2007

# An Expanded Database Structure for a Multi-Period, Optimization Model

Narain Gupta
*Indian Institute of Management, Ahmedabad*, naraingupta@iimahd.ernet.in

Goutam Dutta
*Indian Institute of Management, Ahmedabad*, goutam@iimahd.ernet.in

**Title: An Expanded Database Structure for a Multi-Period, Optimization Model**

Narain Gupta

Indian Institute of Management, Ahmedabad

Vastrapur, Ahmedabad 380015, India

**naraingupta@iimahd.ernet.in**

Goutam Dutta

Indian Institute of Management, Ahmedabad

Vastrapur, Ahmedabad 380015, India

**goutam@iimahd.ernet.in**

## Abstract

We describe a generic database structure of a multi-period two-stage stochastic optimization based decision support system. The model is an extension of earlier work (Dutta & Fourer; 2004). The model could be used with no or little knowledge of OR/MS. The model maximizes expected contribution subject to material balance, facility capacity, facility input, facility output, storage inventory balance, storage area constraints. The model also considers non-anticipativity constraints for first stage decision variables.

## 1. Introduction and Motivation

This research is motivated from Fourer (1997), in which fundamental principles of relational database construction were derived to represent a linear programming (LP) formulation as a database, and the earlier work by Dutta et al. (2000, 2004, and 2006), which extend the work in context with an steel, and pharmaceutical company. An optimization based decision support system (DSS) was developed using the database structure. The DSS was customized for integrated steel plant in North-America. The work resulted in 16-17% potential increase in the bottom line of the company.

In this research, a two stage stochastic linear program (SLP), hence forth called SLP, is conceptualized and represented in the form of a relational database structure. In our model, the fundamental principles of relational database construction established by Fourer are being followed. We state the difficulties of reporting a multi-dimensional data values, and how have we addressed them. We discuss the files and fields of the database, analogous to the sets and parameters of the LP respectively. The one to one correspondence of the parameters of the LP with the fields of the files is presented.

Representing an LP into relational database form is a challenging task. It requires the knowledge of MS/OR (Management Science/Operations Research) to understand the modelers' representation of an LP; knowledge of database management systems to represent the same LP in the form of a relational database; and computer programming to understand and generate the computer readable forms of an LP, and to read and report the computer generated output back to the database. A developer needs to look for possible user friendly features for end user of the DSS such as easier system of data reporting, solution reporting, and updating. He also needs to look at ease, flexibility to use, and user friendliness. We address such issues in this paper in detail.

The organization of the paper is as followed. Section 1 introduces the paper and the motivation for this research. Review of literature on database optimization interface, stochastic optimization, and implementation in industries are presented is section 2. Database design issues and SLP model are discussed in section 3. In this section, we also present one-to-one correspondence of parameters and variables of LP with the fields of the relational data model.  In section 4, we discuss the principle steps of optimization, and the established diagnostic rules for data loading, reporting and update. In this section we also describe generation of variables, constraints, coefficients, and MPS text file using a matrix generator program. We explain how a report writer program read and write the optimal solution to the database and display it to the user. In section 5, we describe the issues related to data storage, retrieval, loading, and update. We show how a multi-dimensional data value of the optimal solution is reported. We also present a unique feature of 4th Dimension, included layouts. We conclude the paper with future directions on this research, and list of references.  Appendix describes the mathematical formulation of the SLP.

## 2. Literature Review

An LP can be represented in several ways as per the need of the user and the system. Fourer (1983) discussed, ways of representing the LP. Murphy et al. (1992) summarized common methods of LP representation schemes in practice (Table 1). Different methods of LP representation schemes were discussed in literature. Geoffrion (1987, 1989) presented the concept of Structured Modeling, A concept of graph-grammar approach is developed by Jones (1990, 1991, and 1992), Glover et al. (1992) described the Netform, and Block Schematic Diagram approach is due to Welch (1987). The matrix generator form of LP representation, which is a translation form, is attempted by several authors. The common matrix generator programs developed in the past includes DATAFORM by Ketron Inc. (1975), UIMP by Mitra et al. (1982), and MODELER by Greenberg (1990). Each of the LP representation scheme has its own merits and demits. Fourer (1983, 1990) recognizes that it is difficult to develop an LP representation form which can be commonly understood by modelers, computers and practitioners.

The review of literature recognizes that attempts are made to represent an LP in almost all forms listed above. Attempts to represent an LP in modelers form using modeling languages, graph-based systems, and block schematic languages were many. Comparative study of Murphy (1992), states that some of the LP representation schemes such as matrix generators, algebraic languages, and block-schematic languages have been implemented in commercial software systems. Other approaches are still at the prototype stage of development. A small number of modeling systems are reported in the literature which uses a relational database representation of an LP.

**Table 1: Categories of LP Representation Schemes (Murphy et al. 1992)**

| **Categories of LP Representation Schemes** | |
|---|---|
| **Class of Representation Schemes** | **Underlying Metaphor** |
| Symbolic | |
| Matrix Generators | MPS Problem Statement |
| Nonprocedural Languages | |
| Algebraic Language | Algebraic Statements |
| **Database-oriented Languages** | **Database of Data Tables** |
| Graphic | |
| Structured Modeling | Structured of Algebra/Data |
| Block-Schematic Languages | Nonzero Block of Resource Flows |
| Network Representations: | |
| Activity-Constraint Graphs | Activity/Constraint Network |
| Netforms | Netform of Recourse Flows |
| Iconic Languages | Analogues of Real World Objects |

Fourer (1997) studied the relational databases in context with the LP formulation. He visualized the subset of Cartesian product of sets of the LP as a relation in mathematical sense. He asserts that, it is not surprising that data indexed over pairs or triples from Cartesian products has a natural representation in relational database construction.

Realizing the broader applicability of representing a mathematical model in the form of a database, we depict, using our stochastic mathematical model, how a SLP model can be represented in the form of a database structure. The fields of the database would have a direct one to one correspondence with the parameters and the variables of the mathematical model.

Displaying the LP data values and optimal solution is not straight in multi-periods, multi-materials, multi-facilities, multi-activities, multi-storages, and multiple scenarios stochastic optimization model. The display has three dimensions, the list of data specific to an activity; time; and a scenario. The data and optimal reporting can be done in several ways. We have made an attempt to address the difficulties in reporting data and optimal solution. We also demonstrate the ease with which we report them and how these difficulties can be eliminated.

This paper also introduces a stochastic optimization based DSS with multi period, multi facility, multi activity, multi storage, multiple times with multiple scenario optimizations planning model. We aver that operating this DSS and generating reports for strategic planning purposes using this database does not require any background of MS/OR or any optimization technique. Users of the DSS require a moderate amount of computer training and knowledge of use of database management systems. The DSS and the mathematical formulation of stochastic model would be addressed in detail in forthcoming paper.

## 3. Database Representation of LP

In the process of translation an LP model can be represented as a database. Murphy et al. (1992) realizes two requirements for such representations. First, there is a need to record information about the structure of the model. Second, it is necessary to provide for the storage and manipulation of the data and the results that are obtained from the optimizer. From database viewpoint, the structure of an LP can be represented by two sets of entities, activities and constraints, together with a many to many relationship that records which activities are associated with which constraint.

The model schema records mapping between the fields of relational database tables and the parameters and variables of model. The key (unique identifies for tuples in the relation) is the set of indices that describe the array position of the data coefficient in the LP matrix. This is all the information needed to generate the algebraic form of LP. To represent non-linear model, stochastic models, and other type of models, the model schema can be expanded. The relation for the coefficients stores only the nonzero elements in the array representation. Thus, the representation conforms closely to the MPS format used for input by most of the optimizers.

Fourer (1997) constructed fundamental principles of database construction for the specific case of large scale mathematical programming. His research emphasized how the development of a database structure with a direct relation to the variables and the constraints of a large scale mathematical programming can lead to a user friendly DSS. In a series of publication Dutta et al. (2000, 2004, 2006) reported database representation of LP and its applications using a user friendly DSS. We will discuss our model with respect to the principles derived by Fourer (1997) in a separate section.

## 3.1 Database Design Issues for Scenario Based Stochastic Optimization Models

The SLP model has, as previously noted, six fundamental elements:

***Times*** are the periods of planning horizon, represented by discrete numbers (1, 2, 3 …). The duration of planning periods can be as short as weeks. Usually the durations of the planning periods are considered as months, quarters or years for operational and strategic planning.

***Scenarios*** are the possible outline of a hypothesized chain of events. Scenarios are represented with a name attached with a probability of occurrence. The sum of probabilities for all the scenarios should add to one.

***Materials*** are the physical items that figure in any of the production stage. Any material can be an input, intermediate, or finished product. Some times any material can be one or more than one of the mentioned type.

***Facilities*** are the collection of machines that produces some material from the other. For example a Hot Mill that produces sheets from slabs is a facility.

***Activities*** are the productive transformation of the materials. Each facility houses one or more activities, which uses one or more input and produces materials in certain proportions. Production of hot metal, production of billets, pickling, and galvanizing are examples of steelmaking activities.

***Storage-Areas*** are the laces where raw materials, intermediates and finished goods can be stored.

Fourer (1997) describes the algebraic formulation of the single period deterministic model and the corresponding database structure. In further extension Dutta and Fourer (2004) presents the multi period deterministic model and the corresponding database structure. We here formulate a two stage SLP, and develop a corresponding database structure. (Appendix I)

## 3.2 Important Characteristics of Fields and Files

For consistency reasons the data type of the keys in any file, corresponding to the sets of the LP viz: MatID, FacID, ActID, StoreID, SceneID, TimeID; are kept as long integer type in general. If user doesn't specify, the unique number is automatically generated by the system. We create another field named Unique Identification Number (UID) as a combination of the above six fields. The number of fields used in UID varies file to file. The elimination of duplicity of the records in the file is ensured using UID. The UID is created whenever a file requires two or more fields in combination as a primary key. The data type of the UID is alpha numeric in all the files, where ever it appears. Each file is facilitated to index using the six key fields, and some other required fields specific to the files.

## 3.3 The Relational Structure of the Data Model

This database structure mainly contains two sub databases. The first one (Figure 2) represents the LP in terms of related objects like materials, facilities, activities, and storages. It stores the related parameters in the database files. It also stores the optimal solution provided by the optimizer. The primary files of this database are [Materials], [Facilities], [Activities], [Storages], [Scenarios], and [Times]. These files are based on six fundamental elements of the model. These are scenario and time independent files. For homogeneity and clarity reasons we have followed an order in writing the basic files. This order remains same while using them for writing subscripts in the parameters and the variables of the SLP model. The followed order for indexes is materials, facilities, activities, storages, scenarios, and times. The unique key of scenario and time dependent files is made of combination of unique keys of the main files for six fundamental elements.

The other database (Figure 1) is for representing an instance of a LP in terms of list of variables, and constraints. It contains all coefficients associated with the constraint-variable combination.

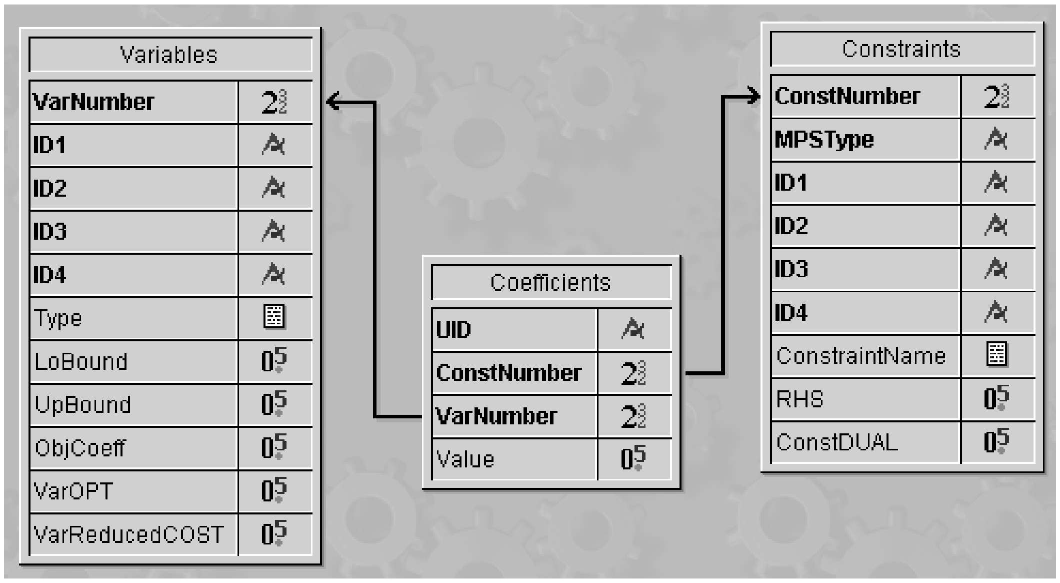


**Figure 1: Variables, Constraints and Coefficients Files of the Relational Database Structure**

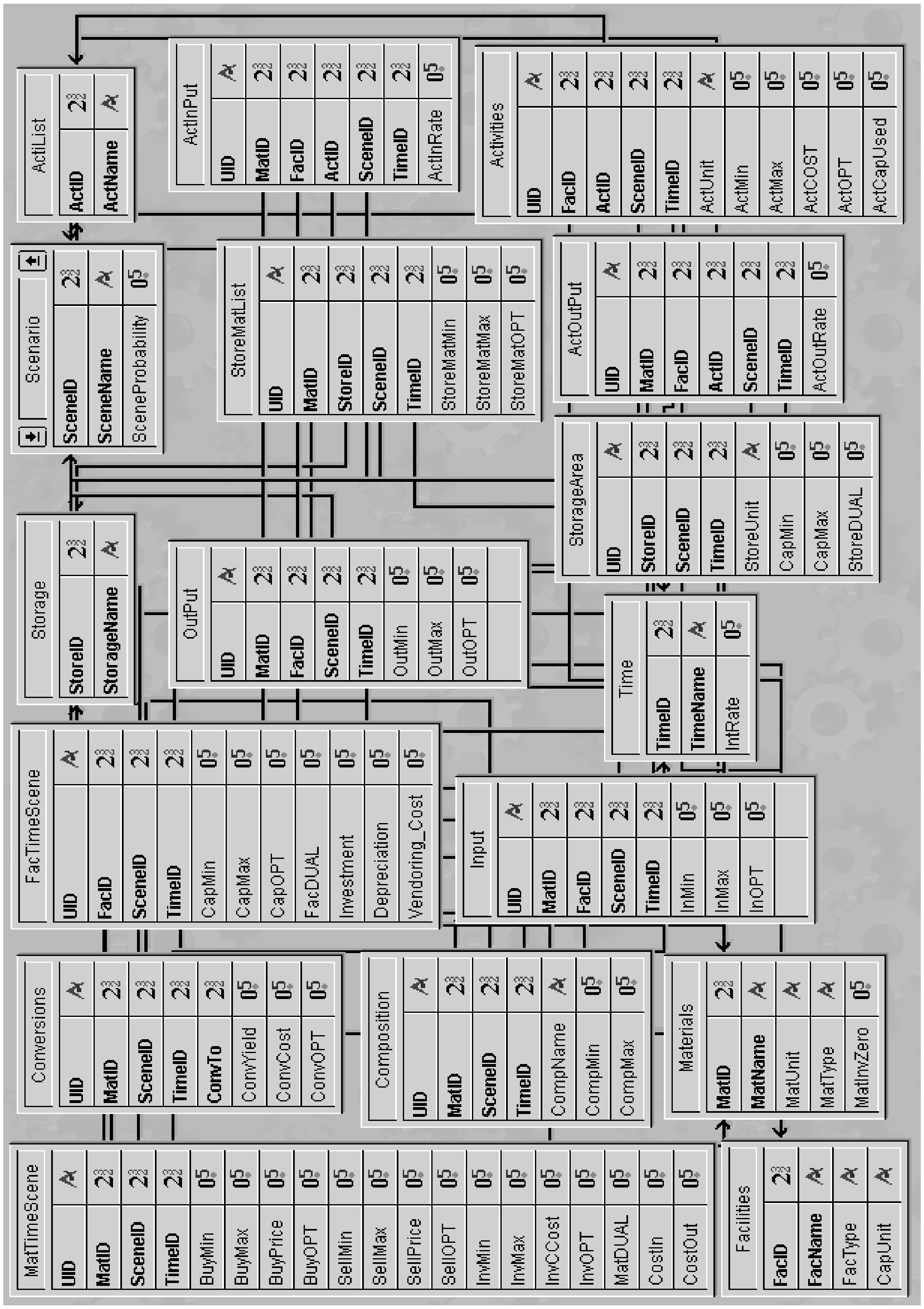


**Figure 2: Relational Database Structure of Stochastic Optimization Model for Strategic Production Planning**

## 3.4 Times and Scenarios Files

There are two files related to fundamental elements of the model (Figure 3). The [Times] file contains the detail of all planning horizons. The [Times] file stores identification number of the time period, name of the period, and interest rates. The corresponding fields are [Times]TimeID, [Times]TimeName, and [Times]IntRate respectively. Similarly the file [Scenarios] contains all information related to identification number, name, and the associated probability of the scenarios. The corresponding fields are [Scenarios]SceneID, [Scenarios]SceneName, and [Scenarios[SceneProbability respectively. The unique keys in [Times] and [Scenarios] files are [Times]TimeID, and [Scenarios]SceneID respectively.

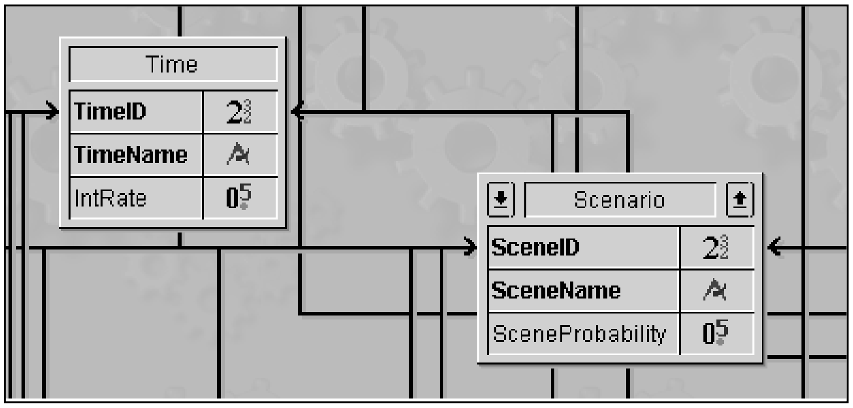


**Figure 3: Scenarios and Times Database Files Sub-Structure**

## 3.5 Materials Related Files

The set *Materials* related files are [Materials], [MatTimeScene], [Conversions], and [Compositions] (Figure 4). These files are related with [Materials] file in Many-To-One relationship. The linking fields are MatID, TimeID, and SceneID. The one to one correspondence of the parameters of the SLP with the fields of the [Materials] file, [MatTimeScene] file, is shown in Table 2. In this paper we do not discuss the conversions and compositions.

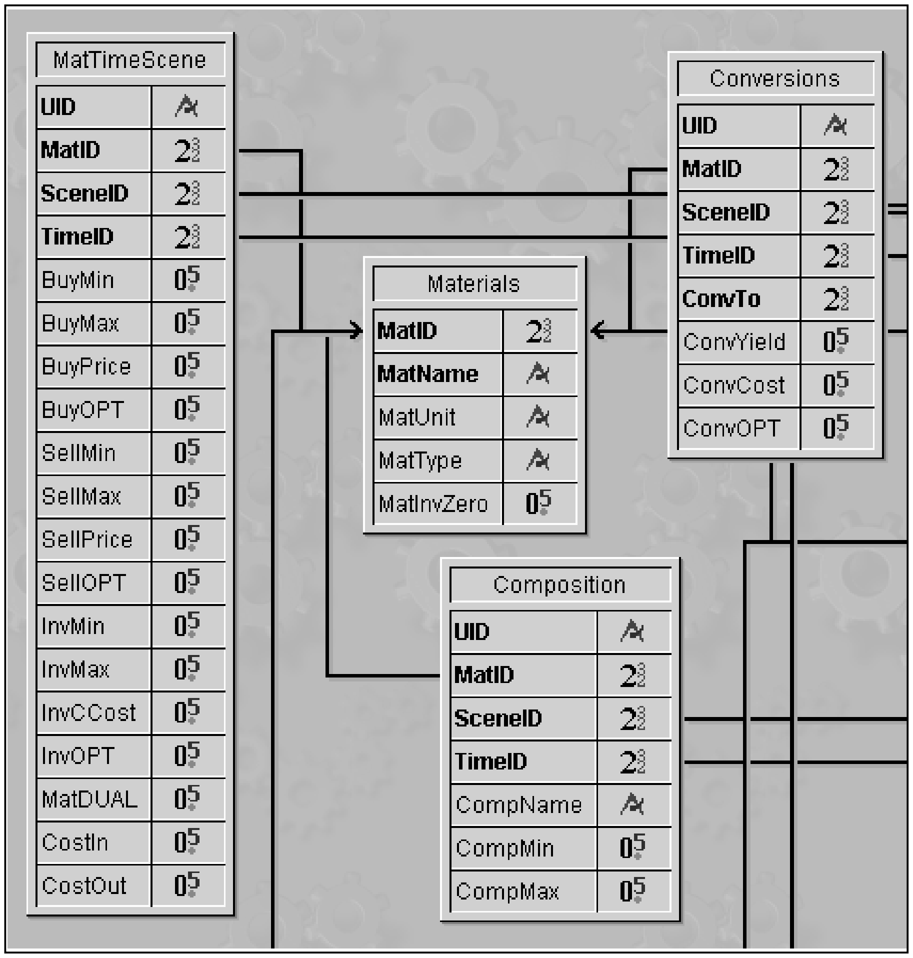
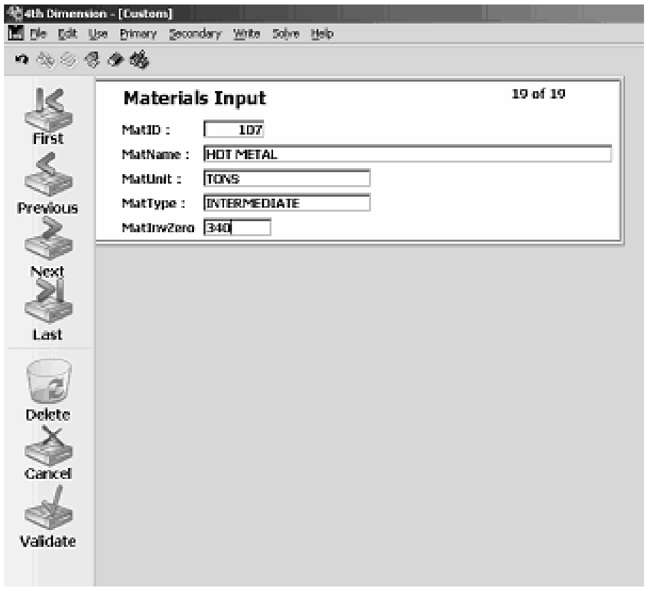


**Figure 4: Materials Related Database Files Sub-Structure**

### 3.5.1 Materials File

The material identification number field [Materials]MatID is a unique number. Materials files is indexed with the two fields [Materials]MatID and [Materials]MatName, on which the file can be indexed. The other fields in the materials files are [Materials]MatUnit, [Materials]MatType, and [Materials]MatInvZero. [Materials]MatInvZero field stores the initial inventory of the respective materials. Following the rules of normalization (parameter is independent of time and scenario) [Materials]MatInvZero is kept in materials file instead of [MatTimeScene] file. The indexed fields of [Materials] file are indexed [Materials]MatID, and [Materials]MatName. Data for these fields is entered using an input layout (Figure 5). List of materials with their attributes is displayed using an output layout (Figure 6).

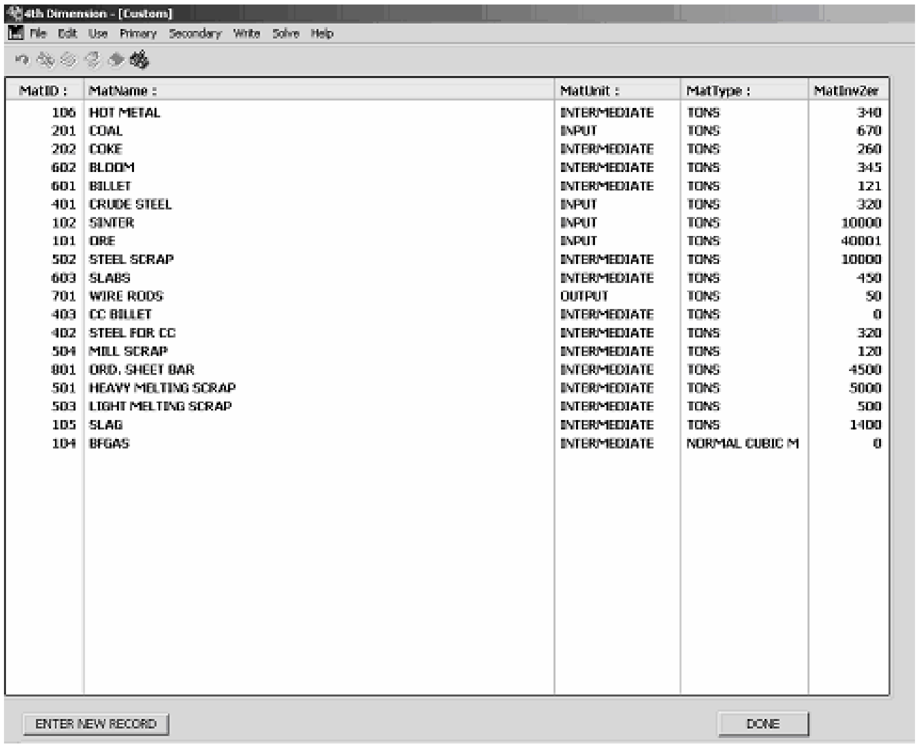


Materials Input

19 of 19

| | |
|---|---|
| MatID : | 107 |
| MatName : | HOT METAL |
| MatUnit : | TONS |
| MatType : | INTERMEDIATE |
| MatInvZero | 340 |

Figure 5: Materials Input Layout

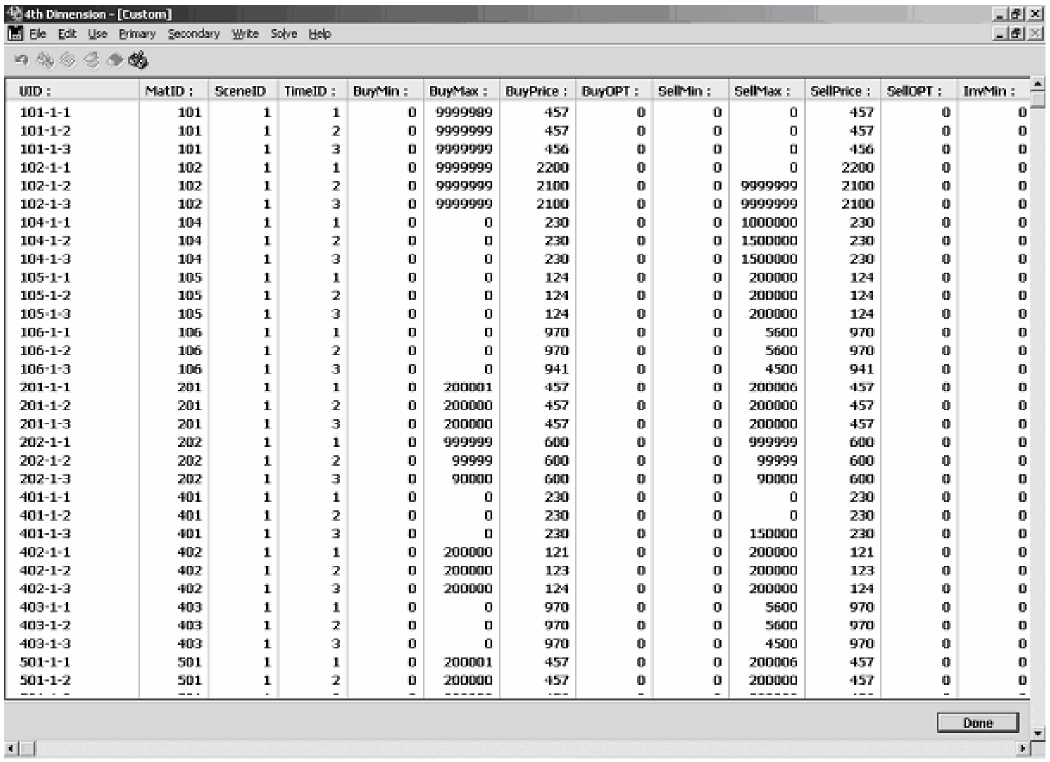


| MatID : | MatName : | MatUnit : | MatType : | MatInvZer |
|---|---|---|---|---|
| 106 | HOT METAL | INTERMEDIATE | TONS | 340 |
| 201 | COAL | INPUT | TONS | 670 |
| 202 | COKE | INTERMEDIATE | TONS | 260 |
| 602 | BLOOM | INTERMEDIATE | TONS | 345 |
| 601 | BILLET | INTERMEDIATE | TONS | 121 |
| 401 | CRUDE STEEL | INPUT | TONS | 320 |
| 102 | SINTER | INPUT | TONS | 10000 |
| 101 | ORE | INPUT | TONS | 40001 |
| 502 | STEEL SCRAP | INTERMEDIATE | TONS | 10000 |
| 603 | SLABS | INTERMEDIATE | TONS | 450 |
| 701 | WIRE RODS | OUTPUT | TONS | 50 |
| 403 | CC BILLET | INTERMEDIATE | TONS | 0 |
| 402 | STEEL FOR CC | INTERMEDIATE | TONS | 320 |
| 504 | MILL SCRAP | INTERMEDIATE | TONS | 120 |
| 801 | ORD. SHEET BAR | INTERMEDIATE | TONS | 4500 |
| 501 | HEAVY MELTING SCRAP | INTERMEDIATE | TONS | 5000 |
| 503 | LIGHT MELTING SCRAP | INTERMEDIATE | TONS | 500 |
| 105 | SLAG | INTERMEDIATE | TONS | 1400 |
| 104 | BFGAS | INTERMEDIATE | NORMAL CUBIC M | 0 |

Figure 6: Materials Output Layout

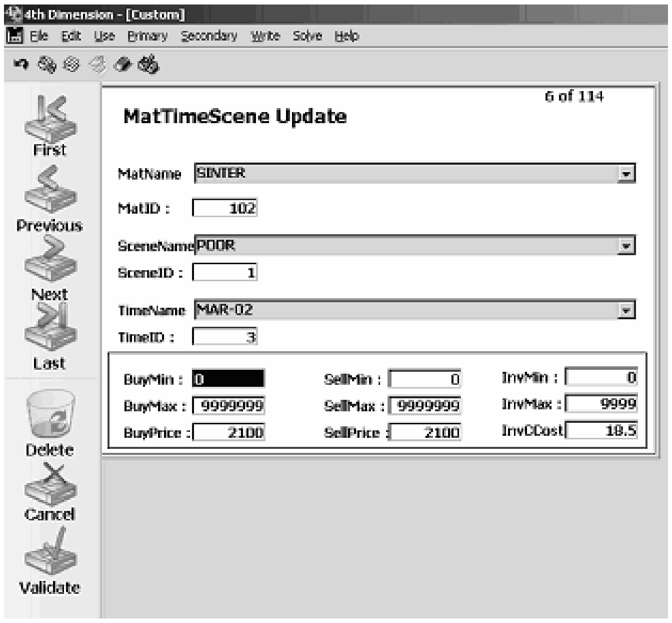
### 3.5.2 Materials Time Scenario File

The unique key in this file is made of the combination of the three fields namely [MatTimeScene]MatID, [MatTimeScene]SceneID, and [MatTimeScene]TimeID. The parameters BuyMin, BuyMax, BuyPrice, SellMin, SellMax, SellPrice, InvMin, InvMax, InvCCost are the time and scenario dependent fields in the [MatTimeScene] file. The data vaues for these fields is displayed in output layout (Figure 7). The optimal values of the program are left in the fields BuyOPT, SellOPT, InvOPT of [MatTimeScene] file. To accommodate fractional data of reasonably high range, we take REAL as the data type for all the fields for parameters and the optimal value in all the files. [MatTimeScene] file is linked with the files [Materials], [Scenarios] and [Times] through their unique fields [Materials]MatID, [Scenarios]SceneID, and [Time]TimeID respectively. The indexed fields of the [MatTimeScene] file are [MatTimeScene]MatID, [MatTimeScene]SceneID, and [MatTimeScene]TimeID.



| UID : | MatID : | SceneID | TimeID : | BuyMin : | BuyMax : | BuyPrice : | BuyOPT : | SellMin : | SellMax : | SellPrice : | SellOPT : | InvMin : |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 101-1-1 | 101 | 1 | 1 | 0 | 9999989 | 457 | 0 | 0 | 0 | 457 | 0 | 0 |
| 101-1-2 | 101 | 1 | 2 | 0 | 9999999 | 457 | 0 | 0 | 0 | 457 | 0 | 0 |
| 101-1-3 | 101 | 1 | 3 | 0 | 9999999 | 456 | 0 | 0 | 0 | 456 | 0 | 0 |
| 102-1-1 | 102 | 1 | 1 | 0 | 9999999 | 2200 | 0 | 0 | 0 | 2200 | 0 | 0 |
| 102-1-2 | 102 | 1 | 2 | 0 | 9999999 | 2100 | 0 | 0 | 9999999 | 2100 | 0 | 0 |
| 102-1-3 | 102 | 1 | 3 | 0 | 9999999 | 2100 | 0 | 0 | 9999999 | 2100 | 0 | 0 |
| 104-1-1 | 104 | 1 | 1 | 0 | 0 | 230 | 0 | 0 | 1000000 | 230 | 0 | 0 |
| 104-1-2 | 104 | 1 | 2 | 0 | 0 | 230 | 0 | 0 | 1500000 | 230 | 0 | 0 |
| 104-1-3 | 104 | 1 | 3 | 0 | 0 | 230 | 0 | 0 | 1500000 | 230 | 0 | 0 |
| 105-1-1 | 105 | 1 | 1 | 0 | 0 | 124 | 0 | 0 | 200000 | 124 | 0 | 0 |
| 105-1-2 | 105 | 1 | 2 | 0 | 0 | 124 | 0 | 0 | 200000 | 124 | 0 | 0 |
| 105-1-3 | 105 | 1 | 3 | 0 | 0 | 124 | 0 | 0 | 200000 | 124 | 0 | 0 |
| 106-1-1 | 106 | 1 | 1 | 0 | 0 | 970 | 0 | 0 | 5600 | 970 | 0 | 0 |
| 106-1-2 | 106 | 1 | 2 | 0 | 0 | 970 | 0 | 0 | 5600 | 970 | 0 | 0 |
| 106-1-3 | 106 | 1 | 3 | 0 | 0 | 941 | 0 | 0 | 4500 | 941 | 0 | 0 |
| 201-1-1 | 201 | 1 | 1 | 0 | 200001 | 457 | 0 | 0 | 200006 | 457 | 0 | 0 |
| 201-1-2 | 201 | 1 | 2 | 0 | 200000 | 457 | 0 | 0 | 200000 | 457 | 0 | 0 |
| 201-1-3 | 201 | 1 | 3 | 0 | 200000 | 457 | 0 | 0 | 200000 | 457 | 0 | 0 |
| 202-1-1 | 202 | 1 | 1 | 0 | 999999 | 600 | 0 | 0 | 999999 | 600 | 0 | 0 |
| 202-1-2 | 202 | 1 | 2 | 0 | 99999 | 600 | 0 | 0 | 99999 | 600 | 0 | 0 |
| 202-1-3 | 202 | 1 | 3 | 0 | 90000 | 600 | 0 | 0 | 90000 | 600 | 0 | 0 |
| 401-1-1 | 401 | 1 | 1 | 0 | 0 | 230 | 0 | 0 | 0 | 230 | 0 | 0 |
| 401-1-2 | 401 | 1 | 2 | 0 | 0 | 230 | 0 | 0 | 0 | 230 | 0 | 0 |
| 401-1-3 | 401 | 1 | 3 | 0 | 0 | 230 | 0 | 0 | 150000 | 230 | 0 | 0 |
| 402-1-1 | 402 | 1 | 1 | 0 | 200000 | 121 | 0 | 0 | 200000 | 121 | 0 | 0 |
| 402-1-2 | 402 | 1 | 2 | 0 | 200000 | 123 | 0 | 0 | 200000 | 123 | 0 | 0 |
| 402-1-3 | 402 | 1 | 3 | 0 | 200000 | 124 | 0 | 0 | 200000 | 124 | 0 | 0 |
| 403-1-1 | 403 | 1 | 1 | 0 | 0 | 970 | 0 | 0 | 5600 | 970 | 0 | 0 |
| 403-1-2 | 403 | 1 | 2 | 0 | 0 | 970 | 0 | 0 | 5600 | 970 | 0 | 0 |
| 403-1-3 | 403 | 1 | 3 | 0 | 0 | 970 | 0 | 0 | 4500 | 970 | 0 | 0 |
| 501-1-1 | 501 | 1 | 1 | 0 | 200001 | 457 | 0 | 0 | 200006 | 457 | 0 | 0 |
| 501-1-2 | 501 | 1 | 2 | 0 | 200000 | 457 | 0 | 0 | 200000 | 457 | 0 | 0 |

**Figure 7: Materials Time Scenario Output Layout**



6 of 114

MatTimeScene Update

| | |
|---|---|
| MatName | SINTER |
| MatID : | 102 |
| SceneName | POOR |
| SceneID : | 1 |
| TimeName | MAR-02 |
| TimeID : | 3 |

| | | | | | |
|---|---|---|---|---|---|
| BuyMin : | 0 | SellMin : | 0 | InvMin : | 0 |
| BuyMax : | 9999999 | SellMax : | 9999999 | InvMax : | 9999 |
| BuyPrice : | 2100 | SellPrice : | 2100 | InvCCost | 18.5 |

**Figure 8: Materials Time Scenario Update Layout**

**Table 2: One To One Correspondence of the Fields of the Materials Related Files with the Parameters and Variables of the SLP**

| Sr. No. | Parameter/Variables of the SLP | Fields of the Tables of the Relational Database |
|---|---|---|
| 1 | $v^{inv}_{jl0}$ | [Materials]MatInvZero |
| 2 | $u^{buy}_{jlt}$ | [MatTimeScene]BuyMax |
| 3 | $c^{buy}_{jlt}$ | [MatTimeScene]BuyPrice |
| 4 | $l^{sell}_{jlt}$ | [MatTimeScene]SellMin |
| 5 | $u^{sell}_{jlt}$ | [MatTimeScene]SellMax |
| 6 | $c^{sell}_{jlt}$ | [MatTimeScene]SellPrice |
| 7 | $l^{inv}_{jlt}$ | [MatTimeScene]InvMin |
| 8 | $u^{inv}_{jlt}$ | [MatTimeScene]InvMax |
| 9 | $c^{inv}_{jlt}$ | [MatTimeScene]InvCCOST |
| 10 | $x^{buy}_{jlt}$ | [MatTimeScene]BuyOPT |
| 11 | $x^{sell}_{jlt}$ | [MatTimeScene]SellOPT |
| 12 | $x^{inv}_{jlt}$ | [MatTimeScene]InvOPT |

## 3.6 Facilities Related Files

The files related to the set Facilities are [Facilities], [FacTimeScene], [Input], and [OutPut] (Figure 9). The structure of the scenario and time dependent parameters of the set *Facilities* is analogous to the set *Materials*. We list the one to one correspondence of the parameters of SLP with the fields of the [FacTimeScene], [Input], and [OutPut] (Table 3). The main file related to facilities is [Facilities]. The unique field of this file is [Facilities]FacID. The other fields are [Facilities]FacName, [Facilities]FacType, [Facilities]CapUnit. The indexed fields of the [Facilities] file are [Facilities]FacID and [Facilities]FacName. The other files that contain the scenario and time dependent parameters related to facilities are [FacTimeScene], [Input], and [Output] files.

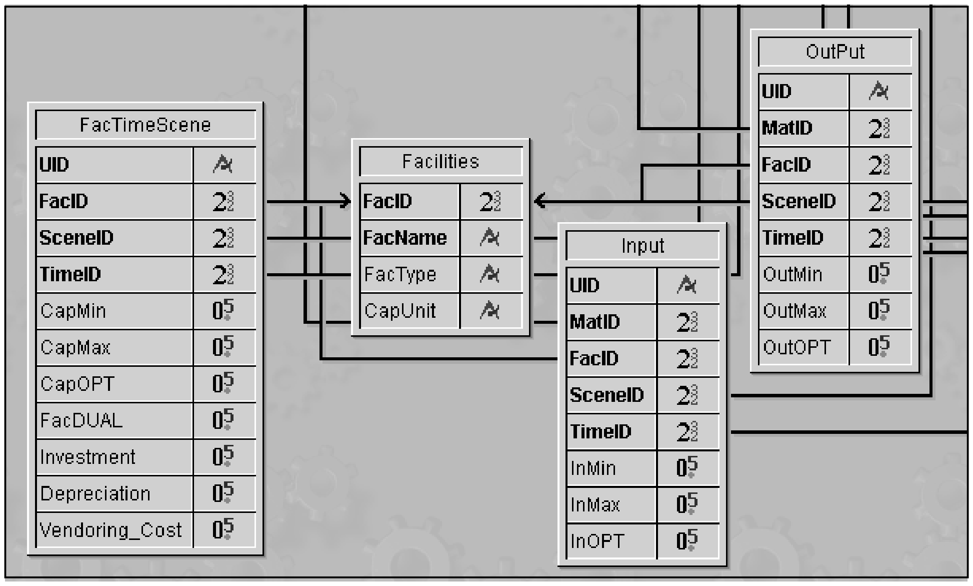


**Figure 9: Facilities Related Database Files Sub-Structure**

### 3.6.1 Facility-Material Input and Facility-Material Output Files

The unique key in files [Input] and [Output] are made of the combination of the fields MatID, FacID, SceneID, and TimeID. The combination is again named as unique identification (UID) with alphanumeric data type. The parameters stored in the [Input] file are the minimum, maximum, and optimum units of input material must be fed into the facility in consideration. These parameters are corresponding to the fields [Input]InMin, [Input]InMax, and [Input]InOPT respectively. Similarly the parameters stored in [OutPut] file are the minimum, maximum, and optimum units of output material must be produced from the facility in consideration. These parameters are corresponding to the fields

[OutPut]InMin, [OutPut]InMax, and [OutPut]InOPT. The files ([Input] and [OutPut]) are related with [Materials], [Facilities], [Scenarios] and [Times] files in a Many-To-One relationship (Figure 2). The linking fields are MatID, FacID, SceneID and TimeID respectively.

### 3.6.2 Facility Time Scenario File

The parameters related to the facilities capacity are structured in [FacTimeScene] file. The fields [FacTimeScene]CapMin, [FacTimeScene]CapMax, and [FacTimeScene]CapOPT, corresponds to the minimum, maximum, and optimum capacity must be used at the facility into consideration respectively. The field [FacTimeScene]Vendoring_Cost is for cost of outsourcing an additional unit of capacity of the facility under consideration in a particular scenario and time. The other fields of the file are [FacTimeScene]FacDUAL, [FacTimeScene]Invenstment, and [FacTimeScene]Depreciation. The file [FacTimeScene] is related with [Facilities], [Scenarios], and [Times] files in a Many-To-One relationship. The linking fields are FacID, SceneID, and TimeID respectively.

**Table 3: One To One Correspondence of the Fields of the Facilities Related Files with the Parameters and Variables of the SLP**

| Sr. No. | Parameter/Variables of the SLP | Fields of the Tables of the Relational Database |
|---|---|---|
| 1 | $l_{ilt}^{cap}$ | [FacTimeScene]CapMin |
| 2 | $u_{ilt}^{cap}$ | [FacTimeScene]CapMax |
| 3 | $c_{ilt}^{cap}$ | [FacTimeScene]Vendoring_Cost |
| 4 | $l_{ijlt}^{in}$ | [Input]InMin |
| 5 | $u_{ijlt}^{in}$ | [Input]InMax |
| 6 | $l_{ijlt}^{out}$ | [OutPut]OutMin |
| 7 | $u_{ijlt}^{out}$ | [OutPut]OutMax |
| 8 | $x_{ilt}^{cap}$ | [FacTimeScene]CapOPT |
| 9 | $x_{ijl}^{in}$ | [Input]InOPT |
| 10 | $x_{ijl}^{out}$ | [OutPut]OutOPT |

## 3.7 Activities Related Files

There are four files related to activities namely [ActiLists], [Activities], [ActInPut], and [ActOutPut] (Figure 10). The main file is the [ActiList] that contains the exhaustive list of all activities on all the facilities. The unique field in this file is [ActiList]ActID. The indexed fields of the file [ActiList] are [ActiList]ActID, and [ActiList]ActName. An [ActiList] file may have similar names of the activities such as PRODUCTION OF BILLET, but the ActID ensures the uniqueness of the characteristics associated with the activities.

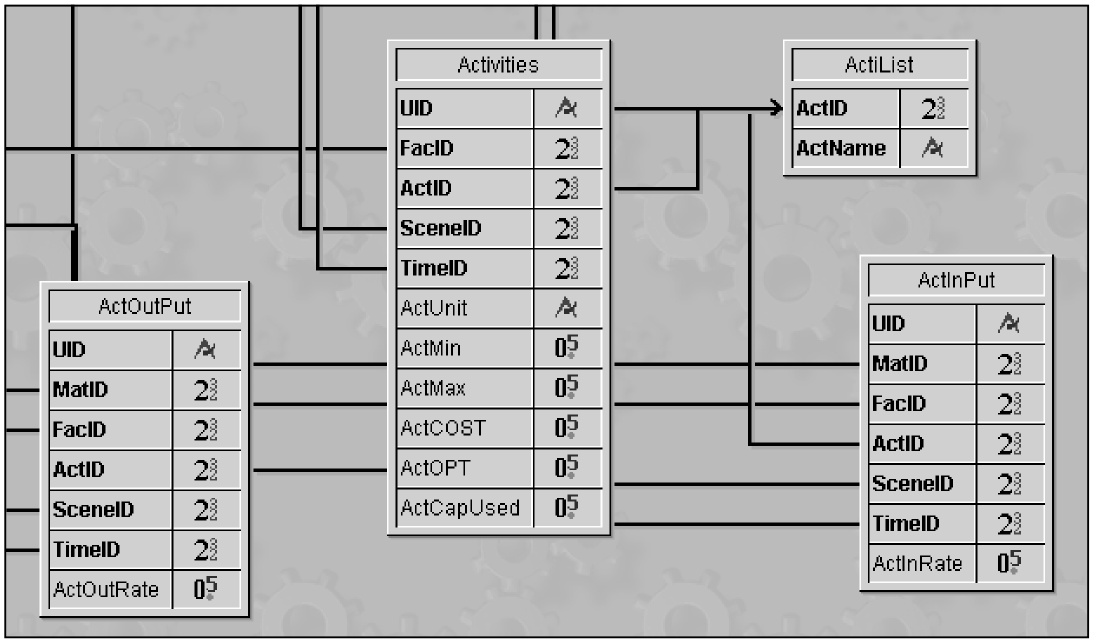


**Figure 10: Activities Related Database Files Sub-Structure**

### 3.7.1 Facility-Activity-Materials Input and Facility-Activity-Materials Output Files

The files [ActInPut] and [ActOutPut] store the rate of input and output of a material on an activity, facility, scenario, and time combination. The unique key in this file is made of the combination of the five fields MatID, FacID, ActID, SceneID, and TimeID. The indexed fields of the files [ActInPut] and [ActOutPut] are the above mentioned five ID's and the UID. The files [ActInPut], and [ActOutPut] are related with [Materials], [Facilities], [Activities], [Scenarios], and [Times] files in Many-To-One relationship. The linking fields are the respective unique fields of these five files.

### 3.7.2 Activities Time Scenario File

Another important file is [Activities] file, which contains the fields [Activities]ActMin, [Activities]ActMax, [Activities]ActOPT, and [Activities]ActCOST. These fields are corresponding to the parameters minimum, maximum, and optimal units of activity that must be operated, and per unit cost of operating the activity on associated facility. The field [Activities]ActCapUsed stores the activity facility ratio for capacity conversion. The indexed fields of the file [Activities] are [Activities]FacID, [Activities]ActID, [Activities]SeneID, and [Activities]TimeID. The combination of these fields makes a unique field named [Activities]UID with alphanumeric data type. The file is related with [Facilities], [Activities], [Scenarios], and [Times] file in Many-To-One relationship. The linking fields are FacID, ActID, SceneID, and TimeID respectively. We confirm the one to one correspondence of the fields of the tables [Activities], [ActInPut], and [ActOutPut] with the parameters of the SLP (Table 4).

**Table 4: One To One Correspondence of the Fields of the Activities Related Files with the Parameters and Variables of the SLP**

| Sr. No. | Parameter/Variables of the SLP | Fields of the Tables of the Relational Database |
|---|---|---|
| 1 | $l_{iklt}^{act}$ | [Activities]ActMin |
| 2 | $u_{iklt}^{act}$ | [Activities]ActMax |
| 3 | $c_{iklt}^{act}$ | [Activities]ActCOST |
| 4 | $r_{iklt}^{act}$ | [Activities]ActCapUsed |
| 5 | $\alpha_{ijklt}^{in}$ | [ActInPut]ActInRate |
| 6 | $\alpha_{ijklt}^{out}$ | [ActOutPut]ActOutRate |
| 7 | $x_{ikl}^{act}$ | [Activities]ActOPT |

### 3.8 Storage Related Files

There are three files related to storage areas namely [Storages], [StorageArea], and [StoreMatList] (Figure 11). The main file [Storage] is indexed over the fields StoreID, and StoreName. The unique key in this files is [Storage]StoreID. This file lists the names of all the storage in the system.

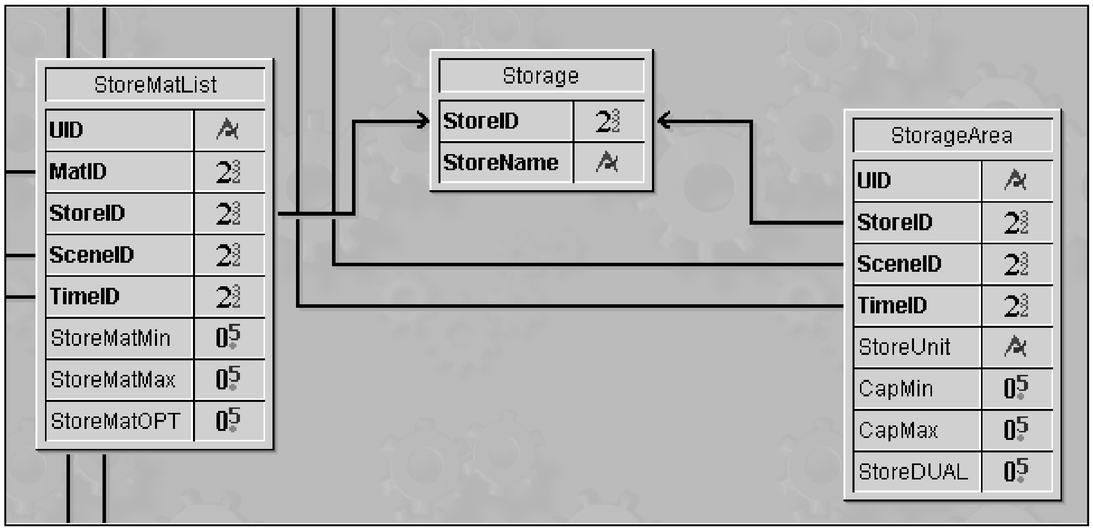


**Figure 11: Storages Related Database Files Sub-Structure**

#### 3.8.1 Storage Areas File

One of the file related to storage is [StorageArea], which is indexed over StoreID, SceneID, TimeID and UID. The unique key is UID with alphanumeric data type, and is made of the combination of the [StorageArea]StoreID, [StorageArea]SceneID, and [StorageArea]TimeID. The file is related with the [Storages], [Scenarios], and [Times] files in a Many-To-One relationship. The linking fields are StoreID, SceneID, and TimeID respectively. The parameters stored in the fields [StorageArea]CapMin, and [StorageArea]CapMax are total minimum and maximum capacity of the storage areas available. The one to one correspondence of the parameters of the SLP and the fields of the files [Storages], [StorageArea], and [StoreMatList] are shown in the Table 5. The field [StorageArea]StoreUnit, contains the unit of measurement of the materials to be stored in the storage areas.

### 3.8.2 Storage Materials File

Another file related to storage is [StoreMatList], which is indexed over MatID, StoreID, SceneID, TimeID, and UID. The unique key of this file is made of the combination of the fields [StoeMatList]MatID, [StoreMatList]StoreID, [StoreMatList]SceneID, and [StoreMatList]TimeID. The file is related with [Materials], [Storages], [Scenarios], and [Times] files in a Many-To-One relationship. The linking fields are MatID, StoreID, SceneID, and TimeID respectively. Also, the file [StoreMatList] is indexed with these linking fields. The parameters saved in this file are minimum, maximum, and optimal unit of material must be stored in the storage areas. The fields corresponding to these parameters are [StoreMatList]StoreMatMin, [StoreMatList]StoreMatMax, and [StoreMatList]StoreMatOPT.

**Table 5: One To One Correspondence of the Fields of the Storages Related Files with the Parameters and Variables of the SLP**

| Sr. No. | Parameter/Variables of the SLP | Fields of the Tables of the Relational Database |
|---|---|---|
| 1 | $l_{slt}^{stor}$ | [StorageArea]CapMin |
| 2 | $u_{slt}^{stor}$ | [StorageArea]CapMax |
| 3 | $x_{jslt}^{stor}$ | [StoreMatList]StoreMatOPT |

## 4. Optimization

In this section we describe the important steps of optimization, generation of variables, constraints and coefficients files, generation of algorithmic representation of LP using a matrix generator program, optimal solution loading and display using report writer program, and notion of soft capacities. This section also discusses diagnostic rules established for data loading, reporting, and updating.

## 4.1 Optimization Steps

A subsequent process of optimization is described in a step by step procedure. The principle steps are as follows (Figure 12):

1. ***Data Collection and Loading***: The set of data describing the production operation at different time periods and potential scenarios is collected and stored at appropriate places in the database.
2. ***Constraints Generation***: The constraints of the SLP are generated in [Constraints] file. The constant terms of the constraint equations, or inequalities, LoRHS and HiRHS (Ranges), are extracted from the database and stored in the [Constraints] file. This step also writes the MPS symbolic notation of constraint type, for example 'N' for objective function, 'L' for less than type of constraints.
3. ***Variables Generation***: The variables of the associated LP are determined, and named in the [Variables] file. The data values for lower bound, upper bound and objective coefficient associated with the decision variables is extracted from the database and stored in [Variables] file.
4. ***Coefficients Generation***: The unique nonzero data values (Technological Coefficients) of each variable and constraint pair is determined and extracted from the database. This data is stored in [Coefficients] file with a paired combination of variable and constraint identification number.
5. ***MPS Text File Generation***: The [Constraints] and [Coefficients] files are scanned for algorithmic form of LP generation. All the essential information about the LP is written in an ordinary text file using a matrix generator program. The file is used an input to the optimizer.
6. ***Solving the LP***: The optimizer reads the LP from input text file. An optimal solution is generated and the output of the optimizer is written in another ordinary text file.
7. ***Optimal Solution Reporting***: the output text file is read by a report writer program. The optimal values are written at appropriate fields in the database tables. This report writer also enables to display the optimal solution and optimal summary of cash flows using output layouts.

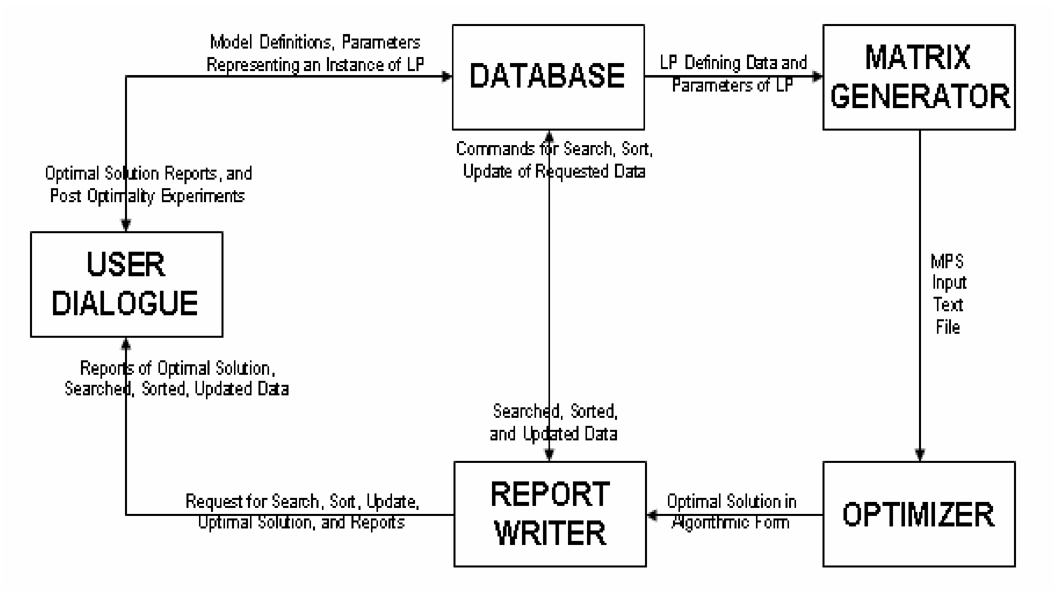

Model Definitions, Parameters Representing an Instance of LP
DATABASE
LP Defining Data and Parameters of LP
MATRIX GENERATOR
Commands for Search, Sort, Update of Requested Data
Optimal Solution Reports, and Post Optimality Experiments
USER DIALOGUE
MPS Input Text File
Reports of Optimal Solution, Searched, Sorted, Updated Data
Searched, Sorted, and Updated Data
Request for Search, Sort, Update, Optimal Solution, and Reports
REPORT WRITER
Optimal Solution in Algorithmic Form
OPTIMIZER

**Figure 12: Object Oriented Conceptual Frame Work of the Stochastic Optimization Based DSS**

## 4.2 Diagnostics Rule

We have established and implemented some diagnostic rules in the database to ensure that the SLP is complete and free from errors. These rules would ensure the feasibility of the problem solution. The rules have been implemented in the DSS with the help of programming code written behind the screen. The few generic diagnostic rules are:

1. **Rule 1**: The upper bound of any variable should never be less than the lower bound of it. The check should be performed immediately before a data would be saved in the data table.
2. **Rule 2**: For any variable and any constraint combination there should not be more than one non-zero value. This non-zero value, the coefficient of the variable, exists at the intersection of the corresponding row and column. The check should be performed immediately before the generation of variables, constraints, and coefficients file.
3. **Rule 3**: The Sum of probabilities attached with all scenarios lead to one. The sum is required to be assessed immediately before user saved the probability for a new scenario entry. If the sum of probability is become more then one due to addition of the new scenario, it is not possible to add more scenarios.

4. **Rule 4**: The optimal decisions resulted from the optimizer should not differ in numeric values for first time period. The reason is that the decisions associated with first period are first stage implement-able decisions, and must be identical. The assessment of this rule is necessary immediately before the RW write the optimal solution to the database tables. This also helps in verification of optimal solution.
5. **Rule 5**: For every constraint the lower right hand side (LoRHS) should not be more than the higher right hand side (HiRHS). The check should be performed immediately before a data would be saved in the data table.
6. **Rule 6**: It is necessary to ensure that the number of records in some files other than the basic six files should be equal to the product of the number of records in the linked files. For example, the records in the [MatTimeScene] file should be equal to the product of the records in [Materials], [Scenario] and [Time] file. Similarly the total number of records in the [StorageArea] file should be equal to the product of the records in [Storage], [Scenario], and [Time] files.
7. **Rule 7**: The total number of records in any file other than the basic six files for fundamental elements should not be more than the product of the total records in the linked files.
8. **Rule 8**: Every file should have a unique key field, which would avoid chances of duplication of records. Also the files which requires a combination of two or more then two fields as a primary key, a composite primary key need to be created for unique field. We created another field named unique identification number (UID) with alpha numeric data type. This field is made of the combination of the fields through which this file is linked with the basic files. The value of the unique field is created by concatenating the linked fields' immediately after their selection in the input box. Also, the values of the UID would automatically be refreshed on entering the data in the input boxes of the linked fields.
9. **Rule 9**: The input layouts of a file should display the potential default values of the parameters in the input text boxes for all possible parameters. It is possible that user may not have a value to enter for some input box. It is also possible that, the user is intending to enter the potential default value. In such situations the defaults values provided in input boxes ease

the process of entering data. This would ensure that the SLP has complete set of the data required. For example, we display 999, 999, 999, 99 and zero numeric values for the upper and lower bounds of all parameters.

We assume that the SLP is complete with data corresponding to every scenario and every time period. If any data is not available in the database, the default values would be used. As a default value, the lower limit for all variables is assumed to be zero, and the upper limit is assumed to be 99,999,999,999. The default value of yield is 100% and of rolling rate is 1tons per hour.

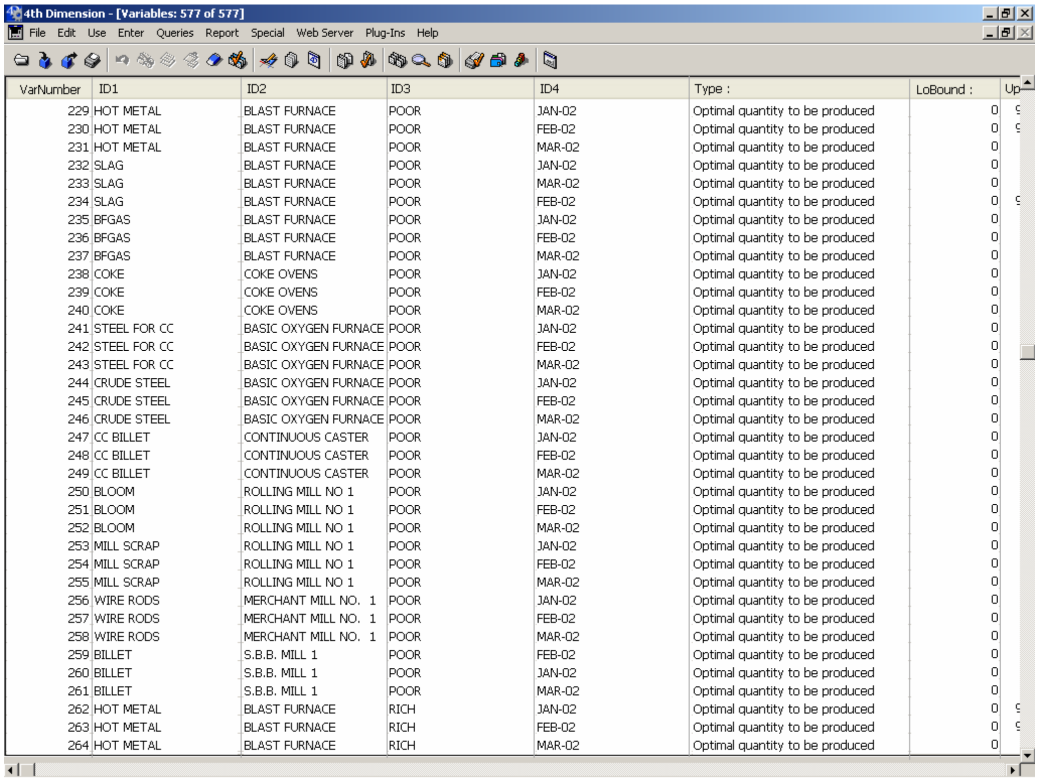
## 4.3 Generating Variables, Constraints and Coefficients File

This relational database contains a sub database which contains three files namely variables, constraints, and coefficients. These three files contain all the information, except lower and upper bounds on the decision variables, to instantiate SLP. The relational nature of the database facilitate locating any individual variable, constraints or the intersecting element in the [Variables], [Constraints], and [Coefficients] file respectively.

### 4.3.1 Variables File

This file stores the information related to the columns of the SLP (Figure 13). The file is indexed over the fields [Variables]VarNumber, [Variables]ID1, [Variables]ID2, [Variables]ID3, and [Variables]ID4. The fields ID1, ID2, ID3, and ID4 denote the indexes of the symbolic form of the variables, and parameters. The total number of sets of the model is six. The number of indexes for all parameters, and variables varies within a range of one to four (Appendix). We recognize a least interaction of the user with output layout of the variables file. Following the rules of normalization we restrict the indexing fields to four. This helps us reducing the size of the database, thereby reduces the storage space requirement.

The field [Variabes]VarNumber with long integer data type, works as the unique key of this file. The size of the SLPs increases exponentially in proportion to the number of scenarios. This leads to a large number of variables and constraints. It is possible that the number of variables and number of constraints may increase beyond the size of integer data type. We make data type of the unique field of the [variables] file and [constraints] file as long integer type. The fields that stores data values related to decision variables are [Variables]LoBound, [Variables]UpBound, [Variables]ObjCoeff. They store the lower and upper bound for the variables value, and the objective coefficient respectively. The field [Variables]Type contains the information regarding type of variable such as optimal quantity to be sold, optimal quantity to be bought, optimal quantity to be produced, optimal quantity to be used at facility, optimal quantity to be inventoried, optimal units of activity to be operated, optimal units of capacity to be vendored, optimal unit of material to be stored, and initial Inventory of material to be kept.


4th Dimension - [Variables: 577 of 577]
File Edit Use Enter Queries Report Special Web Server Plug-Ins Help

| VarNumber | ID1 | ID2 | ID3 | ID4 | Type : | LoBound : | Up |
|---|---|---|---|---|---|---|---|
| 229 | HOT METAL | BLAST FURNACE | POOR | JAN-02 | Optimal quantity to be produced | 0 | 9 |
| 230 | HOT METAL | BLAST FURNACE | POOR | FEB-02 | Optimal quantity to be produced | 0 | 9 |
| 231 | HOT METAL | BLAST FURNACE | POOR | MAR-02 | Optimal quantity to be produced | 0 | |
| 232 | SLAG | BLAST FURNACE | POOR | JAN-02 | Optimal quantity to be produced | 0 | |
| 233 | SLAG | BLAST FURNACE | POOR | MAR-02 | Optimal quantity to be produced | 0 | |
| 234 | SLAG | BLAST FURNACE | POOR | FEB-02 | Optimal quantity to be produced | 0 | 9 |
| 235 | BFGAS | BLAST FURNACE | POOR | JAN-02 | Optimal quantity to be produced | 0 | |
| 236 | BFGAS | BLAST FURNACE | POOR | FEB-02 | Optimal quantity to be produced | 0 | |
| 237 | BFGAS | BLAST FURNACE | POOR | MAR-02 | Optimal quantity to be produced | 0 | |
| 238 | COKE | COKE OVENS | POOR | JAN-02 | Optimal quantity to be produced | 0 | |
| 239 | COKE | COKE OVENS | POOR | FEB-02 | Optimal quantity to be produced | 0 | |
| 240 | COKE | COKE OVENS | POOR | MAR-02 | Optimal quantity to be produced | 0 | |
| 241 | STEEL FOR CC | BASIC OXYGEN FURNACE | POOR | JAN-02 | Optimal quantity to be produced | 0 | |
| 242 | STEEL FOR CC | BASIC OXYGEN FURNACE | POOR | FEB-02 | Optimal quantity to be produced | 0 | |
| 243 | STEEL FOR CC | BASIC OXYGEN FURNACE | POOR | MAR-02 | Optimal quantity to be produced | 0 | |
| 244 | CRUDE STEEL | BASIC OXYGEN FURNACE | POOR | JAN-02 | Optimal quantity to be produced | 0 | |
| 245 | CRUDE STEEL | BASIC OXYGEN FURNACE | POOR | FEB-02 | Optimal quantity to be produced | 0 | |
| 246 | CRUDE STEEL | BASIC OXYGEN FURNACE | POOR | MAR-02 | Optimal quantity to be produced | 0 | |
| 247 | CC BILLET | CONTINUOUS CASTER | POOR | JAN-02 | Optimal quantity to be produced | 0 | |
| 248 | CC BILLET | CONTINUOUS CASTER | POOR | FEB-02 | Optimal quantity to be produced | 0 | |
| 249 | CC BILLET | CONTINUOUS CASTER | POOR | MAR-02 | Optimal quantity to be produced | 0 | |
| 250 | BLOOM | ROLLING MILL NO 1 | POOR | JAN-02 | Optimal quantity to be produced | 0 | |
| 251 | BLOOM | ROLLING MILL NO 1 | POOR | FEB-02 | Optimal quantity to be produced | 0 | |
| 252 | BLOOM | ROLLING MILL NO 1 | POOR | MAR-02 | Optimal quantity to be produced | 0 | |
| 253 | MILL SCRAP | ROLLING MILL NO 1 | POOR | JAN-02 | Optimal quantity to be produced | 0 | |
| 254 | MILL SCRAP | ROLLING MILL NO 1 | POOR | FEB-02 | Optimal quantity to be produced | 0 | |
| 255 | MILL SCRAP | ROLLING MILL NO 1 | POOR | MAR-02 | Optimal quantity to be produced | 0 | |
| 256 | WIRE RODS | MERCHANT MILL NO. 1 | POOR | JAN-02 | Optimal quantity to be produced | 0 | |
| 257 | WIRE RODS | MERCHANT MILL NO. 1 | POOR | FEB-02 | Optimal quantity to be produced | 0 | |
| 258 | WIRE RODS | MERCHANT MILL NO. 1 | POOR | MAR-02 | Optimal quantity to be produced | 0 | |
| 259 | BILLET | S.B.B. MILL 1 | POOR | FEB-02 | Optimal quantity to be produced | 0 | |
| 260 | BILLET | S.B.B. MILL 1 | POOR | JAN-02 | Optimal quantity to be produced | 0 | |
| 261 | BILLET | S.B.B. MILL 1 | POOR | MAR-02 | Optimal quantity to be produced | 0 | |
| 262 | HOT METAL | BLAST FURNACE | RICH | JAN-02 | Optimal quantity to be produced | 0 | 9 |
| 263 | HOT METAL | BLAST FURNACE | RICH | FEB-02 | Optimal quantity to be produced | 0 | 9 |
| 264 | HOT METAL | BLAST FURNACE | RICH | MAR-02 | Optimal quantity to be produced | 0 | |

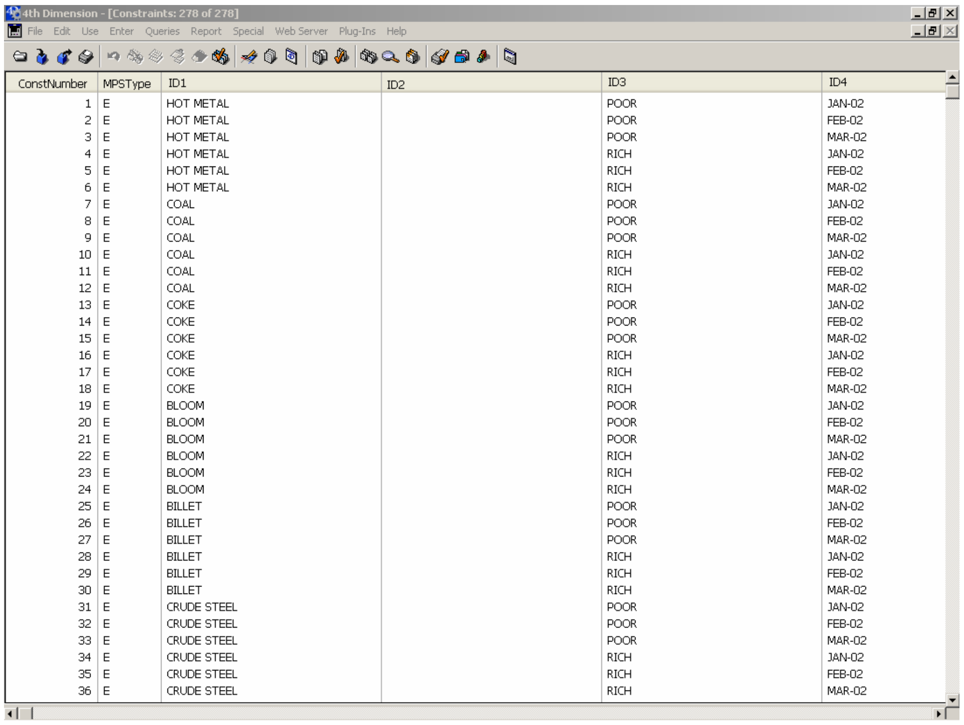
**Figure 13: Output Layout of Variables File**

### 4.3.2 Constraints File

The information related to the rows of the SLP is maintained in this file (Figure 14). The unique field of this file is [Constraints]ConstNumber with long integer data type. Similarly, the number of indexes in case of constraints varies between three to four (Appendix). The file is indexed with the fields [Constriants]ConstNumber, [Constraints]ID1, [Constraints]ID2, [Constraints]ID3, [Constraints]ID4, and [Constraints]MPSType. The fields ID1, ID2, ID3, and ID4 stores information related to the indexes of the constraints. The MPS (Mathematical Programming System) input notations for constraints is stored the field [Constraints]MPSType. Few examples of MPS notation is ‘N’ for objective function, ‘E’ for equivalent, ‘L’ for less than, and ‘G’ for greater than type of constraints. The field [Constraints]RHS contains the right hand side values of the constraints. For consistency and ease of understanding, range type of constraints was changed in two set of constraints. One set if of less than type and another is of greater than type. The field [Constraints]ConstraintName stores the name of the constraints such as material balance, facility inputs, facility outputs, facility capacity - lower bound, facility capacity - upper bound, initial inventory, storage inventory balance, storage capacity - lower bound, and storage capacity - upper bound. The nomenclature used for the first stage implementability constraints is First Stage – *Name of Variable* such as First Stage – Sell, or First Stage – Buy.

### 4.3.3 Coefficients File

This is the most important file of the complete database structure. It supplies data to the matrix generator program to generate an MPS input format for optimizers. The unique key of the file is UID, and is made of the combination of the [Coefficients]VarNumber, and [Coefficients]ConstNumber. The file is indexed over [Coefficients]UID, [Coefficients]ConstNumber, and [Coefficients]VarNumber. The field [Coefficients]Value contains the coefficients of the variable in the related constraint of the SLP. The [Coefficients] file is related with [Constraints] and [Variables] file in a Many-To-One relationship. The linking fields are [Coefficients]ConstNumber, and [Coefficients]VarNumber respectively.



| ConstNumber | MPSType | ID1 | ID2 | ID3 | ID4 |
|---|---|---|---|---|---|
| 1 | E | HOT METAL | | POOR | JAN-02 |
| 2 | E | HOT METAL | | POOR | FEB-02 |
| 3 | E | HOT METAL | | POOR | MAR-02 |
| 4 | E | HOT METAL | | RICH | JAN-02 |
| 5 | E | HOT METAL | | RICH | FEB-02 |
| 6 | E | HOT METAL | | RICH | MAR-02 |
| 7 | E | COAL | | POOR | JAN-02 |
| 8 | E | COAL | | POOR | FEB-02 |
| 9 | E | COAL | | POOR | MAR-02 |
| 10 | E | COAL | | RICH | JAN-02 |
| 11 | E | COAL | | RICH | FEB-02 |
| 12 | E | COAL | | RICH | MAR-02 |
| 13 | E | COKE | | POOR | JAN-02 |
| 14 | E | COKE | | POOR | FEB-02 |
| 15 | E | COKE | | POOR | MAR-02 |
| 16 | E | COKE | | RICH | JAN-02 |
| 17 | E | COKE | | RICH | FEB-02 |
| 18 | E | COKE | | RICH | MAR-02 |
| 19 | E | BLOOM | | POOR | JAN-02 |
| 20 | E | BLOOM | | POOR | FEB-02 |
| 21 | E | BLOOM | | POOR | MAR-02 |
| 22 | E | BLOOM | | RICH | JAN-02 |
| 23 | E | BLOOM | | RICH | FEB-02 |
| 24 | E | BLOOM | | RICH | MAR-02 |
| 25 | E | BILLET | | POOR | JAN-02 |
| 26 | E | BILLET | | POOR | FEB-02 |
| 27 | E | BILLET | | POOR | MAR-02 |
| 28 | E | BILLET | | RICH | JAN-02 |
| 29 | E | BILLET | | RICH | FEB-02 |
| 30 | E | BILLET | | RICH | MAR-02 |
| 31 | E | CRUDE STEEL | | POOR | JAN-02 |
| 32 | E | CRUDE STEEL | | POOR | FEB-02 |
| 33 | E | CRUDE STEEL | | POOR | MAR-02 |
| 34 | E | CRUDE STEEL | | RICH | JAN-02 |
| 35 | E | CRUDE STEEL | | RICH | FEB-02 |
| 36 | E | CRUDE STEEL | | RICH | MAR-02 |

**Figure 14: Output Layout of Constraints File**

## 4.4 Writing LP in MPS Representation Using Matrix Generator

A computer programming code which converts the database representation form of LP into an algorithmic form is defined as matrix generator. We have written a matrix generator (MG) with more than 2500 line of programming code to generate MPS representation of the LP. This programming code is equivalent to more then 10000 line of executable code. This MG generates a text file named 'MPS_INPUT_DATA.txt'. This text file contains the algorithmic representation (computer readable) of LP. The text file is used optimizers as in input format to solve the LP.

In this MG program, we have tried addressing some of the drawbacks associated with MGs, as explained by Fourer (1983). The issues of verifiability, document-ability, simplicity, naming of LP components, and ordering of coefficients are attempted to address. With verification, we ensure that the MPS representation of LP is a correct algorithmic representation of the modelers form. The demerits such as dependency on the back-end SLP model, and non-modifiability, are the inherent drawbacks of MGs, and thus could not be addressed.

## 4.5 Loading Optimal Solution Using Report Writer

The program which writes the optimizer generated optimal solution to the database and report it to the users is defined as report writer. Reading the computer generated output is a tedious task for a user as well as a report writer. The optimal solution generated by the optimizer is written in a text file named 'OPTIMAL.txt'. This file is imported to the database to place the solution values at appropriate fields. Several procedures are written to report the optimal solution to the database, and later to the users. More then 500 lines of programming code is written, which is equivalent to more then 2000 line of executable code. Procedures are written in a way to reuse them by calling from other methods.

## 4.6 Soft Capacities

Facilities may be constrained by the capacity units available to them in a period. A situation may arise when the demand to be met is much higher than the capacity of the plant. In such situation the model would show infeasibility. It is difficult to locate such infeasibilities in the solution. To avoid such possibilities, we introduce a concept of soft capacities. While solving the model, if optimizer encounters some infeasibility due to non-availability of the sufficient capacity, an extra capacity for per unit running of the facility is outsourced on market price. The [FacTimeScene] file contains the field for [FacTimeScene]CapOPT for optimal units of facility outsourced $x_{ilt}^{cap}$. The cost per unit of facility capacity outsourcing $c_{ilt}^{cap}$ is stored in the field [FacTimeScene]Vendor_Cost. The cost per unit of facility capacity outsourcing can be set very high if the capacity from external source is not available.

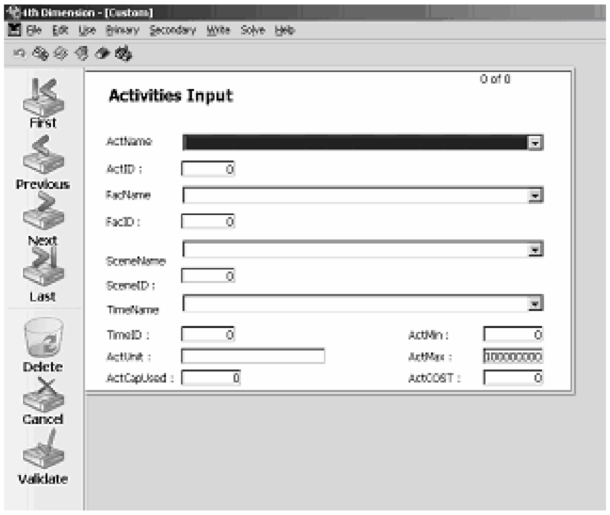
## 5. Data Storage, Retrieval and DSS-Optimizer Interface

Data retrieval and storages procedures are the critical features of this DSS. The core tasks performed by the DSS are generation of variables, constraints, and coefficients files, and thereby the text file in

MPS format; reporting the optimal solution generated by the optimizer back to the database. The DSS work in three different modes Data, Update, and Optimal. The *Data* mode is used for entering and loading of the data. The *Update* mode is used to update the parameter values directly to the variables, constraints, and coefficients files. In *Optimal* mode, a user can see the optimal solution and optimal summary of the cash flows. We describe the issues related to data reporting, data loading, and data update. The optimal summaries and cash flows are discussed in a separate section.

## 5.1 Data Reporting

Parameters of the SLP are entered in the *data* mode. In *data* mode, if any file is opened, an input screen would be shown. Using this screen one can enter the data. For consistency reasons all the alphanumeric data gets automatically converted into uppercase. This task is accomplished immediately before saving into the database. The database has a systematic sequence of entering the data. User need to first, enter the data for six fundamental elements. Once the complete set of data is filled in these files, users need not to touch them any further. This data would automatically be visible in the other input screens with their identification tag and included layouts. This (automatic availability of the data values in other input screen) helps eliminating possible occurrence of inconsistencies in database tables. This also helps eliminating errors due to erroneous data entry. We display an input and output layout of [Activities] file (Figure 15, 16).



Activities Input

0 of 0

ActName

ActID : 0

FacName

FacID : 0

SceneName

SceneID : 0

TimeName

TimeID : 0

ActUnit :

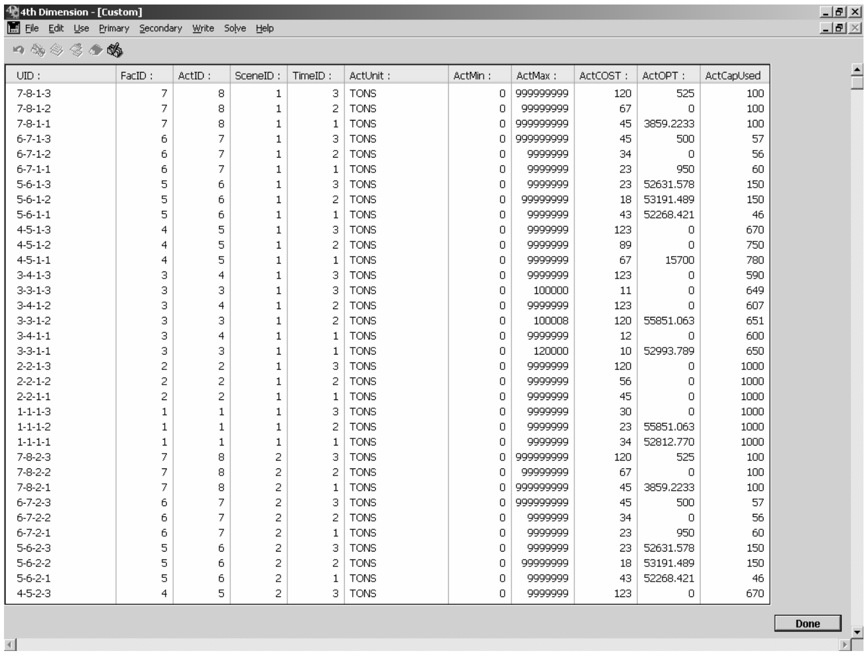
ActCapUsed : 0

ActMin : 0

ActMax : 100000000

ActCOST : 0

Figure 15: Input Layout of Activities File



| UID : | FacID : | ActID : | SceneID : | TimeID : | ActUnit : | ActMin : | ActMax : | ActCOST : | ActOPT : | ActCapUsed |
|---|---|---|---|---|---|---|---|---|---|---|
| 7-8-1-3 | 7 | 8 | 1 | 3 | TONS | 0 | 999999999 | 120 | 525 | 100 |
| 7-8-1-2 | 7 | 8 | 1 | 2 | TONS | 0 | 99999999 | 67 | 0 | 100 |
| 7-8-1-1 | 7 | 8 | 1 | 1 | TONS | 0 | 999999999 | 45 | 3859.2233 | 100 |
| 6-7-1-3 | 6 | 7 | 1 | 3 | TONS | 0 | 999999999 | 45 | 500 | 57 |
| 6-7-1-2 | 6 | 7 | 1 | 2 | TONS | 0 | 9999999 | 34 | 0 | 56 |
| 6-7-1-1 | 6 | 7 | 1 | 1 | TONS | 0 | 9999999 | 23 | 950 | 60 |
| 5-6-1-3 | 5 | 6 | 1 | 3 | TONS | 0 | 9999999 | 23 | 52631.578 | 150 |
| 5-6-1-2 | 5 | 6 | 1 | 2 | TONS | 0 | 99999999 | 18 | 53191.489 | 150 |
| 5-6-1-1 | 5 | 6 | 1 | 1 | TONS | 0 | 9999999 | 43 | 52268.421 | 46 |
| 4-5-1-3 | 4 | 5 | 1 | 3 | TONS | 0 | 9999999 | 123 | 0 | 670 |
| 4-5-1-2 | 4 | 5 | 1 | 2 | TONS | 0 | 9999999 | 89 | 0 | 750 |
| 4-5-1-1 | 4 | 5 | 1 | 1 | TONS | 0 | 9999999 | 67 | 15700 | 780 |
| 3-4-1-3 | 3 | 4 | 1 | 3 | TONS | 0 | 9999999 | 123 | 0 | 590 |
| 3-3-1-3 | 3 | 3 | 1 | 3 | TONS | 0 | 100000 | 11 | 0 | 649 |
| 3-4-1-2 | 3 | 4 | 1 | 2 | TONS | 0 | 9999999 | 123 | 0 | 607 |
| 3-3-1-2 | 3 | 3 | 1 | 2 | TONS | 0 | 100008 | 120 | 55851.063 | 651 |
| 3-4-1-1 | 3 | 4 | 1 | 1 | TONS | 0 | 9999999 | 12 | 0 | 600 |
| 3-3-1-1 | 3 | 3 | 1 | 1 | TONS | 0 | 120000 | 10 | 52993.789 | 650 |
| 2-2-1-3 | 2 | 2 | 1 | 3 | TONS | 0 | 9999999 | 120 | 0 | 1000 |
| 2-2-1-2 | 2 | 2 | 1 | 2 | TONS | 0 | 9999999 | 56 | 0 | 1000 |
| 2-2-1-1 | 2 | 2 | 1 | 1 | TONS | 0 | 9999999 | 45 | 0 | 1000 |
| 1-1-1-3 | 1 | 1 | 1 | 3 | TONS | 0 | 9999999 | 30 | 0 | 1000 |
| 1-1-1-2 | 1 | 1 | 1 | 2 | TONS | 0 | 9999999 | 23 | 55851.063 | 1000 |
| 1-1-1-1 | 1 | 1 | 1 | 1 | TONS | 0 | 9999999 | 34 | 52812.770 | 1000 |
| 7-8-2-3 | 7 | 8 | 2 | 3 | TONS | 0 | 999999999 | 120 | 525 | 100 |
| 7-8-2-2 | 7 | 8 | 2 | 2 | TONS | 0 | 99999999 | 67 | 0 | 100 |
| 7-8-2-1 | 7 | 8 | 2 | 1 | TONS | 0 | 999999999 | 45 | 3859.2233 | 100 |
| 6-7-2-3 | 6 | 7 | 2 | 3 | TONS | 0 | 999999999 | 45 | 500 | 57 |
| 6-7-2-2 | 6 | 7 | 2 | 2 | TONS | 0 | 9999999 | 34 | 0 | 56 |
| 6-7-2-1 | 6 | 7 | 2 | 1 | TONS | 0 | 9999999 | 23 | 950 | 60 |
| 5-6-2-3 | 5 | 6 | 2 | 3 | TONS | 0 | 9999999 | 23 | 52631.578 | 150 |
| 5-6-2-2 | 5 | 6 | 2 | 2 | TONS | 0 | 99999999 | 18 | 53191.489 | 150 |
| 5-6-2-1 | 5 | 6 | 2 | 1 | TONS | 0 | 9999999 | 43 | 52268.421 | 46 |
| 4-5-2-3 | 4 | 5 | 2 | 3 | TONS | 0 | 9999999 | 123 | 0 | 670 |

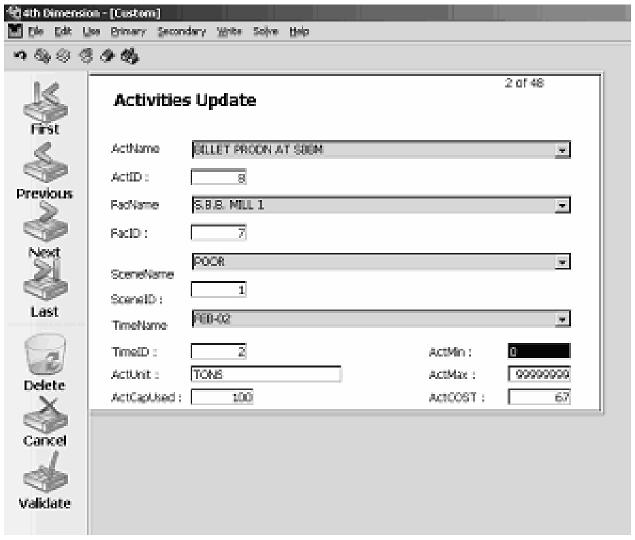
**Figure 16: Output Layout of Activities File**

## 5.2 Data Loading

The DSS allow importing of data from a text files. Data files can be written in a text file using any text editor. Each table of the database can be imported using a separate procedure, specifically written for the table. User is facilitated to import all the files simultaneously in one procedure also. The order of arrangement of the fields in an importing text file must be analogous to the respective database table. The data for different fields must be separated by either a space or a tab. Similarly, each record in the text file must be separated using some character, we use carriage return to separate the records. To maintain a consistency in the data set and also to avoid data entry errors, the imported alphanumeric data automatically get converted into uppercase.

## 5.3 Update Issues

It is observed that the time required to generate the variables, constraints, and coefficients files is much higher in comparison to the solution time of an optimizer. This DSS provide a unique facility of updating the parameters directly to the LP representation. This task is accomplished in *update* mode. The u*pdate* mode is required for updating the parameter values in database files. In this mode opening any file would show the current records of the file (Figure 8, 17). Any change and update of the data would be reflected in the updating file as well as [Variables], [Constraints] and the [Coefficients] file directly. Due to direct update in [Variables], [Constraints] and [Coefficients] files, one need not to generate these files again. It saves a large fraction of optimization time and improves the processing speed of the DSS. Due to direct update to these files one saves on LP generation.



2 of 48

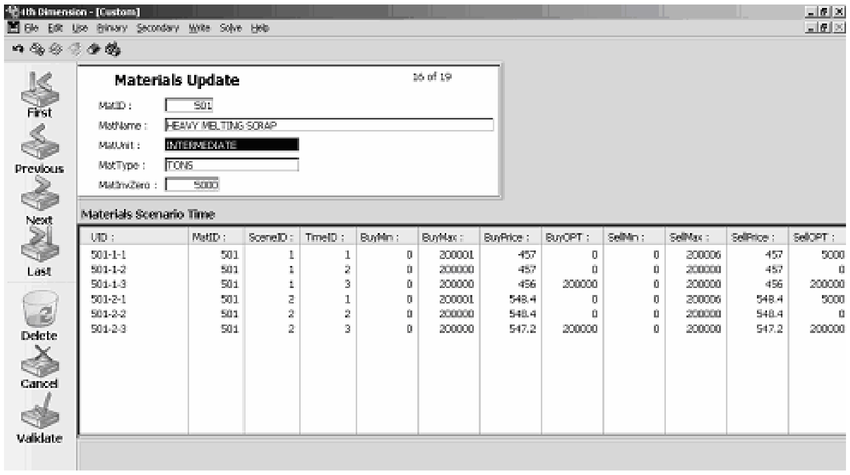
Activities Update

ActName: BILLET PRODN AT SBBM
ActID : 8
FacName: S.B.B. MILL 1
FacID : 7
SceneName: POOR
SceneID : 1
TimeName: FEB-02
TimeID : 2
ActUnit : TONS
ActCapUsed : 100
ActMin : 0
ActMax : 99999999
ActCOST : 67

Figure 17: Update Layout of Activities File

## 5.4 Time-Scenario Dependent Included Layout

In order of user friendliness and better presentation of data, included layout is most important feature of this DSS. Included layout is a unique feature of 4th Dimension, a DBMS, used for the development of this modeling system. In an included layout, the layout of one file can be included in another file. For an example, the materials main layout also displays the time and scenario dependent LP parameters related to materials (Figure 18). Similarly, a facility's main layout displays the time and scenario dependent material-facility input and output LP parameters (Figure 19). We provide included layouts for all sets Materials, Facilities, Activities, Storages. Included layout enables selection and analysis of individual records separately. For security reasons, users are not allowed to modify or update the values using included layouts. Every layout facilitate sorting of records according to the indexed fields of the files.



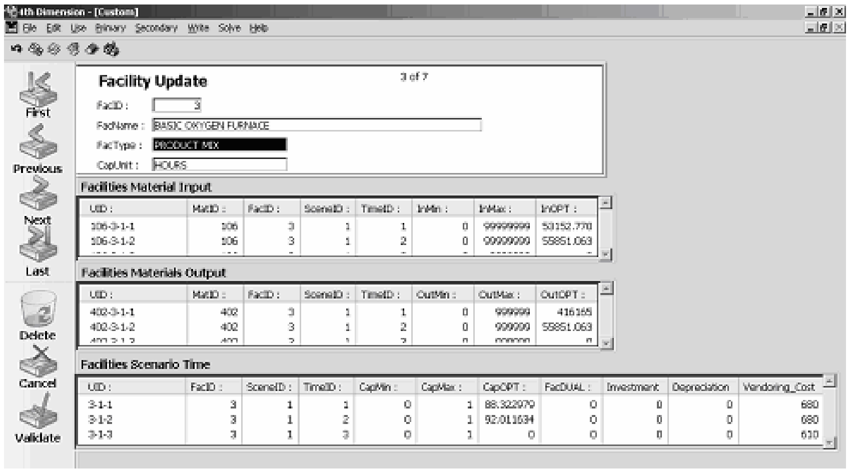
**Materials Update** 16 of 19

| | |
|---|---|
| MatID : | 501 |
| MatName : | HEAVY MELTING SCRAP |
| MatUnit : | INTERMEDIATE |
| MatType : | TONS |
| MatInvZero : | 5000 |

Materials Scenario Time

| UID : | MatID : | SceneID : | TimeID : | BuyMin : | BuyMax : | BuyPrice : | BuyOPT : | SellMin : | SellMax : | SellPrice : | SellOPT : |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 501-1-1 | 501 | 1 | 1 | 0 | 200001 | 457 | 0 | 0 | 200006 | 457 | 5000 |
| 501-1-2 | 501 | 1 | 2 | 0 | 200000 | 457 | 0 | 0 | 200000 | 457 | 0 |
| 501-1-3 | 501 | 1 | 3 | 0 | 200000 | 456 | 200000 | 0 | 200000 | 456 | 200000 |
| 501-2-1 | 501 | 2 | 1 | 0 | 200001 | 548.4 | 0 | 0 | 200006 | 548.4 | 5000 |
| 501-2-2 | 501 | 2 | 2 | 0 | 200000 | 548.4 | 0 | 0 | 200000 | 548.4 | 0 |
| 501-2-3 | 501 | 2 | 3 | 0 | 200000 | 547.2 | 200000 | 0 | 200000 | 547.2 | 200000 |

**Figure 18: Time Scenario Dependent Included Layout of Materials File**



**Facility Update** 3 of 7

| | |
|---|---|
| FacID : | 3 |
| FacName : | BASIC OXYGEN FURNACE |
| FacType : | PRODUCT MIX |
| CapUnit : | HOURS |

Facilities Material Input

| UID : | MatID : | FacID : | SceneID : | TimeID : | InMin : | InMax : | InOPT : |
|---|---|---|---|---|---|---|---|
| 106-3-1-1 | 106 | 3 | 1 | 1 | 0 | 99999999 | 53152.770 |
| 106-3-1-2 | 106 | 3 | 1 | 2 | 0 | 99999999 | 55851.063 |

Facilities Materials Output

| UID : | MatID : | FacID : | SceneID : | TimeID : | OutMin : | OutMax : | OutOPT : |
|---|---|---|---|---|---|---|---|
| 402-3-1-1 | 402 | 3 | 1 | 1 | 0 | 999999 | 416165 |
| 402-3-1-2 | 402 | 3 | 1 | 2 | 0 | 999999 | 55851.063 |

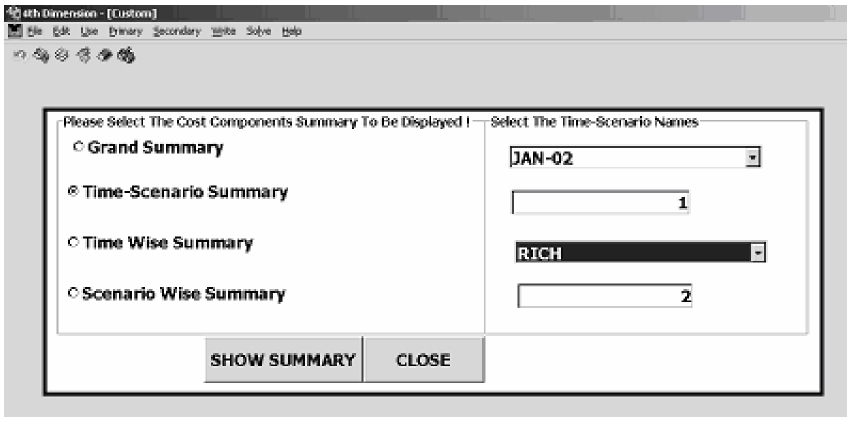
Facilities Scenario Time

| UID : | FacID : | SceneID : | TimeID : | CapMin : | CapMax : | CapOPT : | FacDUAL : | Investment | Depreciation | Vendoring_Cost |
|---|---|---|---|---|---|---|---|---|---|---|
| 3-1-1 | 3 | 1 | 1 | 0 | 1 | 88.322979 | 0 | 0 | 0 | 680 |
| 3-1-2 | 3 | 1 | 2 | 0 | 1 | 92.011634 | 0 | 0 | 0 | 680 |
| 3-1-3 | 3 | 1 | 3 | 0 | 1 | 0 | 0 | 0 | 0 | 610 |

**Figure 19: Time Scenario Dependent Included Layout of Facilities File**

## 5.5 Reporting Optimal Summaries

Reporting optimal solutions, cash flows and optimal summaries of cost components is a complex task. The complexity arises due to the multi-dimensionality of the data such as several cost components, multiple time periods, and multiple scenarios. We provide facility to view cash flow summaries with four options including grand summary, time-scenario summary, time wise summary, and scenario wise summary (Figure 20). We explain the reporting of optimal solution and cash flows in next sections.



Please Select The Cost Components Summary To Be Displayed !

- ○ Grand Summary
- ● Time-Scenario Summary
- ○ Time Wise Summary
- ○ Scenario Wise Summary

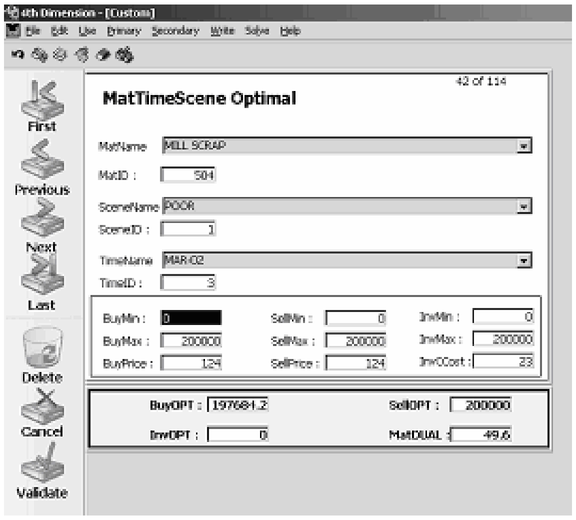
Select The Time-Scenario Names

JAN-02
1
RICH
2

Figure 20: Optimal Summary Reporting Option Layout

### 5.5.1 Reporting Optimal Solution

Optimal solution is reported to the users in *optimal* mode. The optimal values of the decision variables are displayed along with their input parameters (Figure 21). A separate screen along with the input parameters layout is provided to show the optimal values of the variables, reduced costs, and dual values of the constraints. Post optimality analysis can also be performed in the optimal mode for several critical parameters. As a user-friendly tool for strategic planners (to indicate the profit improvement potentials) the dual prices of constraints are also displayed on the output screens.



42 of 114

MatTimeScene Optimal

MatName MILL SCRAP
MatID : 504
SceneName POOR
SceneID : 1
TimeName MAR-02
TimeID : 3

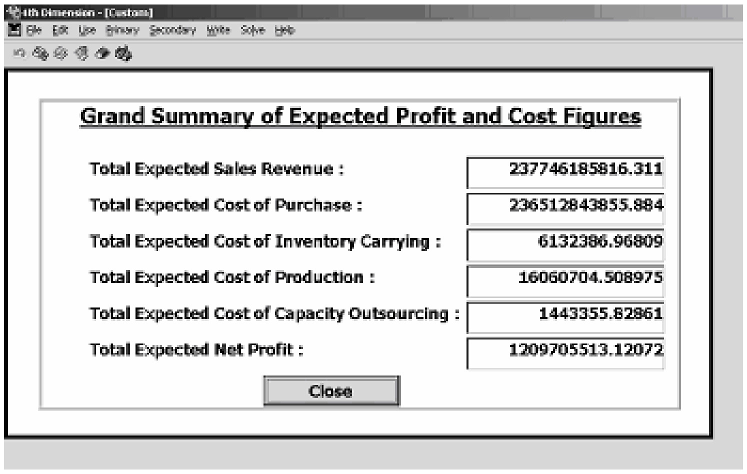
| | | | | | |
|---|---|---|---|---|---|
| BuyMin : | 0 | SellMin : | 0 | InvMin : | 0 |
| BuyMax : | 200000 | SellMax : | 200000 | InvMax : | 200000 |
| BuyPrice : | 124 | SellPrice : | 124 | InvCCost : | 23 |

| | | | |
|---|---|---|---|
| BuyOPT : | 197684.2 | SellOPT : | 200000 |
| InvOPT : | 0 | MatDUAL : | 49.6 |

Figure 21: Optimal Solution Reporting Layout of Material Time Scenario Layout

### 5.5.2 Reporting Cash Flows

We report the cash flows in two forms, nominal cash flows and discounted cash flows. The cash flows are calculated separately for each time and scenario as well as in total as a grand summary (Figure 22). Both the schemes of calculating and reporting are explained below.



Grand Summary of Expected Profit and Cost Figures

| | |
|---|---|
| Total Expected Sales Revenue : | 237746185816.311 |
| Total Expected Cost of Purchase : | 236512843855.884 |
| Total Expected Cost of Inventory Carrying : | 6132386.96809 |
| Total Expected Cost of Production : | 16060704.508975 |
| Total Expected Cost of Capacity Outsourcing : | 1443355.82861 |
| Total Expected Net Profit : | 1209705513.12072 |

**Figure 22: Cash Flow Statement Layout**

#### 5.5.2.1 Nominal Cash Flows

Nominal cash flows are defined as the money flows without considering the time value of money. Nominal cash flow results are generated by maximizing the objective function without applying the interest rate to the sales and cost components. We present the model objective function in nominal form as follows.

$$Z_N \quad = \quad \sum_{l \in L} p_l \sum_{t \in T} Z(l, t)$$

$$Z(l, t) = \sum_{j \in M} c_{jlt}^{sell} x_{jlt}^{sell} - \sum_{j \in M} c_{jlt}^{buy} x_{jlt}^{buy} - \sum_{j \in M} c_{jlt}^{inv} x_{jlt}^{inv} - \sum_{(j,j') \in M^{conv}} c_{jj'lt}^{conv} x_{jj'lt}^{conv} - \sum_{(i,k) \in F^{act}} c_{iklt}^{act} x_{iklt}^{act}$$

$$- \sum_{i \in F} c_{ilt}^{cap} x_{ilt}^{cap}$$

Normally a manager would like to know the sales and cost components in total as well as time and scenario wise. We split this equation into the individual sales and cost components. Following terms of sales and costs are function of time and scenario.

Revenue generate from sales of the finished goods, R (l, t) = $\sum_{j \in M} c_{jlt}^{sell} x_{jlt}^{sell}$

Cost of purchase of raw materials, Cp (l, t) = $\sum_{j \in M} c_{jlt}^{buy} x_{jlt}^{buy}$

Cost of average inventory carrying, Ci (l, t) = $\sum_{j \in M} c_{jlt}^{inv} x_{jlt}^{inv}$

Cost of operating all activities, Ca (l, t) = $\sum_{(i,k) \in F^{act}} c_{iklt}^{act} x_{iklt}^{act}$

Cost of material conversions, Cc (l, t) = $\sum_{(j,j') \in M^{conv}} c_{jj'lt}^{conv} x_{jj'lt}^{conv}$

Cost of capacity outsourcing, Cv (l, t) = $\sum_{i \in F} c_{ilt}^{cap} x_{ilt}^{cap}$

Once we calculate all the six quantities, we can rewrite the net profit as follows:

Z (l, t) = R (l, t) - Cp (l, t) - Ci (l, t) - Ca (l, t) - Cc (l, t) - Cv (l, t)

The terms listed above, can now be represented as a grand summary or individual time and scenario wise cash flows.

### 5.5.2.2 Discounted Cash Flows

The DSS permit maximizing an objective function discounted over future periods. The advantage of the multi-period model is that we can incorporate the time value of money. In a financial analysis if time value of money is considered, we call the results a discounted cash flow. In a discounted cash flow, the user can choose the interest rate. In the case of discounting, the unit of time is very important. If we are planning for small periods, then the effect of discounting on the overall objective function would be insignificant. However, if we are using the DSS for long term strategic purposes, it is essential to take discounting into consideration. (Figure 5.13)

$$Z_D = \sum_{l \in L} p_l (\sum_{t \in T} (1+\rho_l)^{-t} Z(t))$$

The summary statement for each time and the grand summary statement can be converted to the discounted cash flows and discounted summaries.

## 5.6 Capital Budgeting Issues

The model assists in several strategic capital budgeting decisions. It is possible to determine the worth of an investment for capacity expansion. This DSS can help in following ways:

1. Determining the worth of a capacity expansion investment decision. Model can be rerun to solve the problem with addition of new facility capacity. The difference in value of objective function gives the net worth of the new investment.
2. Determining the worth of a sales promotion investment campaign. The expected increase in demand can be forecasted. Again, the model can be rerun with extended limits on units of material to be sold using forecasted demand. The difference in the value of objective function is a clear indication of the worth of this promotion campaign investment.
3. Determining the worth of an outsourced capacity of a facility or a storage area. The model could be rerun with the extended limits on the inventory carrying. The difference in objective function value is the worth of outsourcing a storage area.

Comparison of the differential cash flow with the proposed investment assists finalizing the decision.

## 6. Conclusion and Further Directions

This paper attempted to address the issues in constructing a relational database structure for a LP formulation. Representing an LP in the translations form is explored. We followed the established principles of designing relational database of Fourer (1997). The research gains insights for designing relational databases for generic LP models. We established some generic diagnostic rules of data loading, data reporting and database schema. The rules help avoiding the data entry errors, and eliminate the possibility of infeasibility occurrence due to human errors. It demonstrate that how the multiple dimensional data can be reported into the database as well as to the user.

We recognize that there is ample potential to implement the relational databases in the industry. It requires further studies to explore the issues of relational database construction in context with non linear programming model representation. In our forthcoming paper we will describe the implementation of the DSS, and the stochastic modeling of process industry for strategic planning.

**Appendix**

**Model Formulation**

We first define the data, in six parts: materials, facilities, activities, and storage-areas, scenarios, and times. The notation for the decision variables is then presented. Finally the objective and constraints are described, in both words and formulae.

All quantities of materials are taken to be in the same units, such as kilograms.

**Time data**

$T$= {1… $T$} is the set of time periods in the planning horizon, indexed by t

$\rho_l$ is the interest rate per period in each of the scenario l, taken as zero if there is no discounting

**Materials data**

$M$ is the set of all materials

$l_{jlt}^{buy}$ = lower limit on purchases of material $j$, for each $j \in M$, $l \in L$, and $t \in T$

$u_{jlt}^{buy}$ = upper limit on purchases of material $j$, for each $j \in M$, $l \in L$, and $t \in T$

$c_{jlt}^{buy}$ = cost per unit of material $j$ purchased, for each $j \in M$, $l \in L$, and $t \in T$

$l_{jlt}^{sell}$ = lower limit on sales of material $j$, for each $j \in M$, $l \in L$, and $t \in T$

$u_{jlt}^{sell}$ = upper limit on sales of material $j$, for each $j \in M$, $l \in L$, and $t \in T$

$c_{jlt}^{sell}$ = revenue per unit of material $j$, for each $j \in M$, $l \in L$, and $t \in T$

$l_{jlt}^{inv}$ = lower limit on inventory of material $j$, for each $j \in M$, $l \in L$, and $t \in T$

$u_{jlt}^{inv}$ = upper limit on inventory of material $j$, for each $j \in M$, $l \in L$, and $t \in T$

$v_{j0}^{inv}$ = initial inventory of material *j*, for each $j\in M$

$c_{jlt}^{inv}$ = holding cost per unit of material *j*, for each $j\in M$, l∈ L, and $t\in T$

$M^{conv} \subseteq \{j\in M, j'\in M : j \neq j'\}$ is the set of conversions:

$(j, j')\in M^{conv}$ means that material *j* can be converted to material *j′*

$\alpha_{jj'lt}^{conv}$ = number of units of material *j′* that result from converting one unit of material *j*,

for each $(j, j')\in M^{conv}$, l∈ L, $t\in T$

$c_{jj'lt}^{conv}$ = cost per unit of material *j* of the conversion from *j* to *j′*, for each $(j, j')\in M^{conv}$, l∈ L, $t\in T$

**Facilities data**

***F*** is the set of facilities

$l_{ilt}^{cap}$ = the minimum amount of the capacity of facility *i* that must be used, for each $i\in F$, l∈ L, and

$t\in T$

$u_{ilt}^{cap}$ = the capacity of facility *i*, for each $i\in F$, l∈ L, and $t\in T$

$c_{ilt}^{cap}$ = the cost of vendoring (outsourcing) a unit of capacity at facility *i*, for each $i\in F$,

l∈ L, and $t\in T$

$F^{in} \subseteq FxM$ is the set of facility *i*nputs:

$(i, j)\in F^{in}$ means that material *j* is used as an input at facility *i*

$l_{ijlt}^{in}$ = the minimum amount of material *j* that must be used as input to facility *i*, for

each $(i, j)\in F^{in}$, l∈ L, $t\in T$

$u_{ijlt}^{in}$ = the maximum amount of material *j* that must be used as input to facility *i*, for each $(i, j) \in F^{in}$, $l \in L$, $t \in T$

$F^{out} \subseteq FxM$ is the set of facility outputs:

$(i, j) \in F^{out}$ means that material $j$ is produced as an output at facility $i$

$l^{out}_{ijlt}$ = the minimum amount of material $j$ that must be produced as output at facility $i$,

for each $(i, j) \in F^{out}$, $l \in L$, $t \in T$

$u^{out}_{ijlt}$ = the maximum amount of material $j$ that must be produced as output at facility $i$,

for each $(i, j) \in F^{out}$, $l \in L$, $t \in T$

**Activities data**

$F^{act} \subseteq \{(i, k) : i \in F\}$ is the set of activities:

$(i, k) \in F^{act}$ means that $k$ is an activity available at facility $i$

$l^{act}_{iklt}$ = the minimum number of units of activity $k$ that may be run at facility $i$, for each

$(i, k) \in F^{act}$, $l \in L$, $t \in T$

$u^{act}_{iklt}$ = the maximum number of units of activity $k$ that may be run at facility $i$, for each

$(i, k) \in F^{act}$, $l \in L$, $t \in T$

$c^{act}_{iklt}$ = the cost per unit of running activity $k$ at facility $i$, for each $(i, k) \in F^{act}$, $l \in L$, $t \in T$

$r^{act}_{iklt}$ = the number of units of activity that can be accommodated in one unit of

capacity of facility $i$, for each $(i, k) \in F^{act}$, $l \in L$, $t \in T$

$A^{in} \subseteq \{(i, j, k, t) : (i, j) \in F^{in} (i, k) \in F^{act}, t \in T\}$ is the set of activity inputs:

$(i, j, k, t) \in A^{in}$ means that input material $j$ is used by activity $k$ at facility $i$ during

time period $t$

$\alpha^{in}_{ijklt}$ = units of input material *j* required by one unit of activity *k* at facility *i* in time period *t*, in scenario l, for each l$\in$L, and $(i, j, k, t) \in A^{in}$

$A^{out} \subseteq \{(i, j, k, t): (i, j) \in F^{out}\ (i, k) \in F^{act}, t \in T\}$ is the set of activity outputs: $(i, j, k, t) \in A^{out}$ means that output material *j* is produced by activity *k* at facility *i* during time period *t*

$\alpha^{out}_{ijklt}$ = units of output material *j* produced by one unit of activity *k* at facility *i* in time period *t*, scenario l, for each , l$\in$L , and $(i, j, k, t) \in A^{out}$

**Storage-areas data**

*S* is the set of storage areas

$l^{stor}_{slt}$ = lower limit on total material in storage area s, for each s$\in$*S*, l$\in$L, $t \in T$

$u^{stor}_{slt}$ = upper limit on total material in storage area s, for each s$\in$*S*, l$\in$L, $t \in T$

**Variables**

$x^{buy}_{jlt}$ = units of material *j* bought, for each $j \in M$, l$\in$L, $t \in T$

$x^{sell}_{jlt}$ = units of material *j* sold, for each $j \in M$, l$\in$L, $t \in T$

$x^{stor}_{jslt}$ = units of material *j* in storage area *s*, for each $j \in M$, s$\in$*S*, l$\in$L, $t \in T$

$x^{inv}_{jlt}$ = total units of material *j* in inventory (storage), for each $j \in M$, l$\in$L, $t \in T$

$x^{inv}_{j0}$ = initial inventory of material *j*, for each $j \in M$

$x^{conv}_{jj'lt}$ = units of material *j* converted to material *j′*, for each $(j, j') \in M^{conv}$, l$\in$L, $t \in T$

$x^{in}_{ijlt}$ = units of material *j* used as input by facility *i*, for each $(i, j) \in F^{in}$ , l$\in$L, $t \in T$

$x_{ijlt}^{out}$ = units of material *j* produced as output by facility *i*, for each (*i*, *j*)$\in F^{out}$, l$\in$L, $t\in T$

$x_{iklt}^{act}$ = units of activity *k* operated at facility *i*, for each (*i*, *k*)$\in F^{act}$, l$\in$L, $t\in T$

$x_{ilt}^{cap}$ = units of capacity vendored at facility *i*, for each i$\in F$, l$\in$L, $t\in T$

**First Stage Variables**

$x1_{j}^{buy}$ = units of material *j* bought, for each $j\in M$, in first period

$x1_{j}^{sell}$ = units of material *j* sold, for each $j\in M$, in first period

$x1_{js}^{stor}$ = units of material *j* in storage area *s*, for each $j\in M$, s$\in S$, in first period

$x1_{j}^{inv}$ = total units of material *j* in inventory (storage), for each $j\in M$, in first period

$x0_{j}^{inv}$ = initial inventory of material *j*, for each $j\in M$

$x1_{jj'}^{conv}$ = units of material *j* converted to material *j′*, for each (*j*, *j′*)$\in M^{conv}$, in first period

$x1_{ij}^{in}$ = units of material *j* used as input by facility *i*, for each (*i*, *j*)$\in F^{in}$, in first period

$x1_{ij}^{out}$ = units of material *j* produced as output by facility *i*, for each (*i*, *j*)$\in F^{out}$, in first period

$x1_{ik}^{act}$ = units of activity *k* operated at facility *i*, for each (*i*, *k*)$\in F^{act}$, in first period

$x1_{i}^{cap}$ = units of capacity outsourced at facility *i*, for each i$\in F$, in first period

**Objective**

Maximize the sum over all time periods of revenues from sales less costs of purchasing, holding inventories, converting, operating activities at facilities and vendoring:

$$Z_N \quad = \quad \sum_{l\in L} p_l \sum_{t\in T} Z(l,t)$$

Where,

$$Z(l,t) = \sum_{j\in M} c_{jlt}^{sell} x_{jlt}^{sell} - \sum_{j\in M} c_{jlt}^{buy} x_{jlt}^{buy} - \sum_{j\in M} c_{jlt}^{inv} x_{jlt}^{inv} - \sum_{(j,j')\in M^{conv}} c_{jj'lt}^{conv} x_{jj'lt}^{conv} - \sum_{(i,k)\in F^{act}} c_{iklt}^{act} x_{iklt}^{act}$$

$$- \sum_{i\in F} c_{ilt}^{cap} x_{ilt}^{cap}$$

**Constraints**

For each $j\in M$, l∈ L and $t\in T$, the amount of material $j$ made available by purchases, production, conversions and beginning inventory must equal the amount used for sales, production, conversions and ending inventory:

$$x_{jlt}^{sell} + \sum_{(i,j)\in F^{out}} x_{ijlt}^{out} + \sum_{(j',j)\in M^{conv}} \alpha_{j'jlt}^{conv} x_{j'jlt}^{conv} + x_{jlt-1}^{inv} = x_{jlt}^{sell} + \sum_{(i,j)\in F^{in}} x_{ijlt}^{in} +$$

$$\sum_{(j,j')\in M^{conv}} x_{jj'lt}^{conv} + x_{jlt}^{inv}$$

For each ($i$, $j$)∈ $F^{in}$ , l∈ L and $t\in T$, the amount of input $j$ used at facility $i$ must equal the total consumption by all the activities at facility $i$:

$$x_{ijlt}^{in} = \sum_{(i,j,k,t)\in A^{in}} \alpha_{ijklt}^{in} x_{iklt}^{act}$$

For each ($i$, $j$)∈ $F^{out}$ , l∈ L and t∈ $T$, *t*he amount of output $j$ produced at facility $i$ must equal the total production by all the activities at facility $i$:

$$x_{ijlt}^{out} = \sum_{(i,j,k,t)\in A^{out}} \alpha_{ijklt}^{out} x_{iklt}^{act}$$

For each i$\in F$, l$\in$ L and $t \in T$, the capacity used by all activities at facility $i$ must be within the range given by the lower limit and the upper limit plus the amount of capacity vendored:

$$l_{ilt}^{cap} \leq \sum_{(i,k)\in F^{act}} x_{iklt}^{act} / r_{iklt}^{act} \leq u_{ilt}^{cap} + x_{ilt}^{cap}$$

For each $j\in M$, the amount of material inventoried in the plant before the first time period is defined to equal the specified initial inventory:

$$x_{j0}^{inv} = v_{j0}^{inv}$$

For each $j\in M$, l$\in$ L and $t\in T$, the total amount of material $j$ inventoried is defined as the sum of the inventories over all storage areas:

$$\sum_{s\in S} x_{jslt}^{stor} = x_{jlt}^{inv}$$

For each s$\in S$, l$\in$ L and $t\in T$, the total of all materials inventoried in storage area $s$ must be within the specified limits:

$$l_{slt}^{stor} \leq \sum_{j\in M} x_{jslt}^{stor} \leq u_{slt}^{stor}$$

**Implementability (Non-Anticipativity) Constraints**

$x_{jlt}^{buy} = x1_{j}^{buy}$ for each of the $j\in M$, $l\in L$ and $t = 1$

$x_{jlt}^{sell} = x1_{j}^{sell}$ for each of the $j\in M$, $l\in L$ and $t = 1$

$x_{jslt}^{stor} = x1_{js}^{stor}$ for each of the $j\in M$, $s\in S$, $l\in L$ and $t = 1$

$x_{jlt}^{inv} \quad = \quad x1_{j}^{inv}$ for each of the $j \in M$, $l \in L$ and $t = 1$

$x_{jj'lt}^{conv} \quad = \quad x1_{jj'}^{conv}$ for each $(j, j') \in M^{conv}$, $l \in L$, and $t = 1$

$x_{ijlt}^{in} \quad = \quad x1_{ij}^{in}$ for each $(i, j) \in F^{in}$ , $l \in L$, and $t = 1$

$x_{ijlt}^{out} \quad = \quad x1_{ij}^{out}$ for each $(i, j) \in F^{in}$ , $l \in L$, and $t = 1$

$x_{iklt}^{act} \quad = \quad x1_{ik}^{act}$ for each $(i, k) \in F^{act}$ , $l \in L$, and $t = 1$

$x_{ilt}^{cap} \quad = \quad x1_{i}^{cap}$ for each $i \in F$, $l \in L$, $t = 1$

All variables must lie within the relevant limits defined by the data:

$l_{jlt}^{buy} \leq x_{jlt}^{buy} \leq u_{jlt}^{buy}$, for each $j \in M$, l∈ L and $t \in T$

$l_{jlt}^{sell} \leq x_{jlt}^{sell} \leq u_{jlt}^{sell}$, for each $j \in M$, l∈ L and $t \in T$

$l_{jlt}^{inv} \leq x_{jlt}^{inv} \leq u_{jlt}^{inv}$, for each $j \in M$, l∈ L and $t \in T$

$0 \leq x_{jj'lt}^{conv}$, for each $(j, j') \in M^{conv}$, l∈ L and $t \in T$

$0 \leq x_{ilt}^{cap}$, for each i∈ $F$, l∈ L and $t \in T$

$0 \leq x_{jslt}^{stor}$, for each s∈ $S$, $j \in M$, l∈ L and $t \in T$

$l_{ijl}^{in} \leq x_{ijl}^{in} \leq u_{ijl}^{in}$, for each $(i, j) \in F^{in}$ , l∈ L and $t \in T$

$l_{ijl}^{out} \leq x_{ijl}^{out} \leq u_{ijl}^{out}$, for each $(i, j) \in F^{out}$ , l∈ L and $t \in T$

$l_{ikl}^{act} \leq x_{ikl}^{act} \leq u_{ikl}^{act}$, for each $(i, j) \in F^{act}$ , l∈ L and $t \in T$

**References**

Adams, D. & Beckett, D. (1999), Programming In 4th Dimension, *the Ultimate Guide*, *Automated Solutions Group: Huntington Beach, CA, U.S.A.*

Date, C. J. (1981), Introduction to database systems, 3rd Edition, *Addison-Wesley Publishing Company*

Dennis, D., & Meredith, J. (August 2000), An Empirical Analysis of Process Industry Transformation System, *Management Science*, 46 (8), pp1085 - 1099

Dolk, D. R. (1986), Data as Models: An Approach to Implementing Model Management, *Decision Support Systems*, 2 (1)

Dominguez – Ballesteros, B., Mitra, G., Lucas, C., & Koutsoukis, N- S. (2002), Modeling and Solving Environments for Mathematical Programming (MP): A Status Review and New Directions, *Journal of Operational Research Society*, 53, pp1072 – 1092

Dutta, G., & Fourer, R. (2004), an Optimization-Based Decision Support System for Strategic and Operational Planning In Process Industries, *Optimization and Engineering*, 5, pp 295 – 314

Dutta, G., Sinha, G. P, Roy, P. N, & Mitter, N. (1994), A Linear Programming Model for Distribution of Electrical Energy in a Steel Plant, *International Transactions in Operational Research*, 1 (1), pp 17 – 29

Fourer, R. (1983), Modeling Language Versus Matrix generator for Linear Programming, *ACM Transactions on Mathematical Software*, 9 (2), pp 143 – 183

Fourer, R. (1997), Database Structure for Mathematical Programming Models, *Decision Support System*, 20, pp 317 – 344

Geoffrion, A. M. (1987), an Introduction to Structured Modeling, *Management Science*, 33 (5)

Geoffrion, A. M. (1989), the Formal Aspects of Structured Modeling, *Operations Research*, 31 (1), pp 30 – 51

Geoffrion, A. M. (1989), Computer Based Modeling Environments, *European Journal of Operations Research*, 41, pp 33 – 43

Glover, F., Kingman, D., & Phillips, N. (July – August 1990), Netform Modeling and Applications, *Interfaces*, 20 (4), pp 7 – 27

Jones, C. V. (1992), Attributed Graph – Grammars and Structured Modeling, *Annals of Operations Research*, 38, pp 281 – 324

Jones, C. V. (1990), an Introduction to Graph-Based Modeling Systems, Part I: Overview, *ORSA Journal of Computing*, 2 (2), pp 136 – 151

Jones, C. V. (1991), an Introduction to Graph-Based Modeling Systems, Part II: Graph-Grammars and the Implementation, *ORSA Journal of Computing*, 3 (3), pp 180 – 206

Mitra, G. (1982), UIMP: User Interface for Mathematical Programming, *ACM Transactions on Mathematical Software*, 8 (3), pp 229 – 255

Murphy, F. H., Storh, E. A., & Asthana, A (1992), Representation Scheme for Linear Programming Models, *Management Science*, 38 (7), pp 964 – 991

Robert, F., & David, M. G. (1997), Proposals for Stochastic Programming in the AMPL Modeling Language, *International Symposium on Mathematical Programming Lausanne*

SCH Leung, Y. Wu., & Lai K. K. (2006), A Stochastic Programming Approach for Multi-Site Aggregate Production Planning, *Journal of Operational Research Society*, 57, 123-132

Sen, S. (1999), an Introductory Tutorial on SLP Models, *Interfaces*, 29(2), pp 33 – 61

Welch, J. S. (1987), PAM A – Practitioner's Approach to Modeling, *Management Science*, 33 (5), pp 610 – 625

Wertz, C. J. (1993), Relational Database Design – A Practitioner's Guide, *CRC Press*