311-CD-105-005

EOSDIS Core System Project

Release 4
Systems Management Subsystem Database Design and Schema Specifications for the ECS Project

This document has not yet been approved by the Government for general use or distribution.

March 1999

Raytheon Systems Company
Upper Marlboro, Maryland

Release 4 Systems Management Subsystem Database Design and Schema Specifications for the ECS Project

March 1999

Prepared Under Contract NAS5-60000
CDRL Item #050

RESPONSIBLE ENGINEER

Maureen Muganda /s/ 2/26/99
Maureen Muganda Date
EOSDIS Core System Project

SUBMITTED BY

Mary S. Armstrong /s/ 2/26/99
Mary Armstrong, Development Engineering Manager Date
EOSDIS Core System Project

Raytheon Systems Company
Upper Marlboro, Maryland

311-CD-105-005

This page intentionally left blank.

Preface

This document describes the data design and database specification for the Systems Management Subsystem. It is one of eight documents comprising the detailed database design specifications for each of the ECS subsystems.

The subsystem database design specifications for the as delivered system include:

311-CD-102 Data Management (DM) Subsystem Database Design and Database Schema Specifications for the ECS Project

311-CD-103 Ingest Subsystem Database Design and Database Schema Specifications for the ECS Project

311-CD-104 Interoperability Subsystem (IOS) Database Design and Database Schema Specifications for the ECS Project

311-CD-105 Systems Management Subsystem (MSS) Database Design and Database Schema Specifications for the ECS Project

311-CD-106 Planning and Data Processing Subsystem (PDPS) Database Design and Database Schema Specifications for the ECS Project

311-CD-107 Science Data Server (SDSRV) Subsystem Database Design and Database Schema Specifications for the ECS Project

311-CD-108 Storage Management (STMGMT) Subsystem Database Design and Database Schema Specifications for the ECS Project

311-CD-109 Subscription Server (SUBSRV) Subsystem Database Design and Database Schema Specifications for the ECS Project

This submittal meets the milestone specified in the Contract Data Requirements List (CDRL) of NASA Contract NAS5-60000. It is a formal contract deliverable with an approval code 1. It requires Government review and approval prior to acceptance and use. This document is under ECS contractor configuration control. Once approved, contractor approved changes will be handled in accordance with Class I and Class II change control requirements described in the EOS Configuration Management Plan, and changes to this document shall be made by Document Change Notice (DCN) or by complete revision.

Entity Relationship Diagrams (ERDs) presented in this document have been exported directly from tools and some cases contain too much detail to be easily readable within hard copy page constraints. The reader is encouraged to view these drawings on-line using the Portable Document Format (PDF) electronic copy available via the ECS Data Handling System (ECS) on the world-wide web at http://edhs1.gsfc.nasa.gov.

Any questions should be addressed to:

Data Management Office
The ECS Project Office
Raytheon Systems Company
1616 McCormick Drive
Upper Marlboro, Maryland 20774-5301

Abstract

This document outlines “as-built” database design and database schema of the Systems Management Subsystem database including the physical layout of the database and initial installation parameters.

Keywords: data**,** database, design, configuration, database installation, scripts, security, data model, data dictionary, replication, performance tuning, SQL server, database security, replication, database scripts

This page intentionally left blank.

Change Information Page

List of Effective Pages	
Page Number	**Issue**
Title	Submitted as Final
iii through xii	Submitted as Final
1-1 and 1-2	Submitted as Final
2-1 and 2-2	Submitted as Final
3-1 and 3-2	Submitted as Final
4-1 through 4-68	Submitted as Final
5-1 and 5-2	Submitted as Final
6-1 through 6-4	Submitted as Final
7-1 through 7-6	Submitted as Final
8-1 and 8-2	Submitted as Final
A-1 through A-4	Submitted as Final
AB-1 through AB-8	Submitted as Final

Document History			
Document Number	**Status/Issue**	**Publication Date**	**CCR Number**
311-CD-105-001	Draft	January 1998	97-1755
311-CD-105-002	Draft	May 1998	98-0620
311-CD-105-003	Draft	July 1998	98-0866
311-CD-105-004	Draft	December 1998	98-1267
311-CD-105-005	Submitted as Final	March 1999	99-0166

This page intentionally left blank.

Contents

Preface

Abstract

1. Introduction

1.1 Identification 1-1

1.2 Scope 1-1

1.3 Purpose 1-1

1.4 Audience 1-1

2. Related Documentation

2.1 Applicable Documents 2-1

2.2 Information Documents 2-2

3. Database Configurations

3.1 Server Configurations 3-1

3.2 Storage Device Layouts 3-1

4. Data Design

4.1 Database Overview 4-1

4.1.1 Physical Data Model Entity Relationship Diagram 4-1

4.1.2 Tables 4-2

4.1.3 Columns 4-11

4.1.4 Column Domains 4-30

4.1.5 Rules 4-30

4.1.6 Defaults 4-30
4.1.7 Views 4-30
4.1.8 Integrity Constraints 4-30
4.1.9 Triggers 4-31
4.1.10 Stored Procedures 4-52
4.2 Flat File Usage 4-54
4.2.1 File Descriptions 4-54
4.2.2 Block Specifications 4-57
4.2.3 Field Specifications 4-58
4.2.4 Domain Definitions 4-59

5. Performance and Tuning Factors

5.1 Indexes 5-1
5.2 Segments 5-2
5.3 Named Caches 5-2

6. Database Security

6.1 Approach 6-1
6.2 Initial Users 6-1
6.3 Groups 6-2
6.4 Roles 6-2
6.5 Object Permissions 6-3

7. Replication

7.1 Replication Overview 7-1
7.2 Replication Definitions 7-2
7.3 Replication Subscriptions 7-4
7.4 Replication Database Configuration 7-5
7.5 Replication Server Configuration 7-5

8. Scripts

8.1 Installation Scripts 8-1
8.2 De-Installation Scripts 8-1
8.3 Backup/Recovery Scripts 8-1
8.4 Miscellaneous Scripts 8-1

Appendix. MSS ERDs

Abbreviations and Acronyms

List of Figures

4-1 Sample ERD 4-1
6-1 Sybase Approach to SQL Server Security 6-1

List of Tables

4-1 Flat File Descriptions 4-55
4-2 Flat File Block Descriptions 4-57
4-3 Flat File Field Specifications 4-58
4-4 Flat File Domain Definitions 4-59

This page intentionally left blank.

1. Introduction

1.1 Identification

This Systems Management Subsystem (MSS) Database Design and Database Schema Specification document, Contract Data Requirement List (CDRL) Item Number 050, whose requirements are specified in Data Item Description (DID) 311/DV1, is a required deliverable under the Earth Observing System (EOS) Data and Information System (EOSDIS) Core System (ECS), Contract NAS5-60000.

1.2 Scope

The MSS Database Design and Database Schema Specification document describes the data design and database specifications to support the data requirements of Release 4 Drop 4PX MSS software.

1.3 Purpose

The purpose of the MSS Database Design and Database Schema Specification document is to support the maintenance of MSS data and databases throughout the life cycle of ECS. This document communicates the database implementation in sufficient detail to support ongoing configuration management.

1.4 Audience

This document is intended to be used by ECS maintenance and operations staff. The document is organized as follows:

Section 1 provides information regarding the identification, purpose, scope and audience of this document.

Section 2 provides a listing of the related documents, which were used as a source of information for this document.

Section 3 provides a mapping of data bases to hardware components.

Section 4 contains the MSS physical data model which is the database tables, triggers, stored procedures, and flat files.

Section 5 provides a description of database performance and tuning features such as indexes, caches, and data segments.

Section 6 provides a description of the security infrastructure used and list of the users, groups, and permissions available upon initial installation.

Section 7 contains replication design and implementation details.

Section 8 provides a description of database and database related scripts used for installation, de-installation, backup/recovery, and other miscellaneous functions.

2. Related Documents

2.1 Applicable Documents

The following documents, including Internet links, are referenced in this document, or are directly applicable, or contain policies or other directive matters that are binding upon the content of this volume.

305-CD-100	V 2.0 Segment Design Specification for the ECS Project
920-TDG-009	DAAC Hardware Database Mapping/GSFC
920-TDN-009	DAAC Hardware Database Mapping/NSIDC
920-TDE-009	DAAC Hardware Database Mapping/EDC
920-TDL-009	DAAC Hardware Database Mapping/LARC
920-TDS-009	DAAC Hardware Database Mapping/SMC
920-TDG-010	DAAC Database Configuration/GSFC
920-TDN-010	DAAC Database Configuration/NSIDC
920-TDE-010	DAAC Database Configuration/EDC
920-TDL-010	DAAC Database Configuration/LARC
920-TDS-010	DAAC Database Configuration/SMC
920-TDG-011	DAAC Sybase Log Mapping/GSFC
920-TDN-011	DAAC Sybase Log Mapping/NSIDC
920-TDE-011	DAAC Sybase Log Mapping/EDC
920-TDL-011	DAAC Sybase Log Mapping/LARC
920-TDS-011	DAAC Sybase Log Mapping/SMC
922-TDG-013	Disk Partitions/GSFC
922-TDN-013	Disk Partitions/NSIDC
922-TDE-013	Disk Partitions/EDC
922-TDL-013	Disk Partitions/LARC
922-TDS-013	Disk Partitions/SMC

These documents are maintained as part of the ECS baseline and available on the world-wide web at the URL: http://cmdm.east.hitc.com/baseline. Please note that this is a partial mirror site in that some items are not available (they are identified) since this is OPEN to all. This site may also be reached through the EDHS homepage. Scroll page to the connections line and click on the ECS Baseline Information System link.

2.2 Information Documents

The following documents, although not directly applicable, amplify or clarify the information presented in this document. These documents are not binding on this document.

313-CD-006	Release 4 CSMS/SDPS Internal ICD for the ECS Project
609-CD-003	Release 4 Operations Tools Manual for the ECS Project
611-CD-005	Release 4 Mission Operation Procedures for the ECS Project

These documents are accessible via the EDHS homepage.

3. Database Configurations

3.1 Server Configurations

The database configuration of the server varies from DAAC to DAAC, based on individualized DAAC requirements and hardware availability. These DAAC-specific database configurations are detailed on the following documents:

920-TDG-009	DAAC Hardware Database Mapping/GSFC
920-TDN- 009	DAAC Hardware Database Mapping/NSIDC
920-TDE-009	DAAC Hardware Database Mapping/EDC
920-TDL-009	DAAC Hardware Database Mapping/LARC
920-TDS-009	DAAC Hardware Database Mapping/SMC
920-TDG-011	DAAC Sybase Log Mapping/GSFC
920-TDN-011	DAAC Sybase Log Mapping/NSIDC
920-TDE-011	DAAC Sybase Log Mapping/EDC
920-TDL-011	DAAC Sybase Log Mapping/LARC
920-TDS-011	DAAC Sybase Log Mapping/SMC

These documents are maintained as part of the ECS baseline and available on the world-wide web at the URL: http://cmdm.east.hitc.com/baseline. Please note that this is a partial mirror site in that some items are not available (they are identified) since this is OPEN to all. This site may also be reached through the EDHS homepage. Scroll page to the connections line and click on the ECS Baseline Information System link.

3.2 Storage Device Layouts

Storage Device layouts, disk partitions, vary from DAAC to DAAC based on the amount of data storage expected to be needed to accommodate a particular DAAC's storage requirements. Disk partitions for the PDPS server at each DAAC is detailed in the following documents:

922-TDG-013	Disk Partitions/GSFC
922-TDN-013	Disk Partitions/NSIDC
922-TDE-013	Disk Partitions/EDC
922-TDL-013	Disk Partitions/LARC
922-TDS-013	Disk Partitions/SMC

These documents are maintained as part of the ECS baseline and available on the World Wide Web at the URL: http://cmdm.east.hitc.com/baseline. Please note that this is a partial mirror site in that some items are not available (they are identified) since this is OPEN to all. This site may also be reached through the EDHS homepage. Scroll page to the connections line and click on the ECS Baseline Information System link.

4. Data Design

4.1 Database Overview

The MSS database implements the large majority of the persistent data requirements for the MSS subsystem. The database is designed in such a manner as to satisfy business policy while maintaining data integrity and consistency. Database tables are implemented using the Sybase Relational Database Management system (DBMS). All components of the MSS database are described in the section which follow in sufficient detail to support maintenance needs.

4.1.1 Physical Data Model Entity Relationship Diagram

The Entity Relationship Diagram(ERD) presents a schematic depiction of the MSS physical data model. The ERDs presented here for the MSS database were produced using the S-Designor Data Architect Computer Aided Software Engineering (CASE) tool. ERDs represent the relationship between entities or database tables. On ERDs, tables are represented by rectangles and relationships are represented as arrows (see Figure 4-1).

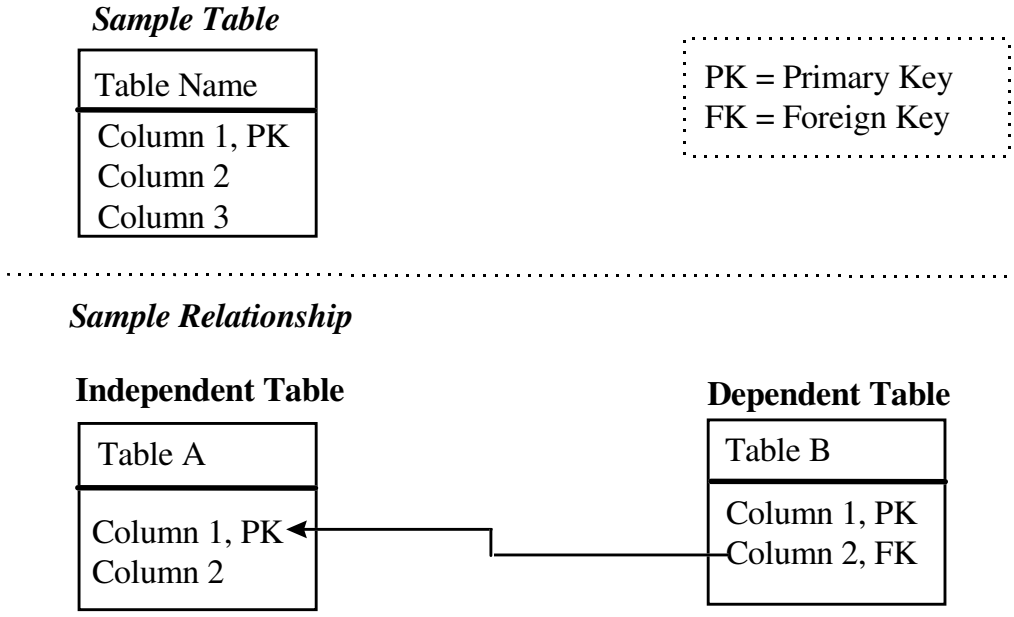

Figure 4-1. Sample ERD

The ERDs for the MSS database are found in the Appendix.

4.1.2 Tables

A listing of each of the tables in the MSS is given here. A brief definition of each of these tables follows including a listing of the columns comprising the table. The Column List indicates if the column is part of the primary key for the table. That is if the columns can be used alone or in combination with other primary key columns to uniquely identify a single row in the table. The column list also indicates whether the column is a mandatory column that must be included in every row.

Table Code
EcAcOrder
EcAcOrderId
EcAcRequest
EcAcRequestId
L_LOCAL_DAAC
MsAcAffiliationCode
MsAcAsterCategory
MsAcDAACCode
MsAcInternetAffiliationCode
MsAcMediaFormatCode
MsAcMediaTypeCode
MsAcPriorityCode
MsAcResearchFieldCode
MsAcStatusCode
MsAcUsrAudit
MsAcUsrProfile
MsAcUsrRequest
MsAcVersions
role_to_cots

Table: EcAcOrder

Description

This table stores the end-user's order information. The data is used to fill and ship the request and to track the status of the order. An order can have many requests, and it may not be associated with a user in the MsAcUsrProfile table. (i.e. the order may ‘belong’ to a guest user) Data is stored indefinitely in the table.

Column List

Name	Code	Type	PK	Mandatory
abortedFlag	abortedFlag	char(1)	No	No
cancelledFlag	cancelledFlag	char(1)	No	No
eMailAddr	eMailAddr	varchar(255)	No	No

Name	Code	Type	PK	Mandatory
finishDateTime	finishDateTime	smalldatetime	No	No
firstName	firstName	varchar(20)	No	No
homeDAAC	homeDAAC	varchar(10)	No	No
lastName	lastName	varchar(20)	No	No
middleInit	middleInit	char(1)	No	No
orderDesc	orderDesc	varchar(50)	No	No
orderDistFormat	orderDistFormat	varchar(64)	No	No
orderGranule	orderGranule	numeric(9)	No	No
orderHomeDAAC	orderHomeDAAC	varchar(10)	No	Yes
orderId	orderId	varchar(10)	No	Yes
orderMedia	orderMedia	varchar(20)	No	No
orderPriority	orderPriority	varchar(10)	No	No
orderSize	orderSize	numeric(9)	No	No
orderSource	orderSource	varchar(12)	No	No
orderStatus	orderStatus	varchar(22)	No	No
receiveDateTime	receiveDateTime	smalldatetime	No	No
shipAddrCity	shipAddrCity	varchar(35)	No	No
shipAddrCountry	shipAddrCountry	varchar(30)	No	No
shipAddrFax	shipAddrFax	varchar(22)	No	No
shipAddrPhone	shipAddrPhone	varchar(22)	No	No
shipAddrState	shipAddrState	varchar(20)	No	No
shipAddrStreet1	shipAddrStreet1	varchar(35)	No	No
shipAddrStreet2	shipAddrStreet2	varchar(35)	No	No
shipAddrZip	shipAddrZip	varchar(15)	No	No
shipDateTime	shipDateTime	smalldatetime	No	No
startDateTime	startDateTime	smalldatetime	No	No
timeOfLastUpdate	timeOfLastUpdate	smalldatetime	No	No
title	title	varchar(5)	No	No
userId	userId	varchar(12)	No	No

Table: EcAcOrderId

Description

This table is used to generate the next orderId for the DAAC identified in the L_LOCAL_DAAC table. At any given time, there must be only one row in this table.

Column List

Name	Code	Type	PK	Mandatory
orderId	orderId	numeric(10)	No	No

Table: EcAcRequest

Description

This table stores the shipping and tracking data for an end-user's request. A user can place one or many requests for a given order.

Column List

Name	Code	Type	PK	Mandatory
destinationDirectory	destinationDirectory	varchar(20)	No	No
destinationNode	destinationNode	varchar(20)	No	No
deviceDensity	deviceDensity	varchar(20)	No	No
deviceId	deviceId	varchar(20)	No	No
eMailAddr	eMailAddr	varchar(255)	No	No
ESDT_Id	ESDT_Id	varchar(20)	No	No
finishDateTime	finishDateTime	smalldatetime	No	No
firstName	firstName	varchar(20)	No	No
ftpAddress	ftpAddress	varchar(128)	No	No
ftpPassword	ftpPassword	varchar(16)	No	No
lastName	lastName	varchar(20)	No	No
mediaType	mediaType	varchar(20)	No	No
middleInit	middleInit	char(1)	No	No
numBytes	numBytes	numeric(9)	No	No
numFiles	numFiles	numeric(9)	No	No
numGranule	numGranule	numeric(9)	No	No
orderHomeDAAC	orderHomeDAAC	varchar(10)	No	Yes
orderId	orderId	varchar(10)	No	Yes
parentId	parentId	varchar(10)	No	No
receiveDateTime	receiveDateTime	smalldatetime	No	No
requestDesc	requestDesc	varchar(50)	No	No
requestDistFormat	requestDistFormat	varchar(64)	No	No
requestHomeDAAC	requestHomeDAAC	varchar(12)	No	Yes
requestId	requestId	varchar(10)	No	Yes
requestPriority	requestPriority	varchar(10)	No	No
requestStatus	requestStatus	varchar(22)	No	No
shipAddrCity	shipAddrCity	varchar(35)	No	No
shipAddrCountry	shipAddrCountry	varchar(30)	No	No
shipAddrFax	shipAddrFax	varchar(22)	No	No
shipAddrPhone	shipAddrPhone	varchar(22)	No	No
shipAddrState	shipAddrState	varchar(20)	No	No
shipAddrStreet1	shipAddrStreet1	varchar(35)	No	No
shipAddrStreet2	shipAddrStreet2	varchar(35)	No	No
shipAddrZip	shipAddrZip	varchar(15)	No	No

Name	Code	Type	PK	Mandatory
shipDateTime	shipDateTime	smalldatetime	No	No
startDateTime	startDateTime	smalldatetime	No	No
tapeFormat	tapeFormat	varchar(20)	No	No
timeOfLastUpdate	timeOfLastUpdate	smalldatetime	No	No
title	title	varchar(5)	No	No

Table: EcAcRequestId

Description

This table is used to generate the next requestId for the DAAC identified in the L_LOCAL_DAAC table. At any given time, there must be only one row in this table.

Column List

Name	Code	Type	PK	Mandatory
requestId	requestId	numeric(10)	No	No

Table: L_LOCAL_DAAC

Description

This table holds the database's DAAC short and long name. This table **must** have only one row, and **must** be the identifier of the DAAC at which the database is installed.

Column List

Name	Code	Type	PK	Mandatory
daac_name	daac_name	varchar(50)	No	Yes
daac_short	daac_short	char(10)	Yes	Yes

Table: MsAcAffiliationCode

Description

This is a lookup table, it defines the list of user affiliations.

Column List

Name	Code	Type	PK	Mandatory
AffiliationCode	AffiliationCode	varchar(16)	Yes	Yes
AffiliationDesc	AffiliationDesc	varchar(255)	No	No

Table: MsAcAsterCategory

Description:

This is a lookup table, it defines the list of Aster categories.

Column List

Name	Code	Type	PK	Mandatory
asterCategory	asterCategory	varchar(40)	No	No
asterCategoryId	asterCategoryId	numeric(2)	Yes	Yes

Table: MsAcDAACCode

Description

This is a lookup table, it list all the DAACs abbreviations and names.

Column List

Name	Code	Type	PK	Mandatory
DAACAbbrv	DAACAbbrv	varchar(3)	Yes	Yes
DAACLongName	DAACLongName	varchar(255)	No	No
DAACShortName	DAACShortName	varchar(10)	No	Yes

Table: MsAcInternetAffiliationCode

Description

This is a lookup table, it list all the internet affiliations.

Column List

Name	Code	Type	PK	Mandatory
InternetAffiliationCode	InternetAffiliationCode	varchar(14)	Yes	Yes
InternetAffiliationDesc	InternetAffiliationDesc	varchar(255)	No	No

Table: MsAcMediaFormatCode

Description

This is a lookup table, it list all the available media format.

Column List

Name	Code	Type	PK	Mandatory
MediaFormatCode	MediaFormatCode	varchar(20)	Yes	Yes
MediaFormatDesc	MediaFormatDesc	varchar(255)	No	No

Table: MsAcMediaTypeCode

Description

This is a lookup table, it list all the media type available.

Column List

Name	Code	Type	PK	Mandatory
MediaTypeCode	MediaTypeCode	varchar(20)	Yes	Yes
MediaTypeDesc	MediaTypeDesc	varchar(255)	No	No

Table: MsAcPriorityCode

Description

This is a lookup table, it defines the list of user request priority levels.

Column List

Name	Code	Type	P	Mandatory
PriorityCode	PriorityCode	varchar(10)	Yes	Yes
PriorityDesc	PriorityDesc	varchar(255)	No	No

Table: MsAcResearchFieldCode

Description

This is a lookup table, it defines the list of user research fields.

Column List

Name	Code	Type	PK	Mandatory
ResearchFieldCode	ResearchFieldCode	varchar(64)	Yes	Yes
ResearchFieldDesc	ResearchFieldDesc	varchar(255)	No	No

Table: MsAcStatusCode

Description

This is a lookup table, it defines the list order statuses.

Column List

Name	Code	Type	PK	Mandatory
StatusCode	StatusCode	varchar(22)	Yes	Yes
StatusDesc	StatusDesc	varchar(255)	No	No

Table: MsAcUsrAudit

Description

TBS

Column List

Name	Code	Type	PK	Mandatory
activityType	activityType	varchar(20)	No	No
DateTime	DateTime	smalldatetime	No	No
hostName	hostName	varchar(30)	No	Yes
location	location	varchar(20)	No	No
program	program	varchar(50)	No	No
status	status	varchar(15)	No	No
userId	userId	varchar(12)	No	Yes

Table: MsAcUsrProfile

Description

This table stores identifying, authenticating, and other data that is used by ECS servers to distribute data to registered users.

Column List

Name	Code	Type	PK	Mandatory
accountNumber	accountNumber	varchar(17)	No	No
affiliation	affiliation	varchar(16)	No	No
asterCategory	asterCategory	numeric(2)	No	No
billAddrCity	billAddrCity	varchar(35)	No	No
billAddrCountry	billAddrCountry	varchar(30)	No	No
billAddrFax	billAddrFax	varchar(22)	No	No
billAddrPhone	billAddrPhone	varchar(22)	No	No
billAddrState	billAddrState	varchar(20)	No	No
billAddrStreet1	billAddrStreet1	varchar(35)	No	No
billAddrStreet2	billAddrStreet2	varchar(35)	No	No
billAddrZip	billAddrZip	varchar(15)	No	No
creationDate	creationDate	smalldatetime	No	No
darExpeditedData	darExpeditedData	bit	No	Yes
ECSAuthenticator	ECSAuthenticator	varchar(32)	No	Yes
eMailAddr	eMailAddr	varchar(255)	No	No
expirationDate	expirationDate	smalldatetime	No	No
firstName	firstName	varchar(20)	No	Yes
GTWYUsrType	GTWYUsrType	varchar(20)	No	No

Name	Code	Type	PK	Mandatory
homeDAAC	homeDAAC	varchar(10)	No	Yes
internetAffiliation	internetAffiliation	varchar(14)	No	No
lastName	lastName	varchar(20)	No	Yes
mailAddrCity	mailAddrCity	varchar(35)	No	No
mailAddrCountry	mailAddrCountry	varchar(30)	No	No
mailAddrFax	mailAddrFax	varchar(22)	No	No
mailAddrPhone	mailAddrPhone	varchar(22)	No	No
mailAddrState	mailAddrState	varchar(20)	No	No
mailAddrStreet1	mailAddrStreet1	varchar(35)	No	No
mailAddrStreet2	mailAddrStreet2	varchar(35)	No	No
mailAddrZip	mailAddrZip	varchar(15)	No	No
middleInit	middleInit	char(1)	No	No
motherMaidenName	motherMaidenName	varchar(20)	No	No
nasaUser	nasaUser	char(1)	No	No
organization	organization	varchar(31)	No	No
privilegeLevel	privilegeLevel	varchar(10)	No	No
projectName	projectName	varchar(30)	No	No
researchField	researchField	varchar(64)	No	No
shipAddrCity	shipAddrCity	varchar(35)	No	No
shipAddrCountry	shipAddrCountry	varchar(30)	No	No
shipAddrFax	shipAddrFax	varchar(22)	No	No
shipAddrPhone	shipAddrPhone	varchar(22)	No	No
shipAddrState	shipAddrState	varchar(20)	No	No
shipAddrStreet1	shipAddrStreet1	varchar(35)	No	No
shipAddrStreet2	shipAddrStreet2	varchar(35)	No	No
shipAddrZip	shipAddrZip	varchar(15)	No	No
telNum	telNum	varchar(22)	No	No
title	title	varchar(5)	No	No
userId	userId	varchar(12)	No	Yes

Table: MsAcUsrRequest

Description

This table stores request for user registration. When a user places a request, the user must be approved first. If a user is approved, the user's data is copy into the MsAcUsrProfile, and deleted from this table, if it is denied ,the user's data remains in this table until it is deleted by the operator.

Column List

Name	Code	Type	PK	Mandatory
accountNumber	accountNumber	varchar(17)	No	No
affiliation	affiliation	varchar(16)	No	No
billAddrCity	billAddrCity	varchar(35)	No	No

Name	Code	Type	PK	Mandatory
billAddrCountry	billAddrCountry	varchar(30)	No	No
billAddrFax	billAddrFax	varchar(22)	No	No
billAddrPhone	billAddrPhone	varchar(22)	No	No
billAddrState	billAddrState	varchar(20)	No	No
billAddrStreet1	billAddrStreet1	varchar(35)	No	No
billAddrStreet2	billAddrStreet2	varchar(35)	No	No
billAddrZip	billAddrZip	varchar(15)	No	No
creationDate	creationDate	smalldatetime	No	No
eMailAddr	eMailAddr	varchar(255)	No	No
expirationDate	expirationDate	smalldatetime	No	No
firstName	firstName	varchar(20)	No	Yes
homeDAAC	homeDAAC	varchar(10)	Yes	Yes
lastName	lastName	varchar(20)	No	Yes
mailAddrCity	mailAddrCity	varchar(35)	No	No
mailAddrCountry	mailAddrCountry	varchar(30)	No	No
mailAddrFax	mailAddrFax	varchar(22)	No	No
mailAddrPhone	mailAddrPhone	varchar(22)	No	No
mailAddrState	mailAddrState	varchar(20)	No	No
mailAddrStreet1	mailAddrStreet1	varchar(35)	No	No
mailAddrStreet2	mailAddrStreet2	varchar(35)	No	No
mailAddrZip	mailAddrZip	varchar(15)	No	No
middleInit	middleInit	char(1)	No	No
motherMaidenName	motherMaidenName	varchar(20)	No	No
nasaUser	nasaUser	char(1)	No	No
organization	organization	varchar(31)	No	No
privilegeLevel	privilegeLevel	varchar(10)	No	No
projectName	projectName	varchar(30)	No	No
researchField	researchField	varchar(64)	No	No
shipAddrCity	shipAddrCity	varchar(35)	No	No
shipAddrCountry	shipAddrCountry	varchar(30)	No	No
shipAddrFax	shipAddrFax	varchar(22)	No	No
shipAddrPhone	shipAddrPhone	varchar(22)	No	No
shipAddrState	shipAddrState	varchar(20)	No	No
shipAddrStreet1	shipAddrStreet1	varchar(35)	No	No
shipAddrStreet2	shipAddrStreet2	varchar(35)	No	No
shipAddrZip	shipAddrZip	varchar(15)	No	No
status	status	varchar(7)	No	No
telNum	telNum	varchar(22)	No	No
title	title	varchar(5)	No	No
userId	userId	varchar(12)	Yes	Yes

Table: MsAcVersions

Description

This table stores ECS Netscape server statistics.

Column List

Name	Code	Type	PK	Mandatory
currentVersion	currentVersion	char(1)	No	Yes
dbInstallDate	dbInstallDate	datetime	No	Yes
mssacctDBName	mssacctDBName	varchar(255)	No	Yes
mssacctDBVersion	mssacctDBVersion	varchar(255)	No	Yes
mssacctServerVersion	mssacctServerVersion	varchar(255)	No	No
mssacctServerVersionInstalled	mssacctServerVersionInstalled	varchar(255)	No	No
schemaVersionID	schemaVersionID	smallint	Yes	Yes

Table: role_to_cots

Description

This defines operator's roles with accessible cots.

Column List

Name	Code	Type	PK	Mandatory
cots_list	cots_list	varchar(36)	No	No
roleID	roleID	varchar(15)	Yes	Yes

4.1.3 Columns

Brief definitions of each of the columns present in the database tables defined above are contained herein.

Column: abortedFlag

Description

This column indicates whether an order has been aborted.

Column: accountNumber

Description

This is the account number that is given by the user when they request a user profile. It becomes associated with the user profile.

Column: activityType

Description

Unknown.

Column: affiliation

Description

This column contains the user's affiliation.

Valid Values:

Gov. Research
Government Other
Univ. Research
Univ. Class Work
Commercial
Kinder.-12 Grade

Column: AffiliationCode

Description

This is the user's affiliation code.

Valid Values:

Gov. Research
Government Other
Univ. Research
Univ. Class Work
Commercial
Kinder.-12 Grade

Column: AffiliationDesc

Description

This column contains the long description of the affiliation code.

Column: asterCategory

Description

MsAcUsrProfile: This column contains the user’s aster category id.

MsAcAsterCategory: The is the description of the aster category id.

Column: asterCategoryId

Description

This column defines an aster category identifier.

Column: billAddrCity

Description

This is the user's city, for billing purposes.

Column: billAddrCountry

Description

This is the user's country, for billing purposes.

Column: billAddrFax

Description

This is the user's fax number, for billing purposes.

Column: billAddrPhone

Description

This is the user's phone number, for billing purposes.

Column: billAddrState

Description

This is the user's state adddress, for billing purposes.

Column: billAddrStreet1

Description

This is the user's street adddress, for billing purposes.

Column: billAddrStreet2

Description

This is the user's street adddress, for billing purposes.

Column: billAddrZip

Description

This is the user's zip code adddress, for billing purposes.

Column: cancelledFlag

Description

This column identifies a the order cancelation status.

Column: cots_list

Description

A list of cots accessible by a roleID.

Column: creationDate

Description

This is the date that the userid was created.

Column: currentVersion

Description

This is the current version of the database schema.

Column: daac_name

Description

This is the DAAC's long name.

Column: daac_short

Description

This is the short home DAAC name.

Column: DAACAbbrv

Description

This is the 3-letters name abbreviation of the DAAC's. (same values a daac_short)

Column: DAACLongName

Description

This is the DAAC's long name. (same values as daac_name)

Column: DAACShortName

Description

This is the short name abbreviation of the DAAC's.

Column: darExpeditedData

Description

This column is false if the user is not allowed to submit DARs that request expedited data. The column is true if the user is allowed to submit DARs that request expedited data

Column: DateTime

Description

User Audit – unknown.

Column: dbInstallDate

Description

This column contains the installation data of the current database schema.

Column: destinationDirectory

Description

This column holds the user's destination directory for ftp acquires.

Column: destinationNode

Description

This column holds the user's destination node for ftp acquires.

Column: deviceDensity

Description

This column holds the request’s device density.

Column: deviceId

Description

This column holds the requests's device ID.

Column: disposition

Description

This table holds the disposition of an advertisement moderation.

Column: disposition_date

Description

This holds the date and time and advertisement disposition.

Column: e_mail

Description

This is the user's email address.

Column: eMailAddr

Description

This is the user's email address.

Column: ESDT_Id

Description

This name will identify the short name associated with the collection or granule.

Column: expirationDate

Description

This is the expiration date of the user’s profile. (i.e. account)

Column: files

Description

This is the number of files.

Column: filesystem

Description

This is the file system identifier.

Column: finishDateTime

Description

EcAcOrder: The column contains the time when all requests for the order have been completed.

EcAcRequest: This column holds the time set by DDIST to the time DDIST finished writing the media for the request (i.e., the request status changed to "waiting for shipment").

Column: firstName

Description

This is ther user's first name.

Column: ftp_browse_count

Description

This is count of ftp browse granules for a user for a given time period.

Column: ftp_bytes

Description

This is the number of bytes that was ftped.

Column: ftp_files

Description

This is the number of files that was ftped.

Column: ftp_granules

Description

This is the number of granules that was ftped.

Column: ftp_order_count

Description

This is the count of acquires sent through ftp for a given user for a given time period.

Column: ftpAddress

Description

This column holds a request's ftp staging.

Column: ftpPassword

Description

This column holds the ftp password for the staging request.

Column: GTWYUsrType

Description

For registered users, the gateway will retrieve their user profile and check this attribute. If is filled, it will use GTWYUsrType and a generated password (V0GwPw) to log the user into DCE (rather than the userID attribute). A DCE account for GTWYUsrType must exist with the current V0GwPwd as its password.

Column: homeDAAC

Description

This is the name of a DAAC, this is where the Request was issued.

Column: internetaffiliation

Description

The column contains the user's internet affiliation.

Column: InternetAffiliationCode

Description

This is the user's internet affiliation.

Column: InternetAffiliationDesc

Description

This column contains a description for an internet affiliation code.

Column: lastName

Description

This column holds the user's last name.

Column: mailAddrCity

Description

This is the user's mailing city address.

Column: mailAddrCountry

Description

This is the user's mailing country address.

Column: mailAddrFax

Description

This is the user's contact fax number.

Column: mailAddrPhone

Description

This is the user's contact phone number.

Column: mailAddrState

Description

This is the user's mailing state address.

Column: mailAddrStreet1

Description

This is the user's mail street address.

Column: mailAddrStreet2

Description

This is the user's mail street address.

Column: mailAddrZip

Description

This is the user's mail zip code address.

Column: media_bytes

Description

This is the number bytes in a media type.

Column: media_files

Description

This is the number of files in a media type.

Column: media_granules

Description

This is the number of granules in a media type.

Column: media_order_count

Description

This is a request type; media order, it holds a count on how many times a user has used this type of request

Column: media_order_count

Description

This is the count of media orders for a given user for a given time period.

Column: media_type

Description

This column holds the media type of request distribution.

Column: MediaFormatCode

Description

This is the type of media format.

Column: MediaFormatDesc

Description

This is the description of the media format.

Column: mediaType

Description

This column holds the media type of request destribution.

Column: MediaTypeCode

Description

This column holds the media type of request destribution.

Column: MediaTypeDesc

Description

This is the description of a media type.

Column: middleInit

Description

This column holds the user's middle name.

Column: motherMaidenName

Description

This is the user's mothers maiden name, for security reasons.

Column: mssacctDBName

Description

The column contains this database’s database name.

Column: mssacctDBVersion

Description

The column contains this database’s schema version identifier.

Column: mssacctServerVersion

Description

The column contains this database’s Sybase Server version id.

Column: mssacctServerVersionInstalled

Description

The column contains this database's Sybase Server version id to which the database should have been installed.

Column: nasaUser

Description

This field identifiers whether a user works for the NASA.

Column: new_state

Description

This column holds the last state of a request.

Column: numBytes

Description

This column contains the number of bytes of a request.

Column: numFiles

Description

This column contains the number of files that fill a request.

Column: numGranule

Description

This column contains the number of granules that fill a request.

Column: ord_orderId

Description

The order identifier is pass from the EcAcOrder.

Column: ord_userId

Description

The user identifier is pass from the EcAcOrder.

Column: orderDesc

Description

This column holds a description of the user's order.

Column: orderDistFormat

Description

This column holds the media format of the user's order.

Valid Values:

tar

Column: orderGranule

Description

This column contains the number of a granules that fill an order.

Column: orderHomeDAAC

Description

This column is pass from EcAcOrder, this is the home DAAC where the order was placed.

Column: orderId

Description

This column is pass from the EcAcOrder table and identiifies an order.

Column: orderMedia

Description

This column holds the media type of the user's order.

Column: orderPriority

Description

This column holds the priority of the user's order.

Column: orderSize

Description

This column holds the size in bytes of the user's order.

Column: orderSource

Description

This column holds the where the source of the order.

Column: orderStatus

Description

This column holds the current status of an order.

Column: organization

Description

This is the user's organization.

Column: parentId

Description

A request can be be broken into subrequests, and this column holds the ID for that request. This is a B1 issue.

Column: permissions

Description

This is the ACL permissions for a logged authentication failure.

Column: priority

Description

This column holds the error priority. This priority will decide how the logged message will be handled.

Column: priority_code

Description

This column holds the priority error code. This priority will decide how the logged message will be handled. The priority code is transulated to an integer code from a priority string by MERGE_HOST_DATA stored procedure.

note: check MERG_HOST_DATA the error code is not being map right.....

Column: PriorityCode

Description

This defines a request priority possible value.

Column: PriorityDesc

Description

This is the description of a request priority code.

Column: privilegeLevel

Description

This column contains the highest priority level a user can give his or her order.

Column: profile_userId

Description

The user identifier is pass from the MsAcUsrProfile table.

Column: program

Description

User Audit – Unknown.

Column: projectName

Description

This is the user's project name.

Column: receiveDateTime

Description

This attibute holds the time the order and/or request was submitted (i.e., created) to the SDSRV, set by the V0 Gateway when it created the EcAcRequest.

Column: req_orderId

Description

The order identifier is pass from the EcAcRequest.

Column: request_id

Description

This column holds the identifier for a request.

Column: request_type

Description

This column identifies the type of request received or the type of request to be triggered by a subscription (e.g., "Notification ftp-pull").

Column: requestDesc

Description

This column holds the resquest's description.

Column: requestDistFormat

Description

This column holds the distribution media format.

Column: requestHomeDAAC

Description

This column holds the home DAAC where the request was placed.

Column: requestId

Description

The requestr identifier is pass from the EcAcRequest.

Column: requestPriority

Description

This column holds the user's request priority.

Column: requestStatus

Description

This column holds the user's request status.

Column: researchField

Description

This is the research field available in the system.

Column: ResearchFieldCode

Description

This is the research field available in the system.

Column: ResearchFieldDesc

Description

This is the research field description.

Column: roleID

Description

The column contains an operator's role.

Column: schemaVersionId

Description

This is the version of the database.

Column: shipAddrCity

Description

This is the user's city address to where the request will be shipped.

Column: shipAddrCountry

Description

This is the user's country address to where the request will be shipped.

Column: shipAddrFax

Description

This is the user's fax number to where the request will be shipped.

Column: shipAddrPhone

Description

This is the user's phone address to where the request will be shipped.

Column: shipAddrState

Description

This is the user's state address to where the request will be shipped.

Column: shipAddrStreet1

Description

This is the user's street address to where the request will be shipped.

Column: shipAddrStreet2

Description

This is the user's street address to where the request will be shipped.

Column: shipAddrZip

Description

This is the user's zip code address to where the request will be shipped.

Column: shipDateTime

Description

This column holds the time the last request for the order was shipped, this time is set by MSS when propagating request status to the order.

Column: st_begin_dt

Description

This column holds the beginning time that an order request was in a particular status (state).

Column: st_end_dt

Description

This column holds the ending time that an order request was in a particular status (state).

Column: startDateTime

Description

This column holds the time set by DDIST to the first time DDIST started to process the request, i.e., start the staging of its data, and the request status changed to "staging".

Column: status

Description

This is the status of a request.

Column: StatusCode

Description

This is the status of a request.

Column: StatusDesc

Description

This is the request status code’s description.

Column: stop_time

Description

This is the time processing of a request ended.

Column: tapeFormat

Description

This column holds the format of the tape for the request.

Column: telNum

Description

This is the user's telephone number.

Column: timeOfLastUpdate

Description

This column holds the time of the last order or request update.

Column: title

Description

This is the title of a user. (i.e., Dr.)

Column: userid

Description

MsAcUsrProfile: This column uniquely identifies a registered user.

All other tables: A registered user id, a guest userid, or an application id.

4.1.4 Column Domains

Domains specify the ranges of values allowed for a given table column. Sybase supports the definition of specific domains to further limit the format of data for a given column. Sybase domains are, in effect, user-defined data types. There are no domains defined for the MSS databases.

4.1.5 Rules

Sybase supports the definitions of rules. Rules provide a means for enforcing domain constraints on a given column. All rules defined in Sybase for the MSS database are described herein.

There are no rules defined in the MSS databases.

4.1.6 Defaults

Defaults are used to supply a value for a column when one is not defined at insert time. All defaults defined in Sybase in the MSS database are described herein.

There are no defaults defined in the MSS databases.

4.1.7 Views

Sybase allows the definition of views as a means of limiting an application or users access to data in a table or tables. Views create a logical table from columns found in one or more tables. All views defined in the MSS databases are described herein.

4.1.8 Integrity Constraints

Sybase allows the enforcement of referential integrity via the use of declarative integrity constraints. Integrity constraints allow the SQL server to enforce primary and foreign key integrity checks without automatically without requiring programming. Sybase 11 is only ANSI-92 compliant, however, therefore its constraints support “restrict-only” operations. This means that a row can not be deleted or updated if their are rows in other tables having a foreign key dependency on that row. Cascade delete and update operations can not be performed if a declarative constraint has been used. There are no declarative integrity constraints defined in the MSS database.

4.1.9 Triggers

Sybase supports the enforcement of business policy via the use of triggers. A trigger is best defined as set of activities or checks that should be performed automatically when ever a row is inserted, updated, or deleted from a given table. Sybase version 11.0.1 allows the definition of insert, update, and delete trigger per table. A listing of each the triggers in the MSS database is given here. A brief definition of each of these triggers follows.

List**Trigger List**

Table	Trigger	Description
EcAcRequest	TrigInsEcAcRequest	Insert EcAcRequest
EcAcRequest	TrigUpdEcAcRequest	Update EcAcRequest
EcAcRequest	TrigDelEcAcRequest	Delete EcAcRequest
MsAcUsrProfile	TrigInsUpdMsAcUsrProfile	Insert MsAcUsrProfile
MsAcUsrProfile	TrigInsUpdMsAcUsrProfile	Update MsAcUsrProfile
MsAcUsrProfile	TrigDelMsAcUsrProfile	Delete MsAcUsrProfile

Trigger: TrigInsEcAcRequest

Trigger Code

```
CREATE TRIGGER TrigInsEcAcRequest
ON EcAcRequest
FOR INSERT as
begin


/* commented out until drop5..
/*
**  Return if no rows affected.
*/
  if @@rowcount = 0
    return

  declare @retval int, @err int
  declare @usertype varchar(20)
  declare @username varchar(30)
  declare @dbname varchar(30)

  select @username = suser_name(), @dbname = db_name()

  exec @retval = RSMaintUserCheck @username, @dbname, @usertype output
  select @err = @@error

  if (@err != 0 or @retval != 0)
  begin
     raiserror 25101 "Error checking for Rep Server maintenance user!"
     return
  end
```

```
 if @usertype = "Rep Server maint"
 begin
   return
 end


/*
** Override the requestHomeDAAC value inserted into this table.
** This will ensure that LOCAL users only insert LOCAL daacs
** into the requestHomeDAAC field.  replication maintenance users
** inserting into this table will not get this far in the trigger.
** Therefore, replication maintenance users will always be able to
** insert rows into this table.
*

 UPDATE EcAcRequest
        set  EcAcRequest.requestHomeDAAC =
               ( select daac_short from L_LOCAL_DAAC)
 from EcAcRequest, inserted
 where EcAcRequest.requestId = inserted.requestId


 declare @insert_cnt     int
 declare @req_user_id    varchar(30)
 declare @req_time       datetime
 declare @req_time_year  int
 declare @req_time_mon   int
 declare @req_time_day   int
 declare @req_time_hour  int
 declare @req_time_min   int

 declare @req_time_yearc varchar(4)
 declare @req_time_monc  varchar(4)
 declare @req_time_dayc  varchar(4)
 declare @req_time_hourc varchar(4)
 declare @req_time_minc  varchar(4)

 declare @time_to_match  datetime
 declare @email          varchar(64)
 declare @esdt           varchar(20)
 declare @media          char(30)
 declare @granules       int
 declare @files          int
 declare @bytes          int
 declare @collection     varchar(30)
 declare @requestId      varchar(10)

 select @requestId = min(requestId)
             from inserted

 while @requestId != null
 BEGIN
```

```
    select @email      = eMailAddr,
           @collection = ESDT_Id,
           @media      = mediaType,
           @req_time   = receiveDateTime
      from inserted
     where requestId = @requestId

    select @req_user_id = Ord.userId
      from inserted ins, EcAcOrder Ord
     where ins.orderId = Ord.orderId
       and ins.requestId = @requestId

    select @req_time_year   = datepart(yy,@req_time)
    select @req_time_mon    = datepart(mm,@req_time)
    select @req_time_day    = datepart(dd,@req_time)
    select @req_time_hour   = datepart(hh,@req_time)
    select @req_time_min    = datepart(mi,@req_time)


    if (@req_time_min  > 30)
      select @req_time_min = 30
    else
      select @req_time_min = 0


    select @req_time_yearc  = convert(varchar(4), @req_time_year)
    select @req_time_monc   = convert(varchar(4), @req_time_mon)
    select @req_time_dayc   = convert(varchar(4), @req_time_day)
    select @req_time_hourc  = convert(varchar(4), @req_time_hour)
    select @req_time_minc   = convert(varchar(4), @req_time_min)


    select @time_to_match =
    convert(datetime, (@req_time_monc + "-" + @req_time_dayc + "-" + @req_time_yearc +
                    " " + @req_time_hourc + ":" + @req_time_minc))


/* Define the "new" record as insert for processing of USER_DAILY_ACCESS_P              */
/* "Insert" defined as this requestid exists in the EcAcRequest table before this record  */
/*     was inserted                                                                      */
/* "Update" defined as all other cases                                                   */



/* Step 2:                                                                               */
/* See if there is a corresponding record in the USER_DAILY_ACCESS_P for this   */
/* User ID and converted time period                                                     */

    IF not exists
    ( select 1
        from USER_DAILY_ACCESS_P
       where @req_user_id = USER_DAILY_ACCESS_P.user_id
```

```
      and @time_to_match = USER_DAILY_ACCESS_P.time_period )
    BEGIN


/* Begin insert of new record into USER_DAILY_ACCESS_P  */
/*  Increment the corresponding count                   */

      if @media = "ftp"
        INSERT  USER_DAILY_ACCESS_P
              (user_id, e_mail, remote_host, collection_name, time_period,
               ftp_order_count, ftp_granules, ftp_files, ftp_bytes)
        /*
        **  Modified 8/8/97 by peter macharrie
        ** receieve_date_time field was populated with time
        ** stamp value that was not on a half hour increment
        ** original statement
        **  VALUES  (@req_user_id, @email, "       ", @collection,  @req_time,
        **         1, 0, 0, 0 )
        */

       VALUES  (@req_user_id, @email, "       ", @collection,  @time_to_match,
              1, 0, 0, 0 )

      else
        INSERT  USER_DAILY_ACCESS_P
              (user_id, e_mail, remote_host, collection_name, time_period,
               media_order_count, media_granules, media_files, media_bytes)
        /*
        **  Modified 8/8/97 by peter macharrie
        ** receieve_date_time field was populated with time
        ** stamp value that was not on a half hour increment
        ** original statement
        **  VALUES  (@req_user_id, @email, "       ", @collection,  @req_time,
        **         1, 0, 0, 0 )
        */

       VALUES  (@req_user_id, @email, "       ", @collection,  @time_to_match,
              1, 0, 0, 0 )
    END



/* BEGIN "update" processing                            */
/* Step 1       : Get selected fields from the inserted record  */

   select @granules = isnull(numGranule,0),
          @files = isnull(numFiles,0),
          @bytes = isnull(numBytes,0),
          @esdt = isnull(ESDT_Id," ")
    from inserted
    where requestId = @requestId

   if @media = "ftp"
```

```
    UPDATE USER_DAILY_ACCESS_P
      SET USER_DAILY_ACCESS_P.ftp_order_count = USER_DAILY_ACCESS_P.ftp_order_count + 1,
          USER_DAILY_ACCESS_P.ftp_granules = USER_DAILY_ACCESS_P.ftp_granules +  @granules,
          USER_DAILY_ACCESS_P.ftp_files = USER_DAILY_ACCESS_P.ftp_files +  @files,
          USER_DAILY_ACCESS_P.ftp_bytes = USER_DAILY_ACCESS_P.ftp_bytes +  @bytes
      FROM USER_DAILY_ACCESS_P
     WHERE USER_DAILY_ACCESS_P.user_id = @req_user_id
       and USER_DAILY_ACCESS_P.time_period = @time_to_match

   else
    UPDATE USER_DAILY_ACCESS_P
      SET USER_DAILY_ACCESS_P. media_order_count = USER_DAILY_ACCESS_P. media_order_count + 1,
          USER_DAILY_ACCESS_P. media_granules = USER_DAILY_ACCESS_P. media_granules +  @granules,
          USER_DAILY_ACCESS_P. media_files = USER_DAILY_ACCESS_P. media_files +  @files,
          USER_DAILY_ACCESS_P. media_bytes = USER_DAILY_ACCESS_P. media_bytes +  @bytes
      FROM USER_DAILY_ACCESS_P
     WHERE USER_DAILY_ACCESS_P.user_id = @req_user_id
       and USER_DAILY_ACCESS_P.time_period = @time_to_match

** End of commented out lines */


   select @requestId = min(requestId)
    from inserted
    where requestId > @requestId
 END



/*
** If a LOCAL user, update the aggregates in the EcAcOrder table.
** Replicate maintenance users will replicate the updates instead
** of recalculating them through the following trigger code.
*/

  declare @requestHomeDAAC varchar(10)
  declare @orderId varchar(10)
  declare @orderHomeDAAC varchar(10)
  declare @rows int

/* update EcAcOrder table aggregate columns */

  select @requestId = min(requestId)
            from inserted

  while @requestId != null
  begin
   select @requestHomeDAAC = min(requestHomeDAAC)
            from inserted
            where requestId = @requestId

   while @requestHomeDAAC != null
   begin
```

```
        select @orderId = orderId,
               @orderHomeDAAC = orderHomeDAAC
          from inserted
         where requestId = @requestId
           and requestHomeDAAC = @requestHomeDAAC

        update EcAcOrder
          set orderGranule = (select sum(numGranule)
                              from EcAcRequest
                              where orderId = @orderId
                                and orderHomeDAAC = @orderHomeDAAC),
              orderSize = (select sum(numBytes)
                              from EcAcRequest
                              where orderId = @orderId
                                and orderHomeDAAC = @orderHomeDAAC)
            from EcAcOrder e1
            where e1.orderId = @orderId
              and e1.orderHomeDAAC = @orderHomeDAAC

        select @rows = @@rowcount, @err = @@error

        if @err != 0
        begin
          print "Error updating EcAcOrder table aggregates."
          rollback transaction
          return
        end

        if @rows > 1
        begin
               print "EcAcRequest insertedTrigger: orderId is not unique."
          rollback transaction
          return
        end

        if @rows = 0
        begin
               print "EcAcRequest inserted Trigger: join to EcAcOrder table failed."
          rollback transaction
          return
        end

      select @requestHomeDAAC = min(requestHomeDAAC)
        from inserted
        where requestId = @requestId
          and requestHomeDAAC > @requestHomeDAAC
    end

    select @requestId = min(requestId)
      from inserted
      where requestId > @requestId
  end
end
```

```
go
```

Trigger: TrigUpdEcAcRequest

Trigger Code

```
create trigger TrigUpdEcAcRequest
on EcAcRequest
for update as
begin


/* commented out lines until drop5…
/*
**  Return if no rows affected.
*/
  IF @@rowcount = 0
     return

  declare @retval int, @err int
  declare @usertype varchar(20)
  declare @username varchar(30)
  declare @dbname varchar(30)

  select @username = suser_name(), @dbname = db_name()

  exec @retval = RSMaintUserCheck @username, @dbname, @usertype output
  select @err = @@error
  if (@err != 0 or @retval != 0)
  begin
    raiserror 25101 "Error checking for Rep Server maintenance user!"
    return
  end

  if @usertype = "Rep Server maint"
  begin
    return
  end


/*
** Override the requestHomeDAAC value inserted into this table.
** This will ensure that LOCAL users only insert LOCAL daacs
** into the requestHomeDAAC field.  replication maintenance users
** inserting into this table will not get this far in the trigger.
** Therefore, replication maintenance users will always be able to
** insert rows into this table.
*/


  UPDATE EcAcRequest
```

```
      set  EcAcRequest.requestHomeDAAC =
           ( select daac_short from L_LOCAL_DAAC)
from EcAcRequest, inserted
where EcAcRequest.requestId = inserted.requestId


declare @insert_cnt     int
declare @req_user_id    varchar(30)
declare @req_time       datetime
declare @req_time_year  int
declare @req_time_mon   int
declare @req_time_day   int
declare @req_time_hour  int
declare @req_time_min   int

declare @req_time_yearc varchar(4)
declare @req_time_monc  varchar(4)
declare @req_time_dayc  varchar(4)
declare @req_time_hourc varchar(4)
declare @req_time_minc  varchar(4)

declare @time_to_match  datetime
declare @email          varchar(64)
declare @esdt           varchar(20)
declare @media          char(30)
declare @granules       int
declare @files          int
declare @bytes          int
declare @collection     varchar(30)

declare @requestId      varchar(10)

select @requestId = min(requestId)
      from inserted

while @requestId != null
BEGIN

  select @email      = eMailAddr,
         @collection = ESDT_Id,
         @media      = mediaType,
         @req_time   = receiveDateTime
    from inserted
   where requestId = @requestId

  select @req_user_id = Ord.userId
    from inserted ins, EcAcOrder Ord
   where ins.orderId = Ord.orderId
     and ins.requestId = @requestId

  select @req_time_year   = datepart(yy,@req_time)
  select @req_time_mon    = datepart(mm,@req_time)
  select @req_time_day    = datepart(dd,@req_time)
```

```
    select @req_time_hour   = datepart(hh,@req_time)
    select @req_time_min    = datepart(mi,@req_time)


    if (@req_time_min  > 30)
      select @req_time_min = 30
    else
      select @req_time_min = 0


    select @req_time_yearc  = convert(varchar(4), @req_time_year)
    select @req_time_monc   = convert(varchar(4), @req_time_mon)
    select @req_time_dayc   = convert(varchar(4), @req_time_day)
    select @req_time_hourc  = convert(varchar(4), @req_time_hour)
    select @req_time_minc   = convert(varchar(4), @req_time_min)


    select @time_to_match =
    convert(datetime, (@req_time_monc + "-" + @req_time_dayc + "-" + @req_time_yearc +
                    " " + @req_time_hourc + ":" + @req_time_minc))


/* Define the "new" record as insert for processing of USER_DAILY_ACCESS_P              */
/* "Insert" defined as this requestid exists in the EcAcRequest table before this record  */
/*     was inserted                                                                      */
/* "Update" defined as all other cases                                                   */



/* Step 2:                                                                               */
/* See if there is a corresponding record in the USER_DAILY_ACCESS_P for this   */
/* User ID and converted time period                                                     */

    IF not exists
    ( select 1
        from USER_DAILY_ACCESS_P
       where @req_user_id = USER_DAILY_ACCESS_P.user_id
         and @time_to_match = USER_DAILY_ACCESS_P.time_period )
      BEGIN


/* Begin insert of new record into USER_DAILY_ACCESS_P  */
/*  Increment the corresponding count                   */

        if @media = "ftp"
           INSERT  USER_DAILY_ACCESS_P
                  (user_id, e_mail, remote_host, collection_name, time_period,
                   ftp_order_count, ftp_granules, ftp_files, ftp_bytes)
         /*
         **  Modified 8/8/97 by peter macharrie
         ** receieve_date_time field was populated with time
         ** stamp value that was not on a half hour increment
         ** original statement
```

```
        **  VALUES  (@req_user_id, @email, "       ", @collection,  @req_time,
        **          1, 0, 0, 0 )
        */


       VALUES  (@req_user_id, @email, "       ", @collection,  @time_to_match,
                1, 0, 0, 0 )

      else
         INSERT  USER_DAILY_ACCESS_P
                (user_id, e_mail, remote_host, collection_name, time_period,
                 media_order_count, media_granules, media_files, media_bytes)

        /*
        **  Modified 8/8/97 by peter macharrie
        ** receieve_date_time field was populated with time
        ** stamp value that was not on a half hour increment
        ** original statement
        **  VALUES  (@req_user_id, @email, "       ", @collection,  @req_time,
        **          1, 0, 0, 0 )
        */
       VALUES  (@req_user_id, @email, "       ", @collection,  @time_to_match,
                1, 0, 0, 0 )
     END



/* BEGIN "update" processing                                    */
/* Step 1       : Get selected fields from the inserted record  */

    select @granules = isnull(numGranule,0),
           @files = isnull(numFiles,0),
           @bytes = isnull(numBytes,0),
           @esdt = isnull(ESDT_Id," ")
      from inserted
     where requestId = @requestId

    if @media = "ftp"
      UPDATE USER_DAILY_ACCESS_P
        SET USER_DAILY_ACCESS_P.ftp_order_count = USER_DAILY_ACCESS_P.ftp_order_count + 1,
            USER_DAILY_ACCESS_P.ftp_granules = USER_DAILY_ACCESS_P.ftp_granules +  @granules,
            USER_DAILY_ACCESS_P.ftp_files = USER_DAILY_ACCESS_P.ftp_files +  @files,
            USER_DAILY_ACCESS_P.ftp_bytes = USER_DAILY_ACCESS_P.ftp_bytes +  @bytes
       FROM USER_DAILY_ACCESS_P
      WHERE USER_DAILY_ACCESS_P.user_id = @req_user_id
        and USER_DAILY_ACCESS_P.time_period = @time_to_match

    else
      UPDATE USER_DAILY_ACCESS_P
        SET USER_DAILY_ACCESS_P. media_order_count = USER_DAILY_ACCESS_P. media_order_count + 1,
            USER_DAILY_ACCESS_P. media_granules = USER_DAILY_ACCESS_P. media_granules +  @granules,
            USER_DAILY_ACCESS_P. media_files = USER_DAILY_ACCESS_P. media_files +  @files,
            USER_DAILY_ACCESS_P. media_bytes = USER_DAILY_ACCESS_P. media_bytes +  @bytes
```

```
       FROM USER_DAILY_ACCESS_P
      WHERE USER_DAILY_ACCESS_P.user_id = @req_user_id
        and USER_DAILY_ACCESS_P.time_period = @time_to_match


    select @requestId = min(requestId)
      from inserted
     where requestId > @requestId
  END

** End of commned out liness */

/*
** If a LOCAL user, update the aggregates in the EcAcOrder table.
** Replicate maintenance users will replicate the updates instead
** of recalculating them through the following trigger code.
*/


  /* if numGranule and numBytes were not updated, then return */

  IF not update(numGranule) and
     not update(numBytes) and
     not update(orderId) and
     not update(orderHomeDAAC)
    return


    /* update EcAcOrder table aggregate columns */

  declare @requestHomeDAAC varchar(10)
  declare @orderId varchar(10)
  declare @orderHomeDAAC varchar(10)
  declare @rows int


  if update (requestId) or
    update (requestHomeDAAC)

  begin
    select @requestId = min(requestId)
          from deleted

    while @requestId != null
    begin

      select @requestHomeDAAC = min(requestHomeDAAC)
            from deleted
            where requestId = @requestId

      while @requestHomeDAAC != null
      begin
```

```
    select @orderId = orderId,
           @orderHomeDAAC = orderHomeDAAC
      from deleted
     where requestId = @requestId
       and requestHomeDAAC = @requestHomeDAAC

    update EcAcOrder
       set orderGranule = (select sum(numGranule)
                             from EcAcRequest
                            where orderId = @orderId
                              and orderHomeDAAC = @orderHomeDAAC),
           orderSize = (select sum(numBytes)
                          from EcAcRequest
                         where orderId = @orderId
                           and orderHomeDAAC = @orderHomeDAAC)
      from EcAcOrder e1
     where e1.orderId = @orderId
       and e1.orderHomeDAAC = @orderHomeDAAC

      select @rows = @@rowcount, @err = @@error
      if @err != 0
      begin
        print "Error updating EcAcOrder table aggregates."
        rollback transaction
        return
      end

      if @rows > 1
      begin
        print "EcAcRequest delete Trigger: orderId is not unique."
        rollback transaction
        return
      end

      if @rows = 0
      begin
        print "EcAcRequest delete Trigger: join to EcAcOrder table failed."
        rollback transaction
        return
      end

      select @requestHomeDAAC = min(requestHomeDAAC)
        from deleted
       where requestId = @requestId
         and requestHomeDAAC > @requestHomeDAAC
    end

    select @requestId = min(requestId)
      from deleted
     where requestId > @requestId
  end
end
```

```
select @requestId = min(requestId)
      from inserted

while @requestId != null
begin
  select @requestHomeDAAC = min(requestHomeDAAC)
    from inserted
   where requestId = @requestId

  while @requestHomeDAAC != null
  begin

     select @orderId = orderId,
           @orderHomeDAAC = orderHomeDAAC
       from inserted
      where requestId = @requestId
        and requestHomeDAAC = @requestHomeDAAC


     update EcAcOrder
       set orderGranule = (select sum(numGranule)
                           from EcAcRequest
                           where orderId = @orderId
                             and orderHomeDAAC = @orderHomeDAAC),
           orderSize = (select sum(numBytes)
                           from EcAcRequest
                           where orderId = @orderId
                             and orderHomeDAAC = @orderHomeDAAC)
         from EcAcOrder e1
         where e1.orderId = @orderId
           and e1.orderHomeDAAC = @orderHomeDAAC

     select @rows = @@rowcount, @err = @@error

     if @err != 0
     begin
       print "Error updating EcAcOrder table aggregates."
       rollback transaction
       return
     end

     if @rows > 1
     begin
           print "EcAcRequest inserted Trigger: orderId is not unique."
       rollback transaction
       return
     end

     if @rows = 0
     begin
           print "EcAcRequest inserted Trigger: join to EcAcOrder table failed."
       rollback transaction
```

```
        return
       end

      select @requestHomeDAAC = min(requestHomeDAAC)
       from inserted
       where requestId = @requestId
        and requestHomeDAAC > @requestHomeDAAC
    end

    select @requestId = min(requestId)
     from inserted
     where requestId > @requestId
  end
end
go
```

Trigger: TrigDelEcAcRequest

Trigger Code

```
create trigger TrigDelEcAcRequest
on EcAcRequest
for
delete
as


/*
**  Return if no rows affected.
*/
IF @@rowcount = 0
  return

/*
**  Check to see if this insert update is being made be replication server
**  maintenance user.  If so, don't bother to do any other checking.
*/
declare @retval int, @err int
declare @usertype varchar(20)
declare @username varchar(30)
declare @dbname varchar(30)

select @username = suser_name(), @dbname = db_name()

exec @retval = RSMaintUserCheck @username, @dbname, @usertype output
select @err = @@error

if (@err != 0 or @retval != 0)
begin
   raiserror 25101 "Error checking for Rep Server maintenance user!"
   return
end
```

```
if @usertype = "Rep Server maint"
begin
  return
end


declare @requestId varchar(10),
            @requestHomeDAAC varchar(10),
        @orderId varchar(10),
        @orderHomeDAAC varchar(10),
            @rows int

/* update EcAcOrder table aggregate columns */

select @requestId = min(requestId)
            from deleted

while @requestId != null
begin

  select @requestHomeDAAC = min(requestHomeDAAC)
            from deleted
            where requestId = @requestId

  while @requestHomeDAAC != null
  begin

      select @orderId = orderId,
            @orderHomeDAAC = orderHomeDAAC
        from deleted
       where requestId = @requestId
         and requestHomeDAAC = @requestHomeDAAC

      update EcAcOrder
        set orderGranule = (select sum(numGranule)
                        from EcAcRequest
                        where orderId = @orderId
                          and orderHomeDAAC = @orderHomeDAAC),
            orderSize = (select sum(numBytes)
                        from EcAcRequest
                        where orderId = @orderId
                          and orderHomeDAAC = @orderHomeDAAC)
          from EcAcOrder e1
          where e1.orderId = @orderId
            and e1.orderHomeDAAC = @orderHomeDAAC

      select @rows = @@rowcount, @err = @@error

      if @err != 0
      begin
        print "Error updating EcAcOrder table aggregates."
        rollback transaction
```

```
        return
      end

      if @rows > 1
      begin
            print "EcAcRequest delete Trigger: orderId is not unique."
        rollback transaction
        return
      end

      if @rows = 0
      begin
            print "EcAcRequest delete Trigger: join to EcAcOrder table failed."
      rollback transaction
      return
      end

   select @requestHomeDAAC = min(requestHomeDAAC)
     from deleted
     where requestId = @requestId
       and requestHomeDAAC > @requestHomeDAAC
  end

  select @requestId = min(requestId)
    from deleted
    where requestId > @requestId
end
go
```

Trigger: TrigInsUpdMsAcUsrProfile

Trigger Code

```
CREATE TRIGGER TrigInsUpdMsAcUsrProfile
ON MsAcUsrProfile
FOR INSERT, UPDATE as
begin

/*
**  Return if no rows affected.
*/
if @@rowcount = 0
  return

/*
** Make sure the homedaac is always for this particular daac
** so that this will not interfere with the replication to smc
*/
declare @retval int, @err int
declare @usertype varchar(20)
declare @username varchar(30)
```

```
declare @dbname varchar(30)

select @username = suser_name(), @dbname = db_name()

exec @retval = RSMaintUserCheck @username, @dbname, @usertype output
select @err = @@error
if (@err != 0 or @retval != 0)
begin
   raiserror 25101 "Error checking for Rep Server maintenance user!"
   return
end

if @usertype = "Rep Server maint"
begin
  return
end

if (select count (*) from inserted
         where homeDAAC not in
         (select daac_short from L_LOCAL_DAAC)) > 0
begin
   declare @daac varchar(10)
   select @daac=daac_short from L_LOCAL_DAAC
   raiserror 25100 "Daac name must be %1!", @daac
   rollback transaction
   return
end
else
begin

if exists (select * from inserted  where eMailAddr is not null)
/* Can only populate where the insert record is valued on eMailAddr  */

BEGIN

/* US Government */

/*  Case 1: the address ends in .gov or .mil */

if  exists ( select right(eMailAddr,4) from inserted
           where right (eMailAddr,4) in (".gov",".mil" ) )
begin
       UPDATE MsAcUsrProfile
         set  MsAcUsrProfile.internetAffiliation = "US Government"
         from MsAcUsrProfile, inserted
         where MsAcUsrProfile.userId = inserted.userId  and
              right(inserted.eMailAddr,4) in (".gov",".mil" )
       return
end

/*  Case 2: the address ends in .us and has .gov. or .mil in string */

if  exists ( select eMailAddr from inserted
```

```
          where right (eMailAddr,3) = '.us' and
               ( charindex(".gov.",eMailAddr) > 0 or
                 charindex(".mil.",eMailAddr) > 0) )
begin
       UPDATE MsAcUsrProfile
         set MsAcUsrProfile.internetAffiliation = "US Government"
         from MsAcUsrProfile, inserted
         where MsAcUsrProfile.userId = inserted.userId  and
              right (inserted.eMailAddr,3) = '.us' and
              ( charindex(".gov.",inserted.eMailAddr) > 0 or
                charindex(".mil.",inserted.eMailAddr) > 0 )
       return
end

/* Educational */

/*  Case 3: the address ends in .edu or .k12 */

if  exists ( select right(eMailAddr,4) from inserted
          where right (eMailAddr,4) in (".edu",".k12" ) )
begin
       UPDATE MsAcUsrProfile
         set MsAcUsrProfile.internetAffiliation = "Educational"
         from MsAcUsrProfile, inserted
         where MsAcUsrProfile.userId = inserted.userId  and
              right(inserted.eMailAddr,4) in (".edu",".k12" )
       return
end

/*  Case 4: the address ends in .us and has .edu. or .k12 in string */

if  exists ( select eMailAddr from inserted
          where right (eMailAddr,3) = '.us' and
               ( charindex(".edu.",eMailAddr) > 0 or
                 charindex(".k12.",eMailAddr) > 0) )
begin
       UPDATE MsAcUsrProfile
         set MsAcUsrProfile.internetAffiliation = "Educational"
         from MsAcUsrProfile, inserted
         where MsAcUsrProfile.userId = inserted.userId  and
             ( charindex(".edu.",inserted.eMailAddr) > 0 or
                        charindex(".k12.",inserted.eMailAddr) > 0 )
       return
end

/*  Commercial */

/*  Case 5: the address ends in .com or .net */

if  exists ( select right(eMailAddr,4) from inserted
          where right (eMailAddr,4) in (".com",".net" ) )
begin
       UPDATE MsAcUsrProfile
```

```
        set MsAcUsrProfile.internetAffiliation = "Commercial"
        from MsAcUsrProfile, inserted
        where MsAcUsrProfile.userId = inserted.userId  and
            right (inserted.eMailAddr,4) in (".com",".net" )
      return
end

/*  Case 6: the address ends in .us and has .com. or .net. in string */

if  exists ( select eMailAddr from inserted
          where right (eMailAddr,3) = '.us' and
              ( charindex(".com.",eMailAddr) > 0 or
                      charindex(".net.",eMailAddr) > 0) )
begin
      UPDATE MsAcUsrProfile
        set MsAcUsrProfile.internetAffiliation = "Commercial"
        from MsAcUsrProfile, inserted
        where MsAcUsrProfile.userId = inserted.userId  and
            right (inserted.eMailAddr,3) = '.us' and
            ( charindex(".com.", inserted.eMailAddr) > 0 or
              charindex(".net.", inserted.eMailAddr) > 0)
      return
end

/* Non-Profit */

/*  Case 7: the address ends in .org */

if  exists ( select right(eMailAddr,4) from inserted
          where right (eMailAddr,4) =  ".org" )
begin
      UPDATE MsAcUsrProfile
        set MsAcUsrProfile.internetAffiliation = "Non-Profit"
        from MsAcUsrProfile, inserted
        where MsAcUsrProfile.userId = inserted.userId  and
            right(inserted.eMailAddr,4) = ".org"
      return
end

/*  Case 8: the address ends in .us and has .com. or .net. in string */

if  exists ( select eMailAddr from inserted
          where right (eMailAddr,3) = '.us' and
              charindex(".org.",eMailAddr) > 0 )
   begin
      UPDATE MsAcUsrProfile
        set MsAcUsrProfile.internetAffiliation = "Non-Profit"
        from MsAcUsrProfile, inserted
        where MsAcUsrProfile.userId = inserted.userId  and
            right (inserted.eMailAddr,3) = '.us' and
                charindex(".org.",inserted.eMailAddr) > 0
      return
   end
```

```
/*  Other USA */

/*  Case 9: the address ends in .us and meets none of the above
    criteria (cases 1-8) */

if  exists ( select eMailAddr from inserted
          where right (eMailAddr,3) = '.us'   and
              charindex(".gov.",eMailAddr) = 0  and
              charindex(".mil.",eMailAddr)  = 0  and
              charindex(".edu.",eMailAddr) = 0  and
              charindex(".k12.",eMailAddr)  = 0 and
              charindex(".com.",eMailAddr) = 0 and
              charindex(".net.",eMailAddr)  = 0  and
              charindex(".org.",eMailAddr)  = 0 )
begin
      UPDATE MsAcUsrProfile
       set MsAcUsrProfile.internetAffiliation = "Other USA"
       from MsAcUsrProfile, inserted
       where MsAcUsrProfile.userId = inserted.userId  and
            right (inserted.eMailAddr,3) = '.us' and
            charindex(".gov.",inserted.eMailAddr)  = 0  and
            charindex(".mil.",inserted.eMailAddr)  = 0  and
            charindex(".edu.",inserted.eMailAddr)  = 0  and
            charindex(".k12.",inserted.eMailAddr)  = 0 and
            charindex(".com.",inserted.eMailAddr) = 0 and
            charindex(".net.",inserted.eMailAddr)   = 0  and
            charindex(".org.",inserted.eMailAddr)   = 0
      return
end

/* Case 10: ends in a period + 3 chars */

if  exists ( select eMailAddr from inserted
          where substring(right(eMailAddr,4),1,1) = '.'   and
              substring(right(eMailAddr,2),2,1) like '[a-z]' and
              substring(right(eMailAddr,2),3,1) like '[a-z]' and
              substring(right(eMailAddr,2),4,1) like '[a-z]' )
begin
      UPDATE MsAcUsrProfile
       set MsAcUsrProfile.internetAffiliation = "Other USA"
       from MsAcUsrProfile, inserted
       where MsAcUsrProfile.userId = inserted.userId  and
            substring(right(inserted.eMailAddr,4),1,1) = '.'   and
            substring(right(inserted.eMailAddr,2),2,1) like '[a-z]' and
            substring(right(inserted.eMailAddr,2),3,1) like '[a-z]' and
            substring(right(inserted.eMailAddr,2),4,1) like '[a-z]'
      return
end

/* Foreign */

/*  Case 11:  ending in decimal plus two chars only (not us) */
```

```
if  exists ( select inserted.eMailAddr from inserted
           where substring(right(inserted.eMailAddr,3),1,1) = '.'   and
                substring(right(inserted.eMailAddr,2),2,2) <> 'us' )
begin
       UPDATE MsAcUsrProfile
         set MsAcUsrProfile.internetAffiliation = "Foreign"
         from MsAcUsrProfile, inserted
         where MsAcUsrProfile.userId = inserted.userId  and
              substring(right(inserted.eMailAddr,3),1,1) = '.'   and
              substring(right(inserted.eMailAddr,2),2,2) <> 'us'
       return
end
end
END

/* end of homedaac ok */
end
go
```

Trigger: TrigDelMsAcUsrProfile

Trigger Code

```
CREATE TRIGGER TrigDelMsAcUsrProfile
ON MsAcUsrProfile
FOR DELETE as
begin

/*
**  Return if no rows affected.
*/
  if @@rowcount = 0
    return

/*
** Make sure the homedaac is always for this particular daac
** so that this will not interfere with the replication to smc
*/
  declare @retval int, @err int
  declare @usertype varchar(20)
  declare @username varchar(30)
  declare @dbname varchar(30)

  select @username = suser_name(), @dbname = db_name()

  exec @retval = RSMaintUserCheck @username, @dbname, @usertype output
  select @err = @@error
  if (@err != 0 or @retval != 0)
  begin
     raiserror 25101 "Error checking for Rep Server maintenance user!"
```

```
    return
  end

  if @usertype = "Rep Server maint"
  begin
    return
  end

  if (select count (*) from deleted
          where homeDAAC not in
          (select daac_short from L_LOCAL_DAAC)) > 0
  begin
     declare @daac varchar(10)
     select @daac=daac_short from L_LOCAL_DAAC
     raiserror 25100 "Daac name must be %1!", @daac
     rollback transaction
     return
  end

end
go
```

4.1.10 Stored Procedures

Sybase also includes support for business policy via the use of stored procedures. Stored procedures are typically used to capture a set of activities or checks that will be performed on the database repeatedly to enforce business policy and maintain data integrity. Stored procedures are parsed and complied SQL code that reside in the database and may be called by name by an application, trigger or another stored procedure A listing of each the stored procedures in the MSS database is given here. A brief definition of each of these stored procedures follows.

List**Procedure List**

Name	Description
ProcIncrementOrderId	Passes back the next order Id to be used as a key for the EcAcOrder.
ProcIncrementEcAcRequestId	Passes the next request Id to be used as a key for the EcAcRequest.

Procedure: ProcIncrementEcAcRequestId

Description

Code

```
CREATE PROCEDURE ProcIncrementEcAcRequestId
          (@RequestIdString varchar(10) output)
AS
BEGIN
     BEGIN TRAN

     DECLARE @rcount int
     SELECT @rcount = count(*) FROM EcAcRequestId
```

```
    IF (@rcount > 1)
    BEGIN
        RAISERROR 93000
        ROLLBACK TRAN
        RETURN 93000
    END
    IF (@rcount = 0)
        INSERT INTO EcAcRequestId VALUES (0)

    UPDATE EcAcRequestId
      SET requestId = requestId + 1

    IF @@error != 0
    BEGIN
        RAISERROR 93000
        ROLLBACK TRAN
        RETURN 93000
    END
SELECT @RequestIdString = convert(varchar(10),MAX(requestId))
     FROM EcAcRequestId

    IF @@error != 0
    BEGIN
        RAISERROR 93000
        ROLLBACK TRAN
        RETURN 93000
    END


    COMMIT TRAN
END
```

Procedure: ProcIncrementOrderId

Description

Code

```
CREATE PROCEDURE ProcIncrementOrderId
        (@OrderIdString varchar(10) output)
AS
BEGIN
    BEGIN TRAN
    DECLARE @rcount int
    SELECT @rcount = count(*) FROM EcAcOrderId
    IF (@rcount > 1)
    BEGIN
        RAISERROR 93001
        ROLLBACK TRAN
        RETURN 93001
    END
    IF (@rcount = 0)
```

```
        INSERT INTO EcAcOrderId VALUES (0)

    UPDATE EcAcOrderId
      SET orderId = orderId + 1

    IF @@error != 0
    BEGIN
        RAISERROR 93001
        ROLLBACK TRAN
        RETURN 93001 rId

    IF @@error != 0
    BEGIN
        RAISERROR 93001
        ROLLBACK TRAN
        RETURN 93001
    END

    COMMIT TRAN
END

 END
```

4.2 Flat File Usage

A flat file is an operating system file that is written and subsequently read, generally independent of other files that exist, and usually static in nature. There are cases when the implementation of persistent data is better suited to a flat file than to a database. MSS Subsystem file usage is detailed in this section via file, block, field, and domain definitions.

4.2.1 File Descriptions

A summary listing of the files in the MSS Subsystem is given in Table 4-1 together with a brief description of the file usage. Many different record formats are used in ECS including ODL, HDF, HDF EOS, block, fixed length, variable length, etc.

Table 4-1. Flat File Descriptions (1 of 3)

File Name	File Type	Record Format	File Description
Accountability component files			
MsAcAffiliation.dat	ASCII	Single line records. One field	Contains the list of valid affiliation names for selection of the user's affiliation.
MsAcAsterCategory.dat	ASCII	Single line records. Two fields.	Contains the list of valid Aster DAR user categories for selection of the user's category.
MsAcCountry.dat	ASCII	Single line records. One field	Contains the list of valid country names for selecting a user's country of residence.
MsAcDceGroup.dat	ASCII	Single line records. One field	Contains DCE account group. (e.g. SCIENTIST, ENGINEER)
MsAcDceOrganization.dat	ASCII	Single line records. One field	DCE account organization. (e.g. NASA)
MsAcGateWayType.dat	ASCII	Single line records. One field	User type. (e.g. DAACOPS, GUEST)
MsAcHomeDAAC.dat	ASCII	Single line records. One field	Contains a list of valid DAAC names for selection of the user's home DAAC.
MsAcNasaUser.dat	ASCII	Single line records. One field	Contains Y (yes) and N (no) to indicate NASA user or not.
MsAcPrimaryAreaStudy.dat	ASCII	Single line records. One field	Contains the list of primary study areas for selecting the user's area.
MsAcPrivilegeLevel.dat	ASCII	Single line records. One field	Contains a list of privilege levels for selecting the user's privilege level. (any other insight into this?)
MsAcState.dat	ASCII	Single line records. One field	Contains the list of states in the USA for selecting a user's state of residence.
MsAcTitle.dat	ASCII	Single line records. One field	Contains the list of valid titles for selecting a user's title.
MsAcType.dat	ASCII	Single line records. One field	Contains USA or NONE to indicate US user or not.

Table 4-1. Flat File Descriptions (2 of 3)

File Name	File Type	Record Format	File Description
MDA Component files			
MsMdMssLogEvents.dat	ASCII	Block: Mult lines with Mult fields	Contains block record that define how to select and how to process MSS log events
MsMdHpovLogEvents.dat	ASCII	Block: Mult lines with Mult fields	Contains block record that define how to select and how to process HPOV log events
MsMdHttpLogEvents.dat	ASCII	Block: Mult lines with Mult fields	Contains block record that define how to select and how to process HTTP log events
MsMdTivLogEvents.dat	ASCII	Block: Mult lines with Mult fields	Contains block record that define how to select and how to process TIVOLI log events
MsMdTableList.dat	ASCII	Block: Mult lines with Mult fields	Contains list of valid MDA database table for MDA GUI archive and retrieval
MsMd_EventCategory.cfg	ASCII	Block: Mult lines with Mult fields	Contains list of valid event categories for MDA GUI
MsMd_EventCsci.cfg	ASCII	Block: Mult lines with Mult fields	Contains list of valid CSCIs for MDA GUI
MsMd_EventMode.cfg	ASCII	Block: Mult lines with Mult fields	Contains list of valid modes for MDA GUI
MsMd_EventPerformance.cfg	ASCII	Block: Mult lines with Mult fields	Contains list of valid performance types for MDA GUI
MsMd_EventPriority.cfg	ASCII	Block: Mult lines with Mult fields	Contains list of valid event priorities for MDA GUI
MsMd_EventType.cfg	ASCII	Block: Mult lines with Mult fields	Contains list of valid event types for MDA GUI
MsMd_EventTuple.cfg	ASCII	Block: Mult lines with Mult fields	Contains list of valid tuple IDs for MDA GUI
Subagent component files			
MsAgEventsHoldingFile	binary	EcAgEvent objects	If subagent is not connected to the deputy agent, the events holding file is used to store events to be processed upon connection to deputy.

Table 4-1. Flat File Descriptions (3 of 3)

File Name	File Type	Record Format	File Description
MsAgBindingVectorFile	binary	MsAgMgmtHandle object	The binding vector file contains binding information to ecs servers such as the uuid and the mode. Whenever a running ecs server is detected by subagent, it adds an entry to the binding vector file. Similarly, when a ecs servers is shutdown or dies for any other reason, its entry is removed from the binding vector. If the subagent is restarted, it uses the information in the binding vector to reconnect with the servers that it was previously monitoring.
MsAgInstanceFile	binary	integers	The instance ID file contains the number that was last assigned to a server by subagent. This number is saved to ensure a unique instance ID across different servers. Whenever subagent needs to assign an instance ID to a server, it reads the value in this file, increments it by one, and then assigns it to the server. The incremented value is written back to the instance ID file.
MsCmActiveModesFile	binary	list of strings	This file contains a list of all the active modes that subagent should discover. The list is maintained by the mode manager GUI.
MsCmAvailableModesFile	binary	list of strings	This file is a master list of all the modes available on all the ecs hosts within a cell. Available modes file is created by each subagent in the cell adding all the installed modes on its host. The purpose of this file is to provide a list of all the modes that can be inserted into the active modes file.

4.2.2 Block Specifications

Table 4-2 identifies the block formats used in MSS files.

Table 4-2. Flat File Block Descriptions (1 of 2)

File Name	Block Name	Block Description
Accountability component file block descriptions		
MsAcAffiliation.dat	(standard)	Single line records; contain 1 field.
MsAcAsterCategory.dat	(standard)	Single line records; contain 2 fields.

Table 4-2. Flat File Block Descriptions (2 of 2)

File Name	Block Name	Block Description
MsAcCountry.dat	(standard)	Single line records; contain 1 field.
MsAcDceGroup.dat	(standard)	Single line records; contain 1 field.
MsAcDceOrganization.dat	(standard)	Single line records; contain 1 field.
MsAcGateWayType.dat	(standard)	Single line records; contain 1 field.
MsAcHomeDAAC.dat	(standard)	Single line records; contain 1 field.
MsAcNasaUser.dat	(standard)	Single line records; contain 1 field.
MsAcPrimaryAreaStudy.dat	(standard)	Single line records; contain 1 field.
MsAcPrivilegeLevel.dat	(standard)	Single line records; contain 1 field.
MsAcState.dat	(standard)	Single line records; contain 1 field.
MsAcTitle.dat	(standard)	Single line records; contain 1 field.
MsAcType.dat	(standard)	Single line records; contain 1 field.

4.2.3 Field Specifications

Brief specifications of the fields present within the MSS Subsystem flat files are contained in Table 4-3. The fields are ordered alphabetically by File Name.

Table 4-3. Flat File Field Specifications

File Name/Block Name	Field Name	Data Type	Field Description
MsAcAffiliation.dat	affiliation	String	Valid affiliation name.
MsAcAsterCategory.dat	Aster category ID	Char 2	Two digit Aster DAR category ID
	Aster category mnemonic	String	Mnemonic corresponding to category ID.
Accountability component field descriptions			
MsAcCountry.dat	country	String	Valid country name.
MsAcDceGroup.dat	DCE group	String	DCE group
MsAcDceOrganization.dat	DCE organization	String	DCE organization
MsAcGateWayType.dat	gateway type	String	Gateway type
MsAcHomeDAAC.dat	home DAAC	String	Valid DAAC name.
MsAcNasaUser.dat	NASA user indication	Char 1	NASA user flag
MsAcPrimaryAreaStudy.dat	primary area of study	String	Valid primary study area
MsAcPrivilegeLevel.dat	privilege level	String	Valid privilege level.
MsAcState.dat	state	String	Valid state within USA.
MsAcTitle.dat	title	String	Valid user title
MsAcType.dat	type	String	User is in USA or not.

4.2.4 Domain Definitions

Domain definitions specify the data type and valid content of fields within a file (e.g., specific values for a limited set of data, ranges of numeric data, units of measure for applicable data). This information is generally used by software to edit incoming data for validity prior to writing or changing data within the file. Use of domain values in updating (adding and changing) records within files preserves the integrity of the data within the file. The domain definitions for the MSS Subsystem are presented in Table 4-4.

Table 4-4. Flat File Domain Definitions

File Name/Block Name	Field Name	Domain Description
Accountability component field domains		
MsAcAffiliation.dat	affiliation	String: K-12 Commercial Government University Other
MsAcAsterCategory.dat	Aster category ID	Integer: 0-99
	Aster category mnemonic	String: (category 0 is not an Aster DAR user) MITI/NASA EOS member IEOS agencies ASTER Science Team Leader US Team Leader ASTER Science Working Groups ASTER Science Team Member AO User Special-Priority Japan user EOS Science Project Office ASTER Science Project (SSSG) ASTER CDS/ESDIS Project ASTER Instrument Team Category 14 Category 15 (through) Category 99

Table 4-4. Flat File Domain Definitions (cont.)

File Name/Block Name	Field Name	Domain Description
MsAcCountry.dat	country	Afghanistan Albania Algeria American Samoa Andorra Angola Anguilla Antarctica Antigua and Barbuda Argentina Armenia Aruba Australia Austria Azerbaijan Bahamas Bahrain Bangladesh Barbados Belarus Belgium Belize Benin Bermuda Bhutan Bolivia Bosnia-Hercegovina Botswana Bouvet Island Brazil British Indian Ocean Territory Brunei Darussalam Bulgaria Burkina Faso Burundi Cambodia Cameroon Canada Cape Verde Cayman Islands Central Africa Republic Chad

Table 4-4. Flat File Domain Definitions (cont.)

File Name/Block Name	Field Name	Domain Description
		Chile China Christmas Island Cocos(Keeling) Islands Colombia Comoros Congo Cook Island Costa Rica Cote d'Ivoire(Ivory Coast) Croatia Cuba Cyprus Czech Republic Denmark Djibouti Dominica Dominican Republic East Timor Ecuador Egypt El Salvador Equatorial Guinea Estonia Ethiopia Falkland Islands Faroe Islands Fiji Finland France French Guiana French Polynesia French Southern Territories Gabon Gambia Georgia Germany Ghana Gibraltar Greece Greenland Grenada

Table 4-4. Flat File Domain Definitions (cont.)

File Name/Block Name	Field Name	Domain Description
		Guadeloupe Guam Guatemala Guinea Guinea-Bissau Guyana Haiti heard and McDonald Islands Honduras Hong Kong Hungary Iceland India Indonesia Iran Iraq Ireland Israel Italy Jamaica Japan Jordan Kazakhstan Kenya Kiribati North Korea South Korea Kuwait Kyrgyzstan Lao People's Democratic Republic Latvia Lebanon Lesotho Liberia Libyan Arab Jamahiriya Liechtenstein Lithuania Luxembourg Macau Madagascar Malawi Malaysia

Table 4-4. Flat File Domain Definitions (cont.)

File Name/Block Name	Field Name	Domain Description
		Maldives Mali Malta Marchall Islands Martinique Mauritania Mauritius Mexico Micronesia Moldovia Monaco Mongolia Montserrat Morocco Mozambique Myanmar Namibia Nauru Nepal Netherlands Netherlands Antilles Neutral Zone New Caledonia New Zealand Nicaragua Niger Nigeria Niue Norfolk Island Northern Mariana Islands Norway Oman Pakistan Palau Panama Papua New Guinea Paraguay Peru Phillippines Pitcairn Island Poland Portugal

Table 4-4. Flat File Domain Definitions (cont.)

File Name/Block Name	Field Name	Domain Description
		Puerto Rico Qatar Reunion Island Romania Russian Federation Rwanda St. Helena St. Kitts and Nevis St. Lucia St. Pierre and Miquelon St. Vincent and the Grenadines Samoa San Marino Sao Tome and Principe Saudi Arabia Senegal Seychelles Sierra Leone Singapore Slovak Republic Slovenia Solomon Islands Somalia South Africa Spain Sri Lanka Sudan Suriname Svalbard and Jan Mayen Islands Swaziland Sweden Switzerland Syrian Arab Republic Taiwan Tajkistan Tanzania Thailand Togo Tokelau Tonga Trinidad and Tobago Tunisia

Table 4-4. Flat File Domain Definitions (cont.)

File Name/Block Name	Field Name	Domain Description
		Turkey Turkmenistan Turks and Caicos Islands Tuvalu Uganda Ukraine United Arab Emirates United Kingdom United States Uruguay Uzbekistan Vanuatu Vatican City Venezuela Vietnam Virgin Islands(British) Virgin Islands(U.S.) Wallis and Fortuna Islands Western Sahara Yemen Yogoslavia(former) Zaire Zambia Zimbabwe
MsAcDceGroup.dat	DCE group	???
MsAcDceOrganization.dat	DCE organization	???
MsAcGateWayType.dat	gateway type	String: DAACOPS ECSDEV V0CERES GUEST
MsAcHomeDAAC.dat	user's home DAAC	String: ASF CSN EDC GSF JPL LAR MSF NSC ORN

Table 4-4. Flat File Domain Definitions (cont.)

File Name/Block Name	Field Name	Domain Description
MsAcNasaUser.dat	NASA user	String: Y N
MsAcPrimaryAreaStudy.dat	user's primary area of study	String: Air-Sea Interaction JPL Atmospheric Aerosols LaRC Biogeochemical Dynamics ORNL Biological Ocenanography JPL Cryospheric Studies NSIDC Geophysics NSIDC Global Biosphere GSFC Human Dimensions of Global Change SEDAAC Hydrologic Cycle GSFC Land Processes EDC Physical Oceanography JPL Polar Processes ASF Radiation Budget LaRC Sea Ice ASF Tropospheric Chemistry LaRC Upper Atmosphere Composition GSFC Upper Atmosphere Dynamics GSFC
MsAcPrivilegeLevel.dat	privilege level	String: XPRESS Vhigh HIGH NORMAL LOW
MsAcState.dat	user's state of residence	Alabama Alaska Arizona Arkansas California Colorado Connecticute Delaware District of Columbia Florida Georgia Hawaii Idaho Illinois Indiana

Table 4-4. Flat File Domain Definitions (cont.)

File Name/Block Name	Field Name	Domain Description
		Iowa Kansas Kentucky Louisana Maine Maryland Massachusetts Michigan Minnesota Mississippi Missouri Montana Nebraska Nevada New Hampshire New Jersey New Mexico New York North Carolina North Dakota Ohio Oklahoma Oregon Pennsylvania South Carolina South Dakota Tennessee Texas Utah Vermont Virginia Washington West Virginia Wisconsin Wyoming
MsAcTitle.dat	user's title	String: Dr Mr Ms Miss Mrs Rev Sr

Table 4-4. Flat File Domain Definitions (cont.)

File Name/Block Name	Field Name	Domain Description
MsAcType.dat	User location	String: USA NONE

5. Performance and Tuning Factors

5.1 Indexes

An index provides a means of locating a row in a table based on the value of specific a columns, without having to scan each row in the table. If used appropriately, indexes can significantly increase data retrieval. Sybase allows the definition of two types of indexes, clustered and non-clustered. In a clustered index, the rows in a tables are physically stored in the sort order determined by the index. Clustered indexes are particularly useful, when the data is frequently retrieved in order. Non-clustered indexes differ from their clustered counterpart, in that data is not physically stored in sort order. Only one clustered index may be defined per table. A list of all the indexes defined against tables in the MSS database is given here along with a description of each index.

Index List

Table Code	Index Code	P	F	U	C
L_LOCAL_DAAC	L_LOCAL_DA_5815771101	Yes	No	Yes	Yes
MsAcDAACCode	PK_MSACDAACCODE	Yes	No	Yes	Yes
MsAcInternetAffiliationCode	PK_MSACINTERNETAFFILIATIONCODE	Yes	No	Yes	Yes
MsAcMediaTypeCode	PK_MSACMEDIATYPECODE	Yes	No	Yes	Yes
MsAcPriorityCode	PK_MSACPRIORITYCODE	Yes	No	Yes	Yes
MsAcResearchFieldCode	PK_MSACRESEARCHFIELDCODE	Yes	No	Yes	Yes
MsAcStatusCode	PK_MSACSTATUSCODE	Yes	No	Yes	Yes
MsDbVersions	PK_MSDBVERSIONS	Yes	No	Yes	Yes
EcAcOrder	EcAcOrderPkIdx EcAcOrderOrderIdIdx EcAcOrderUserIdIdx	Yes No No	No No No	Yes No No	No No No
EcAcRequest	EcAcRequestPkIdx EcAcRequestIdIdx	Yes No	No No	Yes No	No No
MsAcAffiliationCode	PK_MSACAFFILIATIONCODE	Yes	No	Yes	Yes
MsAcAsterCategory	PK_MSACAsterCategory	Yes	No	Yes	Yes
MsAcDAACCode	PK_MSACDAACCODE	Yes	No	Yes	Yes
MsAcInternetAffiliationCode	PK_MSACINTERNETAFFILIATIONCODE	Yes	No	Yes	Yes
MsAcMediaFormatCode	PK_MSACMEDIAFORMATCODE	Yes	No	Yes	Yes
MsAcMediaTypeCode	PK_MSACMEDIATYPECODE	Yes	No	Yes	Yes
MsAcPriorityCode	PK_MSACPRIORITYCODE	Yes	No	Yes	Yes
MsAcResearchFieldCode	PK_MSACRESEARCHFIELDCODE	Yes	No	Yes	Yes
MsAcStatusCode	PK_MSACSTATUSCODE	Yes	No	Yes	Yes

Table Code	Index Code	P	F	U	C
MsAcUsrAudit	MsAcUsrAuditActivityTypeIdx	No	No	No	No
	MsAcUsrAuditDateTimeIdx	No	No	No	No
	MsAcUsrAuditHostNameIdx	No	No	No	No
	MsAcUsrAuditLocationIdx	No	No	No	No
	MsAcUsrAuditProgramIdx	No	No	No	No
	MsAcUsrAuditStatusIdx	No	No	No	No
	MsAcUsrAuditUserIdIdx	No	No	No	No
MsAcUsrProfile	MsAcUsrProfilePkIdx	No	No	Yes	No
	UserProfileAlt_1	No	No	Yes	No
	MsAcUserProfileNameIdx	No	No	No	No
	MsAcUsrProfileUserIdIdx	No	No	No	No
MsAcUsrRequest	MsAcUsrRequestPkIdx	Yes	No	Yes	No
MsAcVersions	PK_MSACVERSIONS	Yes	No	Yes	Yes
role_to_cots	role_to_co_20320102701	Yes	No	Yes	Yes

5.2 Segments

Sybase supports the definition of segments. A segment is a named pointer to a storage device or devices. Segments are used to manually place database objects onto particular storage devices. All database tables and indexes are placed in the default segment. The transaction log is in the log segment segment. System tables are placed in the system segment.

5.3 Named Caches

A cache is a block of memory that is used by Sybase to house data pages that are currently being accessed. A named cache is a named block of memory that the SQL server can use to house frequently accessed tables. Assigning a table to cache causes it to be loaded into memory. This greatly increases performance by eliminating the time expense normally associated with disk i/o. Named caches used in the MSS databases are described herein.

The are no named caches defined for the MSS database.

6. Database Security

6.1 Approach

The database security discussed within this section is bounded to security implementation within the Sybase SQL Server RDBMS. A Sybase general approach to security is adopted as illustrated in Figure 6-1.

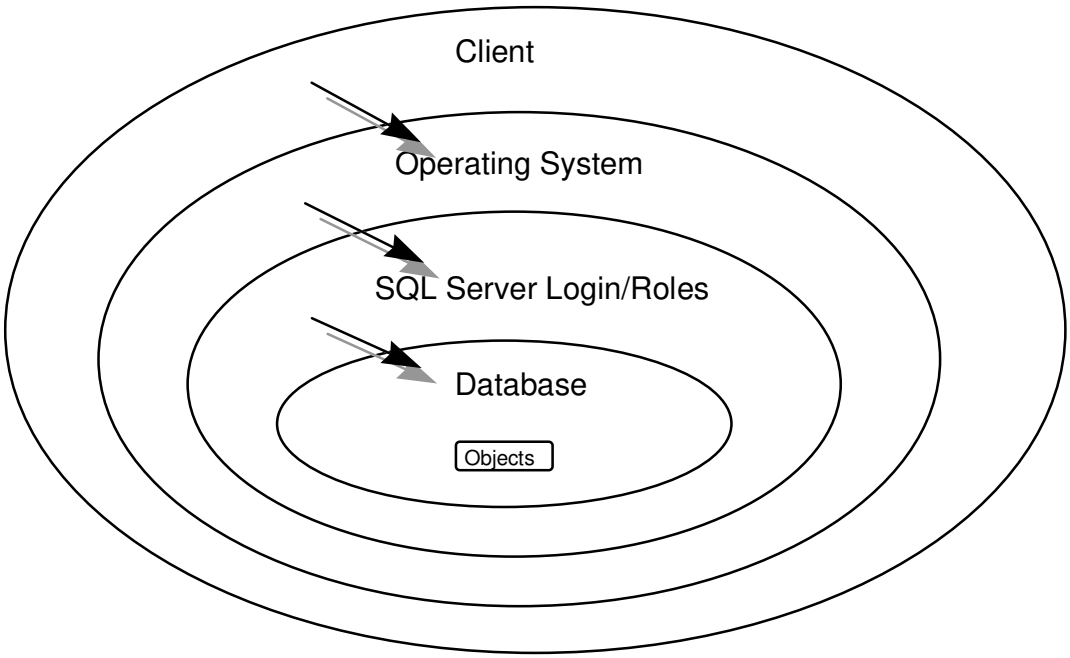

Figure 6-1. Sybase Approach to SQL Server Security

The client/user requires a SQL Server login to access the RDBMS. The System Administrator may grant or revoke various roles to any logins, i.e., *sa, sso, oper*. These roles are identified in Section 6.4. The login needs to be assigned to a user in the database and permissions must be assigned for the user to gain access to the objects within the database. Examples of objects associated with the database include tables, views, commands.

6.2 Initial Users

Upon initial installation the following users will have access to MSS database. The level of access is limited to that associated with their assigned group and/or role. A complete definition of each of these groups and roles is given below.

User
EcAcOrderManager
EcClDtDesktopDaacUser
EcMsAcOrderGUI
EcMsAcOrderSrvr
EcMsAcRegUserGUI
EcMsAcRegUserSrvr
EcMsBaBAASMgr
MsAcManger

6.3 Groups

Groups are a means of logically associating users with similar data access needs. Once a group has been defined, object and command permissions can be granted to that group. A user who is member of a group inherits all of the permissions granted to that group. Several group have been defined in the MSS database upon initial installation. A listing of these groups is contained herein.

Group	Assigned Users
AcctGroup	EcAcOrderManager EcMsAcOrderSrvr EcMsAcRegUserSrvr MaScManager
ClientGroup	EcClDtDesktopDaacUser
MDAGroup	EcMsMdMdaServer (mss_db) MsMdaServer (mss_db)
RepGroup	
public	EcMsAcOrderGUI EcMsAcRegUserGUI EcMsBaBAASMgr

6.4 Roles

Roles were introduced in Sybase 10 to allow a structured means for granting users the permissions needed to perform standard database administration activities and also provide a means for easily identifying such users. There a six pre-defined roles that may be assigned to a user. A definition of each of these roles follows as well as a description of the types of activities that may be performed by each role.

System Administrator (sa_role) - This role is used to grant a specific user to permissions needed to perform standard system administrator duties including:

a. installing SQL server and specific SQL server modules

b. managing the allocation of physical storage

c. tuning configuration parameters

d. creating databases

Site Security Officer (sso_role) - This role is used to grant a specific user the permissions needed to maintain SQL server security including:

a. adding server logins

b. administrating passwords

c. managing the audit system

d. granting users all roles except sa_role

Operator (oper_role) - This role is used to grant a specific user the permissions needed to manage backup and recovery of the database including;

a. dumping transactions and databases

b. loading transactions and databases

Navigator (navigator_role) -This role is used to grant a specific user the permissions needed to manage the navigation server.

Replication (replication_role) - - This role is used to grant a specific user the permissions needed to manage the replication server.

Sybase Technical Support (sybase_ts_role) - This role is used to grant a specific user the permissions needed to perform database consistency checker (dbcc), a sybase supplied utility, commands that are considered outside of the realm of normal system administrator activities.

6.5 Object Permissions

Group	Object	Grant				
Acct Group		Select	Insert	Update	Delete	Execute
	ProcIncrementEcAcRequestId					G
	ProcIncrementOrderId					G
	EcAcOrder	G	G	G	G	
	EcAcRequest	G	G	G	G	
	MsAcAffiliationCode		G	G	G	
	MsAcDAACCode		G	G	G	
	MsAcInternetAffiliationCode		G	G	G	
	MsAcMediaFormatCode		G	G	G	
	MsAcMediaTypeCode		G	G	G	
	MsAcPriorityCode		G	G	G	
	MsAcResearchFieldCode		G	G	G	
	MsAcStatusCode		G	G	G	
	MsAcUsrAudit		G	G	G	
	MsAcUsrProfile	G	G	G	G	

	MsAcUsrRequest	G	G	G	G	
ClientGroup	role_to_cots		G	G	G	
RepGroup	MsAcUsrProfile	G	G	G	G	
public	EcAcOrderId	G				
	EcAcRequestId	G				
	L_LOCAL_DAAC	G				
	MsAcAffiliationCode	G				
	MsAcAsterCategoryCode	G				
	MsAcDAACCode	G				
	MsAcInternetAffiliationCode	G				
	MsAcMediaFormatCode	G				
	MsAcMediaTypeCode	G				
	MsAcPriorityCode	G				
	MsAcResearchFieldCode	G				
	MsAcStatusCode	G				
	MsAcUsrAudit	G				
	role_to_cots	G				

7. Replication

7.1 Replication Overview

Replication as the name implies is a set of Sybase products that allow replication of data from one database to another. The MSS database employs replication to support its data distribution requirements. In order for replication to be accomplished the data source must defined the tables and columns that may be replicated to a data recipient. These definitions are referred to replication definitions. In the same manner a data recipient must specify the replication definitions in which he is interested. These specifications are referred to as replication subscriptions. In addition the replication database and server must be configured to support the potentially large volumes of data that will be transferred between the source and recipient databases. Each of these important parameters is outlined in detail below.

The Replication Definition and Subscription scripts for MSS were developed as templates. These templates will be installed at the DAAC site or SMC. Since peer to peer configuration for MsAcUsrProfile is required, the template approach was decided so that only those scripts that need to be implemented at each site are configured. The template files provide the necessary generic functions needed to configure the MSS MsAcUsrProfile replication environment. Unix shell scripts have been developed to allow installers to pass the appropriate site specific information (i.e. database name, replicate replication server name, etc.) when prompted. The Unix script then customizes the template script files for the specific site. Listed below are the templates for Replication definitions and subscriptions for MSS's MsAcUsrProfile table.

The DD&M group has created naming conventions for all replication scripts and naming conventions for the subscriptions, replication definitions, and other objects related to the ECS replication environment. Script names consist of the following conventions:

<action>.<SUBSYS>.<Replication Object>.<Primary Site ID>.sql.<*Replicate Site ID*>

Action - Replication command to run on a particular object (i.e. drop, alter, check, etc.)
SUBSYS - The CSCI Subsystem being replicated.
Replication Object - The type of replication object being acted upon. (i.e. Subscription, replication definition, etc.)
Primary Site ID - The site ID identified as the Primary site for this particular subsystem's data. This field is named "PRIME" for the template scripts and is changed via a Unix install script during installation and configuration activities at the site.
sql - convention for identifying that this script is an SQL (Replication) script.
Replicate Site ID - This field does not apply to every script convention. This field is defined on scripts that act upon replication subscription objects. The "REP" field name for the template script is changed during installation and configuration at the site. This suffix identifies the replicate site to which the data will be replicated.

7.2 Replication Definitions

Replication definitions that have been defined against MSS tables and columns are detailed herein.

create.mss.repdefs.sql.PRIME

```
create replication definition <pSite>_MsAcUsrProfile_rd
        with primary at <pDs>.<pDb>
        with all tables named 'MsAcUsrProfile'
                (userId varchar(12),
                 homeDAAC varchar(10),
                 title varchar(5),
                 firstName varchar(20),
                 middleInit varchar(1),
                 lastName varchar(20),
                 motherMaidenName varchar(20),
                 telNum varchar(22),
                 ECSAuthenticator varchar(32),
                 GTWYUsrType varchar(20),
                 eMailAddr varchar(255),
                 internetAffiliation varchar(14),
                 organization varchar(31),
                 projectName varchar(30),
                 affiliation varchar(16),
                 researchField varchar(64),
                 accountNumber varchar(17),
                 privilegeLevel varchar(10),
                 creationDate datetime,
                 expirationDate datetime,
                 mailAddrStreet1 varchar(35),
                 mailAddrStreet2 varchar(35),
                 mailAddrCity varchar(35),
                 mailAddrState varchar(20),
                 mailAddrZip varchar(15),
                 mailAddrCountry varchar(30),
                 mailAddrPhone varchar(22),
                 mailAddrFax varchar(22),
                 billAddrStreet1 varchar(35),
                 billAddrStreet2 varchar(35),
                 billAddrCity varchar(35),
                 billAddrState varchar(20),
                 billAddrZip varchar(15),
                 billAddrCountry varchar(30),
                 billAddrPhone varchar(22),
                 billAddrFax varchar(22),
                 shipAddrStreet1 varchar(35),
                 shipAddrStreet2 varchar(35),
                 shipAddrCity varchar(35),
                 shipAddrState varchar(20),
                 shipAddrZip varchar(15),
                 shipAddrCountry varchar(30),
                 shipAddrPhone varchar(22),
```

```
                    shipAddrFax varchar(22),
                    asterCategory numeric,
                    darExpeditedData bit,
                    nasaUser varchar(1))
        primary key (homeDAAC, userId)
        searchable columns (homeDAAC)
go
create replication definition <pSite>_EcAcRequest_rd
        with primary at <pDs>.<pDb>
        with all tables named 'EcAcRequest'
                (orderId varchar(10),
                 orderHomeDAAC varchar(10),
                 requestId varchar(10),
                 requestHomeDAAC varchar(12),
                 parentId varchar(10),
                 title varchar(5),
                 firstName varchar(20),
                 middleInit varchar(1),
                 lastName varchar(20),
                 eMailAddr varchar(255),
                 requestDesc varchar(50),
                 requestStatus varchar(22),
                 requestDistFormat varchar(64),
                 numFiles numeric,
                 numBytes numeric,
                 numGranule numeric,
                 deviceId varchar(20),
                 deviceDensity varchar(20),
                 tapeFormat varchar(20),
                 mediaType varchar(20),
                 ESDT_Id varchar(20),
                 requestPriority varchar(10),
                 shipAddrStreet1 varchar(35),
                 shipAddrStreet2 varchar(35),
                 shipAddrCity varchar(35),
                 shipAddrState varchar(20),
                 shipAddrZip varchar(15),
                 shipAddrCountry varchar(30),
                 shipAddrPhone varchar(22),
                 shipAddrFax varchar(22),
                 receiveDateTime datetime,
                 startDateTime datetime,
                 finishDateTime datetime,
                 timeOfLastUpdate datetime,
                 shipDateTime datetime,
                 ftpAddress varchar(128),
                 ftpPassword varchar(16),
                 destinationNode varchar(20),
                 destinationDirectory varchar(20))
        primary key (requestHomeDAAC, requestId)
        searchable columns (requestHomeDAAC)
go
create replication definition <pSite>_EcAcOrder_rd
```

```
        with primary at <pDs>.<pDb>
        with all tables named 'EcAcOrder'
                (orderId varchar(10),
                 orderHomeDAAC varchar(10),
                 userId varchar(12),
                 homeDAAC varchar(10),
                 title varchar(5),
                 firstName varchar(20),
                 middleInit varchar(1),
                 lastName varchar(20),
                 eMailAddr varchar(255),
                 orderSource varchar(12),
                 orderStatus varchar(22),
                 orderDesc varchar(50),
                 orderDistFormat varchar(64),
                 orderMedia varchar(20),
                 orderSize numeric,
                 orderGranule numeric,
                 orderPriority varchar(10),
                 shipAddrStreet1 varchar(35),
                 shipAddrStreet2 varchar(35),
                 shipAddrCity varchar(35),
                 shipAddrState varchar(20),
                 shipAddrZip varchar(15),
                 shipAddrCountry varchar(30),
                 shipAddrPhone varchar(22),
                 shipAddrFax varchar(22),
                 receiveDateTime datetime,
                 startDateTime datetime,
                 finishDateTime datetime,
                 timeOfLastUpdate datetime,
                 shipDateTime datetime,
                 cancelledFlag varchar(1),
                 abortedFlag varchar(1))
        primary key (orderHomeDAAC, orderId)
        searchable columns (orderHomeDAAC)
go
```

7.3 Replication Subscriptions

Replication subscriptions that have been defined against MSS tables and columns are detailed herein.

define.mss.subs.PRIME.sql.REP

```
/* ============================================================ */
/*   DAAC Table : MsAcUsrProfile                                */
/* ============================================================ */

define subscription <pSite>2<rSite>_MsAcUsrProfile_sub
for <pSite>_MsAcUsrProfile_rd
with replicate at <rDs>.<rDb>
go
```

7.4 Replication Database Configuration

Replication Database Configuration specifications applicable to MSS replication are contained herein.

MSS - Uses the following parameters:

dsi_keep_triggers (This parameter enables or disables the
ability for replicated transactions to execute triggers at the replicated database. (Off for MSS)
dsi_replication (This parameter enables or disables the
ability to replicate transactions executed by the maintenance user.
(Off for MSS)

7.5 Replication Server Configuration

Replication Server Configuration specifications applicable to MSS replication are contained herein.

Configure the number of threads "num_threads" to 70 from the default of 50. This allows more open server connections which are necessary when support a large number of LTMs on the same host as the replication server instance resides.

This page intentionally left blank.

8. Scripts

8.1 Installation Scripts

Any scripts used to support installation of the MSS database are described herein

Script File	File Location	Description
EcMsDbBuild	/ecs/formal/MSS/ddm_mss_support/scripts	Installs MSS DAAC databases.
EcMsDbPatch	/ecs/formal/MSS/ddm_mss_support/scripts	Runs MSS patches between drops.
Install_mss_permissions	/ecs/formal/MSS/ddm_mss_support/users	Installs MSS user's permission.

8.2 De-Installation Scripts

Any scripts used to support de-installation of the MSS database are described herein.

EcMsDbDrop	/ecs/formal/MSS/ddm_mss_support/scripts	Drops MSS databases.

8.3 Backup/Recovery Scripts

Any scripts used to facilitate backup or recovery of the MSS database are described herein.

EcMsDbDump	/ecs/formal/MSS/ddm_mss_support/scripts	Dumps MSS databases.
EcMsDbLoad	/ecs/formal/MSS/ddm_mss_support/scripts	Loads MSS databases.
EcMsDbDumpTran	/ecs/formal/MSS/ddm_mss_support	Dumps MSS transactions.
EcMsDbLoadTran	/ecs/formal/MSS/ddm_mss_support	Loads MSS transactions.

8.4 Miscellaneous Scripts

Miscellaneous scripts applicable to the MSS database are described herein.

None.

This page intentionally left blank.

Appendix. MSS ERDs

This page intentionally left blank.

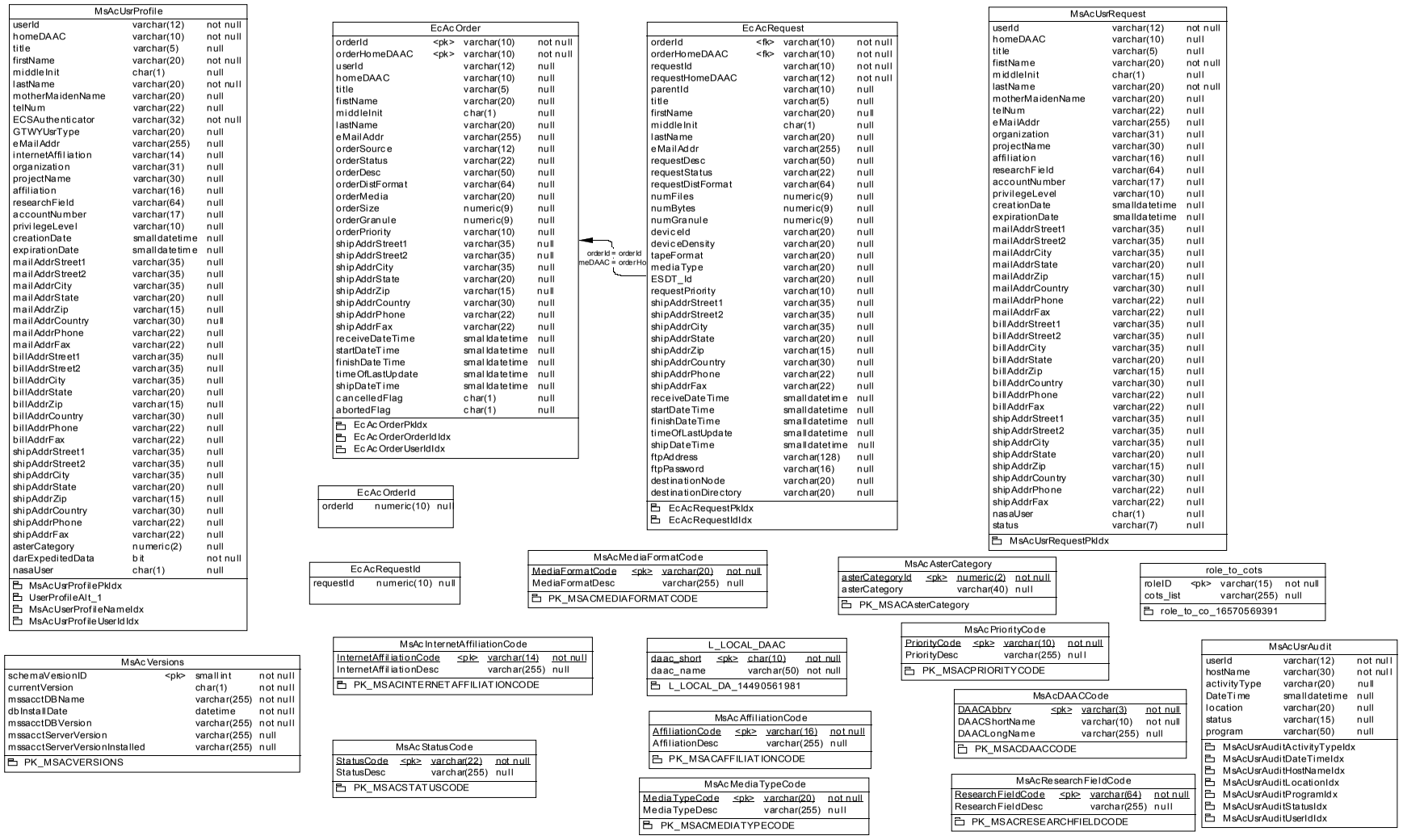

Figure A-1. Accountability Database

Abbreviations and Acronyms

ACL	Access Control List
ACMHW	Access and Control Management HWCI
ADC	affiliated data center
ADSHW	Advertising Server HWCI
ADSRV	Advertising Service CSCI
AI&T	algorithm integration and test
AITHW	Algorithm Integration and Test HWCI
AITTL	Algorithm Integration and Test CSCI
AM-1	EOS AM Project spacecraft 1, morning spacecraft series -- ASTER, CERES, MISR, MODIS and MOPITT instruments
ANSI	American National Standards Institute
API	application program (or programming) interface
APID	application's process ID
AQAHW	Algorithm QA HWCI
ASCII	American Standard Code for Information Exchange
ASTER	Advanced Spaceborne Thermal Emission and Reflection Radiometer (formerly ITIR)
AVHRR	Advanced Very High-Resolution Radiometer
BER	bit error rate
BUFR	binary universal format for representation of data
CASE	Computer Aided Software Engineering
CCSDS	Consultative Committee for Space Data Systems
CD	contractual delivery 214-001
CD-ROM	compact disk -- read only memory
CDR	Critical Design Review
CDRL	contract data requirements list
CERES	Clouds and Earth's Radiant Energy System
CI	configuration item

COTS	commercial off-the-shelf (hardware or software)
CPU	central processing unit
CSCI	computer software configuration item
CSDT	Computer Science Data Type
CSMS	Communications and Systems Management Segment (ECS)
CSS	Communications Subsystem
DAAC	Distributed Active Archive Center
DAN	data availability notice
DAO	Data Assimilation Office
DAR	data acquisition request
DAS	data availability schedule
DBMS	Database Management System
DDICT	Data Dictionary CSCI
DDIST	Data Distribution Services CSCI
DDSRV	Document Data Server CSCI
DESKT	Desktop CSCI
DID	data item description
DIM	distributed information manager (SDPS)
DIMGR	Distributed Information Manager CSCI
DIPHW	Distribution and Ingest Peripheral Management HWCI
DMGHW	Data Management HWCI
DMS	Data Management Subsystem
DMWG	Data Management Working Group
DP	Data Provider
DPR	data processing request
DPREP	Science Data Preprocessing CSCI
DPS	Data Processing Subsystem
DRPHW	Data Repository HWCI
DSS	Data Server Subsystem
ECS	EOSDIS Core System

EDC	EROS Data Center
EDHS	ECS Data Handling System
EDOS	EOS Data and Operations System
EOS	Earth Observing System
EOS-AM	EOS Morning Crossing (Descending) Mission -- see AM-1
EOSDIS	Earth Observing System Data and Information System
EROS	Earth Resources Observation System
ESDIS	Earth Science Data and Information System (GSFC)
ESDT	Earth science data types
ESN	EOSDIS Science Network (ECS)
FDDI	fiber distributed data interface
FDF	flight dynamics facility
FDFEPHEM	FDF-generated definitive orbit data
FGDC	Federal Geographic Data Commuittee
FK	Foreign Key
FOO	Flight of Opportunity
FOS	Flight Operations Segment (ECS)
GB	gigabyte (10^9)
GNU	(recursive acronym: "GNU's Not Unix"); a project supported by the Free Software Foundation dedicated to the delivery of free software
GPCP	Global Precipitation Climatology Project
GPCP	Global Precipitation Climatology Project
GPI	GOES Precipitation Index
GRIB	GRid In Binary
GSFC	Goddard Space Flight Center
GTWAY	Version 0 Interoperability Gateway CSCI
GUI	graphic user interface
GV	ground validation
HDF	hierarchical data format
HDF-EOS	an EOS proposed standard for a specialized HDF data format

HIPPI	high performance parallel interface
HMI	human machine interface
HTML	HyperText Markup Language
HTTP	Hypertext Transport Protocol
HWCI	hardware configuration item
I&T	integration and test
I/F	interface
I/O	input/output
ICD	interface control document
ICLHW	Ingest Client HWCI
ID	identification
IDE	Interactive Development Environments
IDG	Infrastructure Development Group
IDR	Incremental Design Review
IERS	International Earth Rotation Service
IMS	Information Management System (obsolete ECS element name)
INGST	Ingest Services CSCI
IOS	Interoperability Subsystem
IP	international partners
IR-1	Interim Release 1
IRD	interface requirements document
ISO	International Standards Organization
ISS	Internetworking Subsystem
IV&V	independent verification and validation
JPL	Jet Propulsion Laboratory
L0-L4	Level 0 (zero) through Level 4
LaRC	Langley Research Center (DAAC)
LIM	local information manager (SDPS)
LIMGR	Local Information Manager CSCI
LIS	Lightning Imaging Sensor

LSM	local system management (ECS)
MB	megabyte (10^6)
MDT	mean downtime
MDT	mean downtime
MFLOPS	mega (millions of) floating-point operations (10^6) per second
MISR	Multi-Angle Imaging SpectroRadiometer
MODIS	Moderate-Resolution Imaging Spectrometer
MOPITT	Measurements of Pollution in the Troposphere
MSFC	Marshall Space Flight Center
MSS	Systems ManagementSubsystem
MTBF	mean time between failure
MTPE	Mission to Planet Earth
MTTR	mean time to restore
N/A	not applicable
NAS	National Academy of Science
NASA	National Aeronautics and Space Administration
NESDIS	National Environmental Satellite Data and Information Service
NMC	National Meteorological Center (NOAA)
NOAA	National Oceanic and Atmospheric Administration
NSIDC	National Snow and Ice Data Center (DAAC)
O/A	orbit/altitude
ODC	other data center
OSI	Open System Interconnect
PDPS	Planning and Data Processing Subsystem
PDR	Preliminary Design Review
PDS	production data set
PGE	Product Generation Executive
PGS	Product Generation System (obsolete ECS element name) (ASTER)
PK	Primary Key
PLANG	Production Planning CSCI

PLNHW	Planning HWCI
PLS	Planning Subsystem
POSIX	Portable Operating System Interface for Computer Environments
PR	Precipitation Radar (TRMM)
PRONG	Processing CSCI
QA	quality assurance
RMA	reliability, maintainability, availability
RTF	rich text format
SAA	satellite active archive
SAGE	Stratospheric Aerosol and Gas Experiment
SCF	Science Computing Facility
SDP	Science Data Processing
SDPF	Sensor Data Processing Facility (GSFC)
SDPS	Science Data Processing Segment (ECS)
SDPTK	SDP Toolkit CSCI
SDSRV	Science Data Server CSCI
SeaWIFS II	Sea-Viewing Wide Field-of-View Sensor II
SFDU	Standard Format Data Unit
SMC	System Management Center (ECS)
SPRHW	Science Processing HWCI
SRS	software requirements specification
SSM/I	Special Sensor for Microwave/Imaging (DMSP)
SST	sea surface temperature
STMGMT	Storage Management
STMGT	Storage Management Software CSCI
SUBSRV	Subscription Server
TMI	TRMM Microwave Image
TOMS	Total Ozone Mapping Spectrometer
TONS	TDRS On-board Navigational System
TRMM	Tropical Rainfall Measuring Mission (joint US-Japan)

TSDIS	TRMM Science Data and Information System
USNO	US Naval Observatory
UT	universal time
UTC	universal time code
V0	Version 0
VIRS	Visible Infrared Scanner (TRMM)
WAIS	Wide Area Information Server
WKBCH	Workbench CSCI
WKSHW	Working Storage HWCI
WWW	World-Wide Web

This page intentionally left blank.