nonull

GO


PRINT 'DATA TYPE : typ_cost_category'
GO


execute sp_addtype typ_cost_category, 'nchar(8)', nonull

GO


PRINT 'DATA TYPE : typ_debit_credit_sw'
GO


execute sp_addtype typ_debit_credit_sw, 'nchar(1)', nonull

GO

execute sp_bindrule rul_debit_credit_sw, 'typ_debit_credit_sw'
```

```
GO


PRINT 'DATA TYPE : typ_description'
GO


execute sp_addtype typ_description, 'nchar(40)', nonull

GO


PRINT 'DATA TYPE : typ_docyr'
GO


execute sp_addtype typ_docyr, 'smallint', nonull

GO

execute sp_bindrule rul_doc_yr, 'typ_docyr'

GO


PRINT 'DATA TYPE : typ_exp_bud_flag'
GO


execute sp_addtype typ_exp_bud_flag, 'nchar(1)', nonull

GO

execute sp_bindrule rul_exp_bud_flag, 'typ_exp_bud_flag'

GO


PRINT 'DATA TYPE : typ_fipcd'
GO


execute sp_addtype typ_fipcd, 'nchar(2)', nonull

GO


PRINT 'DATA TYPE : typ_harmcommcd'
GO


execute sp_addtype typ_harmcommcd, 'nchar(11)', nonull

GO

execute sp_bindrule rul_harmcommcd, 'typ_harmcommcd'

GO


PRINT 'DATA TYPE : typ_liability_sw'
GO


execute sp_addtype typ_liability_sw, 'nchar(1)', nonull

GO
```

```
execute sp_bindrule rul_liability_sw, 'typ_liability_sw'

GO


PRINT 'DATA TYPE : typ_line_status'
GO


execute sp_addtype typ_line_status, 'nchar(1)', nonull

GO

execute sp_bindrule rul_line_status, 'typ_line_status'

GO


PRINT 'DATA TYPE : typ_mkt_sm'
GO


execute sp_addtype typ_mkt_sm, 'nchar(3)', nonull

GO

execute sp_bindrule rul_mrkt_share_measure, 'typ_mkt_sm'

GO


PRINT 'DATA TYPE : typ_mktcd'
GO


execute sp_addtype typ_mktcd, 'nchar(16)', nonull

GO


PRINT 'DATA TYPE : typ_mktmodat'
GO


execute sp_addtype typ_mktmodat, 'datetime', nonull

GO


PRINT 'DATA TYPE : typ_orgcd'
GO


execute sp_addtype typ_orgcd, 'nchar(10)', nonull

GO


PRINT 'DATA TYPE : typ_orgid'
GO


execute sp_addtype typ_orgid, 'nchar(8)', nonull

GO


PRINT 'DATA TYPE : typ_orgtyp'
GO
```

```
execute sp_addtype typ_orgtyp, 'nchar(5)', nonull

GO


PRINT 'DATA TYPE : typ_partid'
GO


execute sp_addtype typ_partid, 'nchar(8)', nonull

GO


PRINT 'DATA TYPE : typ_phonenbr'
GO


execute sp_addtype typ_phonenbr, 'nchar(20)', nonull

GO


PRINT 'DATA TYPE : typ_progtype'
GO


execute sp_addtype typ_progtype, 'nchar(3)', nonull

GO

execute sp_bindrule rul_progtype, 'typ_progtype'

GO


PRINT 'DATA TYPE : typ_progyr'
GO


execute sp_addtype typ_progyr, 'smallint', nonull

GO

execute sp_bindrule rul_program_yr, 'typ_progyr'

GO


PRINT 'DATA TYPE : typ_swn'
GO


execute sp_addtype typ_swn, 'nchar(1)', nonull

GO

execute sp_bindrule rul_y_n_sw, 'typ_swn'

GO

execute sp_bindefault def_swn, 'typ_swn'

GO


PRINT 'DATA TYPE : typ_swy'
```

```
GO


execute sp_addtype typ_swy, 'nchar(1)', nonull

GO

execute sp_bindrule rul_y_n_sw, 'typ_swy'

GO

execute sp_bindefault def_swy, 'typ_swy'

GO


PRINT 'DATA TYPE : typ_trancd'
GO


execute sp_addtype typ_trancd, 'smallint', nonull

GO


PRINT 'DATA TYPE : typ_transaction_nbr'
GO


execute sp_addtype typ_transaction_nbr, 'int', nonull

GO


PRINT 'DATA TYPE : typ_usa_for'
GO


execute sp_addtype typ_usa_for, 'nchar(3)', nonull

GO

execute sp_bindrule rul_usa_for, 'typ_usa_for'

GO


PRINT 'DATA TYPE : typ_usernbr'
GO


execute sp_addtype typ_usernbr, 'smallint', nonull

GO


PRINT 'DATA TYPE : typ_vol_um'
GO


execute sp_addtype typ_vol_um, 'nchar(5)', nonull

GO


PRINT 'DATA TYPE : typ_year'
GO
```

```
execute sp_addtype typ_year, 'smallint', nonull

GO


PRINT 'DATA TYPE : typ_zip1'
GO



execute sp_addtype typ_zip1, 'nchar(5)', nonull

GO


PRINT 'DATA TYPE : typ_zip2'
GO



execute sp_addtype typ_zip2, 'nchar(4)', nonull

GO


/* RULES FOLLOW */

/*
  Rule(s)
*/

PRINT 'RULE : rul_100_percent'
GO

create rule rul_100_percent


--
--CREATE RULES
--
AS @percent between 0 and 100

GO


PRINT 'RULE : rul_act_app_stat'
GO

create rule rul_act_app_stat

AS @act_app_stat in ( 'A', 'D', 'P', 'R' )

GO


PRINT 'RULE : rul_activity_program'
GO

create rule rul_activity_program

AS @activity_program in ("M","F","C","E","N","I","D","W","H","S","G","P","Q","A","X","T")


GO


PRINT 'RULE : rul_activity_progyr'
GO

create rule rul_activity_progyr

AS @activity_progyr like "[0-9][0-9]"

GO
```

```
PRINT 'RULE : rul_activity_type'
GO

create rule rul_activity_type

AS @activity_type in ("G","V","R","B","A","F","C","T","I","Q","M")

GO


PRINT 'RULE : rul_approp_yr'
GO

create rule rul_approp_yr

AS @appropriations_yr > 1900

GO


PRINT 'RULE : rul_brand_gener_flag'
GO

create rule rul_brand_gener_flag

AS @brangenfl in ('BRANDED','GENERIC')

GO


PRINT 'RULE : rul_ceiling_type'
GO

create rule rul_ceiling_type

AS @ceiling_type in (  'ADMINISTRATIVE',
                                            'EVALUATION',
                                            'PROMOTIONAL',
                                            'PUBLICATIONS',
                                            'REG_CONSTRAINT',
                                            'REG_EVALUATION',
                                            'RESEARCH',
                                            'STATE_REGIONAL',
                                            'UNCOMMITTED'
       )

GO


PRINT 'RULE : rul_claim_status'
GO

create rule rul_claim_status

AS @claim_status in ('A','D','I','P','R')

GO


PRINT 'RULE : rul_cnty_stat_flag'
GO

create rule rul_cnty_stat_flag

AS @statflag in ('I','A')

GO


PRINT 'RULE : rul_company_size'
GO

create rule rul_company_size
```

```
AS @company_size in ( 'L', 'S' )

GO


PRINT 'RULE : rul_debit_credit_sw'
GO

create rule rul_debit_credit_sw

AS @debcresw in ('D','C')

GO


PRINT 'RULE : rul_doc_yr'
GO

create rule rul_doc_yr

AS @docyr between 0 and 99

GO


PRINT 'RULE : rul_docstat'
GO

create rule rul_docstat

AS @docstat in ('D','A','P','L','I','S','F','K','R','C')

GO


PRINT 'RULE : rul_exp_bud_flag'
GO

create rule rul_exp_bud_flag

AS @exp_bud_flag in ('B', 'E')

GO


PRINT 'RULE : rul_fas_area_cd'
GO

create rule rul_fas_area_cd

AS @fasareacd in ('I','II','III','IV')

GO


PRINT 'RULE : rul_harmcommcd'
GO

create rule rul_harmcommcd

AS @typ_harmcommcd like
     '[MX][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'
   or @typ_harmcommcd like
     '[B][0-9][0-9][0-9][0-9][0-9][0-9][0-9]   '

GO


PRINT 'RULE : rul_int_month'
GO

create rule rul_int_month

AS @int_month between 1 and 12
```

```
GO


PRINT 'RULE : rul_liability_sw'
GO

create rule rul_liability_sw

AS @liabsw = 'L'

GO


PRINT 'RULE : rul_line_status'
GO

create rule rul_line_status

AS @line_status in ('A','I','R','D','F')

GO


PRINT 'RULE : rul_market_share'
GO

create rule rul_market_share

AS convert(smallint,@mrkshr) between 0 and 100

GO


PRINT 'RULE : rul_mrkt_share_measure'
GO

create rule rul_mrkt_share_measure

AS @mrkt_share_measure in ("VOL", "VAL")

GO


PRINT 'RULE : rul_positive_amount'
GO

create rule rul_positive_amount

AS @amount >= 0

GO


PRINT 'RULE : rul_program_yr'
GO

create rule rul_program_yr

AS @progyr > 1900

GO


PRINT 'RULE : rul_progtype'
GO

create rule rul_progtype
as @progtype in ('MPP', 'FMD', 'EIP', 'QSP', 'EMP', 'TSC')

GO


PRINT 'RULE : rul_req_app_level'
GO
```

```
create rule rul_req_app_level

AS @req_app_level in ( 'AA', 'DIV' )

GO


PRINT 'RULE : rul_stg_of_dev'
GO

create rule rul_stg_of_dev

AS @stgofdev in ('LDC','DC','FORMER USSR/EE','???')

GO


PRINT 'RULE : rul_usa_for'
GO

create rule rul_usa_for

AS @usa_for in ( 'FOR', 'USA' )

GO


PRINT 'RULE : rul_user_orgtype'
GO

create rule rul_user_orgtype

AS @user_orgtype in ("FAS/W","FAS/F","PART","OTHER")

GO


PRINT 'RULE : rul_y_n_sw'
GO

create rule rul_y_n_sw

AS @sw in ("Y", "N")

GO


PRINT 'RULE : typ_progyr'
GO

create rule typ_progyr

AS @typ_progyr >= 1900

GO
```