Summarizability in OLAP and Statistical Data Bases

Hans-J. Lenz[1] and Arie Shoshani
Lawrence Berkeley National Laboratory
hjlenz@CCMailer.WiWiss.FU-Berlin.DE, shoshani@lbol.gov

Abstract

Summarizability of OLAP and Statistical Databases is an a extremely important property because violating this condition can lead to erroneous conclusions and decisions. In this paper we explore the conditions for summarizability. We introduce a framework for specifying precisely the context in which statistical objects are defined. We use a three step process to define normalized statistical objects. Using this framework, we identify three necessary conditions for summarizability. We provided specific tests for each of the conditions that can be verified either from semantic knowledge, or by checking the statistical database itself. We also provide the reasoning for our belief that these three summarizability conditions are sufficient as well.

1. Introduction

The purpose of both OLAP databases and Statistical Data Bases (SDBs) is to provide a framework for performing summarization operations on multidimensional databases. In most respects relevant to this paper OLAP and SDBs are similar in their structure, and therefore we use the two terms synonymously here. However, we point out a few subtle differences in the paper where appropriate. In order to introduce some basic terminology consider a typical example of an SDB: the population (i.e. number of residents) by counties by race by age by sex for a certain year. The multidimensional space in this example refers to counties, race, sex and age (of dimensionality 4), and the measured element is the population. The summarization operation (in this case, sum) can then be applied to each of these dimensions so as to reduce the dimensionality of the multidimensional space. For example, summarizing over race and sex produces the data set: the population by counties by age, which has a dimensionality smaller by 2 than the previous data set. Similarly, one can summarize this data set over all counties of each state to produce the population of states by age. Note that in this case, we did not reduce the dimensionality of the multidimensional space, because the summarization was done along the same dimension (i.e. counties and states are both geographical dimensions), but the granularity of this dimension has increased to the state level. The relationship between "states" and "counties" is referred to as a "classification structure". In section 3, we will describe a model that captures these concepts more precisely.

The purpose of this paper is to explore under what conditions such summarization operations produce the correct result. We call such conditions summarizability conditions. To illustrate this point, consider the case in the previous example, where the SDB contained population of cities (instead of counties) by race by age by sex for a certain year. If we summarize the population to the state level, this will produce an incorrect result, since states usually includes regions other than cities, such as farms and villages.

In a previous paper [3] the concept of summarizability was introduced, and two necessary conditions for summarizability were identified. These two conditions were rather limited in that they were applied over classification structures only. In this paper, we further apply these conditions to elements of the multidimensional space. In addition, we find another important condition that along with the previous conditions provides a much richer framework for determining summarizability. We discuss these conditions in detail and use examples to illustrate these concepts. We argue informally that the three condition together are also sufficient for summarizability.

Summarizability is an extremely important property, in that an incorrect (and not always obvious) summarization can lead to erroneous analysis, conclusions, and decisions. Thus, it is necessary to inform users when they attempt to perform a non-summarizable operation. We will show that a data set may be summarizable relative to some

[1] This work was performed while on leave from Free University Berlin, visiting Lawrence Berkeley National Laboratory

dimensions but not others. We will show further that summarizability depends on the type of summary measure and the summarization function. To introduce the problems we start in section 2 with several examples where summarizability does not hold; that is, if the summarization operation is applied, then the result is incorrect. In section 3, we describe a model for representing summary data as the basic structure on which summarizability can be tested. In section 4, we introduce two necessary conditions for summarizability, and show that these conditions apply to both the classification structures and the multidimensional space. In section 5, we describe a third condition, which depends on the type of measured elements. We show that the type of element as well as the summarization function can determine whether an operation is summarizable. In section 6, we discuss dependency analysis over summary data that may be erroneous even when summarizability conditions hold. Finally, we discuss future work in section 7, and provide informal reasons as to why we believe the three summarizability conditions are necessary and sufficient. We end with a conclusion section.

2. Illustrative examples

In this section we discuss several examples that illustrate non-summarizability. Our main purpose is to expose various conditions that are required for summarizability.

2.1 Example 1

Consider a data set that provides a count of the number of graduate students in a certain university organized by department and year, as shown in figure 1. Suppose that we wish to find out the number of students that attended each department over the 3 year period. Adding the figures for each year (e.g. 15+17+13=45) is incorrect if students attended the department for more than one year, which is usually the case. Indeed the column of totals on the right hand side of the table reflects the fact that the departments started the graduate program in 1994, and each student attended a 2 year program. Thus, in the computer science department there were 15 students in 1994 who also attended in 1995. Consequently, there were only 2 new students in 1995, for a total of 17 students. These two students continued into 1996, and 11 new students joined, for a total of 13. Thus the total number of students that attended over the 3 year period is: 15+2+11=28. A similar computation can be made for the statistics department to show that the total number of students attending over the 3 year period is 21. Note that the grand total of 49 at the bottom right corner of Figure 1 also reflects the fact that students attend a 2-year program.

In general, it is not possible to compute summaries for a non-summarizable measure, because the semantic rules may not be known. For the above example, we assumed that all students attend a 2-year program, but in reality some students may drop after one year, or some stay to take extra credit for a third year. Thus, for non-summarizable measures the totals can be extracted only from the base data (also called the "micro-data") which has the information about each individual student.

year / dept	1994	1995	1996	totals
computer science	15	17	13	28
statistics	10	15	11	21
totals	25	32	24	49

Figure 1: number of students by department and year assuming that students attend a 2-year program

The non-summarizability property of the measure "number of students" is not absolute but relative to the dimension being summarized. As can be seen from Figure 1, the "number of students" is summarizable relative to departments. That is, the totals at the bottom of Figure 1 of the number of students per year over both departments is correct. However, as we will show in example 2, this is only true under certain assumptions.

We note that if the measures in Figure 1 show only first year graduate students, then the result of summing the figures over 3 year (e.g. total of 45 for the computer science department) is correct. The problem here is one of characterizing the type of measure being taken. As will be discussed in section 5, the former is of type "stock", while the later is of type "flow". The former is not summarizable relative to the year dimension, while the later is.

This phenomenon is prevalent in SDBs and OLAP databases, and not simply a contrived example. Other typical examples exist for socio-economic data. For example, if the measure is "number of accident deaths by month", then it is summarizable over the months, but if the measure is "number of children attending school by month", then it is not summarizable. Similar examples

exist for OLAP databases. For example, the "number of items sold per month" is of type flow, and therefore it is summarizable over months, while "number of items remaining on the shelf per month" is of type "stock" and is not summarizable over months. This property will be expanded upon in section 5.

2.2 Example 2

Consider again the data set in Figure 1. Suppose that students can choose to attend a joint program of computer science and statistics. Thus, a student that attends a joint program is counted twice when we summarize over departments. An example of that is shown in Figure 2, where we know that 2 students attended the joint program in 1994, and continued to do so in 1995. We also know that no students attended the joint program in 1996. Note that the grand total at the bottom right corner of Figure 1 reflects both the 2-year program condition and the shared program condition. As was mentioned above, the totals in this example can be extracted, in general, only directly from the micro-data. They cannot be calculated from the table because the measure the "number of student" is not summarizable relative to both the year and department dimensions.

The reason for the non-summarizability relative to the department dimension stems from the fact that the students attend a joint program and therefore are not partitioned into disjoint sets relative to the departments. We will discuss this condition further in section 4.

year / dept	1994	1995	1996	totals
computer science	15	17	13	28
statistics	10	15	11	21
totals	23	30	24	47

Figure 2: number of students by department and year assuming that students attend a 2-year program AND can take joint department program

2.3 Example 3

Consider the data set shown in Figure 3, of the "number accident deaths per year per city in the USA" (the figures are not actual figures). As mentioned above, the "number of accident deaths" is a measure of type "flow", and it is therefore summarizable relative to the year dimension. Suppose that we wish to summarize the number of accidents for each state. Is that summarizable? As was pointed out in the introduction, states include areas other than cities, such as rural areas, villages, etc. and therefore summarizing over cities alone will yield an incorrect result.

state	city / year	1994	1995	1996	totals
Alabama	Birmingham	57	68	36	161
	Decatur	15	24	11	90
	⋮				
	Tuskegee	11	8	17	36
⋮					
California	Berkeley	15	13	8	36
	Eureka	3	0	2	5
	⋮				
	San Francisco	25	36	41	102
⋮					

Figure 3: number of accident deaths by city by year

The reason for non-summarizability in this case is that the mapping between cities and states is "incomplete" relative to the measure "number of accident deaths". To illustrate that "incompleteness" depends on the measure, consider the same table as in Figure 3 where the measure is the "number of museums" in each city. Given that we have the knowledge that museums only exist in cities, then this measure will be summarizable to the state level. This is a purely semantic condition that has to be determined ahead of time in order to declare the measure summarizable relative to a certain dimension. Note that in contrast to "number of accidents" the measure "number of museums" will not be summarizable relative to the year dimension. This is because of its type (called "stock", which will be discussed in Section 5).

Often times, this problem of "incompleteness" is dealt with by adding artificial instances to a dimension. For

example, if we added to the list of cities for each state an instance called "other" (where "other" is a reserved system keyword, similar to "null"), and assign to it the number of accident deaths in all regions in the state other than cities, then the summarizability condition will hold. This is often practiced in cases where a more detailed breakdown of the dimension instances is unknown or uninteresting. An example of that is the breakdown of a "race" dimension into the instances: "white", "black", "asian", and "other".

We note here that in this example, "cities" and "states" belong to the same dimension. Because there is a hierarchical relationship between the elements of cities and the elements of states, it is possible to summarize over that hierarchy. This hierarchical relationship in the same dimension is often referred to as the "classification hierarchy". We will discuss in the next section (section 3) the methodology for determining which elements make up the dimensions of the statistical data set, and which constitute a hierarchical relationship in the same dimension.

2.4 Example 4

Consider again the table in Figure 3, where the measure is now "average income of households". Obviously, the measures were originally obtained by adding up all the household income and dividing by the count of households. As it is well known, if is impossible to summarize this measure over the dimensions; that is, it is incorrect to apply the average function to the result of the previous average. This is a property of the statistical function. However, in the case of "average", it is possible to make this measure summarizable if we carry both the "sum" and "count". Assuming that both the "sum of income" and the "count of households" are summarizable relative to the dimension, then the average can be summarized as well for that dimension.

Another example of a measure that can be summarizable by using multiple components is "range". If the "min" and "max" of the range are kept, then the correct summarization can take place. In general, any complex statistics that can be expressed as component aggregates is summarizable. For example, "standard deviation" is summarizable if we carry forward the sum of squares, the sum and the count.

There are functions that are inherently non-summarizable. Examples are "median", "percentiles" or "trimmed mean". In such cases one has to generate the summary values directly from the micro-data. This is true regardless of the measure being taken, and regardless of the dimension we wish to summarize over.

3. Constructing the statistical object

We used in the previous section several examples that illustrate various conditions for summarizability. In order to determine summarizability, it is necessary to understand how a statistical object is defined. We describe in this section a methodology for determining what are the dimensions of a statistical object, which elements form the classification hierarchies, and the dependencies between the attributes. This somewhat lengthy process of defining the "statistical object" schema is absolutely necessary in order to apply the summarizability conditions.

We use the term "statistical object" to refer to the structure that we wish to apply summarization to. Inevitably, a statistical object is derived from some base data, or "micro-data". The micro-data is the data about the individuals or objects over which we wish to apply summarization statistics. The resulting data set as represented in the structure of the "statistical object" is often referred to as the "macro-data". For example, population statistics, birth rates, income levels, etc., are the result of some summarization function over the individuals making up this population. Often, the individual data is not available for reasons of privacy or sheer size. In such cases we say that only the macro-data is available, and the micro-data is not. Nevertheless, it is necessary to describe how the "statistical object" was derived from the micro-data structure in order to capture the semantics of that object. There are three steps involved in determining the structure of a statistical object.

3.1 Step 1: specifying the micro-data framework

We start by considering the framework (or context) of the base data. It has three parts:

(1) the individual/object class
(2) the attributes/descriptors associated with the individuals/objects
(3) the text description of these individuals/objects

A database may have several object classes. The reason for selecting a single object class for the micro-data framework is that it is necessary to determine ahead of time the focus of the summarization process. For example, the statistical object in example 1 (number of graduate students by department by year), may have come from a base data that has the object classes "students" "graduate students", "professors" "departments", etc. By focusing on "graduate students" the context of the statistical object is determined.

Here are the details of the for micro-data framework of example 1:

(1) individual/object class: graduate_student
(2) attributes/descriptors: student_id, year_attended, department, sex, height, GPA_per_year
(3) We consider only graduate students for ABC university, which has only two department: computer science and statistics. Graduate students follow a 2-year program. In a particular year, they can be enrolled in a single department only.

We note that the attributes include some identifier for the student, and may include attributes that are irrelevant to the statistical object we are interested in. The information in part (3) will help in determining summarizability. For example, the statement "Graduate students can be enrolled in a single department only" says that a student may not appear in the counts of two different department for the same year".

Given this framework, a statistical object is defined by specifying the elements described in step 2 below.

3.2 Step 2: specifying the elements of the statistical object

We use below a terminology commonly used in the Statistical Database literature [5]. We refer to the measured attribute as the "summary attribute", as it is this attribute that summarization is applied to. We refer to the attributes that make up the dimensions as "category attributes". This is because typically they have a finite set of disjoint discrete values (also called "categories" by statisticians). It is necessary that the categories are disjoint for summarizability. For example, age-groups should not overlap.

This step involves 4 parts that need to be identified:

(1) the summary attribute
(2) the category attributes and their values
(3) the statistical function associated with the statistical object
(4) the implicit category attributes and their values

We explain each of these parts next.

(1) The selection of the summary attribute requires the specification of the statistical function applied to generate the statistical object, and the attribute it applies to. In addition, the name and type of the summary attribute, its range of values, and its units (if any) need to be specified. In the example of Figure 1, we specify:

(a) summary attribute name: "number of graduate students" (acronym: "student_totals").
(b) function applied: "count"
(c) attribute that the function is applied to: "student_id" (or "null" - see comment below)
(d) summary attribute type: "positive integer"
(e) summary attribute range of values: 0 <= value <= 500
(f) summary attribute unit: "null"

There are several points that we wish to note. First, "the attribute that the function is applied to" can be "null" in the case that the function specified is "count" as in the case above. This is interpreted as the count of the individuals/objects (similar to "count *" in SQL). Second, with the exception of "count", the attribute that the function is applied to can be any of the numerical attributes. For example, we could have chosen to apply "average" to the GPA_per_year in the above example, thus having the average GPA for all students in each department and year. Third, the category attributes we select can be a subset of the attributes in the micro-data framework. In the example above, we left out "sex" and "GPA". Thus, the statistical object includes students of both sexes, and the GPA is irrelevant.

The selection of the summary attribute is crucial to the meaning of the statistical object. For the micro-data of the above example, we could have chosen the summary attribute to be "number of graduate student years" (i.e. the number of registrations"). This would effect the summarizability tests discussed in the next sections.

It is worth mentioning here a distinction that can be made between OLAP databases and SDBs relative to Step 2. In SDBs, it is usually the case that some statistical function is applied to the micro-data to generate the statistical object. In the example above the function "count" was applied. In the case of OLAP databases, the micro-data can be treated as the statistical object. For example, if we have a database of the cost of items organized by stores and day of sale, this can be treated as a statistical object where "day" and "store" are declared category attributes, and "cost of item" being the summary attribute. This is achieved simply by specifying the "function applied" to be "null", and "the attribute it applies to" to be "null" as well.

(2) Similar to the summary attribute, the name, acronym, type, and units need to be specified for each category attribute. But, in addition, the specific category values for each category attribute must be specified. The category values are considered part of the metadata of the statistical

object, and are often stored in the statistical database as well. For the above example, we have:

category attributes: "department", "year"

"department" type: text
"department" units: "null"
"department" categories: "computer_science", "statistics"

"year" type: integer
"year" units: year
"year" categories: "1994", "1995", "1996"

(3) The statistical function associated with the statistical object is usually the same as the function used to generate the statistical object. For example, if an "average" function was used to generate the summary attribute "average income", then it makes sense to associate the same function with the statistical object. However, there is no inherent reason that these should be the same. For example, one can get a "maximum" value for a traded stock over each month, and then associate an "average" function with the statistical object to generate "averages over maximum values".

The function associated with the statistical object cannot have the value "null". For the example of Figure 1, that function is "sum" (while the statistical object was generated with "count"). We note that even in the case that the function to generate the statistical object is "null", as was the case with the OLAP example above, the function associated with the statistical object cannot be "null". A function, such as "sum", "avg", etc. must be specified as the function associated with the statistical object in this case.

(4) The implicit category attributes are attribute hidden in the label of the statistical object. This is usually done when a category attribute has only a single value. For example, suppose that we have a 3-dimensional table containing "the number of accident deaths by city by age by race" for California in 1996. Obviously "city", "age" and "race" are category attributes. But, in addition, there are two additional implicit category attributes: "state" and "year". These attribute have a single value each for their categories which is the reason that they are not part of the table structure. However, they have to be considered as part of the definition of the statistical object.

Identifying the implicit category attributes is especially important for combining multiple statistical objects. The discussion of this capability is beyond the scope of this paper. However, we only note that implicit category attributes need to be analyzed as to their summarizability before combining multiple statistical objects.

3.3 Step 3: determining the category hierarchies

The number of category values is usually small (such as 5 values for race). However, in case that the number of values is large (such as the list of all counties in the USA, or the number of different items sold in a supermarket), it is convenient to group them into higher level categories (such as the types of item sold). This hierarchical organization may have multiple levels, such as cities within counties and counties within states. In some cases, it is convenient to group the values into bins, such as age-groups, to reduce the number of categories of a category attribute. It is the purpose of this step to identify such category hierarchies.

After the category attributes have been selected and specified, we need to apply a functional dependency analysis to the category attributes to determine which belong to the multidimensional space, and which belong to a classification hierarchy. The analysis is similar to functional dependency analysis in relational databases to normalize the database [2]. This can be thought of as a normalization process for the statistical object.

We start with a simple example. Consider the statistical object "number of cars sold per dealership per month". The category attributes selected are shown below in the parenthesis:
number_of_cars_sold (car_make, month, year, dealer, county, state).

It is clear that there exist functional dependencies between month and year, and between county and state (assuming that county names are globally unique). We use the notation "month year" and "county state" to mean month determines year and county determines state, respectively. The question is whether there exists a functional dependency between dealer and county. If we know that a dealer can operate in more than one county, no functional dependency exists. Otherwise, we have "dealer county". In the first case, car_make, month, dealer, and county are dimensions of the statistical object, and "month year" and "county state" form the classification hierarchies. In the second case, only car_make, month, and dealer are dimensions, and "month year", "county state", and "dealer county" form classification hierarchies. As can be seen this analysis is the same for determining multi-attribute keys in relational databases, so we do not need to expand on this here. The results of this analysis is a set of the category attributes that participate

in the multi-dimensional space, and a list of all functional dependencies that form classification structures.

These two cases are illustrated in Figure 4, using the graph notation in [3], where an X node represents the cross product of the multi-dimensional space, a C node represents a category attribute, and an S node represents a summary attribute. Note that the leaf nodes in this graph form the multi-dimensional space, while nodes above them form classification structures.

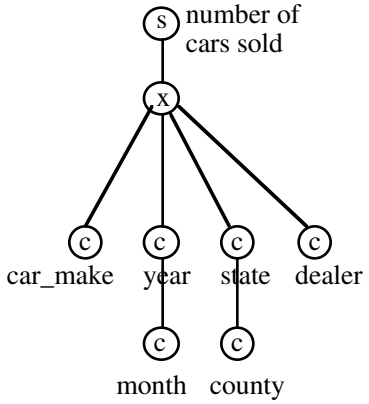

Fig 4, case 1: dealer -/-> county

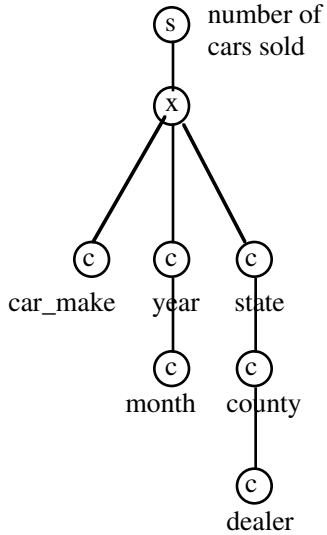

Fig 4, case 2: dealer ---> county

Figure 4: illustration of the effect of functional dependencies the structure of the statistical object

The classification structure itself is considered also part of the metadata of the statistical object. The metadata can be very extensive for category attributes with many values (such as the "state-county" classification), and is stored in the database system supporting the statistical object.

The functional dependency analysis is necessary to determine which attributes form the independent dimensions of the multidimensional space. The summarizability conditions discussed in the next section require the independence between category attributes of the multi-dimensional space.

4. Two necessary conditions for summarizability

In this section we discuss two of the three necessary conditions for summarizability. We discuss the third condition in the next section because of its complexity and the need for some terminology. These two conditions were already introduced with the examples of section 2. We discuss here more precisely how to test or evaluate these conditions.

4.1 Condition 1: disjointness of category attributes

The disjointness condition states that the category attributes must form disjoint subsets over the individuals/objects. For example, if students can be assigned to a single department only, we say that the department clusters the set of students into disjoint subsets. In this case the unit of interest is represented by "students", and the category attribute is "department". Note that testing this disjointness condition is done at the instance level of the micro-data, i.e. we examine the individual students relative to categories of the department. This condition is shown schematically in Figure 5. The test for this condition can be made either by examining the semantic knowledge as described in the "micro-data framework", or can be actually applied to the micro-data if the semantic information is unknown. In case that the test is made on the data, it applies only to the current data set. Any changes to the data will require a new test to be made.

We need to examine this condition for two different cases: a category attribute that is part of the multi-dimensional space, and a category attribute that belongs to a classification hierarchy only.

4.1.1 case 1: the category attribute is a leaf node

Leaf nodes in the graphical representation of the statistical object form the multi-dimensional space. In this case the

disjointness test is relative to the individuals of the "micro-data framework". For example, the car_make in the above example forms disjoint sets over the individual cars, as we know that no car can be labeled as manufactured by two or more manufacturers. If this is not the case, then summarizability condition 1 is violated, and no summarization can be applied to this dimension.

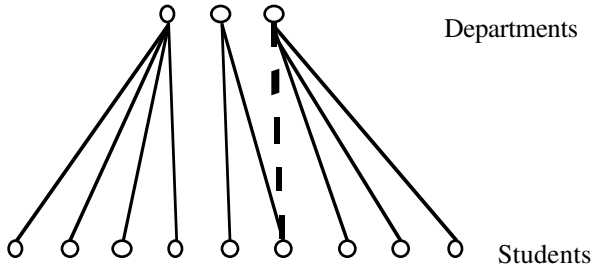

Figure 5: non-disjoint clustering. The bold broken line violates the disjointness condition

4.1.2 case 2: the category attribute is a non-leaf node

Non-leaf nodes in the graphical representation of the statistical object form the category hierarchies. In this case the disjointness test is made relative to the node below it. For example, "state" is a non-leaf node. We can test whether states form disjoint sets over counties. If it turns out that counties have globally unique names, then this condition holds. However, if we consider the relationship between cities and states, where cities do not have unique names (i.e. the same city name is sometimes used in two or more states), this summarizability condition is violated. In many cases, this situation can be fixed easily by renaming the offending instances or assigning an extension to names (e.g. extending the city name with the state name, such as Rome-NY).

We note that normally the functional dependency analysis in step 3 (section 3.3) will detect this condition for non-leaf nodes. However, often functional dependencies are made at the semantic level only, and mistakes can be made. For example, it is natural to specify "city state", assuming that city names are unique. Thus, it is advisable to check this condition at the data instance level. As mentioned before the associations between instances of classification hierarchies are often outside the statistical database as they are part of the metadata. This condition needs to check the metadata.

4.2 Condition 2: completeness

Given that the previous condition is satisfied, it is necessary to test whether the grouping of individuals/objects into clusters is "complete". The term "complete" in this context has two parts. The first is that all the elements of the clusters exist. That is, the union of all clusters constitutes the entire set of the individuals/objects. For example, in the case of students assigned to departments, we need to know that all the students are accounted for. The second condition is that each individuals/objects is assigned to some category. For example, if some student is not assigned to a department, this test is violated as well. These two parts of the test are shown schematically in Figure 6.

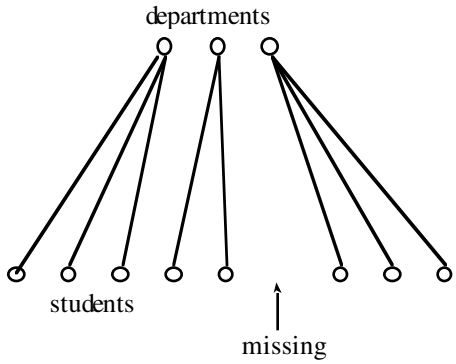

Fig 6, part 1: missing element

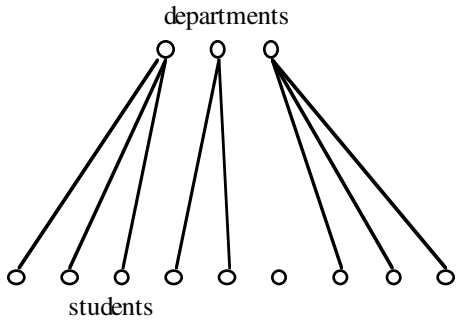

Fig 6, part 2: element not assigned

Figure 6: conditions for incompleteness

An important aspect of this condition is that it depends on the summary attribute. The same completeness test can be satisfied for one summary attribute and not for another. For example, suppose that the summary attribute is the average GPA per department. If the university permits

external students to audit classes, these students do not matter for computing the average GPA. In this case, it is ok to leave out the external students from the set of all students, since the incompleteness condition is not violated relative to the summary attribute. A similar situation was mentioned in example 3, where we made the case that the "city-state" hierarchy is incomplete relative to the summary attribute "number of accident death", but is complete relative to the summary attribute "number of museums". In this case, we use semantic knowledge that museums exist only in cities. A test over the database is also possible given that all museums are listed and their location is known.

As was the case for condition 1, the tests for the two types of category attributes (i.e. leaf and non-leaf nodes) are different.

4.2.1 case 1: the category attribute is a leaf node

As before, this test is applied relative to the individuals/objects of the "micro-data framework". Thus, for car_make of Figure 4, we need to make sure that all the cars are accounted for. In addition, if certain cars have unknown makes and are not assigned to a car_make category, then this second condition is violated. Here again, it is possible to fix such a problem by adding a car_make category named "unknown", so that the clustering is complete.

4.2.2 case 2: the category attribute is a non-leaf node

Here again, the completeness test is done for a non-leaf node i relative to the node j below it. Thus, we check this condition between two category attributes that are functionally dependent (j i) . We make the completeness test between the clusters of categories of node j and the categories of node i. Note that here again we need to make this test relative to the summary attribute. As discussed above for the case of city state, its summarizability depends on whether the summary attribute is "number of accident deaths" or "number of museums".

In the next section we describe the third condition for summarizability. It deals with the type of the summary attribute (stock, flow, etc.) and how this effects summarizability over different types of category attributes. The three conditions together must hold for summarizability.

5. The third necessary condition for summarizability

The third necessary condition for summarizability depends on the type of summary attribute as well as the statistical function applied.

Summarization is performed quite differently with respect to temporal categories (such as months or years) as opposed to non-temporal categories (such as sex, race, or spatial regions). In statistics, the term "temporal aggregation" is used. The summarizability property depends on the type of a summary attribute. Summary attributes can be classified as either a "flow" (also called "rate"), a "stock" (also called "level") or a "value-per-unit". We illustrate the type of a summary attribute by some examples:

flow: monthly number of births, annual income
stock: inventory of cars, number of citizens
value-per-unit: item price, cost per unit manufactured, exchange rate.

Flows refer to a periods and are recorded at the end of these periods, while stocks are measured and recorded at a particular points of time. Value-per-units are determined for a fixed time, too, but are not similar to stocks in so far as their unit is different. For instance, the unit of price per car is $/car, while the unit of an inventory of cars is given either by cars or $, dependent whether the number of cars or their value are considered.

A word of caution regarding "flows" and Stocks". Since an attribute of type "flow" is recorded at the end of a period (such as monthly number of births recorded at the end of each month) this may cause a confusion with type "stock" which may also be recorded at the end of a period (such as monthly inventory of cars). The distinction between these two types should be made on *what* is recorded, not *when* it is recorded. "Flows" record the commulative effect over a period, while "stocks" record the state at specific points in time.

While temporal summarizability of flows is semantically meaningful, temporal summarizability for stocks and values-per-unit depends on the statistical function being applied. For instance, monthly income can be summed to annual income, but it makes no sense to sum monthly inventories of cars (at the end of each month) into years.

The following table shows how the type of function and the type of attribute effect temporal summarizability:

	stock	flow	value-per-unit
min	ok	ok	ok
max	ok	ok	ok
sum	not ok	ok	not ok
avg	ok	ok	ok
range	ok	ok	ok

Figure 7: Temporal Summarizability by the type of function and summary attribute

If we consider summarizability with respect to non-temporal categorical attributes like race, region or industrial sectors, then the conditions of summarizability are different. They are presented below.

	stock	flow	value-per-unit
min	ok	ok	ok
max	ok	ok	ok
sum	ok	ok	not ok
avg	ok	ok	ok
range	ok	ok	ok

Figure 8: Non-Temporal Summarizability dependent upon type of function and attribute

We make couple of comments relative to the tables above. First, as mentioned above, a "count" must be carried along with the "sum" in order to support the "avg" function. Similarly, we assume that for the "range" function, the "min" and "max" are carried along. Second, for temporal summarization, while stock is not summarizable for "sum", it is summarizable for "avg". This is because of the semantics of attributes of type "stock". Third, value-per-unit is not summarizable under "sum" for both the temporal and non-temporal cases.

5.1 Condition 3: type compatibility

This third condition checks for the compatibility of the 3 parts of a summarization function: the category attribute, the summary attribute, and the statistical function. One has to determine the type of the category attribute (temporal, non-temporal), the type of the summary attribute (stock, flow, value-per-unit), and determine for the statistical function associated with the statistical object, whether summarizability holds. The tables above give the answers for the most common functions.

As was mentioned in Section 2.4, some functions are not summarizable in all cases, such as "median" or "percentile". In such cases any summarization operation must be performed directly on the micro-data. For very large databases, such as those considered in OLAP, precomputation of all possible combinations may be prohibitive. In a recent paper, the problem addressed was which precomputed summarizations should be stored, given a limited space, in order to answer any ad-hoc query [1]. We note here that such an approach only applies to statistical objects that are summarizable relative to all category attributes.

6. Dependency structures

In this section we wish to point out that even when the summarizability conditions are satisfied, there might be loss of information when performing dependency analysis. Thus, additional conditions are need for dependency analysis.

Summarizing over a category attribute of a multi-dimensional statistical object may cause erroneous conclusions with respect to the dependency structure of the remaining category attributes . This can be seen when the dependency structures of the original and summarized objects are compared. This happens even in cases when the summarizability conditions listed above are fulfilled. The distortion of the dependency structure of the remaining attributes caused by summing over a given (set of) attribute is called the Simpson paradox [4].

As an example, consider the category attribute set $A=\{a_1, a_2, a_3\}=\{\text{sex, year, age-group}\}$, where each attribute has only two values. We use counts ("absolute frequency") as a statistical function in the 3-way table below.

		a_3			
		1		2	
		$a_2=1$	$a_2=2$	$a_2=1$	$a_2=2$
a_1	1	4	8	2	12
	2	3	5	3	15

Figure 9: A 2x2x2 table, use to show that summarization does not preserve proportions (Source: [Whitener 78]))

Computing sex proportions $Pr(a_1=1|a_2,a_3)$ for males conditioned upon all values of the remaining attributes year and age-group, we get

a) $Pr(a_1=1|a_2=1,a_3=1) = 4/7 < Pr(a_1=1|a_2=2,a_3=1) = 8/13$ and

b) Pr (a_1=1|a_2=1a_3=2) = 2/5 < Pr (a_1=1|a_2=2,a_3=2) = 12/27.

Line a) relates the proportion of males of age-group 1 from year 1 to year 2. Similarly, line b) relates the proportion of males of age-group 2 from year 1 to year 2. As can be seen, the sex proportion of males increased in both age groups over time.

When we summarize over attribute a_3 (age-group), the following 2-way table is produced.

	a_2	
	1	2
a_1 1	6	20
2	6	20

Figure 10: 2-way table derived from 3-way table of Fig 9 by summing over a_3

The corresponding sex proportions Pr (a_1=1|a_2) of males are equal to

a) Pr (a_1=1|a_2=1) = 6/12 = 1/2 and
b) Pr (a_1=1|a_2=2) = 20/40=1/2.

It is contradictive to say, that in both age-groups there is an increasing effect while in the whole population there is no effect at all.

The Simpson paradox in this case can be explained by the fact, that the two-attribute interaction effect of attributes a_1 and a_2 in the 3-way table is different from the corresponding effect computed from the 2-way table. We use the notation $u_{12(3)}$ for the two attribute interaction where the subscript 12 in $u_{12(3)}$ refers to attributes a_1 and a_2, and the 3 in parenthesis refers to the 3-way table. Similarly, in the term $u_{12,3}(2)$, the subscript 12,3 says that this 2-way table of a_1 and a_2 was produced by summarizing over a_3. Using this notation, the Simpson paradox in this case can be expressed as $u_{12(3)} \neq u_{12,3}(2)$.

Therefore, a 3-way table of counts is called strictly collapsible over an attribute k with respect to the set (i , j) of remaining attributes, if

(i) 2-attribute interactions are not affected, i.e. $u_{i,j(3)} = u_{ij,k}(2)$ for i,j,k=1,2 and
(ii) 3-attribute interaction is zero, i.e. $a_{ijk}(3)$=0.

Simpson (1951) proved for 3-way tables with binary value sets as above that such tables are collapsible over an attribute k=3, iff $u_{13}(3) = 0$ or $u_{23}(3) = 0$ or $u_{13(3)} = u_{23}(3) = 0$. This is equivalent to the following conditional independence statements (where '⊥' means 'is independent of' and ' |' means ' under the condition given'

a) $a_3 \perp a_2|a_1$ or
b) $a_3 \perp a_1|a_2$ or
c) $a_3 \perp a_2|a_1$ and $a_3 \perp a_1|a_2$.

A theorem for general m-way tables is given in [6].

Thus, one should be careful in performing dependency analysis on summarized data. It is necessary to check these collapsibility conditions.

7. Future work - proving sufficiency of the three summarizability conditions

We have shown that the three conditions presented in this paper are necessary for summarizability. However, are they sufficient as well? Through examples, one can intuitively see the semantic reasoning for the "correctness" of operations over summary data. However, In order to construct a formal proof one has to define precisely when a summarization operation is correct, and then show that summarizability always holds if these 3 conditions are met.

We have not developed such a formal proof, but wish to give the reasoning as to why we believe these 3 conditions are indeed sufficient. Informally, summarization is "correct" if the result of the summarization on the statistical object is precisely the same as the summarization made directly from the micro-data. As we have seen from the examples given, some of the violations of summarizability stem from structural deficiencies (e.g. a city that maps to two states), and some from semantic deficiencies (e.g. applying "sum" to a summary attribute of type "stock"). Thus, there are two aspects to the correctness of statistical object: structural correctness, and semantic correctness.

In the tradition of object modeling, statistical databases have been treated in this paper as having a schema definition of objects (or object classes) and object instances. We argue below that we have applied both the structural correctness and the semantic correctness tests to the object schema and the object instances levels, and thus we covered all possible cases. The four possibilities are explained next.

(1) The structural correctness of the statistical object schema is achieved in the process of defining the normalized statistical object in section 3. In particular, the third step of analyzing functional dependencies identify the category attributes that make up the multidimensional space and those that make up the classification hierarchies. This guarantees that there are no functional dependencies between the dimensions of the statistical object, and each can be analyzed separately. It also guarantees that the classification hierarchies have only a one-sided dependency, thus forming true hierarchies.

(2) The structural correctness of the object instances is tested in condition 1. The disjointness test verifies at the instance level that the assignment of instances of the micro-data into categories of a category attribute are unique; that is no instance was assigned to more than one category. If this is not the case (such as a student assigned to 2 departments) then summarizability does not hold.

(3) The semantic correctness of the statistical object schema is tested in condition 3. This is achieved by checking that the statistical function associated with the statistical object is summarizable for the type of summary attribute (stock, flow, value-per-unit) and the type of category attribute (temporal, non-temporal). Indeed, by checking jointly all the components of the summarization operation (summary attribute, category attribute, statistical function) we guarantee that the semantics of the operation are satisfied.

(4) The semantic correctness of the object instances is tested in condition 2. The completeness condition goes beyond the structural disjointness, by checking that there are no missing instances, and that all instances have been assigned to some category.

The above arguments are informal, but we believe that the reasoning for the sufficiency of these summarizability conditions can lead to a formal proof.

8. Conclusions

Summarizability of OLAP and Statistical Databases is an a extremely important property because violating this condition can lead to erroneous conclusions and decisions. In this paper we explored the conditions for summarizability. We introduced a framework for specifying precisely the context in which statistical objects are defined. We have used a three step process to define normalized statistical objects. Using this framework, we identified three necessary conditions for summarizability. We provided specific tests for each of the conditions that can be verified either from semantic knowledge, or by checking the statistical database itself.

While we do not have a formal proof for that, we believe that the three conditions are not only necessary, but also sufficient conditions for summarizability. We hope to develop a formal proof in the future. We challenge readers to do the same, or to come up with counter examples that show additional necessary conditions.

Acknowledgment

We thank the anonymous referees for their helpful and insightful comments. This work is supported by the of Energy Research, U.S. Department of Energy under Contract DE-AC03-76SF00098.

References

[1] Harinarayan, Rajaraman, A, Ullman, J. D., Implementing Data Cubes Efficiently, Proceedings of the International Conference on Management of Data (SIGMOD), 1996, pp. 205-216.

[2] Maier, D., The Theory of Relational Databases, Computer Science Press, 1983.

[3] Rafanelli, M. and Shoshani, A. STORM: A Statistical Object Representation Model, Statistical and Scientific Data Base Management Conference (SSDBM) 1990, pp. 14-29.

[4] Simpson, C.H., The interpretation of interaction in contingency tables, J.R. Statist. Soc. Series B, vol. 13, 1951, pp. 238-241.

[5] Shoshani, A., Statistical Databases: Characteristics, Problems, and Some Solutions, Proceedings of the International Conference on Very Large Data Bases (VLDB) 1982, pp. 208-222.

[6] Whittemore, A. S., Collapsibility of multidimensional Contingency Tables, J.R. Statist. Soc. Series B, vol. 40, 1978, pp. 328-340.