An Analysis of Additivity in OLAP Systems

John Horner
College of Information Science and Technology
Drexel University
Philadelphia, PA 19104

Il-Yeol Song
College of Information Science and Technology
Drexel University
Philadelphia, PA 19104

Peter P. Chen
Department of Computer Science
Louisiana State University
Baton Rouge, LA 70803

ABSTRACT
Accurate summary data is of paramount concern in data warehouse systems; however, there have been few attempts to completely characterize the ability to summarize measures. The sum operator is the typical aggregate operator for summarizing the large amount of data in these systems. We look to uncover and characterize potentially inaccurate summaries resulting from aggregating measures using the sum operator. We discuss the effect of classification hierarchies, and non-, semi-, and fully-additive measures on summary data, and develop a taxonomy of the additive nature of measures. Additionally, averaging and rounding rules can add complexity to seemingly simple aggregations. To deal with these problems, we describe the importance of storing metadata that can be used to restrict potentially inaccurate aggregate queries. These summary constraints could be integrated into data warehouses, just as integrity constraints and are integrated into OLTP systems. We conclude by suggesting methods for identifying and dealing with non- and semi- additive attributes.

Categories and Subject Descriptors
H.2.7 [**Database Management**]: Database Administration – *data warehouse and repository.*

General Terms
Design, Reliability

Keywords
Data Warehouse, OLAP, Additivity, Metadata, Summarization

1. INTRODUCTION
Accurate and efficient summarized data are essential in data warehouses and online analytical processing (OLAP) systems. Data warehouses contain historical, integrated, and relatively static data, and are often magnitudes larger than online transactional processing (OLTP) systems. Effective summarizability of these enormous disparate data sources is of paramount concern, and until recently, has been a widely ignored challenge in data warehousing. [13] argues that summarizability "is an extremely important issue that is largely ignored in OLAP literature". Because of the size of data warehouses and the strategic purpose of these systems, query outputs are nearly always aggregated sets of the base data, and inaccurate aggregate query outputs could result in misinterpretation of the data resulting in poor strategic decisions. [11] describes three design criteria necessary for all database systems, consisting of correctness, efficiency, and usability, and argues that data correctness is of utmost importance.

DOLAP'04, November 12–13, 2004, Washington, DC, USA.

Data warehouses are typically conceptualized as facts and dimensions, whereby facts are measures of interest, and dimensions are attributes used to identify, select, group, and aggregate measures of interest. Attributes that are used to aggregate measures are labeled classification attributes, and are typically conceptualized as hierarchies. Figure 1 shows an example of a classification hierarchy along the time dimension, illustrating that measures can be aggregated from the lowest level of granularity, dates, into progressive higher months, quarters, and years. For example, a profit measure may be aggregated from the daily profit to the monthly, quarterly, or yearly profit.

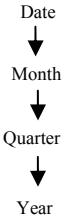

Figure 1: A Classification Hierarchy

Typically, data is aggregated along multiple hierarchies, summarizing data along multiple dimensions. For example, a summary may show the total sales in the year 2004 at all branch locations in Pennsylvania. In this case, the sales measure is rolled up along the time dimension and location dimension. Because of the enormous size of the data sources, operations are performed to summarize measures in a meaningful way. The typical operations include the following:

- **Roll-up**, which increases the level of aggregation along one or more classification hierarchies;
- **Drill-down**, which decreases the level of aggregation along one or more classification hierarchies;
- **Slice-Dice,** which selects and projects the data;
- **Pivoting**, which reorients the multi-dimensional data view to allow exchanging facts for dimensions symmetrically; and,
- **Merging**, which performs a union of separate roll-up operations.

In this paper, we are concerned with the accurate summarization of measures, and therefore we will concentrate on roll-up and merge operations, which both use aggregate operators to combine finer-grained measures into summary data. The most common aggregate operator used to summarize measures is the summation operator, and is generally considered the default aggregate operator when rolling-up data. However, in certain instances, using the sum operator to summarize data can result in inaccurate summary outputs. Therefore, it is of vital importance to recognize the potential for inaccurate summaries resulting from using the summation operator. The ability to use the summation aggregate operator has been labeled *additivity*. A measure is:

- **additive** along a dimension if the sum operator can be used to meaningfully aggregate values along all hierarchies in that dimension;

- **fully-additive** if it is additive across all dimensions;
- **semi-additive** if it is only additive across *certain* dimensions; and,
- **non-additive** if it is not additive across any dimension.

[9] argues that "the most useful facts are numeric and additive". Yet, data warehouses are littered with non-additive measures. One reason for this abundance is that it is often advantageous to store non-additive data in a data warehouse. Organizations may store non-additive data to alleviate data management issues. [2] describes that reduced information means reduced data management, and "a reality of data warehousing is that many companies struggle to build, maintain, and query large databases". [9] states that optimized inventory levels in stores can have a major impact on chain profitability and state that snapshots, which are inherently non-additive across different time spans, are a viable category of data warehouse. This illustrates a fine distinction of data that *could* present summarization problems if incorrectly analyzed, and those that are more likely to present problems in real systems. For instance, temperature is a non-additive attribute that cannot be meaningfully added with other temperatures; however, it is unlikely that someone will mistakenly misinterpret a query that sums temperatures together. On the other hand, duplicate, incorrect, and missing values in a classification attribute can result in undetected anomalous queries, resulting in poor decisions.

While several authors have discussed the importance of accurate summarizations, we have not found a comprehensive taxonomy of measures that could potentially result in inaccurate summaries. In this paper, we look to characterize inaccurate summaries, specifically those resulting from using the sum aggregate operator. We show various examples of inaccurate summary outputs, and suggest rules for identifying and handling these situations. We also contend that these issues should be considered in all phases of data warehouse design.

Section 2 discusses types of summarization problems, Section 3 discusses potential solutions, and Section 4 concludes our paper.

2. CHARACTERIZATION OF SUMMARY DATA IN OLAP SYSTEMS

In the previous section, we described the importance of accurate summaries in data warehouse systems. The first step towards ensuring accurate summaries is recognizing when and how the sum operator can be used to meaningfully aggregate data. For this purpose, we have created a taxonomy that can be used to facilitate this recognition. Figure 2 illustrates our taxonomy of summation constraints.

The following subsections describe the importance of classification hierarchies, followed by subsections detailing examples and underlying causes for the non- and semi-additive data illustrated in Figure 2.

2.1 Classification Hierarchies

With respect to summarization, classification hierarchies are of central concern because the primary method of rolling-up and drilling-down data is along these pre-defined hierarchies. Additionally, data warehouse design often involves materializing aggregate views along these hierarchies. As we have already described, a classification hierarchy is a ladder of increasingly high level attributes, each composing data from the lower levels. Hierarchies can be further classified into *strict*, *non-strict*, *complete*, *incomplete*, *multiple path*, and *alternate path* hierarchies.

A *strict hierarchy* is one where each object at a lower level belongs to only one value at a higher level. For instance, US states and Canadian Provinces can be considered to form a strict hierarchy because each state or province belongs to only one country. A *non-strict hierarchy* can be thought of as a many-to-many relationship between a higher level of the hierarchy and the lower level. Non-strict hierarchies can result in multiple or alternate path hierarchies, whereby the lower object splits into two distinct higher level objects. For instance, sales could be rolled up along the location hierarchy either by area code, zip code, or city and county. This is considered an alternate path hierarchy because the hierarchy joins again at a higher level. If the hierarchy does not come back together it can be considered a multiple path hierarchy.

Alternate and multiple path hierarchies are important when summarizing measures, and can specifically present problems when merging data. An inaccurate summarization can result if summaries from different paths of the same hierarchy are merged. Specifically, data cannot be merged among classification attributes that have overlapping data instances. For instance, you cannot merge total sales from an area code with the total sales in a city that the area code served. This is because data instances would be rolled up into both categories, and measures would be added twice if the summaries were merged.

Classification	Examples
1.0 Non-Additive	
1.1 Fractions	
1.1.1 Ratios	GMROI[1], Profitability ratios
1.1.2 Percentages	Profit margin percent, return percentage
1.2 Measurements of intensity	Temperature, Blood pressure
1.3 Average/Maximum/Minimum	
1.3.1 Averages	Grade point average, Temperature
1.3.2 Maximums	Temperature, Hourly hospital admissions, Electricity usage, Blood pressure
1.3.3 Minimums	Temperature, Hourly hospital admissions, Electricity usage, Blood pressure
1.4 Measurements of direction	Wind direction, Cartographic bearings, Geometric angles
1.5 Identification attributes	
1.5.1 Codes	Zip code, ISBN, ISSN, Area Code, Phone Number, Barcode
1.5.1 Sequence numbers	Surrogate key, Order number, Transaction number, Invoice number
2.0 Semi-Additive	
2.1 Dirty Data	Missing data, Duplicated data, Ambiguous and unusable data, Incorrect data
2.2 Changing data	Area codes, Department names, customer address
2.3 Temporally non-additive	Account balances, Quantity on hand,
2.4 Categorically non-additive	Basket counts, Quantity on hand, Quantity sold
3.0 Additive	

[1] ***GMROI is Gross Margin Return On Inventory investment***

Figure 2: Taxonomy of Summation Constraints

It becomes especially important to recognize which merges will result in duplicate values in summary data in complex data warehouses with numerous alternate and multiple path hierarchies, some of which are materialized. In addition to merges along different paths of the same hierarchy, summaries cannot be merged with summaries at a different level of the same hierarchy. This is because the higher levels contain data at the lower levels. For instance, you could not merge the total sales from New York City with those of the United States of America, since the Sales from New York City will be included in the sales from the United States.

*Completeness in hierarchie*s means that all members belong to one higher-class object, which consists of those members only. For instance, the US states hierarchy would be considered complete if there were data instances for each of the 50 US states. In other situations, it may not be possible to capture all of the possible values. Therefore, if a classification hierarchy is incomplete, incorrect summaries may result. More on this topic will be discussed in Section 2.3, when we discuss dirty data in the semi-additive section.

Many concepts from statistical databases (SDB) are transferable to OLAP systems, and vice versa; however, due to the differences in users and domains, the literatures in each area do not cite each other. Both SDBs and OLAP systems are structured in a multi-dimensional format and provide statistical summaries. The main difference between SDB and OLAP systems is the area of emphasis, with SDB literature emphasizing conceptual modeling, and OLAP literature emphasizing query performance [14]. Therefore, it is important for OLAP systems to borrow some areas of SDB systems, such as strict classification hierarchies and the need to distinguish summary data from attributes [13]. [14, 15] discuss strictness and completeness in data warehouses, but only from a conceptual modeling perspective. While it is important to consider these aspects in conceptual modeling, it is not sufficient. Considering these properties in later stages of design can be useful for restricting potentially inaccurate summaries.

There is a fundamental difference between classification attributes and dimensional attributes within the same dimension [10]. Classification attributes are those attributes that are used to summarize measures, while dimensional attributes are used as descriptors. [12] argues for the need to rigorously classify hierarchies and detailed characteristics of hierarchies, such as completeness and multiplicity.

2.2 Non-Additive Measures

Non-additive facts are defined as measures where the sum operator cannot be used to meaningfully aggregate values. In this section we will detail various non-additive facts, and describe methods for dealing with these measures.

2.2.1 Ratios and Percentages

In [9], several types of measures that are inherently non-additive are described. They state that percentages and ratios, such as gross margin, are non-additive, and therefore, when designing systems, both the numerator and denominator should be stored in the fact table. Additionally, they note that it is important to

remember when summing a ratio, it is necessary to take the ratio of the sums and not the sums of the ratios. In other words, the numerator and denominators should first be aggregated separately using the sum operator, and then the totals should be divided, yielding the appropriate ratio values.

2.2.2 Measures of Intensity

[9] state that all measures that record a static level are inherently non-additive across the date dimension, and possibly over other dimensions as well. *Measurements of intensity* are another type of static measurement that is inherently non-additive, and includes measures, such as temperature and blood pressure. These attributes are often indirect measurements of additive base data. And, often, it may not be possible to store the data from which these attributes are derived. For instance, temperature is a measurement of the kinetic energy, or average speed of the molecules, in the substance that is being measured. Since it probably is not feasible to capture and store the average speed of molecules, much less the additive attributes, distance and time, for each molecule from which this speed is derived, the non-additive measure, temperature, is stored.

While measurements of intensity are non-additive, they are often still important measures, and therefore cannot be "designed out" of a system. It is also unlikely that non-additive intensity attributes would be misinterpreted, primarily because most professionals recognize that measures of intensity cannot be meaningfully added, and the results of large scale aggregation of these values would often be well-outside what is considered normal. However, blindly using sum as the default roll-up operator for all facts will make systems that rely on measures of intensity less efficient because the analyst will need to take time to determine reset the aggregate operator for the query. Therefore, it is important for systems to allow default aggregate operators for measures to be set and be made visible to analysts performing queries.

2.2.3 Average/Maximum/Minimum

Often, snapshot fact tables will include measures that are derived by taking the average, maximum, or minimum values from an operational system. For instance, in the health care domain, it is often important to analyze the number of patients admitted to a hospital. So, a snapshot fact table could be created at the daily grain to capture the number of hospital admissions. In this case, it also may make sense to store other aggregates about daily hospital admissions, such as the maximum number of admissions in an hour, the minimum number of admissions in an hour, when these extremes took place, and average number of hourly admissions. These values may be useful for providing information about finer grain peaks in a snapshot. Unlike total daily number of admissions, which is fully-additive, maximums, minimums, and averages are non-additive; however, other aggregate functions can be used on these measures. For instance, it may be useful to look at the highest maximum or lowest minimum.

2.2.4 Other Non-additive Facts

In addition to ratios and measurements of intensity, there are numerous other facts that cannot be meaningfully added because they would violate arithmetic rules. For instance, text fields, certain arbitrarily assigned numbers (i.e. zip code, ISBN, etc…), and measures of direction (such as 35 degrees east of north), cannot meaningfully be added, and often cannot be meaningfully aggregated using other aggregate operators as well. While there are few plain English queries that would result in someone mistakenly adding zip codes together and not realizing the error, it is possible that systems developed to view this summarized data may not provide the appropriate flexibility and efficiency of use if these constraints are not made apparent.

2.3 Semi-additive Measures

Semi-additive measures are those measures where the sum operator can be used for aggregation along certain dimensions, but not all. In this section we detail several semi-additive measures and methods for handling these measures.

2.3.1 Dirty Data

The data that populates data warehouses is pulled from multiple disparate data sources, often from legacy databases and non-standardized data and schema formats. Data cleansing is typically performed during the extraction, transformation, and loading (ETL) process and can involve eliminating duplicate records, correcting data errors, and ensuring the integrity of the relations among data sets. Once data makes it through the ETL process, it is assumed to be thoroughly cleaned; however, it is often impossible to eliminate all possible anomalies in the data.

Inaccurate data is often termed *Dirty Data*, and has been characterized as missing data, wrong data, correct but unusable data, and compound data with different representations. These data can occur when there are data entry errors, non-enforced integrity constraints, different values of the same data in different databases, ambiguous representations of data, or nulls are mistakenly permitted [8]. Of particular concern are dimensions that have missing, duplicate, ambiguous, or inaccurate data in classification attributes, because these could result in inaccurate summarizations of rolled-up data along the associated hierarchies.

[9] suggests creating an arbitrary code for dimensional instances that are unknown or unavailable. For instance, a retail chain may track customers based on store discount cards that are swiped every time a customer checks out. Customers who do not have discount cards would not be tracked in the system. In the sales fact table, these transactions would be included, but would be given an arbitrary missing data code for the customer dimensional fields. When summarizing these sales along the customer location hierarchy, the sales for customers with cards would be in the appropriate group. However, the sales for the customers without cards would be included in an ambiguous missing data group. If the majority of the customers' locations are not tracked, then an analysis of sales by location would be meaningless. In this case, facts rolled-up along this dimension would have inherently ambiguous data, and the resulting summaries would be imprecise.

This example demonstrates that summarization problems can have an impact on decision-making. The effect is likely to be more significant as the percentage of missing data increases. Therefore, it is important to provide a measure of precision when rolling up the data. In this situation, analysts should determine whether measures are systematically missing from the dataset.

Another problem occurs when customer data are stored incorrectly or duplicated with different information in each instance. With respect to aggregate summarizations, the most vital issues occur when instances have incorrect or anomalous information in classification attributes. In these cases, it is difficult to gauge the degree of the problem because these errors are not typically known.

For example, suppose a customer is listed twice in the data warehouse with the mistakenly transposed zip codes of 19050 and 19005. When rolled up to the zip code level, the sales from this customer would be associated with the wrong group. However, since both zip codes are in the state, the anomaly would disappear when rolled up to the state level. Since there is nothing in the system to indicate that measures associated with these zip codes is incorrect, hidden summarization problems can occur. High cardinality attributes with non-systematic capture techniques are more prone to incorrect or duplicate data instances.

2.3.2 Changing Data

Incomplete and changing dimensional attribute instances can result in inaccurate summarizations. Data warehouses are typically created from transactional systems meant to support operational needs. Many of these systems were not designed with long term analysis of data in mind. Therefore, it is not uncommon for more data to have been captured later in the lifetime of the database, which is not available earlier. Additionally, external changes in the environment of the data instances can result in new data becoming available. For instance, in the past 15 years, there has been an explosion of area codes in the United States. Area codes have been split into new area codes and overlaid on top of existing area codes. For instance, in the Philadelphia area, the original area code, 215, was split into two area codes, 215 and 610. Additionally, area code 484 was overlain on top of the existing area codes; so, what once was in the area code 215 may now be either in 215, 610, or 484. A query may show that the population in the 215 area code decreased from 200 to 150 people, when in fact the population of the area coved by the original 215 area code increased from 200 to 400 people. Needless to say, this can make data analysis across area codes very complex, and potentially can lead to inaccurate summarizations of the data.

Additionally, merges, splits, overlapping, and incomplete data are dependent on the data instances and often are a result of changing business needs or a changing environment from which the data was held. Therefore, it is important to track changes, especially those that affect classification hierarchies, as the characteristics of the data and environment change. This example illustrates how data could be misinterpreted, especially if only area code 215 population trends were analyzed, especially when the formation of new area codes is not known to the analyst.

[9] describe a similar idea of slowly changing dimensions, and described several methods for dealing with this issue, including simply overwriting data (type 1), storing the new data instance in a new row, but with a common field to link the dimensions as being the same (type 2), or adding a new attribute to the dimension table to store both the new and old values (type 3). Other methods for dealing with changing dimension include versioning with current value flags, or versioning with event lengths. There are disadvantages to using the methods associated with slowly changing dimensions for dealing with data that merges, splits, or is overlain. Specifically, the methods do not explicitly state how and why the data has changed, but rather only show that a specific value has changed. Additionally, the splitting, merging, or overlaying of data is, conceptually, information about the data, and should therefore be considered metadata. In other words, storing the changing data in the dimension tables can make them conceptually complex, and may not provide the full explanation of the change needed. Therefore, it is more appropriate to store merges, splits, and overlain data in metadata tables, which can be linked to the dimension tables.

These techniques only address how to handle changes in the dimensional data instances, but do not focus on changes that affect the schema or summation constraints. For instance, at one time area codes formed a strict alternate path hierarchy in a location dimension, whereby one location was assigned one area code; however, area codes have been created, split, and overlain on top of the original area codes. In this situation, what was initially a strict hierarchy is now a non-strict hierarchy. None of the methods for slowly changing dimensions would recognize the change in this factor, which could affect the accuracy of summaries; therefore, it is also important to store metadata pertaining to changes in the characteristics of classification hierarchies.

2.3.3 Temporally Non-additive Measures

Temporally non-additive measures are those measures that are not additive along the time dimensions. One example of a temporally non-additive measure is a bank account balance (Figure 3). Account balances from different times cannot meaningfully be added together and therefore can be considered temporally non-additive. Another typical example is Quantity_on_hand.

Temporally non-additive measures are only meaningful when grouped by the snapshot date because they include duplicate transaction level data that is not apparent in the fact table. Account balances are determined from numerous transactions, and adding account balances over different times counts these underlying transactions multiple times.

Dimension	Additive
Time	*No*
Customer	Yes
Branch	Yes
Account Type	Yes

Figure 3: Dimensions across which account balances are additive

For instance, suppose a customer opens a checking account with a balance of $1000 in January. The customer may then make a deposit in February of $500, deposit $3000 in March, and deposit $1000 in April. Figure 4 depicts the non-additive nature of the monthly account balances derived from these transactions.

As Figure 4 shows, summing the monthly account balances across time would result in certain transactions being duplicated. Specifically, all of the previous transactions are duplicated when summing the monthly account balance. While this balance is determined by a number of finer grained transactions that are not stored in the OLAP system, they are still included in the summary result. The figure demonstrates that adding balances across time would result in certain transactions being duplicated.

Month	Old Transactions	New Transactions	Balance
January	-	+$1000	$1000
February	$1000	+$500	$1500
March	$1000+$500	+$3000	$4500
April	$1000+$500+$3000	+$1000	$5500
Total	**$7000**	**$5500**	**$12500**

Figure 4: Account Balances

This example demonstrates that although the individual transactions are not stored in the database, they are included in the account balances, and summing the account balances includes transactions multiple times. Therefore, any snapshot fact measures that are derived in this fashion will be inherently non-additive across the time dimension. However, aggregating all of the account balances at a single time along other dimensions can provide a meaningful measure of the total amount of money in the organization at any one time.

While snapshots are inherently semi-additive, they can provide useful information, including the average amount of money held over a period of time, the times when there is a maximum or minimum amount of money held, and the variation of money held. Since these are summaries that may need to be queried from the system, it is often advantageous to store these snapshots. It may also be beneficial to continue to store additional attributes for more robust querying capabilities, such as net amount of deposits, net amount of withdrawals, number of deposits, number of withdrawals, and total number of transactions.

2.3.4 Categorically Non-additive Measures

Categorically non-additive measures are measures that are not additive on the dimension that holds the different types of information. If a fact table includes a measure that is derived from counting diverse data, such as basket counts, the measure will be semi-additive (Figure 5). The reason for this is that summing the total count results in diverse products being added up in the same summarization number.

For example, a store may sell 50,000 products per week; however, this number means relatively little for most business purposes, especially if numerous and diverse products are sold. Therefore, basket counts are non-additive along the product dimension. In other words, when aggregating basket count, it must be grouped by product. Unlike inventory levels and account balances, the quantity of products sold can be aggregated along lower levels of a hierarchy. Account balances can only be grouped by the lowest grain of the time dimension, which also matches the snapshot interval; however, quantity of products sold can, in some cases, meaningfully be aggregated along lower levels of a hierarchy. For example, Figure 6 depicts the number of products sold from a general purpose store and shows 3 levels of the product family hierarchy.

Dimension	Additive
Time	Yes
Product	*No*
Customer	Yes
Store	Yes

Figure 5: Dimensions across which basket counts are additive

In this example, a store may sell 100 different models of televisions and 200 different models of telephone. Showing the quantity of each different model would result in an overload of information; therefore, basket counts can meaningfully be aggregated at the *class*, and possibly even *family* level. Unlike temporally non-additive measures, it is possible that categorically non-additive measures may be *partially* rolled up along the classification hierarchy to reduce the problem of too much information.

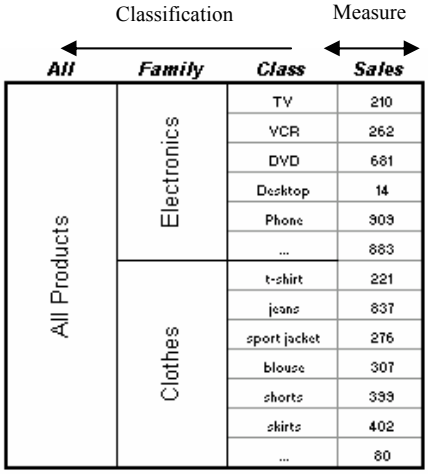

Figure 6. Basket Count Measures

Inventory levels, such as quantity on hand, are another type of categorically non-additive measures (Figure 7). Inventory measures are derived from counting different types of items, and are also typically snapshot levels. We note that most books mention inventory levels are only non-additive across time [9, 1]. Because units of different products may imply different meaning, adding quantities of different products may not be meaningful. For example, adding quantities of TVs sold and Batteries are not informative. Inventories demonstrate that our characterization of additive attributes is not mutually exclusive, as they can be both categorically and temporally non-additive.

Dimension	Additive
Time	*No*
Product	*No*
Customer	Yes
Store	Yes

Figure 7: Dimensions across which quantity-on-hand measures are additive

2.4 Additive Measures

Measures that do not meet any of the previously described rules for non-additivity or semi-additivity can be assumed to be additive. However, just because a measure is additive across all dimensions does not ensure that a query will be correct. Inaccurate summaries can arise when pre-specified rules for rolling up data are prescribed. For instance, using non-standardized rules for averaging and rounding data can result in inconsistencies in summary data. These types of problems are most apparent when sharp decision cutoff points exist, and are based on aggregate functions. An example is the averaging rules for particle air pollution. Congress passed a regulation [16] that prescribes regulatory decisions based on the 3-year average concentration of particle pollution using a pre-specified averaging rule. This value is obtained by averaging together the quarterly and yearly averages. This result from

averaging the averages can vary from traditional averaging whereby all data points are averaged together. This results because different quarters can have a different number of samples resulting in unequal weighting of instances. While it may not be a momentous change, if the summarization result is near the cut-point for determining whether to impose regulations, then the averaging mechanism becomes quite significant.

Similarly, rounding rules can result in differences in data output. If data is pre-rounded to a set number of digits before averaging, it can result in different results than if it is averaged and then rounded. These are just a few examples of how business rules for rounding and averaging can result in the misrepresentation of data. These situations are not unique to the examples given, and also extend beyond averaging and rounding. Regardless of the operation, it is important to store these business rules as metadata in the database.

2.5 Other Aggregate Operators

While the sum operator is the most common aggregation operator, data from data warehouses can provide much richer information than simple summations can provide. Often, analysts are concerned with the central tendency, range of data, or use more robust tools to mine patterns or perform what-if scenarios using the data. As data warehouses are implemented in non-traditional domains and are integrated with analysis tools, it is more common to have non-additive attributes, non-traditional aggregate operators, and new types of logical errors. Many organizations and agencies in the scientific community and government have been capturing enormous amounts of data, and are recognizing the benefits of integrated data warehouses for strategic planning, and more rich and complete data analysis. For instance, [4] describe the deployment of a data warehouse in a health care domain. And, [6] describe the application of data warehousing in non-traditional domains such as environmental monitoring and research, biomedical applications, and other types of digital libraries.

3. DATA DICTIONARY AND SUMMARY CONSTRAINTS

In this section we will describe previously suggested approaches for dealing with summarization issues, and briefly describe the need for future techniques to expand upon these approaches.

[7] argues that data should be normalized into what they propose as General Multidimensional Normal Form (GMNF), whereby aggregation anomalies are avoided through a conceptual modeling approach that emphasizes sorting out dimensions, dimensional hierarchies, and which measures belong where. While using their approach to structure data in GMNF may help avoid certain aggregation anomalies, it does not guarantee correctness of all queries, such as those associated with categorically non-additive measures, and measures rolled-up along changing dimensions. Additionally, [9] argues that the most useful facts are additive because they provide the most robust querying capabilities and lessen the opportunities for inaccurate summaries. The approach described in [7], however, makes no attempt to ensure all facts are additive. Rather they only ensure that all facts are summarizable by focusing on adequately structuring the dimensions and dimensional hierarchies.

Prior research has suggested incorporating notations or appendices and glossaries into conceptual models to address these concerns. [5] proposes a conceptual model that explicitly depicts hierarchies and aggregation constraints along hierarchies, and argue that a fact glossary should be developed describing how each fact was derived from an ER model. [7] contends that conceptual design is the most important phase of data warehouse design, and during this phase it is important to sort out dimensions, dimensional hierarchies, and which attributes belong where.

While we agree these ideas are important, they are not sufficient. The literature focuses on eliminating or documenting potential summarizability constraints. Non-additive data, however, can cause summarizability problems that may not be feasible to eliminate from a data warehouse. Simply documenting the associated constraints is not sufficient for mitigating the associated problems. The accuracy of summary data is a principal concern in OLAP systems, and this information should not be stored in glossaries where it may be glanced over or ignored. Rather, it is important to include summarizability constraints in a conceptual model in a manner that facilitates the transfer to later stages of design. Specifically, data warehouses should store the dimensions across which measures are additive, and should alert or constrain analysts to potentially inaccurate summaries. We term these safeguards *summary constraints* because they restrict inappropriate aggregate summaries. Imagine the ubiquity of update anomalies if primary and foreign key constraints were only documented in an appendix of a conceptual model.

In order to deal with dynamic summarizability constraints at query time, it is important to store and integrate metadata pertaining to these constraints into data warehouses. In data warehouses, there are 3 kinds of metadata, consisting of administrative, business, and operational metadata [3]. *Administrative metadata* consist of descriptions of source databases, back end and front end tools, definitions in the schema, data mart locations and contents, derived data, dimensions and hierarchies, pre-defined queries and reports, physical organization, ETL rules, refresh and purging policies, and user profiles, authorization, and control. *Business metadata* consist of business terms and definitions, ownership of data, and charging policies.

And, *operational metadata* consists of information collected during the operation of the warehouse, including the lineage of migrated and transformed data, whether it is active, archived or purged, usage statistics, error reports, and audit trails. It is important to store certain metadata from each category to handle summarizability constraints. In addition to the metadata described above, [12] states the importance of storing information about hierarchy completeness. [7] states that the distinction between meaningful and meaningless aggregation data should be stored in an appendix. And, [13] describes the importance of tracking splits and merges in the database.

Non-additive Categories		Identification	Suggestions
Fractions	Ratios	Look at derivation process and business rules for derived fields with numerators and denominators	Store the numerator and denominator in separate fields; remember to take the ratio of the sums, not the sum of the ratios
	Percentages		If possible, store the base data from which the percentages are derived; track constraints in metadata
Measurements of Intensity		Review business rules to determine meaning of attributes	Use alternate aggregate functions (average, maximum, minimum); track constraints in metadata
Averages		Look at mapping process and business rules for derivation of measures from operational systems	Use alternate aggregate functions; track constraints and in metadata
Maximums			
Minimums			
Measurements of Direction		Review business rules to determine meaning of attributes	Perform vector averaging; use mode to look for central tendency
Identification Attributes	Codes	Review business rules to determine meaning of attributes; Normally are dimensional attributes	Codes with meaningful components may be useful if decomposed into their parts
	Sequence Numbers		Sequence numbers may be useful for measuring the number of transactions between events and identifying milestones

Figure 8: Identification and Suggestions for dealing with Non-additive Attributes

Semi-additive Categories		Identification	Suggestions
Dirty Data	Missing	Look for facts that are linked to dimensional instances representing missing or unknown value	Use the aggregate summation of facts linked to missing or unknown dimensional instances as a measure of precision; store completeness properties in metadata
	Duplicate	High cardinality attributes with non-systematic data capturing techniques are prone to duplicate and incorrect data instances	Estimate effect of duplicates on summarizations from cardinality, ETL method, and knowledge of base data
	Wrong		Estimate effect of incorrect data from cardinality and method of capture; The effect of incorrect data on summaries will be less numerous when data is rolled up to at higher levels of hierarchies
Changing Data		Look for duplicate natural keys in dimension table with different descriptions and dimension keys, or dimensional instances with a non-null *historical data* attribute. The identification of changes where data was replaced require in-depth understanding of data	Track changes to strictness and completeness of classification hierarchies; store history of changes and reasons for changes in metadata
Temporally Non-additive		Snapshot fact tables of levels may be *temporally non-additive*	Do not add across time spans; aggregate queries must be grouped by time interval equal to that of the snapshot interval; divide sum by number of time intervals to get temporal average
Categorically Non-additive		Data derived from counting different types of instances in an operational system may be *categorically non-additive*	Do not add across different types of items; determine level of hierarchy where diverse data becomes meaningless and ensure aggregate queries are grouped at or below that level

Figure 9: Identification and Suggestions for dealing with Semi-additive Attributes

Designing systems to accurately summarize data is a primary concern in data warehouse design, and classifying hierarchies, creating normal forms for summarization, and tracking summary constraints in conceptual models are important first steps. However, it is important the data warehouses be built flexible enough to handle business querying needs. This involves allowing non- or semi additive measures, which raises the potential for inaccurate summaries. We suggest that since all of the issues cannot be designed out of OLAP systems, the constraints must be included in metadata.

4. CONCLUSIONS

Accurate summarizability is a chief concern in OLAP systems. In this paper, we examined the effects of non- and semi-additive facts on the accuracy and meaning of summary data. From this, we developed a taxonomy of semi- and non- additive attributes. We were also derived several rules that can help with the recognition and handling of potentially inaccurate summaries. Figure 8 and Figure 9 depict the methods for identifying and working with non- and semi- additive attributes.

There is no simple solution to resolve problems associated with inaccurate summary data. Approaches should include a combination of considering summarization constraints from the conceptual phases through implementation. This could be better achieved by storing summarization characteristics in a machine readable data dictionary and by creating select query triggers that ensure summary constraints are not violated.

5. ACKNOWLEDGEMENT

This research of Peter Chen was partially supported by National Science Foundation grant: ITR-0326387 and AFOSR grants: F49620-03-1-0238, F49620-03-1-0239, and F49620-03-1-0241.

6. REFERENCES

[1] Adamson, C., Venerable, M. (1998). *Data Warehouse Design Solutions,* John Wiley and Sons, Inc.

[2] Bedell, J. (1998). "Outstanding Challenges in OLAP." Data Engineering. *Proceedings of 14th ICDE*, 23-27 Feb. 1998. 178 – 179.

[3] Chaudhuri, S., and Dayal, U. (1997). "An Overview of Data Warehousing and OLAP Technology". *SIGMOD Record*. 65 – 74. ACM Press, New York, NY.

[4] Ewen, E. F., Medsker, C. E., and Dusterhoft, L. E. (1998). "Data Warehousing in an Integrated Health System; Building the Business Case". *DOLAP* '98, Washington, DC.

[5] Golfarelli, M., Maio, D., and Rizzi, S. (1998). "Conceptual Design of Data Warehouses from E/R Schemes". *Proceedings of the Thirty-First Hawaii International Conference*, 6-9 Jan. 1998, 7, 334 – 343.

[6] Holowczak, R., Adam, N., Artigas, J., and Bora, I. (2003). "Data Warehousing in Environmental Digital Libraries". *Communications Of The ACM,* September 2003, 46. 172 – 178.

[7] Hüsemann, B., Lechtenbörger, J, and Vossen, G. (2000). "Conceptual data warehouse design". *Proc. Of International Workshop on Design and Management of Data Warehouses*, 2000.

[8] Kim, B., Choi, K., Kim, S., and Lee,D. (2003). "A Taxonomy of Dirty Data". *Data Mining and Knowledge Discovery*". 81-99.

[9] Kimball, R. and Ross, M. (2002). *The Data Warehouse Toolkit*: Second Edition. John Wiley and Sons, Inc..

[10] Lehner, W. (1998). "Modeling Large Scale OLAP Scenarios". In *Proceedings of the Sixth International Conference on Extending Database Technology*, 153—167.

[11] Martyn, T. (2004). "Reconsidering multi-dimensional schemas". *SIGMOD Record*. 33 - 1. 83 - 88 ACM Press New York, NY.

[12] Pourabbas, E. and Rafanelli, M. (1999). "Characterizations of hierarchies and some operators in OLAP environments". *Proceedings of the 2nd ACM international workshop on Data warehousing and OLAP (DOLAP'99)*. Kansas City, Missouri. 54 – 59.

[13] Shoshani, A. (1997). "OLAP and statistical databases: Similarities and differences". *Proceedings of the sixteenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*. Tucson, Arizona. 185 – 196. ACM Press New York, NY.

[14] Trujillo, J., Palomar, M. Gomez, J., and Song, I. (2001). "Designing Data Warehouses with OO Conceptual Models". *IEEE Computer*. V34, No 12, 66-75.

[15] Tryfona, N., Busborg, F., and Borch Christiansen, J. (1999). "StarER: A Conceptual Model for Data Warehouse Design". ACM, *DOLAP* '99 Kansas City, MO. USA.

[16] United States Environmental Protection Agency (US EPA). 1997. National Ambient Air Quality Standards for Particulate Matter, Final Rule, US EPA, Part 50 of Title 40 of the Code of Federal Regulations.