# The Aggregate Data Problem: a System for their Definition and Management

M. Rafanelli, A. Bezenchek, L.Tininini

Istituto di Analisi dei Sistemi ed Informatica - CNR, Viale Manzoni 30, 00185 Roma, Italy
e-mail: rafanelli, bezenchek, tininini@iasi.rm.cnr.it

## Abstract

In this paper we describe the fundamental components of a database management system for the definition, storage, manipulation and query of aggregate data, i.e. data which are obtained by applying statistical aggregations and statistical analysis functions over raw data. In particular, the attention has been focused on: (1) a data structure for the efficient storage and manipulation of aggregate data, called ADaS; (2) the graphical structures of the aggregate data model ADAMO for a more user-friendly definition and query of aggregate data; (3) a graphical user interface which enables a straightforward specification of the ADAMO structures; (4) a textual declarative query language to retrieve data from the aggregate database, called ADQUEL.

## 1 Introduction and related works

Over the last few years many researchers have studied the problems related to aggregate (statistical) data modelling and have proposed various data models, operators, query languages and also systems for defining, storing, querying and manipulating such data. Aggregate data are also called *macro data* and are obtained by applying statistical aggregations and statistical analysis functions over raw data (also called *micro data*). The data which enable a correct interpretation of both micro and macro data are usually called *metadata* and their importance has been stressed by several authors (see [StaJ 93]). The relational model and the modified operators of the relational algebra have been the basis of most approaches to aggregate data modelling, but some authors have also experimented alternative approaches by introducing new representation paradigms and operators.

Aggregate data are often represented by tables, but bar-charts, histograms, thematic maps, pie-charts, relations, graphs, are also widespread representation forms. In this paper all the above representations for aggregate data will be denoted by the generic term of *statistical objects*, so as to prescind from the particular representation technique of the same (abstract) data type, namely the statistical object type.

In literature statistical objects have often been described by distinguishing two different types of attributes [ChSh 81], [RaRi 83], [Su 83], i.e.:

(a) one or more *summary attributes*, each representing the result of the application of an aggregation function on micro data, and whose numerical values are called *summary data*;

(b) a set of *category attributes*, which provide a qualitative description of the summary attributes and are part of the above mentioned metadata.

The statistical object establishes a mapping from the domains of the category attributes (independent variable) to the numerical domains of the summary attributes (dependent variable). Since the independent variable ranges over an n-dimensional space (the space of the n-tuples of category attribute instances) the concept of *multidimensionality* of aggregate data were introduced [ShWo 85]. Many authors have proposed to model aggregate data by means of particular relations where the category attributes form the key. In other words, the category attributes C and the summary attributes S are the building blocks of a relation scheme $R = C \cup S$, where the functional dependency $C \rightarrow S$ holds. A statistical table is represented by a relation r over R such that each tuple in r corresponds to an entry in the table. The problems and drawbacks related to the modelling of aggregate data by means of such a "flat" relation scheme have been exposed by several authors. In [BeRT 96b] a new data structure, called ADaS (acronym of **A**ggregate **D**ata **S**tructure), which overcomes many of these problems, has therefore been proposed. One of the innovative features of this structure has been the introduction of a third type of attributes, namely the *implicit* category attributes, which can considerably enlarge and enhance the manipulation capabilities of an aggregate DBMS. Among the metadata of a statistical object we will consider in particular:

- the *aggregation type*, that is the type of the aggregation function applied on micro data (e.g. count, sum, average, etc.) [Tans 87], [RaRi 93];
- the *summary type*, that is the type of the summary attribute (e.g. real, integer, non-negative real, non-negative integer) [Malv 93];
- the *phenomenon* described by the statistical object under consideration (e.g. production, population, income, life-expectancy) [BeMR 94].

One of the predominant operations on aggregate data is the one which allows a category attribute to be "removed" from a given statistical object (obtaining, for example, a "Population by year" from a "Population by year and sex"). Such an operation is often called *aggregation* [ChSh 81], [Su 83], [ShWo 85] and also *Attribute Removal by Aggregation* [OzOM 85], [OzOM 87]. However, in this paper, such an operation will be referred to as *summarization*, as in [RaRi 93], and we will also consider its generalized form, called *G-summarization* [BeRT 96b]. The

summary values obtained by summarizing one or more category attributes are known as *marginal values* [MaMR 91].
In section 2 we present the formal description of the aggregation process and define the Aggregate Data Structure (ADaS). In Section 3 we introduce the graphical structures of the model ADAMO, and show that they can be used both for defining and for querying aggregate data. The system prototype implementation is in progress and the graphical interface of this prototype is outlined in Section 4. Section 5 briefly introduces the declarative aggregate data query language ADQUEL. In Section 6 the main advantages of the ADaS approach with respect to the relational ones are shown and Section 7 concludes.

## 2 The Aggregation Process and the Aggregate Data Structure

In this section we briefly introduce some definitions taken from [BeRT 96b]. Our goal is to provide the reader with the (minimum) necessary formal instruments, in order to avoid a misinterpretation of the following sections and not to provide an exhaustive discussion on the argument.

### 2.1 The Aggregation Process: an Informal Introduction

An aggregation process is a collection of activities that produce a statistical object starting from a given set of raw data. Among these activities we will consider in particular:

- the definition of a *phenomenon of interest* $\mathcal{P}$ (e.g. population, production, income), which selects, through a mapping $\varphi$, a corresponding set of micro data, called *phenomenon universe* $\Omega$;
- the definition of the category attributes and of the corresponding instances; (e.g. Nations with instances Italy, France and Germany);
- the definition of some subsets of $\Omega$, called *aggregation sets*, characterized by the category attributes and the instances expressed above; the collection of all aggregation sets of a given statistical object will be denoted by $\mathcal{A}$;
- the application of an aggregation function F (e.g. count, average) to the above defined aggregation sets, which associates a numerical (summary) value to each of them;
- the (optional) application of a *composition function* to the newly created statistical object, together with one or more statistical objects already in the database, to produce a *composite* statistical object (a statistical object representing two or more distinct aggregation processes).

### 2.2 Base Definitions

***Definition 1*** Let $\Omega$ be a phenomenon universe, C be a category attribute and i one of its instances. The logical predicate (C=i) defined on the micro data of $\Omega$ is called *base predicate.*

***Definition 2*** The *base set* of the base predicate (C=i) is the subset of $\Omega$ consisting of all micro data which satisfy the base predicate. In the following, such subset will be denoted by $\mathfrak{B}_{C=i}$.

The single aggregation sets are always expressible by means of unions and intersections among base sets. However the following definitions allow a more compact expression of the entire set $\mathcal{A}$ to be obtained.

***Definition 3*** The *classification set* $\mathfrak{C}_C$ of the category attribute C with domain $\mathfrak{D}(C)$ is the set whose elements are the base sets defined on $\Omega$ by C:

$$\mathfrak{C}_C = \left\{ \mathfrak{B}_{C=i} : i \in \mathfrak{D}(C) \right\}$$

***Definition 4*** We denote by $\mathfrak{U}_C$ the *union of all base sets* defined on $\Omega$ by the instances of the category attribute C:

$$\mathfrak{U}_C = \bigcup_{i \in \mathfrak{D}(C)} \mathfrak{B}_{C=i}$$

and we will call *union set* of C the set $\{\mathfrak{U}_C\}$

***Definition 5*** Let H and K be two sets whose elements are sets; we define the *cross intersection* of H and K, and denote it by $H \otimes K$, as the set of all the possible intersections between each element in H and each element in K:

$$H \otimes K = \{ j : j = h \cap k, h \in H, k \in K \}$$

In order to clarify the practical impact of these definitions, let us consider an example taken from [BeRT 96b], a simple table representing cereal production data in some European countries in the period 1980-'81 (Table 1).

**Table 1.** A simple statistical table

| Cereal production in the period 1980-'81 | wheat | rice |
|---|---|---|
| Italy | val_1 | val_2 |
| France | val_3 | val_4 |
| Germany | val_5 | val_6 |

The phenomenon in consideration is "production". The set $\mathcal{A}$ of the aggregation sets can be concisely expressed in terms of cross intersections among classification and union sets:

$\mathcal{A} = \{A_1, A_2, A_3, A_4, A_5, A_6\} =$

$= \mathfrak{C}_{country} \otimes \mathfrak{C}_{cereal} \otimes \{\mathfrak{U}_{year}\}$

with $\mathfrak{D}$ (country) = {Italy, France, Germany},

$\mathfrak{D}$ (cereal) = {wheat, rice},

$\mathfrak{D}$ (year) = {1980, 1981}.

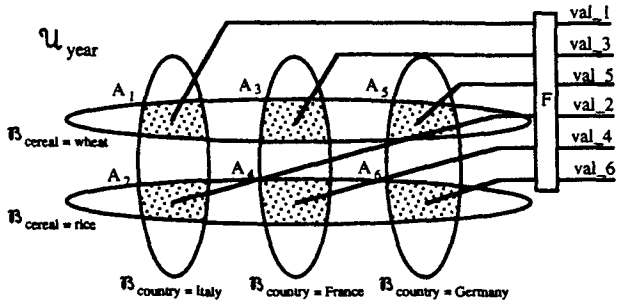


**Fig. 1.** Generation of the summary values by applying F on the aggregation sets

As shown in Fig. 1, the summary function associates a summary value to each aggregation set in $\mathcal{A}$ (for simplicity $\mathcal{U}_{year}$ is not explicitly drawn and the entire plan of the sheet is considered to represent $\mathcal{U}_{year}$).
The attributes like "year" are defined *implicit*, since they do not explicitly express a classification in the statistical object, though they are necessary for a correct definition of the aggregation sets. Note that an implicit category attribute takes part in the definition of $\mathcal{A}$ with its union set, rather than with its classification set, the constraint it expresses being the same for all aggregation sets in the statistical object, regardless of the number of its instances.

### 2.3 The Aggregate Data Structure (ADaS)

In the following we give the formal definitions of simple, complex and composite ADaS.

***Definition 6*** A *simple* ADaS is a data structure defined by the six-tuple $< \mathcal{P}, \mathcal{N}, \mathcal{T}, \mathcal{E}, \mathcal{I}, \mathcal{S} >$, where:

- $\mathcal{P}$ is the phenomenon described by the ADaS.
- $\mathcal{N}$ is the numerical domain where the summary attribute is defined (e.g. $\mathbb{R}$, $\mathbb{Z}$, $\mathbb{Z}^+$).
- $\mathcal{T}$ is the aggregation type (e.g. count, sum, average).
- $\mathcal{E}$ is the ordered set of the *explicit category attributes*, each of them with its corresponding ordered instance domain (M represents the number of explicit category attributes, while $P_j$ ( j = 1, ... , M ) the number of instances of the j-th attribute):
  $\mathcal{E} = \{ < E_1, \{ i_{E_1,1}, i_{E_1,2}, , ... , i_{E_1,P_1} \} >, \ ...$
  $< E_M, \{ i_{E_M,1}, i_{E_M,2}, , ... , i_{E_M,P_M} \} > \}$
- $\mathcal{I}$ is the ordered set of the *implicit category attributes*, each of them with its corresponding ordered instance domain (N represents the number of implicit category attributes, while $Q_j$ ( j = 1, ... , N ) the number of instances of the j-th attribute):
  $\mathcal{I} = \{ < I_1, \{ i_{I_1,1}, i_{I_1,2}, ... , i_{I_1,Q_1} \} >, \ ...$
  $< I_N, \{ i_{I_N,1}, i_{I_N,2}, ... , i_{I_N,Q_N} \} > \}$
  The collection $\mathcal{A}$ of the aggregation sets is expressed by:
  $\mathcal{A} = \mathcal{C}_{E_1} \otimes ... \ \mathcal{C}_{E_M} \otimes \{ \mathcal{U}_{I_1} \} \otimes ... \{ \mathcal{U}_{I_N} \}$
  and consists of $P_1 * P_2 * ... * P_M = (\Pi_{(j=1,M)} P_j)$ elements (aggregation sets).
- $\mathcal{S}$ is an ordered set of $(\Pi_{(j=1,M)} P_j)$ *summary values* in bijective correspondence with the aggregation sets of $\mathcal{A}$. More precisely, the function which maps from the generic aggregation set of $\mathcal{A}$
  $$\mathcal{B}_{E_1=i_{E_1,h_1}} \cap ... \cap \mathcal{B}_{E_M=i_{E_M,h_M}} \cap \mathcal{U}_{I_1} \cap ... \cap \mathcal{U}_{I_N}$$
  to the corresponding k-th element of $\mathcal{S}$ is expressed by the following formula:
  $k = (h_1 - 1) * (\Pi_{(j=2,M)} P_j) + (h_2 - 1) * (\Pi_{(j=3,M)} P_j) + ... + (h_{M-1} - 1) * P_M + h_M$

A complex ADaS needs to be introduced when the aggregation sets of a statistical object are not simply expressible as a cross intersection among classification and union sets:

***Definition 8*** A *complex* ADaS is a data structure defined by the six-tuple $< \mathcal{P}, \mathcal{N}, \mathcal{T}, \underline{\mathcal{E}}, \underline{\mathcal{I}}, \underline{\mathcal{S}} >$, where the former three components have already been described in Definition 7, while the latter three are *ordered sets* having the same cardinality R:

- $\underline{\mathcal{E}} = \{ \mathcal{E}_1, \mathcal{E}_2, ..., \mathcal{E}_R \}$
- $\underline{\mathcal{I}} = \{ \mathcal{I}_1, \mathcal{I}_2, ..., \mathcal{I}_R \}$
- $\underline{\mathcal{S}} = \{ \mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_R \}$

The single members $\mathcal{E}_j$, $\mathcal{I}_j$, $\mathcal{S}_j$ (j=1, ..., R) of each set are ordered sets of the same type described for a simple ADaS.

A composite ADaS needs to be introduced when a statistical object collects data from two or more aggregation processes (*composite* statistical object):

***Definition 9*** A *composite* ADaS is a data structure defined by the ordered set $\{\alpha_1, \alpha_2, ..., \alpha_S\}$, where each member $\alpha_j$ is a complex (possibly simple) ADaS.

## 3 The Graphical Definition and Query of Aggregata Data: the ADAMO Tree-Structures

The ADaS solves many problems related to the description and logical definition of aggregate data, nevertheless its nested structures may be rather complex to understand from a user's point of view. Consequently, a more user-friendly (possibly graphical) representation needs to be introduced, which allows comprehension of the statistical object "at a glance", maintaining, however, the correspondence with the logical representation provided by the ADaS.
Several authors have proposed hierarchical structures (see for example the models Subject [ChSh 81], SAM* [Su 83] and Storm [RaSh 90]) to provide a semantic representation of statistical objects, but none of them clearly establish a distinction between implicit and explicit category attributes. Consequently, our approach has been based on the tree-structures of the model for aggregate data ADAMO (acronym of Aggregate DAta MOdel) [BeRT 96a], where such a distinction is clearly stated. In ADAMO the conceptual description of statistical objects is achieved by means of tree-structures with variously shaped nodes, each of them in strict correspondence with one of the concepts introduced in the previous section, namely:

(1) a *simple circle node* ( ○ ) represents a classification set (and therefore an explicit category attribute);
(2) a *slashed circle node* ( ∅ ) represents a union set (and therefore an implicit category attribute);
(3) a *triangle node* ( △ ) represents a cross-intersection among its child-nodes (note that the associativity of the operator ⊗ allows a triangle node to have more than two child-nodes);
(4) a *butterfly node* ( ⋈ ) represents the set-union among its child-nodes;
(5) a *square node* ( □ ) represents the mapping f between the aggregation sets and the corresponding summary values, and also several other properties regarding the statistical object, such as: a) the phenomenon; b) the aggregation

type; c) the summary type; d) the information source; e) the unit of measure; f) the counting unit.

For example, let us reconsider the Table 1; in Subsection 2.2 we showed that the set $\mathcal{A}$ of the aggregation sets is expressed by $\mathcal{A} = \mathcal{C}_{country} \otimes \mathcal{C}_{cereal} \otimes \{\mathcal{U}_{year}\}$. Hence such a table can be graphically represented as in Fig. 2(a). Fig. 2(b) stresses the semantics of the various nodes. The advantages of this graphical representation are more evident when the ADaS is complex or composite. For instance, let us consider an example of a bar-chart reported in Fig. 3.

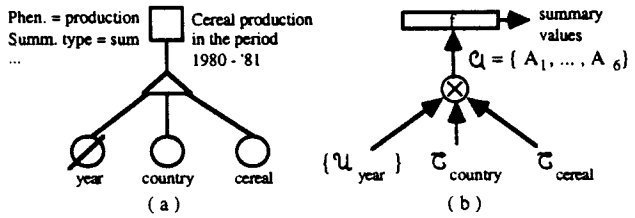


**Fig. 2.** The ADAMO tree-structure of the Table 1

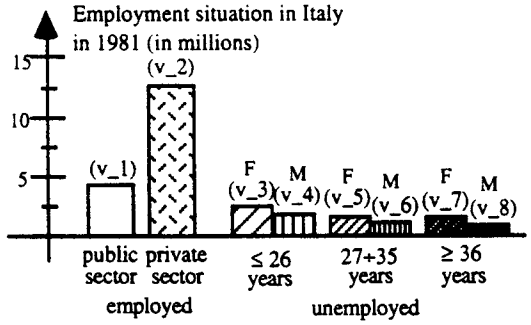


**Fig. 3.** Bar-chart of a complex statistical object

The correspondent ADAMO structure is shown in Fig. 4. Note that some of the edges linking the terminal nodes have an orthogonal dash: such a notation represents the summarizability of the corresponding (explicit) category attribute. In order to clarify the advantages with respect to the ADaS description of the same object, let us consider the corresponding ADaS representation.

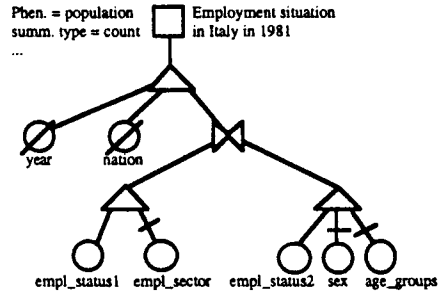


**Fig. 4.** The ADAMO tree-structure of the complex statistical object of Fig. 3

$\mathcal{A}$ can be expressed by $\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2$, where:

$\mathcal{A}_1 = \mathcal{C}_{empl\_status1} \otimes \mathcal{C}_{empl\_sector} \otimes \{\mathcal{U}_{year}\} \otimes \{\mathcal{U}_{nation}\}$

and

$\mathcal{A}_2 = \mathcal{C}_{empl\_status2} \otimes \mathcal{C}_{sex} \otimes \mathcal{C}_{age\_groups} \otimes \otimes \{\mathcal{U}_{year}\} \otimes \{\mathcal{U}_{nation}\}$.

Consequently:

$\alpha$ = < population, $\mathbb{Z}^+$, count,
{ { < empl_status1, { employed } >,
< empl_sector, { public, private } > },
{ < empl_status2, { unemployed } >,
< sex, { F, M },
< age_groups, { ≤26, 27+35, ≥ 36 } > } },
{ { < year = { 1981 } >,
< nation = { Italy } > },
{ < year = { 1981 } >,
< nation = { Italy } > } },
{ { v_1, v_2 },
{ v_3, v_5, v_7, v_4, v_6, v_8 } } >

What makes the ADAMO tree-structures particularly interesting is that they can be used both as the Data Definition Language and as the Query Language of the system. When the ADAMO structures are used to query the database, the summary values, and possibly the instances of some explicit category attributes, remain unspecified. For example, by specifying the instance sets of the attributes year, country, and cereal in the structure of Fig. 2(a) the user gives a conceptual description of the Table 1. If they want to insert a new statistical object having the indicated structure, the user will also introduce the corresponding summary values, along with some additional information regarding the visualization of the object, e.g. that it is displayed in tabular form with the attribute "country" in rows and the attribute "cereal" in columns. If, instead, the user wants to extract a statistical object with the indicated structure from the aggregate database, they will leave the summary values unspecified and the system will try to retrieve the requested information. In this case some of the instance sets of the circle nodes may also be unspecified, because the user normally needs a certain classification, but does not want to express explicitly the single corresponding values. For example, if we need population data about USA in 1994 classified by sex and age_groups, we would prefer not to insert explicitly both the instances of sex, as they are obvious, nor the instances of age_groups, since they may mismatch the available classification in the database.

## 4 The ADAMO Structure Editing Graphical Interface

Since the ADAMO tree-structures are fundamental for both defining and querying the aggregate database, the system needs a simple and effective user interface to insert, retrieve and modify these structures. In Fig. 5 the structure editing window of the system prototype is shown (the displayed structure is that of the bar-chart in Fig. 3). Notice that the user can choose the instances of the category attributes that has to be shown or hidden within the window: in this case the instances displayed are those of nation and sex.

The creation of an ADAMO structure from scratch may be difficult for "occasional" users, but these users will seldom (if ever) have to create one, rather they will edit the structure of an already inserted statistical object. They will usually browse the database, examine some of the available statistical objects and display them in their favourite graphical form.

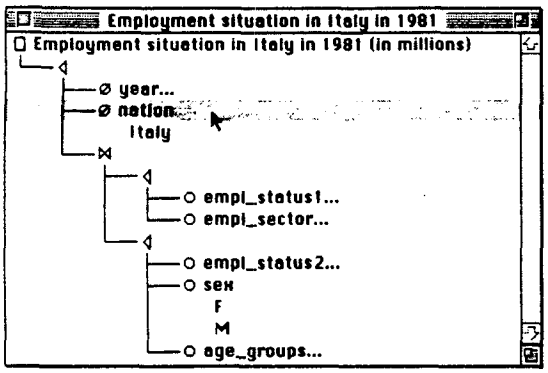


**Fig. 5.** ADAMO tree-structure editor

When an object similar to the one desired is found, they will extract its structure, modify it (for example by replacing category attribute "age_groups" with "quali-fication" and removing classification by sex) and finally query the system to obtain such an object from the database.
As stated above, the summary values (and possibly some category attribute domains) are left unspecified. The system will fill them with the available summary values and category attribute instances retrieved from the database. Once the data have been collected and/or calculated, the user can display them in the default tabular form proposed by the system, or by opportunely reorganizing the rows and columns or even in another representation form.

## 5 ADQUEL

In Section 3 and 4 we have shown that the ADAMO structures provide an effective way to formulate queries to the aggregate DBMS. However, users already accustomed to formulating queries in a traditional declarative language like SQL may prefer to express their query in textual form. ADQUEL is the response to such requirements.
ADQUEL is a declarative query language for aggregate data that arises from the formalization of the aggregation process exposed in Section 2, in particular from the concepts of phenomenon, classification set, union set and aggregation set. The query is formulated by specifying the phenomenon under consideration and the constraints defining the desired aggregation sets.
The domain of a category attribute (either in explicit or implicit form) is declared with a statement of the form (in the following definitions the literals are denoted in boldface):
<attribute-domain-declaration> ::=
**domain of** <attribute> **is** <list-of-instances>
where
<list-of-instances> ::= <instance> {, <instance>}
The basic form of an ADQUEL query is
<ADQUEL-query> ::= **adretrieve** (<phenomenon>)
**for asets =** <attribute-expression>
[**where** <condition>]
The attributes in the <attribute-expression> may appear in explicit or implicit form. In the (default) explicit form, an attribute name represents the corresponding classification set, while implicit (<attribute>) represents the union set.
The attributes in the <attribute-expression> are connected by the '*' and '+' operands, representing the cross intersection and the set union, respectively.
The <condition> may involve other ADaS components, for example the information source and may express that only data from the specified information source(s) are admissable.
The query to retrieve the bar-chart of Fig. 3, with the additional condition that the information source must be OCSE, is the following:

**domain of** year **is** 1981
**domain of** nation **is** "Italy"
**domain of** empl_status1 **is** "employed"
**domain of** empl_status2 **is** "unemployed"
**domain of** empl_sector **is** "public, private"
**adretrieve** (population)
**for asets = implicit** (year) * **implicit** (nation) *
((empl_status1 * empl_sector) +
(empl_status2 * sex * age_groups))
**where infosource** = "OCSE"

Note that the domains of sex and age_groups have been left unspecified, so that they will be filled with the domains of the retrieved ADaS.

## 6 ADaS vs. Relational Approach

Let us reconsider the Table 1 and suppose that another table is also available representing the corn production classified by country (Italy, France and Germany) and by year (1980 and 1981). Some distinct relational representations of such a table have been proposed in literature. The simplest approach is that of considering a relation (or a G-relation, as in [Su 83]) named "Cereal_production_in_the_period_ 1980_1981" on the relation scheme (country, cereal, prod_value) and a relation named "Corn_production" on the (necessarily distinct) relation scheme (country, year, prod_value). If the user needs a table representing cereal production in 1980-1981 (global value) classified by cereal (wheat, rice and corn) and country (Italy and France), all required data are stored in the database, but the system lacks the necessary information to automatically perform the data retrieval process. The user will have to browse the entire database (probably millions of tables), performing the necessary summarizations, creating intermediate relations and finally joining the obtained results properly, for example they may express this intermediate query to extract the needed values from the second table:

SELECT SUM(prod_value)
FROM Corn_Production
WHERE country = Italy OR country = France
GROUP BY country

In order to enhance the automatic manipulation capabilities of the system, some authors have proposed the introduction of universal schemes. For example, a universal scheme for the above mentioned tables is (country, year, cereal, prod_value). Both tables will be in the same relation defined on such a relation scheme. There are several problems related to universal schemes: the relation scheme must include all possible classifying attributes for the considered summary attribute and this implies large redundancies and null-value problems. Moreover, as stressed in [Malv 93], the

correctness of the obtained results can be guaranteed only if all tables are obtained from the same *population* of micro data and if some other assumptions hold. For this reason Malvestuto suggests collecting only *homogeneous* tables (i.e. regarding the same population) in a single relation. In this way, however, the formulation of the query becomes difficult, since a database may contain thousands of populations and the user has to know their names to correctly formulate the query. The ADaS allows us to determine in an automatic way the population $\pi$ of the object, which is expressed by:

$$\pi = \mathcal{U}_{E_1} \cap ... \cap \mathcal{U}_{E_M} \cap \mathcal{U}_{I_1} \cap ... \cap \mathcal{U}_{I_N}$$

Moreover, each statistical object is associated with a single ADaS and this allows a more precise expression of the manipulation metadata connected with such an object, above all of its summarizability characteristics (e.g. that the "male" and "female" values can be summarized to obtain the correct marginal values). The user simply needs to express the desired information, without being concerned with *how* to obtain it. The ADQUEL query to retrieve the desired data is as follows:

**domain of** year **is** 1980, 1981
**domain of** cereal **is** "wheat", "rice", "corn"
**domain of** country **is** "Italy", "France"
**adretrieve** (production)
**for asets = implicit** (year) * cereal * country

The system will extract all ADaSs having $\mathcal{P}$ = production and also the specified attributes and instances in the $\mathcal{C}$ and $\mathcal{I}$ components. The necessary operations will be performed according to the manipulation metadata expressed for each extracted ADaS. The major disadvantage of Aggregate Data Structures with respect to relations is the less-immediate mapping with the physical storage, since existing relational DBMS do not support the needed complex data types.

## 7 Conclusions

This paper presents the fundamental components of a database management system for the definition, storage, manipulation and query of aggregate data. We have introduced a data structure for the efficient storage and manipulation of aggregate data, called ADaS, and have shown that it is preferable with respect to relational approaches. In order to obtain a more user-friendly definition and query of aggregate data, the graphical structures of the ADAMO model have been introduced. The graphical user interface which enables the definition and editing of the ADAMO structures has been presented and it is one of the most innovative features of the implemented prototype. This prototype is in progress on a Macintosh computer using the MPW programming environment, the C++ language and a commercial relational DBMS library. We have also proposed a textual declarative query language, called ADQUEL, which enables an alternative formulation of the ADAMO graphical queries.

## Bibliography


[BeMR 94] A.Bezenchek, F.Massari, M.Rafanelli, "STORM$^+$: Statistical Data Storage and Manipulation System" in *11th Symp. on Computational Statistics, Compstat '94*, Vienna, Austria, Aug. 22-26, 1994.

[BeRT96a] A.Bezenchek, M.Rafanelli, L.Tininini, "ADAMO: A Conceptual Model for the Graphical Definition and Manipulation of Aggregate Data", Technical Report IASI, July 1996, in press

[BeRT96b]A.Bezenchek, M.Rafanelli, L.Tininini, "A Data Structure for Representing Aggregate Data", *Proceed. 8th Intern. Confer. on SSDBM*, Stockholm, June 18-20, 1996, IEEE Press.

[ChSh 81] P.Chan, A.Shoshani, "SUBJECT: A Directory Driven System for Organising and Accessing Large Statistical Databases", *Proceed. of the 7th Intern. Confer. on Very Large Data Bases*, 1981, Cannes, France, Pages 553-563

[Malv 93] F.M.Malvestuto, "A universal-scheme approach to statistical databases containing homogeneous summary tables, *ACM Transactions on Database Systems*, Vol. 18, No.4, Dec. 1993, pp. 678-708.

[MaMR91]F.M.Malvestuto, M.Moscarini, M.Rafanelli, "Suppressing marginal cells to protect sensitive information in a two-dimensional statistical table", *Proceed. 10th ACM Symposium on Principles of Database Systems - PODS '91*, ACM Press, Denver, Colorado, May 29-31, 1991.

[OzOM85] G.Ozsoyoglu, M.Ozsoyoglu, F.Mata, "A language and a physical organization technique for summary tables", *Proceed. 4th ACM International Conference on Management of Data - SIGMOD '85*, Austin, Texas, May 28-31, 1985, pp. 3-16.

[OzOM87] G.Ozsoyoglu, M.Ozsoyoglu, V.Matos, "Extending Relational Algebra and Relational Calculus with Set-valued Attributes and Ag-gregate Functions" *ACM Trans. on Database Syst.*, 1987, Vol. 12, No. 4, pp. 566-592.

[RaRi 83] M.Rafanelli, F.L.Ricci, "Proposal of a Logical Model for Statistical Data Base", *Proc. 2th Int. Work. on Statistical Database Management*, Los Altos, CA, 1983.

[RaRi 93] M.Rafanelli, F.L.Ricci, "Mefisto: a functional model for statistical entities", *IEEE Trans. on Knowledge and Data Engin.*, Aug. 1993, V.5, N.4.

[RaSh 90] M.Rafanelli, A.Shoshani, "STORM: A STatistical Object Representation Model", *Proc. 5th Int. Confer. on SSDBM, Charlotte, NC, April 1990, Lecture Notes in Computer Science*, Springer Verlag Pub., Vol. 420.

[ShWo 85] A.Shoshani, H.K.T.Wong, "Statistical and Scientific Database Issues" *IEEE Trans. on Software Engineering*, October 1985, Vol.SD-11, No.10.

[StaJ 93] *Statistical Journal of the United Nations ECE*, Vol. 10, Number 2, IOS Press, pp. 91-226.

[Su 83] S.Y.W.Su, "SAM*: A Semantic Association Model for Corporate and Scientific Statistical Databases", *J. of Information Sciences*, Vol. 29, No. 2-3, May-June 1983, Pages 151-199.

[Tans 87] A.V.Tansel, "A Statistical Interface for Historical Relational Databases", *Proc. 3th Int. Conf. on Data Eng.*, Los Angeles, Febr. 3-5, 1987, Comp. Soc. Press, pp. 538-546.