

---

**Project # 01.016, Task 2**

**DRAFT ENVIRONMENTAL DATABASE DESIGN**

---

*Prepared for Review*

Marine Corps Air Station (MCAS) Cherry Point  
Cherry Point, North Carolina 28533-0006

*Prepared by*

URS Corporation  
Post Office Box 13000  
Research Triangle Park, North Carolina 27709

July 2001

## TABLE OF CONTENTS

1.0	Introduction .....	3
2.0	Meeting Scope of Work .....	3
3.0	Design Description .....	4
3.1	Design Assumptions .....	4
3.2	Database Table Descriptions .....	5
3.2.1	Tables e_Item and e_ItemProperty .....	8
3.2.2	Tables e_Substance and e_SubstanceProperty .....	8
3.2.3	Tables e_Constant and e_ConstantProperty .....	9
3.2.4	Tables e_Event and e_EventResult .....	10
3.2.5	Table e_Algorithms .....	12
3.2.6	Tables e_Valid_Records and e_Valid_Values .....	13
3.2.7	Table e_Program_Inputs .....	14
3.2.8	Table e_Admin_Log .....	14
3.3	Database Business Rules .....	14
3.3.1	Summary of Main Database Business Rules .....	14
3.4	Database Linkages, Foreign and Native .....	15
3.5	Additional Table Structure Anticipated .....	16
4.0	Potential Database Design Enhancements .....	16
4.1	Enhanced Audit Capabilities .....	16
4.2	Processing Improvements .....	18
5.0	Glossary of Terms and Acronyms Used .....	20

## LIST OF FIGURES

Figure 1:	Overall Environmental Data Management System .....	5
Figure 2:	Entities Relationship Diagram (ERD) of the Database System .....	7
Figure 3:	Audit Enhancement and Process .....	17
Figure 4:	Horizontal Table Splitting Example .....	18

## LIST OF TABLES

Table 1:	SOW Requirements and Corresponding Database Tables .....	3
Table 2:	Database Tables Descriptions .....	6

## 1.0 INTRODUCTION

All Federal government agencies and organizations are required to comply with environmental laws and regulations. Military organizations have policies to comply with these laws and regulations by integrating environmental considerations into military operations and training. The objective of Project 01.016 is to review the reports required under the environmental Clean Air Act (CAA) and Superfund Amendments and Reauthorization Act (SARA) and develop a data structure to facilitate production of these reports. The Draft Environmental Database Design is the product of Task 2 and meets these scope of work requirements:

- Support the legal reporting requirements of the CAA and SARA regulations by maintaining all appropriate and necessary records required for compliance; and
- Provide a means to relate to the SDSFIE/FMSFIE data sets.

This design builds on the analysis of Task 1, where existing datasets were compared to current requirements. The design addresses the records determined to be database appropriate.

## 2.0 MEETING SCOPE OF WORK

CAA and SARA reporting revolves around emission estimates and materials inventory estimates for regulated and permitted sources. Frequently, the estimates must be accompanied with production measures (such as, fuels combusted) in order to justify the estimates and help a regulator understand the estimation process. At a minimum, these data lists must be maintained and are met in this database design as shown in the following table.

**Table 1: SOW Requirements and Corresponding Database Tables**

Requirement	Database Tables
Regulated Sources List	e_Item & e_ItemProperty
Emission Factors, Rates, Constants, etc.	e_Constant & e_ConstantProperty
Production Measures used in estimation	e_Event & e_EventResults
Relationships between the above lists	e_Algorithms
SDSFIE/FMSFIE linkage/utility	e_Item & e_ItemProperty used to store value of airasp_id from table ehairasp.

Additionally, the design facilitates the inclusion of foreign database information and information from native database tables without formally defined linkages.

All designs exhibit the goals of the designers. The main goals for this design are:

- Meet the SOW;

- Provide a recordkeeping structure with extensive recording flexibility to meet changing recordkeeping/reporting needs; and
- Create a design that could be integrated within an existing database instance (i.e., SDSFIE/FMSFIE) or be the foundation of a new instance.

### 3.0 DESIGN DESCRIPTION

This database design is an “object-oriented” design. An object-oriented design is an efficient, effective, and scalable approach to database design. There are 13 tables total, 10 of which are in parent-child pairs in which the identity record is the parent, and property record(s) is the child. The eleventh table records emission algorithm definitions and the remaining two serve audit and administrative recordkeeping functions. Figure 1 illustrates the expected system incorporating this database. Table 1 lists the tables.

The database design is efficient at two levels because the minimal table count: 1) limits database application overhead and server processing, and 2) allows a user (i.e., the SME) to more quickly understand and manipulate the records.

The database design is scalable in two ways: 1) unlimited number of records, and 2) the design allows for incorporation of new record requirements without changing the structure by adding a new table or new column in a table.

### 3.1 Design Assumptions

The main assumption is that this database will not stand as an isolated record structure. User interface application(s) provide practical utility to this design; it does not stand-alone. The following list of assumptions provides an expected “boundary” for the design of the database record structure.

- There will be an application user interface (AUI) for data entry and maintenance. The AUI will programmatically follow the same business rules as the database and will enforce the main rules based on data instructions from database records.
- There will be an application report interface (ARI) for users to obtain suitable reports to meet internal report and regulation report requirements. The ARI may be several reporting systems integrated to serve as a single application.
- Identity records will have two levels of categorization: fields category1 and category2<sup>1</sup>.
- Child records have one level of categorization: field category1.
- Foreign data sources will supply records, as needed, particularly records that are not associated with environmental reporting. (e.g., personnel name and email.)
- The presentation of records to general users and reports of the ARI will generally be managed using Views (database processors) or embedded SQL scripts. At this level foreign data,

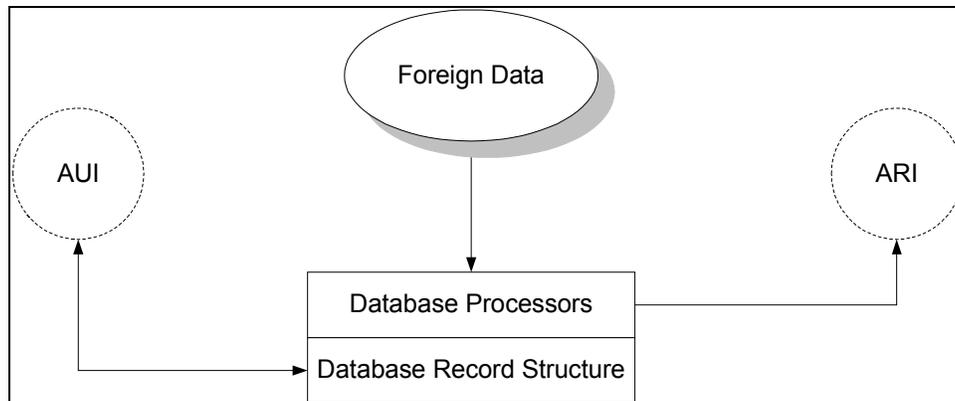
---

<sup>1</sup> There has been comment by the peer reviewer that if two levels of categorization is good, a third would be better. The designers have not been able to identify a third level to justify changing the current design. Note that the property child records provide a near infinite capability to describe a parent record. The categorization levels have more importance in the business rules logic than in describing the parent record.

sources would be incorporated seamlessly into the system processing and presentation. See Exhibit #2 as an example.

- Stored Procedures and Triggers<sup>2</sup> (database processors) or the AUI will be used to facilitate data entry, foreign data source interfacing, record maintenance, and record presentation to different applications. See Exhibit #1 as an example.

Figure 1 is a graphical presentation of the anticipated overall Environmental System. It shows the database relationship to system components.



**Figure 1: Overall Environmental Data Management System**

### 3.2 Database Table Descriptions

There are three classes of tables:

- Functional data records—Functional tables store records directly concerning CAA and SARA recordkeeping and reporting requirements. This is “real” data such as equipment names, fuel usage, chemicals, etc.
- Definitional data records—Definitional tables store information on managing processing input or output with the functional data. Here properties, categorizations, etc. are defined, as well as relationships between sources, emission factors, and production measures for emission estimate purposes.
- Administrative data records<sup>3</sup>—Administrative tables provide a means and aid for the system SME to maintain correct system functionality. Audit traces, input parameters for system programs, default values, etc. are recorded here.

<sup>2</sup> In the software industry, there is strong argument about using database level programming on the basis that the application then becomes tied to a specific database software product. These advocates argue that the AUI/ARI should manage all data processing. Of course these advocates usually are AUI/ARI programmers. Database level programming does offer efficiency benefits and can provide data processing features not readily possible at the AUI level (notably the immediate response and actions of a database trigger). It is noted that large systems such as those provided by SAP, do use database level programming. As the AUI/ARI are not a design product of this contract, the designers of this database merely inform of the options available for data management and processing.

The following table summarizes the tables in the design.

**Table 2: Database Tables Descriptions**

Table Name	Class	Relationship	Purpose
e_Item	Functional	Parent to e_ItemProperty	Stores identity records for all CAA and SARA items.
e_ItemProperty	Functional	Child to e_Item	Stores property records for defined items.
e_Substance	Functional	Parent to e_SubstanceProperty	Stores identity records for all substances being regulated.
e_SubstanceProperty	Functional	Child to e_Substance	Stores property records for all substances..
e_Constant	Functional	Parent to e_ConstantProperty	Stores identity records for constants.
e_ConstantProperty	Functional	Child to e_Constant	Stores property records for all constants.
e_Event	Functional	Parent to e_EventResult	Stores identity records for time-related events.
e_EventResult	Functional	Child to e_Event	Stores measure records for all time-related events.
e_Algorithms	Definitional	Indirect to all functional tables	Provides definitions for emissions estimate processing.
e_valid_records	Definitional	Parent to e_Valid_Values	Stores identity records for picklist/other AUI processing.
e_valid_Values	Definitional	Child to e_Valid_Records	Stores values for picklist/other AUI processing.
e_Admin_Log	Administrative	None	Stores data processing activity records for audit purposes.
e_Program_Inputs	Administrative	None	Stores input values for stored procedures, etc.

Figure2 is the entities relationship diagram (ERD) of the design.

<sup>3</sup> Section 3.5 discusses the anticipated need for additional administrative tables to support AUI/ARI requirements.

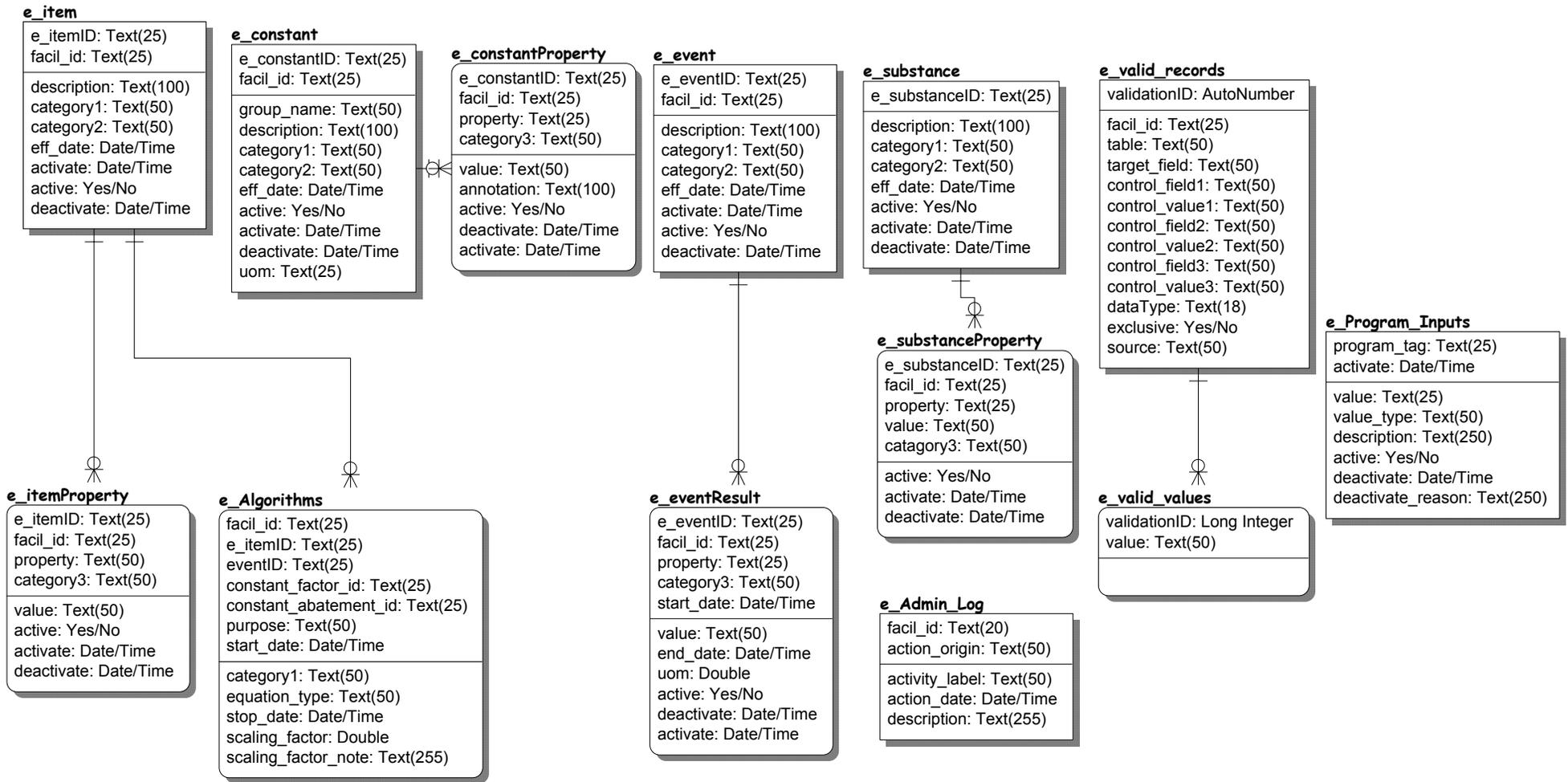


Figure 2: Entities Relationship Diagram (ERD) of the Database System

### 3.2.1 Tables e\_Item and e\_ItemProperty

This table pair is designed to store any item of interest for CAA or SARA recordkeeping. An item can be *a record of* any of the following:

- Facility
- Geographic Area
- Equipment
- Permit
- Permit Condition
- Task

This design recognizes the common features of these records so does not create individual tables for each class. The list can be expanded, although this list is reasonably inclusive of items regulated under CAA or SARA. Note that non-regulation related items could be recorded, also.

Any property of any item can be stored in the child table as a “labeled” record. The flexibility of this design is such that properties of interest can be recorded when they become a concern without the prerequisite of modifying the table.

Example Records<sup>4</sup>:

facil_ID	e_Item	Description	Category1	Category2	eff_date
Cherry_Point	Cherry_Point	Cherry Point Air Station	FACILITY		05/01/1950
Cherry_Point	Boiler1	Facility Boiler # 1	EQUIPMENT	BOILER	05/01/1961
Cherry_Point	Air_Permit1	Title V Air Permit	PERMIT	TITLE V	05/01/1999

facil_ID	e_Item	Category3	Property	Value
Cherry_Point	Air_Permit1	DATE	Expiration	05/01/2004
Cherry_Point	Air_Permit1	ID	Permit Number	1000123V1
Cherry_Point	Air_Permit1	LINK	WebLink	//Abc.gov/CPT5.pdf
Cherry_Point	Boiler1	LINK	GIS Link	GIS123456
Cherry_Point	Boiler1	RATING	Nameplate Rating	100 MMBTU/Hr
Cherry_Point	Boiler1	ID	SCC	104-252-254
Cherry_Point	Cherry_Point	PERSONELL	Base Commander	Gen. Smith
Cherry_Point	Cherry_Point	ID	EPA SARA ID	555-444-333
Cherry_Point	Cherry_Point	ID	NC Facility ID	987654321

### 3.2.2 Tables e\_Substance and e\_SubstanceProperty

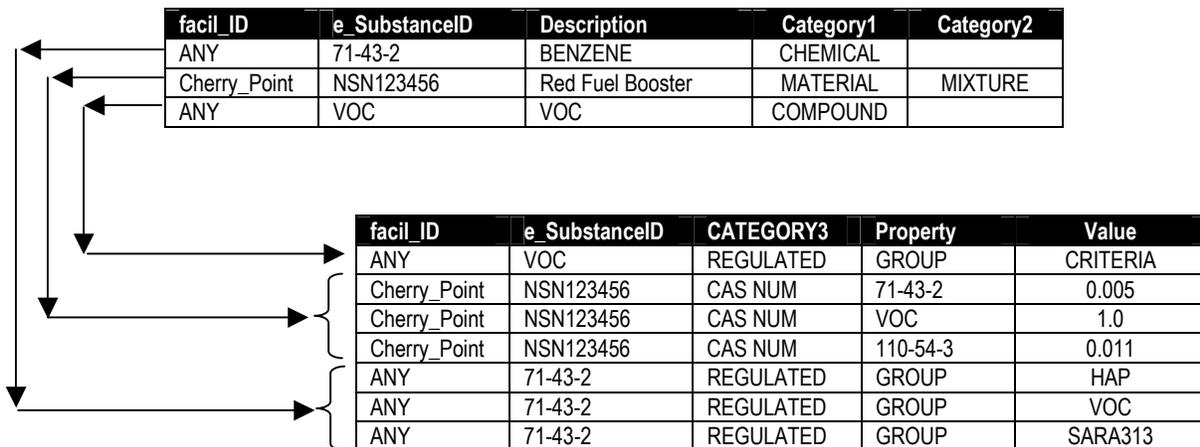
This table pair is designed to store any chemical or material of interest for CAA or SARA recordkeeping. The “substance” concept is inclusive and can be *a record of* any of the following items:

<sup>4</sup> The record audit fields, activate, active, and deactivate are not displayed as they are not pertinent to the example.

- Chemical Element
- Chemical Compound
- Pollutant Grouping
- Material

Any property of any substance can be stored in the child table as a “labeled” record. The flexibility of this design is such that properties of interest can be recorded when they become a concern without the prerequisite of modifying the table.

Example Records<sup>5</sup>:



### 3.2.3 Tables e\_Constant and e\_ConstantProperty

This table pair is designed to store data of a semi-static variety, usually with a numeric value, of interest for CAA or SARA recordkeeping. Typical data foreseen for these tables:

- Emission Factors
- Emission Rates
- Abatement Factors
- Conversion Factors/Ratios
- Permit Limitation

The list can be expanded; this list is reasonably inclusive.

Any property of any constant can be stored in the child table as a “labeled” record. The flexibility of this design is such that properties of interest can be recorded when they become a concern without the prerequisite of modifying the table.

<sup>5</sup> The record audit fields are not displayed.

Example Records<sup>6</sup>:

facil_ID	e_ConstantID	Description	Category1	Category2	UOM
ANY	NG<100	NG Boiler Comb. < 100 MMBTU/Hr	EMISSION FACTOR		LB/MMBTU
ANY	TAS_101	Technical Approach Specification 101	EMISSION RATE		LB/HR
Cherry_Point	Air_Permit1_B1	Boiler #1 Limits	PERMIT LIMIT	Boiler1	HR/YR
Cherry_Point	BL1_NGRATE	Boiler # 1 NG Rating	RATE	Boiler1	MMBTU/HR

facil_ID	e_ConstantID	CATEGORY3	Property	Value	Annotation:
Cherry_Point	BL1_NGRATE	EQUIP	RATE	75	Manufacturer Nameplate
ANY	AP42_NG<100	EFACTOR	VOC	.0054	AP-42 Table 3.2-1, 7/97
ANY	AP42_NG<100	EFACTOR	PM10	.000062	AP-42 Table 3.2-1, 7/97
ANY	AP42_NG<100	EFACTOR	NOX	.021	AP-42 Table 3.2-1, 7/97
ANY	AP42_NG<100	EFACTOR	50-00-0	.000007	AP-42 Table 3.2-2, 7/97
ANY	TAS_101	EFACTOR	108-95-2	0.0000821	TAS 101 Calculation
Cherry_Point	Air_Permit1_B1	PLIMIT	HOURS	5800	Title V Section A, 3.1

### 3.2.4 Tables e\_Event and e\_EventResult

This table pair is designed to store all time-related events of interest for CAA or SARA recordkeeping.

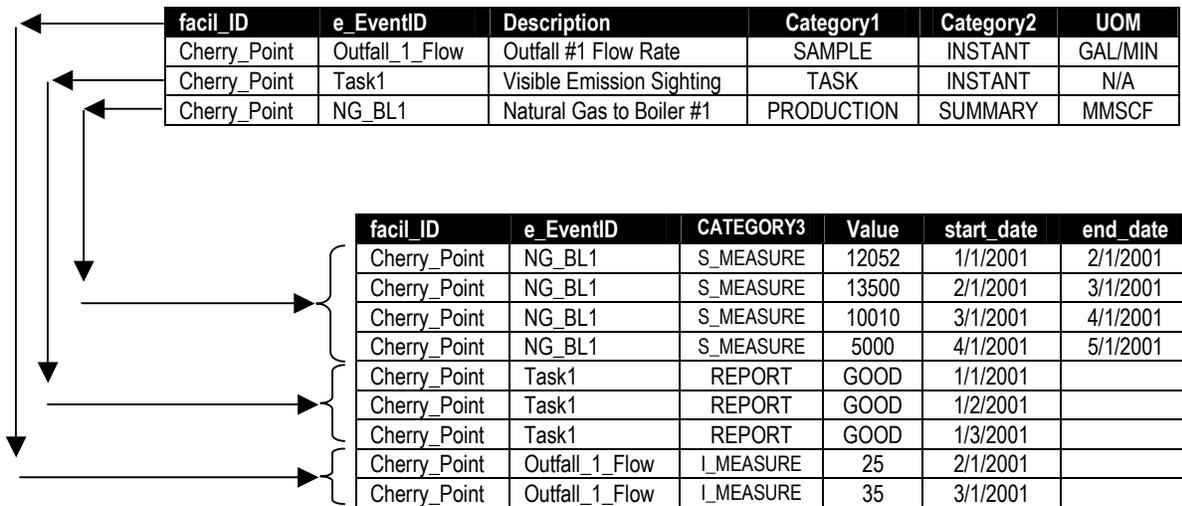
The “event” concept is very inclusive and can be *a record of* any of the following items:

- Production measure (instantaneous or span/summary)
- Sample measure (of any sort)
- Task completion
- Inventory readings

The time-stamped result of an event is stored in the child table.

<sup>6</sup> The record audit fields are not displayed.

Example Records<sup>7</sup>:



<sup>7</sup> The instantaneous time rule is shown as “end\_date is null.” The record audit field are not displayed.

### 3.2.5 Table e\_Algorithms

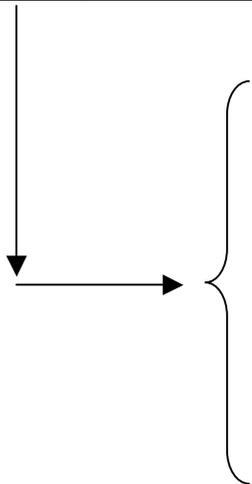
This table provides the definition of an emission estimate. Here all sources as items, constants as factors or rates, and production events are related logically. Also, the algorithm record allows for scaling to affect a unit-of-measure conversion, emission stream splitting, etc.

facil_ID	e_ItemID	Constant_Factor_ID	Constant_Abatement_ID	EventID	Category1	Equation_Type	Purpose	start_date	Stop_Date	Scaling_Factor	Scaling_Factor_Note
Cherry_Point	Boiler1	NG<100	N/A	NG_BL1	EVENT	FACTOR	ACTUALS	1/1/1996		0.001	Convert MMSCF to MMBTU
Cherry_Point	Boiler1	TAS_101	SCRUBBER1	BL1_HR	EVENT	RATE	ACTUALS	1/1/1996		1	
Cherry_Point	Boiler1	NG_OIL	SCRUBBER1	OIL_BL1	EVENT	FACTOR	ACTUALS	1/1/1996		6.3	Convert Barrel to MMBTU
Cherry_Point	Boiler1	NG<100	N/A	BL1_NGRATE	CONSTANT	FACTOR	POTENTIALS	1/1/1999	1/1/2004	1	1

### 3.2.6 Tables e\_Valid\_Records and e\_Valid\_Values

These tables provide instructions and values for the AUI and ARI. The result is that certain fields, as defined, are constrained to a value set with a single level of dependency allowed. (e.g., the value set of a table's field category2 is constrained by the inputted value of field category1). Here the database rules are largely defined as "table driven" inputs.

facil_ID	ValidationID	Table	Target_Field	Control_Field1	Control_Valu e1	Control_Field 2	Control_Valu e2	Control_Field 3	Control_Valu e3	DataType	Exclusive	Source
ANY	100000001	e_Item	Category1							TEXT	NO	VALUE LIST
ANY	100000002	e_Item	Category2	Category1	EQUIPMENT					TEXT	NO	VALUE LIST
ANY	100000003	e_Algorithms	Category1							TEXT	NO	VALUE LIST
Cherry_Point	100000004	e_Algorithms	EventID	Category1	EVENT					TEXT	YES	e_Event.e_EventID
Cherry_Point	100000005	e_Algorithms	EventID	Category1	CONSTANT					TEXT	YES	e_Constant.e_ConstantID
ANY	100000006	e_ConstantProperty	Property	e_Constant.cat egory1	EMISSION FACTOR			Category3	EFACTOR	NUMBER	NO	e_Substances.e_substanceID
ANY	100000007	e_SubstanceProperty	Property	e_substance.ca tegrity1	MATERIAL	e_substance.c ategrity2	MIXTURE	Category3	CAS NUM	TEXT	YES	e_Substances.e_substanceID



ValidationID	Value
100000001	FACILITY
100000001	EQUIPMENT
100000001	PERMIT
100000002	BOILER
100000001	AREA
100000001	STACK
100000002	ENGINE
100000002	PAINT BOOTH
100000002	TANK
100000003	EVENT
100000003	CONSTANT
100000004	category1 = 'PRODUCTION' and category2 = 'SUMMARY'
100000005	category1 = 'RATE'
100000007	category1 IN ('CHEMICAL', 'COMPOUND')

### 3.2.7 Table e\_Program\_Inputs

This table provides the various system programs with input when needed. The objective for the overall system is that no variables are “hard-wired” within the code. Instead, all program variables are table driven allowing the SME direct control over the system behavior without the need of a PL/SQL programmer.

Program_Tag	Value	Value_Type	Description	Activate	Active	Deactivate	Deactivate_Reason
RETIRE_TIME	72	Number	Number of months to maintain a record	7/9/2001	TRUE		

### 3.2.8 Table e\_Admin\_Log

This table provides a trace of system activities for the SME to review.

facil_ID	Activity_Origin	Activity_Label	Action_Date	Description
Cherry_Point	Child_Deactivation_Item	Row Trigger	6/12/2001	Table e_Item: Boiler 2 Deactivation date 6/1/2001
Cherry_Point	Production_Shield	Row Trigger	7/15/2001	Table e_Event_Shield; 100 items processed

## 3.3 Database Business Rules

Database business rules are the rules, constraints, etc. that bind the data in the database. These rules should not be confused with the implementation rules. The implementation rules are a product of an implementation plan and must conform to the database business rules. It is an advantage to have database business rules that are flexible and do not constrain the implementation plan.

The database design features a structure that shifts a large portion of what would normally be database business rules into the implementation plan. This structure is realized in the tables e\_Valid\_Records and e\_Valid\_Values. These tables store “table-driven” rules and field relationships.

### 3.3.1 Summary of Main Database Business Rules<sup>8</sup>

- All indirect relationships and constrained field values are defined by the tables e\_Valid\_Records and e\_Valid\_Values and enforced by the AUI and/or system processors.
- Table e\_Valid\_Records is structured to recognize that any field in a given table can be constrained in value sets by up to three other fields’ values. Additional constrains by foreign table field values (especially parent tables) are managed by source (table) identification and SQL text commands.
- All functional data class tables’ records must be constrained to a specific site. (It is recommended that a “reserved word” value of “ANY” be defined in the e\_Valid\_Records and e\_Valid\_Values along with AUI and processor implementation to provide the system with a global understanding of the record.)

<sup>8</sup> The defined structures of the database, i.e., primary keys, foreign keys, field data types, etc., are not summarized here. These are provided in the Conceptual Design Model (ERWIN file).

- System processors do not to have defined constants in the PL/SQL DECLARE sections. Such values must be read from the e\_Program\_Inputs table.
- The AUI or auto-sequencer will define the e\_ValidationID field value in the e\_Valid\_Records table.
- The identity level tables have two categorization fields; category1 must be defined and category2 may be null.
- The property tables have one categorization field, category3, and it must be defined.
- The user will define facil\_ID and e\_ItemID in the e\_Item table. The facil\_ID is constrained by first defined as an e\_ItemID where category1 = 'FACILITY'.
- The user will define e\_EventID in table e\_Event.
- The system will implicitly recognize two types of time-related events in the e\_EventProperty table, instantaneous and span. An instantaneous time event is a measure taken at a moment in time and in this case a rule must be in place of either start\_date = end\_date or end\_date is null<sup>9</sup>. A span is recognized as both start\_date and end\_date as populated with end\_date > start\_date. Implementation plan rules can address the minimum allowed span and be set into the system using the e\_Valid\_Records table structure.
- The user will define e\_SubstanceID in table e\_Substance. For chemicals and chemical compounds with CAS numbers, these values would be the recommended IDs.
- The user will define e\_ConstantID in table e\_Constant.
- All fields named “value” in all tables will be a varchar2 data type even if the value is numeric. In this case, the processors will convert either explicitly or implicitly.
- All identity records (parent table) of tables in the functional data class will have two levels of record categorization. The first categorization requires data, but the second categorization can be optional and is dependent on the value assigned for the first categorization.
- The e\_Algorithms table has one categorization (required).
- The e\_Algorithms table Scaling\_Factor field default value is 1. When the default is not used, the AUI is to enforce a notation in field Scaling\_Factor\_Note to explain the scaling factor.
- In tables with a field named “eff\_date”, this field refers to a time property of the article described by the record.
- In tables with fields “activate”, “active”, and “deactivate”, these fields refer to the record. Field active is a processing convenience and is either -1 (TRUE) or 0 (FALSE); it is defaulted to -1. If deactivate (a date type) is not null then active is constrained by the AUI and system processors to 0. Field activate is set by the AUI at the time of record creation in this manner: activate = *sysdate*.

### 3.4 Database Linkages, Foreign and Native

It is anticipated that, like most institutions, there is a universe of databases with useful data for meeting recordkeeping and reporting requirements. Typical and past practice is to transcribe data from one database to another to provide the data to the AUI or, more likely, the ARI. The result is multiple copies

---

<sup>9</sup> In this design paper the rule is presented as end\_date is null.

of what should be the same data in several different storage structures. In time this data becomes unsynchronized leading to inconsistency in reporting data.

The database design and business rules address this problem by not requiring formal key linkages between tables and by allowing an existing storage structure to supplant functional data tables of this design. For example, the tables e\_Substance and e\_SubstanceProperty must be defined as part of this database to ensure necessary data availability of chemical data. Such a table set must be part of any environmentally oriented database system. Within SDSFIE/FMSFIE there is not a suitable structure but one might be constructed by reviewing the available data and constructing SQL script views that provides the right structure and information. Alternately, another system (HMMS or HSMS) may have similar tables that can be leverage for this system's use. In either case, the necessary data, presented in a suitable structure, could be provided from another database.

### **3.5 Additional Table Structure Anticipated**

It is anticipated that additional table structure would be required to service the AUI. To manage the CAA and SARA records into reportable information, some level of data processing must occur. At a minimum, a table will contain information for the AUI or ARI processing engines. The most obvious example is information on emission equation formulation.

System access, security, and such administrative functions are part of the AUI design. These features will likely require one or more tables to support the AUI.

It is not possible to define any tables or structure to service an unknown AUI/ARI system. The design of this table(s) would be driven by the needs of the AUI/ARI programming logic.

## **4.0 POTENTIAL DATABASE DESIGN ENHANCEMENTS**

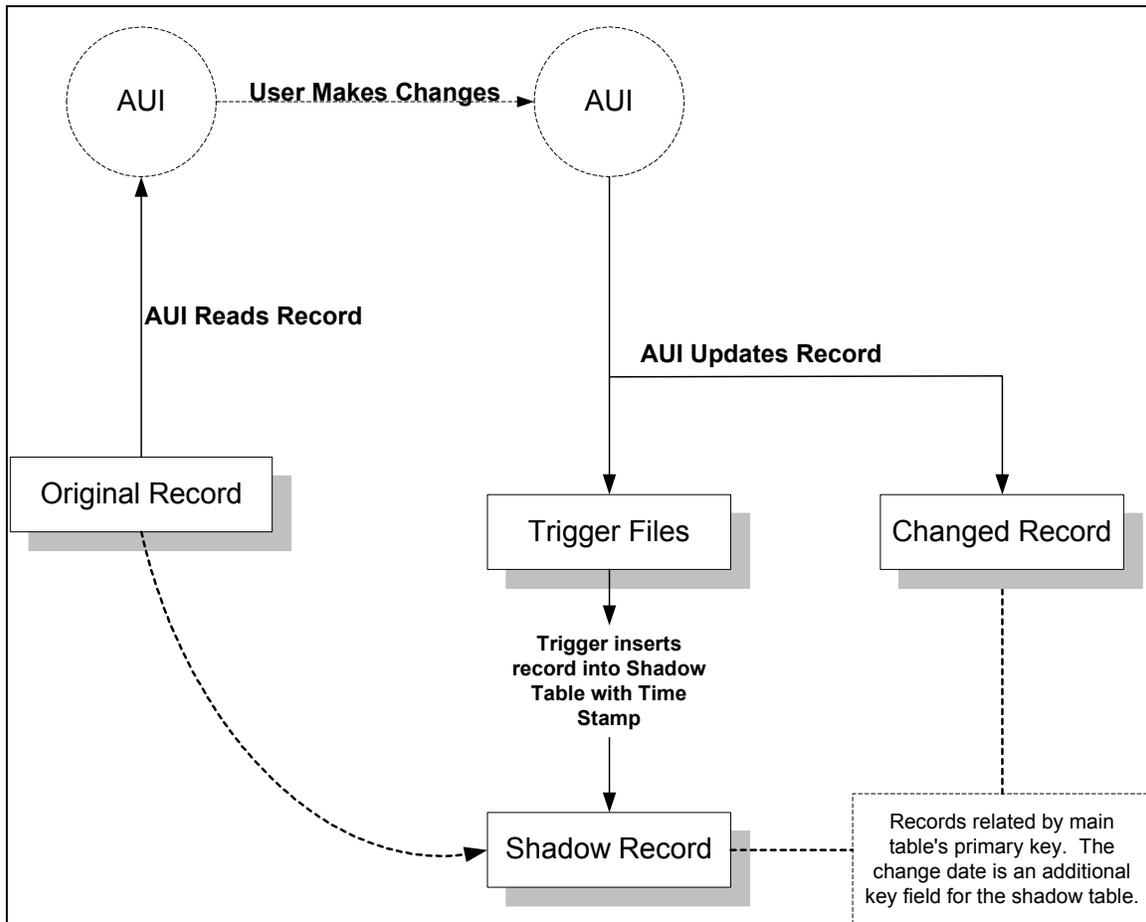
Two enhancements are presented for consideration. Both enhancements offer a benefit but to require more system management. Their value needs to be judge against their cost.

### **4.1 Enhanced Audit Capabilities**

A minimal audit capability is provided with the e\_Admin\_Log table and the implied database triggers represented by Exhibit #1. The e\_Admin\_Log table allows for a running trace of all record changes. However, the trace is minimal and does not allow a SME to completely reverse a change in all cases. A more sophisticated system can be provided though the usage of "shadow" tables.

A shadow table is a replica of the main table with the addition of two more field—a date field and a text field (e.g., Change\_Date date, User varchar2(50)). This table stores a copy of the main table's record to-

be-changed (i.e., the record prior to any changes made by a user or process). The two additional fields provide a time stamp and user/process identity making the change.



**Figure 3: Audit Enhancement and Process**

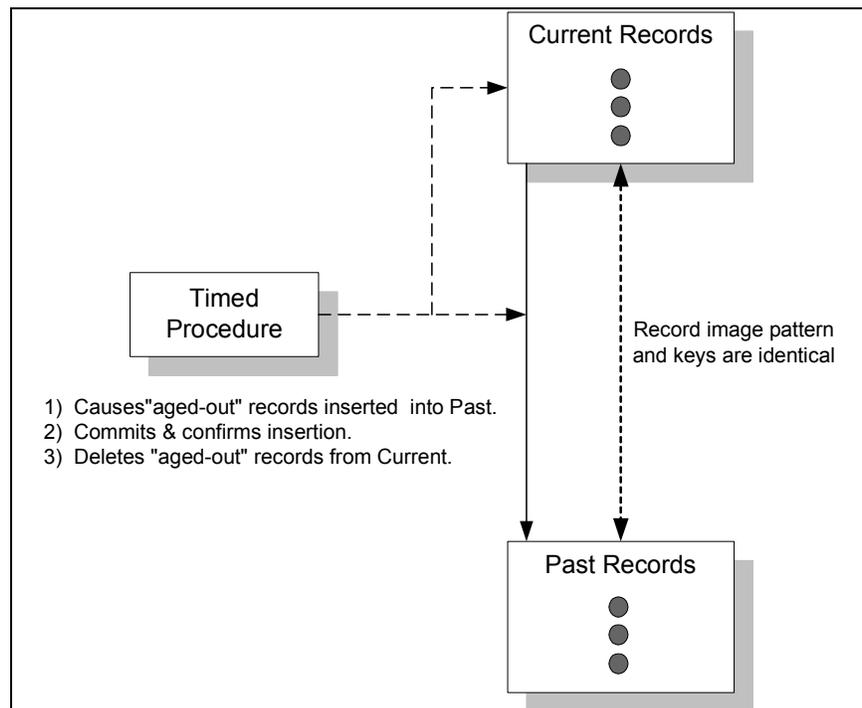
The size of a shadow table can expand significantly and needs a regular purge cycle. A purge cycle can be automated such that records of a certain age are deleted on a regular basis (e.g., all records of 2 years age are deleted automatically using a procedure that runs once per month). In the meantime, the SME has the capability to review all changes *and recompose* any records as necessary. This shadow system can eliminate the process logic surrounding the Admin\_Log table, or run as augmentation in parallel.

## 4.2 Processing Improvements

Tables with large recordsets require proportionally more processing resources, possibly producing a user noticeable time lag. In this design, the table that may exhibit this situation is the table e\_EventsResults. This table records all time dependent records. In a given year, it is reasonable to believe that thousands of time dependent actions will be recorded. Over several years (CAA would require up to 6 years of event records to be compliant with an inclusive 5-year record requirement) this table *may* become processing burdens.

To manage such a potential problem, horizontal table splitting can be performed. This enhancement of the design results in two tables instead of one. The e\_EventsResults table structure would be copied into an identical table. The new table would be assigned all records of a certain age and the original table would only contain current records.

Considering the reporting cycles of CAA in NC and SARA, it would be logical to define the current period as 18 months, making the past period all records older than 18 months. With such logic, normal reporting would only be processing records from the current period and not involving past period records. However, past period records are being maintained, as is required, and could be processed if necessary.



**Figure 4: Horizontal Table Splitting Example**

The illusion of a single table and data construction can be managed using an ANSI standard SQL script view similar to this example:

```
Create or Replace View e_EventResults_View as
Select * from e_EventResults
UNION
Select * from e_EventResults_Past;
```

## 5.0 GLOSSARY OF TERMS AND ACRONYMS USED

Term	Meaning
ARI	Application Report Interface: program(s) that allow a user to selectively obtain records from a database.
AUI	Application User Interface: program interface that allows users to interact in a controlled fashion with records in the database.
CAA	Clean Air Act
ERD	Entities Relationship Diagram: A graphical presentation of a database structure.
ERWIN	Software Program Aiding Database Design
HMMS	Hazardous Materials Management System
HSMS	Hazardous Substance Management System
NC	North Carolina, generally referring to the environmental agency.
Oracle	A provider of a major database product or the product itself (i.e., Oracle).
PL/SQL	Procedure Language/Structured Query Language: A procedural database language used to manipulate, control, and process records in a database.
Reserved Word	In programming, a word that has a defined meaning to system interpreter and may not be used by programmer or SME in any other fashion.
SARA	Superfund Amendments and Reauthorization Act
SDSFIE/FMSFIE	Spatial Data Standards for Facilities, Infrastructure and the Environment Facility Management Standards for Infrastructure and the Environmental
SME	Subject Matter Expert: person(s) responsible for the maintenance of data within the system.
SQL	Structured Query Language: A non-procedure database language chiefly used to request records from a database.
Sysdate	The Oracle database system's function to obtain the current date/time information from the system clock.
View	A database stored SQL query.

**Exhibit #1: Example of a database trigger procedure used to maintain data logic synchronization.**

The following PL/SQL trigger procedure updates records in table e\_ItemProperty following a change in the table e\_Item's Deactivate field from null to a value. The trigger marks all e\_ItemProperty records related by similar e\_Item primary key values with the same value.

```

Rem   Name:      Child_Deactivation_Item
Rem
Rem   Purpose:   Automatically deactivates child records of the parent record in table
Rem              e_Item when the parent record is deactivated
Rem
Rem   Author    Patrick G. Hecht
Rem
Rem   Date:     July, 2001
Rem
Rem   Actions   1) Updates all child records in e_Item_Property by e_Item primary
Rem              key.
Rem              2) Records action in Admin_Log Table.
Rem
Rem   Remarks   None.
Rem

```

```

Create or Replace Trigger Child_Deactivation_Item
BEFORE UPDATE on e_Item
FOR EACH ROW WHEN (new:Deactivate is not null)

```

```

DECLARE

```

```

v_Action_Label varchar2(25) := 'Row Trigger';
v_Description varchar2(2000) := 'Table e_Item: `';

```

```

BEGIN

```

```

UPDATE e_ItemProperty
Set Deactivate = new:Deactivate
Where Site_ID = new:Site_ID and Item_ID = new:Item_ID;

```

```

v_Description = v_Description || 'Item = ` || new:Item_ID || ` Deactivated Date: ` ||
to_char(new:Deactivate, 'mm/dd/yyyy')';

```

```

INSERT INTO Admin_Log (facil_ID, Action_Origin, Activity_Label, Action_Date,
Description) VALUES
new:facil_ID, 'Child_Deactivation_Item', v_Action_Label, sysdate, v_Description;

```

```

END

```

**Exhibit #2: Example Emission Estimation Views (Query) For ARI.**

The following ANSI SQL script views prepare data and process it into emission estimates for reporting purposes.

```
--      Processes a list of active emission factors

CREATE or REPLACE VIEW e_FACTORS_VIEW as
SELECT facil_ID, e_ConstantID, e_Property as CAS_Num
      , to_number(value) as Factor, annotation, UOM
FROM e_constants a, e_constantProperty b
WHERE a.facil_ID = b.facil_ID and a.e_ConstantID = b.Constant_ID and a.active = -1
      AND category1 in ('EMISSION FACTOR', 'EMISSION RATE');

--      Processes a list of active abatement factors
--      Creates pseudo list of 0 level abatement factors

CREATE or REPLACE VIEW e_ABATEMENT_VIEW as
SELECT DISTINCT facil_ID, e_ConstantID, e_Property as CAS_Num
      , to_number(value) as Abatement
FROM e_constants a, e_constantProperty b
WHERE a.facil_ID = b.facil_ID and a.e_ConstantID = b.Constant_ID and a.active = -1
      AND category1 = 'ABATEMENT FACTOR'
UNION
SELECT DISTINCT 'ANY' 'NO_ABATEMENT', e_substanceID, 0
FROM e_substance
WHERE category1 in ('CHEMICAL', 'COMPOUND');

--      Processes e_algorithm instructions into a list of emissions

CREATE or REPLACE VIEW AIR_EMISSIONS_VIEW as
SELECT a.siteID
      , a.item_ID
      , a.purpose
      , d.description, d.substanceID
      , SUM(scaling_factor * b.Factor * (c.Abatement / (1-c.Abatement))) * f.value)
as LBS
      , f.start_date
      , f.stop_date
FROM   e_algorithms a          -- Emission estimate definitions
      , e_FACTORS_VIEW b      -- Emission Factors/Rates Values
      , e_ABATEMENT_VIEW c    -- Emission Abatement Values
      , e_substance d         -- Provides common name
      , e_event e             -- Event "ON" check, UOM check
      , e_eventresults f      -- Event values
      , e_item g              -- Item "ON" check
WHERE  (a.siteID = b.siteID OR b.siteID = 'ANY')
      AND c.constant_Factor_ID = b.constantID
      AND (a.siteID = c.siteID OR b.siteID = 'ANY')
      AND b.CAS_NUM = c.CAS_NUM
      AND b.CAS_NUM = d.substanceID AND d.active = -1
      AND a.siteID = e.siteID AND a.eventID = e.e_eventID and e.active = -1
```

```
AND f.siteID = e.siteID AND f.eventID = e.e_eventID
AND f.start_date >= a.eff_date AND f.start_date < a.eff_date
AND f.end_date <= a.stop_date AND f.end_date > a.eff_date
AND a.siteID = g.siteID AND a.item_ID = g.e_ItemID and g.active = -1
GROUP BY a.siteID
       , a.item_ID
       , d.description
       , d.substanceID
       , f.start_date
       , f.stop_date
       , purpose
;
```