International Journal of Engineering and Techniques - Volume 3 Issue 3, May-June 2017

RESEARCH ARTICLE OPEN ACCESS

A SURVEY OF LOGICAL MODELS FOR OLAP DATABASES

Dr.T.Geetha[1], R.Karthikeyan[2], Adarsh S[3], Bijo BM[4]

[1,2] *Asso.Prof, Dept of MCA, Gnanamani college of Technolgy, Namakkal, INDIA.*

[3,4] *P.G.Scholar, Dept of MCA, Gnanamani college of Technolgy, Namakkal, INDIA.*

Abstract:

Most of us understand spreadsheets and table data.Few can understand databases but only very few can really process the data into meaningful information. Most companies professionals acquaint themselves with pivot tables which summarise data and provide the performance figures at a glance. But for advanced analysis of data, professionals are hired to run the data marts and warehouses. For faster results data in warehouses is often aggregated and presummarised and so called OLAP Cubes are constructed. The start point is usually a flat database(s) or relational databases which are usually designed from the perspective of transactional data. For any OLAP based work, it is said that 90% of efforts are made in construction of the Cube. This paper is written with the perspective of easy understanding of OLAP and allied concepts. Two Sample databases have been referred; one the Food Mart sample database supplied as part of SQL Analysis Services and second is the Global Terror Data (GTD) available on University of Maryland website as a flat file MS Excel Sheet.

Keywords — **OLAP, Terminology, Conversion, Structures,Concepts**.

I. INTRODUCTION

In 1993, E.F. Codd coined the term OLAP [1]. 1To get started with OLAP, one has to invariably bring data from relational database(s) into cubes. Knowledge of the structure of OLAP data is thus important. One has to find the OLAP structures in a relational database. For that one has to describe the OLAP structures themselves – that one is trying to create in an OLAP database While academic researchers propose formal mathematical models of multidimensional databases, the Industry uses concrete software tools that implement OLAP [2,3] such as Analysis Services in various database servers like SQL Server 2000 or SQL Server 2005. The latter method has been referred to present a gist for easy understanding. The understanding improves if one has browsed an OLAP cube to experience what it's like. There are few websites, one referred here[4] which present dashboards displaying OLAP cubes in a user friendly manner. One needs to see how dimensions, levels, measures, etc. actually look to an OLAP end-user. OLAP presents data in a different way. One has to befamiliar with the look and feel of OLAP data and acquaint with some OLAP concepts. Some important ones have been defined/ described as best understood by the authors. It is felt that these concepts are needed by everybody who use OLAP as well as for people who are developing an OLAP system.

II. THE MOST FREQUENT TERMS/ CONCEPTS

The terms/concepts described herein include OLAP, Browsing, Calculated Measures, Cell, Child Members, Cube, Current Member, Data Mining, Dicing, Dimension, Dimension Table, Drilling Down, Drilling Up, Drill-Through, Fact Table, Hierarchy, Level, Local Cube, MDX, Measure (Fact), Member, Member Property (Attribute), Ragged Hierarchy, Set, Slicing, Snowflaking, Star Schema, Tuple, Unbalanced Hierarchy. A short-hand definition is highlighted followed by a lengthier description.

A. OLAP

1) ***The Acronym:*** **OnLine Analytical Processing** – logically speaking, would seem to include any computer application that is used to analyze data, but nobody uses it like that. Thus as an acronym it is hazy – when you tell someone what the letters stand for, one still doesn't have the slightest idea what the software actually looks like.

2) ***Popular Definition:*** **A Million Spreadsheets in A Box** – This is what best described to a beginner. OLAP basically evolved as a powerful spreadsheet tool and today it is prescribed in a big way. It is flexible, but its basic purpose is to show spreadsheets with key ability to navigate to different views of the data. One does not have to bank on technical people every time to view the data in a new way from one perspective to another, helping in tracing some information of interest, either in details or a more general perspective.

3) ***Technical Definition: Fast, Interactive Browsing of Multidimensional, Multi-Level Data and OLAP Browsing–*** OLAP always involves multiple dimensions, which should have multiple levels. In the absence of levels, OLAP browsing does not have much power.

4) OLAP is interactive: OLAP browsing is something done by a human analyst. As the analysts view the data, they can ask new questions of the data and receive immediate answers. If one has a data analysis application that does not return results of new queries immediately, one cannot claim to have OLAP. To make this possible, data is organized in multidimensional structures, with some of the summary (aggregated) views in the data pre-calculated ahead of time. For a common man thus, OLAP could be replaced with terms

omewhat like Interactive Spread sheeting. "OLAP Browsing" may be more aptly called "Data Browsing".

B. Data Mining

Non-Interactive Computer Analysis of Data: While Analysis Services can be utilised for both OLAP and data mining. Both analyze data but difference is that OLAP is used interactively but in. Data Mining the computer analyzes data and reports significant results back to the analyst.

C. Cube

Multidimensional Way to Look at Data: Cube is the primary OLAP structure used to view data. It is analogous to a table in the relational database world. The term cube implies three dimensions, but OLAP extends the concept further. Modern day Analysis Services cubes can have 128 dimensions (Although the term implies three dimensions, a cubecan theoretically have any number of dimensions; infact, most real-world cubes have four to 12 dimensions [5,6]. Cubes contain all the dimensions on which users base their analyses of fact data. An instance of a database dimension in a cube is called a cube dimension and relates to one or more measure groups in the cube. A database dimension can be used multiple times in a cube. For example, a fact table can have multiple time-related facts, and a separate cube dimension can be defined to assist in analyzing each time-related fact. [7] However, only one time-related database dimension needs to exist, which also means that only one time-related relational database table needs to exist to support multiple cube dimensions based on time.

D. Dimensions

The Perspectives Used for Looking at the Data or simply the categories Used for Columns and Rows in the Spreadsheet :

Dimensions are the answer "How do you want to see your data?" Some more widely understood examples of dimensions as can be seen in commercial data warehouses are: –

Prod
uct
Tim
e
Store
Customer Age
Customer
Income
Employee

Dimensions have levels which are used for Drilling Down to get a detailed view and Drilling Up to get a generalized view. Dimensions have to be carefully designed, for in them lies the value of OLAP. For GTD database we propose these contenders for dimensions as under :- Time or the incident date with year as top level dimension and month, day at subordinate levels. (fields iyear, imonth, iday refer). In this case if one notices that incidents are higher in a particular year, one may want to drill down to see if they were higher in a particular month. If one has to see how different incidents have recurred in different time periods, one can put the different incidents on the columns and the different time periods on the rows (or the other way around).

Location from the 'region_txt' fields as top level field and 'country_txt' fields as the subordinate level. One can add other subordinate levels of province/state and further city, but these entries have not been standardized in the GTD hence our proposed cube shall be sub-dimensioned only to the country level. Incident subcategorized by various criteria coded under 'crit1', 'crit2', 'crit3', 'doubterr', 'multiple', 'conflict; fields. The three criteria fields could be independent dimensions with 'doubter' at lower level of dimension hierarchy to make out ambiguous cases under each criteria. This would be more useful for incidents which do not meet all the three criteria. Target Information represented by 'targtype_txt' fields. Attack types represented by the field 'attacktype_txt' with 'success' at the lower level of dimension. Hostage/Kidnapping Information represented by 'ishostkid' with lower level of dimension as out come of the incident 'hostkidoutcome'field. Ransom information represented by 'ransom' field sub categorized by other ransom related fields.

Perpetrator Information represented by 'gname' i.e group names. For looking at the perpetuators who are causing the occurrence of incidents at different sites one may use a different row for each group and a separate column for each site. One may drill up to a higher level, to see if the data patterns found by data mining are valid on a wider scale. Weapon Information represented by 'weaptype' as top level dimension and 'weapsubtype' field as a child dimension.

E. Hierarchies

Levels are Organized Into Hierarchies: Dimensions have hierarchies and hierarchies have levels. But often dimensions have single hierarchy, and in that case hierarchy can be ignored, for most part. But there are times when a single dimension has more than one hierarchy – such as when the levels of a Time dimension are organized by Calendar Year and by Fiscal Year. In such a case Time dimension can be thought of as organized by Calendar Year hierarchy and Fiscal Year hierarchy. In our sample data as far as time dimension is concerned there is only one hierarchy iyear – imonth – iday. If an OLAP tool does not handle hierarchies well, they may not be seen, or if seen they might appear as separate dimensions. For e.g. if we set to view confirmed/ doubtful cases of terrorism based on three point criteria fields crit1', 'crit2', 'crit3', then the 'doubtterr' and the three criteria fields may appear to a beginner to be cases of multiple hierarchy, whereas they are really separate dimensions.

F. Members

1) Levels Have Members: Members may be best understood as the Labels for the Columns and Rows in the Spreadsheet –

For eg. for a dimension called Product, there could be a level called Department, and the members of the level Department could be Hardware, Plumbing, Lumber, etc. In our sample GTD data, for the Time dimension, one of the Levels is the 'iyear' with sub level as 'imonth' might be called Month, and the members of the level Month would be the members 1,2,3, etc standing for Jan , Feb, Mar etc.

2) Drilling Down to Child Members: This implies looking at data at a particular level and when we want to see data in details. When drilled down to the next lower level, the members seen are called child members. For example, in case of an OLAP cube

made as discussed, if one drills down on 'iyear', the child members 01-12 meaning January to December will be seen. Relationship between members within a dimension are described by a variety of family terms
– parent, sibling, cousin, descendants, ancestors, etc.

3) ***Drill-Through (Jumping from OLAP Back to the Source Data):*** While drill up and drill down operations allow viewing different levels in the data, some OLAP systems permit jumping back to source data. One can select data of interest in the cube and then drill through to the source data to view extra detail. In our proposed OLAP cubes we shall aim at using such an OLAP tool so that the results of data mining can be analysed in light of detailed descriptions recorded in the members 'addnotes' (additional notes), 'summary' (Incident summary) , 'location' (description of the location) etc.

4) Set (A Collection of Members): One usually views more than one column and more than one row in a spreadsheet. The group of members that is shown on the columns and the group of members that is shown on the rows are called sets. We shall be using the following example representations in our OLAP descriptions.

By listing each member – {April, May, June}

By a family relationship – the children of Quarter 1 – [Quarter 1].Children

All the members of a level – Time.Month.Members

All the members of a dimension – Time.Members

5) ***Attributes (Properties of Members):*** Extra information about members can be seen by creating member properties. An Incident level could have member properties such as Incident ID, place, type etc. For e.g. In our proposed GTD cube the property level can have propextent_txt (values unknown, minor, major, catastrophic), propvalue (defining damage in $) and 'propcomment' (Comments/description) as attribute members. A beginner in OLAP may be inclined to view nkillus (Limited only to US fatalities.), nkill (Confirmed fatalities including victims), nkillter (Perpetrator Casualties), nwound (Confirmed non fatal injuries incl victims and perpetrators), nwoundus (Limited only to US nonfatal injured.) nwoundte (Perpetrators wounded) as attributes of the dimension Casualty but it may be more appropriate to view them as measures.

6) ***Slicing (OLAP Filtering):*** A dimension can be used for slicing when it isn't being displayed on the columns or rows of the spreadsheet. One can slice on any member from any level of the dimension. Slice on the member '04' in the 'imonth' level of the Time dimension, will only show data from April.

7) Dicing (Combining of Dimensions): More than one dimension can be put on the rows or on the columns. One will then see one row for every combination of the members from each of the dimensions. When data is cut with one member it's called slicing. When data is cut with sets of members from two or more dimensions, it's called dicing, akin to cutting up vegetables from every direction. For e.g. In our proposed GTD OLAP cube slicing Time dimension at level 'imonth' on member '04' as above in combination with slicing location dimension on member 'India' in the 'country_txt' level shall show incidents occurred in India from April.

8) Tuple (Members Defining a Row or Column): Just as in geometry, a point is defined by its x-y coordinates (x, y) which can be thought of as x and y being members of the x-dimension and the y-dimension. In OLAP, each row or column is defined by the members that are used for that row or column.

9) ***Set (Group of Tuples):*** A group of tuples makes a set. In the simplest case, those tuples only have a single member.

10) ***Current Member:*** Every cell in a cube is defined by one member from every dimension in the cube. That one member is called the current member for that dimension. The current member for each dimension is determined as follows:

- If the dimension is used on the columns or on the rows, the current member is the member for that dimension used in the tuple defining the column or row.
- If the dimension is used for slicing, the current member is the member used in the slice.
- If the dimension is not used for columns, rows, or slicing, the current member is the default member for the dimension. The default member is often the All level member. When the All level member is used, the practical effect is to ignore that particular dimension in the display of the data.
- *To summarize members, tuples, and sets:*

  Rows and Columns of an OLAP Spreadsheet are defined by a set of tuples. Each row or column is defined by a single tuple.

  Each tuple is defined by a single member from one or more dimensions. In a simple case, when only a single dimension is used for a row or column, the tuple has only a single member.

G. Measures (or Facts):

1) The numbers in the OLAP spreadsheet: When setting up OLAP cubes, these values are also often called facts. Data warehouses commonly include three types of facts [5]: Events, at least at the finest granularity, typically model real-world events, with one fact representing the same instance of an underlying phenomenon. Examples include sales, clicks on a Web page, or movement of goods in and out of a warehouse. The examples cited in this paper are mostly based on this model. Snapshots model an entity's state at a given point in time, such as store and warehouse inventory levels and the number of Web site users. The same instance of the underlying real-world phenomenon—such as a specific can of beans on a shelf may occur in several facts at different points in time. Cumulative snapshots handle information about activity up to a certain moment. For example, the total sales up to and including the current month this year can be easily compared to the figure for the corresponding month last year.

2) **Typical measures would be:** nkillus (Limited only to US fatalities.), nkill(Confirmed fatalities including victims), nkillter (Perpetrator Casualties), nwound (Confirmed non fatal injuries incl victims and perpetrators), nwoundus (Limited only to US nonfatal injured.) nwoundte (Perpetrators wounded).

3) Cells of a Cube: **Where the facts/ measures/ numbers are displayed** – The spaces in the OLAP spreadsheet are called cells (like the cells in an Excel spreadsheet).

4) *Calculated Measures:* **Applying Formulas to Multidimensional Data** - Some measures are more interesting when combined with other measures or analyzed from perspective of a particular dimension. With calculated measures one can view data across different time periods or calculate the percentage each incident contributed to total casualty figures.

H. Dimension Tables

1) Store Dimension Information: The information used for OLAP dimensions is contained in relational database tables which are often called dimension tables. Dimension tables have the following types of fields:

- **Key fields** join the dimension tables to the fact table (and to other dimension tables when snow flaking). For e.g in case og GTD if we consider attack information dimension table comprising fields 'success', 'suicide', 'attacktype_txt', then the latter could serve as a keyfield join to the fact table.
- **Level name fields**, which are used to store the member names for the levels. For example, the Time dimension table could have a field called Month, which would have values such as April, May, Jun etc which in our proposed implementation are stored as numeric values from 1-12 depicting months. Likewise for the 'attacktype_txt' field the GTD, its various values in the field attacktype_txt field of the could act as level name fields.
- **Level Order Key fields**, used to store integer values used to order the members of the levels (if necessary). For example, the Time dimension table could have a field called Month Order Key, which could have a value of 1 for January, 2 for February, 3 for March, etc. Likewise 'attacktype' field could be used to store integer values for ordering the 'attacktype_txt' members of the attack information dimension.
- Member Property fields, used to store the member property information. A Time dimension could have a field called Day Count, which would store the number of days for each month.

I. Fact Table

1) Stores Information for the Measures: The information used for the measures is contained in a relational database table called the fact table. Whenever a new cube is made in the Analysis Manager, the first question one has to answer is to identify the fact table. Fact tables have the following types of fields:

- Key fields, connecting the fact table to a dimension table for each of the dimensions. Every fact must be connected to one particular member from the lowest level of every one of the dimensions in the cube.
- Measure fields, containing the numeric values used for the measures.

2) For e.g. in our proposed cubes if we have Attack Information, Weapon Information as dimension tables then the selected Key fields in fact table could be attacktype and weapsubtype from dimension tables and measure fields could be fields from Casualty dimension with 'nkill' (Confirmed
fatalities including victims), 'nkillter' etc. as measures depending upon the cube of interest.

J. The Star Schema

A Multidimensional Data Structure in a Relational Database:

The combination of a fact table with its dimension tables is called a star schema. The fact table is at the center of the star, while each dimension table represents a point of the star.

K. The Snowflake Schema

Snow Flaking the Star Schema: The basic star schema has a simple structure where all the information for each dimension is contained in a single table. Snow flaking is the process of dividing information for one dimension among two or more tables. The resulting data structure is often called a snowflake schema.

L. MDX

Querying Language for OLAP: A special language developed for OLAP. This language looks similar to SQL, but it has many unique features. MDX is used for several purposes as listed below:

- ✓ To create calculated members – members which are not in the data, but can be derived from the existing members and measures.

- ✓ To define security rules for accessing data. To retrieve data from a cube.
- ✓ To create calculated sets.

M. Unbalanced Hierarchy

Dimension with levels that can appear or disappear: Also termed Parent-Child Dimension. Most dimensions in Analysis Services have a fixed number of levels. But a parent-child dimension can display levels that have an indefinite depth. This is useful for hierarchical structures that are constantly changing, such as who's manager – worker relation. In the GTD data we could not locate any such unbalanced hierarchy. Summarizability requires distributive aggregate functions and dimension hierarchy values [7]. Informally, a dimension hierarchy is strict if no dimension value has more than one direct parent, onto if the hierarchy is balanced, and covering if no containment path skips a level. Intuitively, this means that dimension hierarchies must be balanced trees. As Figure below shows, in the case of irregular dimensions, some lower-level values will be either double-counted or not counted when reusing intermediate query results.

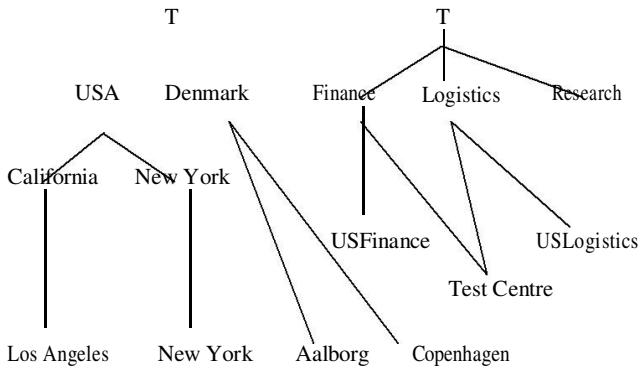

Figure 1. Irregular dimensions. The location hierarchy to the left is noncovering because Denmark has no states. The hierarchy to the right is nonstrict as Finance and Logistics share the TestCenter.

N. Ragged Hierarchy

A Dimension Where Some Parents Have Missing Children (But Do Have Grandchildren): It's fine in the hierarchy that has the Country level, the State level, and the City level. But for small countries not having states a ragged hierarchy allows skipping levels where there are no real values as in case of Denmark depicted in Figure 1 above. Our proposed cubes with Incident location can have a ragged hierarchy of as it covers all the countries of the world and not all of them have states/provinces. Irregular dimension hierarchies occur in many contexts, including organization hierarchies [8], medical diagnosis hierarchies [9] and concept hierarchies for Web portals such as Yahoo! One solution is to normalize irregular hierarchies, a process that pads non-onto and noncovering hierarchies with dummy dimension values to make them onto and covering, and fuses sets of parents to remedy the

O. Actions

Beyond OLAP Browsing , Doing Something: OLAP Browsing is a fairly passive activity, but Analysis Services allow actions right from inside the cubes such as opening a web page for the person browsing. The person finding a product that needed ordering, could order it on the spot. If one can expand the cubes formed from GTD with Terrorist Organisation activities in details, then actions which could be taken include finding a potential sleeper cell getting into action and sending a warning e-mail to security agencies. This could be done all with actions inside the cubes.

P. Local Cubes

Files that can be used for disconnected access to cube data:

OLAP cubes are usually stored in an Analysis Server. Applications can browse these cubes by connecting to the Analysis Server. But cube data can also be put into local cube files, which can be used when users do not have a connection to the Analysis Server. Browsing a local cube file appears to the users to be the same thing as browsing a cube stored in an Analysis Server. Local cubes are particularly useful in a pilot project, when everyone is not yet hooked up to an Analysis Server. We shall try and follow this approach for piecemeal analysis of proposed cube.

V. FINDING THE MEASURES, DIMENSIONS, LEVELS, AND MEMBER PROPERTIES

Coming on to real world situations, it can be a complex exercise finding measures, dimensions, and levels in a relational database. It is easy to construct cubes if data is already organized into a star schema. However if data is in a normal relational database, it is not so obvious what should be used for measures, dimensions, levels, and member properties.

To our mind there could be three ways of deciding on the structures to be included in the cubes as follows:

- ✓ By looking at the GTD spreadsheet currently supplied by the UMD website.
- ✓ By looking at the elements in the source data.
- ✓ By anticipating what the users really want to see.

All ways are important. If the spreadsheets are currently in use in reports, the information in them should probably be included in the OLAP cubes. But then it is quite possible that the source data may have additional information which has been ignored in existing reports. When one is building an OLAP system, he has to snatch the opportunity to give the users more of the data than they saw before.

VI. BY LOOKING AT SPREADSHEETS CURRENTLY IN USE

The best way to discover what is needed in the cubes may be to see what an analyst writes in the margins of their spreadsheets as perceptive analysts realize that the reports they receive do not show all the significant information. They take the data in the reports and do additional calculations or comparisons in the margins and those notes can tell us what is important. For

example in a business scenario a report may show information about orders, including an order date, a required date, and a delivery date - but doesn't show whether or not an order was overdue, or the number of days an order was overdue, or the number of days allowed between order date and required date. An analyst concerned about on-time delivery for a particular customer would then have to scan through the report to find that customer and then make notes regarding the difference between the various dates and the content of the particular orders. Maybe the customer had overdue orders because of unrealistic delivery times. Maybe the overdue orders were caused by some particular item included in the order.

Thus while designing the cube, one can include the number of days between the various dates kept precalculated. Because it's in a cube, the analyst can easily find the overdue orders and analyze them from any of the available dimensions - by product, by time period, by customer, by customer geographical location, etc. In case of GTD similar scenarios can be constructed by finding the difference between dates of incidents claimed by individual terrorist organizations to make out the activeness of groups and predict their next moves.

VII.BY LOOKING AT ELEMENTS IN THE SOURCE DATA

Our fundamental goal when developing OLAP structures is to find out what perspectives, calculations, and comparisons are important to our users and then to include those elements in our cubes.

VIII. BY ANTICIPATING WHAT THE USERS WANT TO SEE

It's a little hard to imagine users asking for everything from a relational database, but it's not at all unusual for users to give a long list of items that should be included in the cubes. We should be able to include any data in our OLAP structures - no matter what it is but if one cannot use a table or a field one should explain why it can't be used. Depending on what the users want, the analyst may end up designing cubes in either of the following ways:

Simple ones containing only one hierarchy per dimension. Those which contain multiple hierarchies within several of the dimensions Those which contain all the hierarchies in separate dimensions These cubes can be created on various platforms and stored in separate cube files. These can then be used on any computer that has the Office XP or Office 2003 Office Web Components installed. Local cube files can also be zipped to enable downloading and browsing with a different OLAP browser.

IX. KEY POINTS IN THE PROCESS OF FINDING OLAP STRUCTURES [11]

Following key points may be adhered to while designing the OLAP Cubes:-

A. Key Point 1

Almost every piece of data in a source database can be used in an OLAP structure - and some of them should be used more than once, either to create a measure, a level, or an additional attribute (member property in SQL Server 2000 onwards). Some fields can be used both as measures and as attributes.

B. Key Point 2

Even key fields in the spreadsheet or relational database can be used in the cubes – by including them as member properties or attributes. These fields, which in normal course are used to identify an individual record or to maintain referential integrity between tables, often in some systems may have meaning to the users. E.g. they may refer to a product, for example, by the Product ID number. But even if they don't have any meaning, these key fields can be valuable for connecting/ interfacing the OLAP cubes to other systems - opening existing reports or to initiate a process, or creating an interface from the cubes to the transaction systems. In our case the GTD ID 'eventid' can be used for connecting to other systems such as Terrorist databases.

C. Key Point 3

Every non-key numeric field is a candidate for creating a measure - either by itself or together with other fields in a calculation that creates a measure. E.g. 'nkill' (This measure would normally be used either to sum up annual casualties or as an Average per month or per year). In a typical business scenario Quantity, Discount could be used to create calculated measures such as average price, extended price after discount, Extended Price Before Discount, Discount Percent, Discount Amount, Extended Price After Discount, Average Item Count, Average Price, Average Margin, Profit etc. For e.g. In our proposed GTD non-key numeric fields in the GTD database are those pertaining to property damages and casualties such as 'propvalue' (defining damage in $) and 'propcomment' (Comments/description) as attribute members. The casualty related measures include nkillus (Limited only to US fatalities.), nkill (Confirmed fatalities including victims), nkillter (Perpetrator Casualties), nwound (Confirmed non fatal injuries incl victims and perpetrators), nwoundus (Limited only to US nonfatal injured.) nwoundte (Perpetrators wounded).

D. Key Point 4

All measures used in a single cube have to use the same level of granularity. E.g Freight as a measure to analyse different shippers, would be easy to use in a cube where Orders table is used as the primary basis for the fact table in the cube as the granularity of freight matches orders. But it is harder to use Freight in a cube where Order Details is used as the basis for the fact table. In this case to ensure that all the measures in the cube are on the same level of granularity, Freight would have to be divided out among the Order Details, equally or proportionally or some method . For GTD database there are no such granularity issues anticipated except when using aggregated measures with different dimensions.

E. Key Point 5

Measures can also be created by counting values in fields - but Distinct Count is often the kind of counting that's needed. Usually Analysis platforms allow defining one Distinct Count measure per cube, and one can define additional ones

using calculated measures. It is not hard to identify such candidates in GTD such as if our cubes were to comprise aggregated number of cases of ransom, suicide bombing, etc, these shall be calculated measure that create a Distinct Count of 'ransom', 'suicide', though it will slow down cube browsing in larger cubes.

F. Key Point 6:

Non-null values can be counted in fields that allow null values.

G. Key Point 7

Measures can be created by comparing the values in date fields. E.g. a measure named OrderedShippedDiff would record the time taken for dispatch of ordered stores as a difference between RequiredDate and ShippedDate. For related terror incidents by some pattern the difference of incident dates claimed by same group can serve as the measure of activeness of the group.

H. Key Point 8

Finding the potential measures in a database can help find the potential fact tables around which cubes can be built. E.g. measures in the Order Details table give transactional information. This table could be used as the basis for a fact table for a cube called Orders, which could have dimensions identifying the Product sold, the Customer, the Employee, and the Time of the transaction. In GTD numerous cubes can be visualized using this approach. If we focus on successful incidents we could have distinct count on 'success' with dimensions such as country, perpetrators, target info.

I. Key Point 9

Every table in the relational database should be considered as a candidate for a dimension. Join tables usually do not qualify for the purpose. Since GTD data is available as an excel sheet all fields can be categorized under different dimensions each of which can be visualized as a separate table with one to one join with 'eventid' as the key field.

J. Key Point 10

Two or more candidate dimensions could be merged into a single dimension if the data in them is highly correlated (or is perceived by the end users as being correlated). One can make out if tables are related, by looking at their foreign key relationships.We have to consider how the tables fit into cognitive groupings which is not possible simply by looking at the structure of the database. Certain common sense questions which may be considered are whether certain types of data belong together or some of the data fit together logically in a hierarchy. Some examples of candidates to common dimensions are:

- ***Example 1:*** Customer and Customer Demographic – The latter presents additional information about former. Likewise in GTD the Info on consequences could merged with Attack info as it presents additional info about the Attack.
- ***Example 2:*** In GTD the Regions, Country and City could be made to form a hierarchy in a single dimension.
- ***Example 3:*** Target and Attack – The latter could be used as a higher level in the former's dimension. A similar case in Weapon and Attack. One has to weigh the advantages and disadvantages when deciding what entities should be combined together in a dimension. Here is a more detailed analysis of the decisions regarding Attack, Target and Weapons used.
  - **Consideration 1** for Example 3. Creating two separate dimensions (or hierarchies) for Attack (AttackByWeapon and AttackByTarget) gives end users two different ways to summarize information for Attacks.
  - **Consideration 2** for Example 3. Users may want to put AttackByTarget on columns and AttackByWeapon on rows. This will look fine unless the user drills down to the Attack level of both dimensions. They can look at Attacks and Weapons while they're looking at Targets or they can look at Attacks and Targets while they're looking at Weapons.
  - **Consideration 3** for Example 3. If both dimensions are used at the same time and both are drilled down to the Attack level, most of the cells of the grid will be empty. It's not logical to have the same level on the rows and on the columns at the same time.

K. Key Point 11

Numeric values can be converted into dimensions by employing them individually or by dividing the values into ranges. SQL Server Analysis Services in provide a way for the server to generate the desired ranges while in SQL Server 2000 ranges have to be created in the source database. Extent of Property Damages or number of casualties could be divided into such ranges to be converted to dimensions.

L. Key Point 12

Tables that contain measures may or may not also have values which can be used to create a dimension.

M. Key Point 13

If a table like Order Details is used as the fact table, the Orders table can be used to create a simple dimension that allows users to view individual orders - but this often doesn't work out very well. Order doesn't make a very good dimension, because it most likely doesn't have any natural levels. But then many people want to have this kind of dimension to be able to view the individual orders as they are browsing the cube. In such a case some kind of artificial levels may be added e.g. a dimension which divides the OrderID values into ranges of 100 values (10200-10299, 10300-10399, etc.) It is also helpful to create a set of local cubes, each with a subset of the Order Details to improve browsing speed performance. Similar subsets could be created in GTD by using the 'eventID' field wherein the date is encompassed in the form 'yyyymmdd' and the first 4 digits could be used to form subsets by years or if we use 6 digits then subsets could be formed with granularity defined upto months.

N. Key Point 14

OLAP cubes nearly always have a Time dimension as a hallmark. Exceptions are rare and GTD is no exception either.

O. Key Point 15

Almost any date field in the relational database can be made into a Time dimension. The date field in the GTD can be constructed from a combination of 'iyear', 'imonth', 'iday'. The 'resolution' field or the date of resolution of incident is another candidate for Time dimension.

P. Key Point 16

The data load date can also be used as the basis for a Time dimension. While building a cube which is focussed on transactions, the date fields in the data are likely to be used for the Time dimension. But while building an inventory type of cube, the Time dimension will often be based on the date the data is loaded into the data mart. In Inventory scenario UnitsInStock and UnitsOnOrder fields can be used for tracking inventory . If the values of these fields in the source database were saved on a daily or weekly basis, a cube could be built that could be used to analyze the change in the inventory, basing the Time dimension on the stock entering date. In GTD the incident occurrence date can be used as Time Dimension.

Q. Key Point 17

The levels of a Time dimension should match the time periods that analysts want to use to analyze their data. One date field can be used to create all the levels of a Time dimension - All Time, 5-Year Period, Year, Half-Year, Quarter, Month, WeekInAllTime, WeekInYear, WeekInMonth, DayOfYear, DayOfMonth, and DayOfWeek. One should use the particular levels that the analysts want to see. There are many cases where the analysts want two or more separate hierarchies in the Time dimension, because they like to drill down into the Time dimension in different ways.

R. Key Point 18:

Except for the Time dimension, each standard level is usually built from the data in a single field. For example, one could build a dimension for the Product dimension consisting of Category on a higher level and Product at a lower level. This works well only if each Product is a member of one and only one Category. In GTD for the data in MultiCategory fields such as 'attacktype' can be used to build the 'Attack' dimension in the GTD cubes while data in the 'Region' and 'Country' fields can form a two-level dimension called 'Location'.

S. Key Point 19:

Geography is a convenient source for levels in a dimension. Geography works well for levels because lower geographical levels are often fully contained in higher ones. Most cities are contained in a particular state and most states are contained within a particular country. Districts and regions are often more of a problem, because their definition can vary within an organization and they can change over time. Metropolitan areas are much harder, because they often don't fit in a hierarchy with states - or even with countries. The location dimension hierarchy in the GTD could be formed from region, country and city fields.

T. Key Point 20:

Sometimes end users may need less obvious levels into a hierarchy for particular s needs. The following could be such example levels for the Attack dimension; not so logical but may be exactly the way analysts want to see their data:

Attackty
pe
Success
Group

While an analyst can suggest a better solution that users will like than what they originally wanted. One option would be to pull Group out of the hierarchy and use it as a separate dimension, with two levels (the All Level and the Success Level) and two members at the lowest level (Yes and No). When browsing the cube, this dimension could be used as a slicer, so that the users could see only the successful attacks or the ones that had not been successful or successful groups. It could also be used together with the Attacktype on the rows or columns so that the users would see the following levels:

Attackty
pe
Success

OR

Attackty
pe
Group
Success

U. Key Point 21:

Parent-Child Dimensions vs. Standard Dimensions – A need for such dimensions arises when hierarchies have an indefinite depth and defined by key relationships. Employees table is a typical example of such a parent-child dimension there is a key field, relationship ReportsTo, that links back to the primary key field EmployeeID, and defines a supervisory chain. This hierarchy has an indefinite depth. In SQL Analysis Services if we draw out the organization chart, it could have 7 levels of supervision. But if a new employee is hired and placed under an employee at the 7th level it shall be making a new 8th level. This could be done in the source database simply by putting the appropriate value in the ReportsTo field.

Employees could be organized with a fixed, non-parent-child hierarchy. If, for example, the employee table had fields for Immediate Supervisor, Supervisor, and Shift Lead, we would use those three fields to form fixed levels in a standard dimension. But in Analysis Services, it is better to use a standard than a parent-child dimension because of performance considerations. Given a choice, a standard dimension should be built. But if we have a situation where the hierarchy has an indefinite depth, a parent-child dimension should be used.

In GTD sample the need for Parent-Child dimensions does not arise. The terror groups and subgroups could be coded as standard dimensions. However if the cubes are extended to include details of personnel belonging to the groups then these terrorists could be coded in Parent-Child dimensions to code their hierarchy of reporting.

V. Key Point 22:

When one has multiple ways within a dimension that one can drill down, multiple hierarchies can be created for that dimension. E.g. in the Employee dimension, few possible hierarchies could be as follows:

✓ ***Group. Identity Hierarchy:*** Simple listing of Group Names.
EmployeeName

✓ ***Terror Groups Hierachy:*** Where groups mostly operate.
Region->Country-> Group

✓ ***Group Hierarchy:***

Standard Hierarchy: Group and their sub-groups.
ParentChild: Group and Reports To.

✓ ***Groups.Territory Hierarchy:*** The regions and territories where groups are assigned.

✓ Alternatively one can create separate/ independent dimensions, with each dimension having a single hierarchy. Examples as under:

GroupIdentity
GroupGeography
GroupMembersReportTo
GroupTerritory

W. Key Point 23:

Data should sometimes be excluded if there is a many-to-many relationship between that data and the fact table. Group.Territory can't be used because there may be many territories assigned to each group for different operations and there is no obvious way of determining which of those territories is represented by each order. To use this dimension, there would need to be an additional field in the fact table to identify the territory for the order.

X. Key Point 24:

All other fields (almost) can be used as member properties or attributes that provide additional information. They are displayed in different ways by different OLAP client applications. One typical way is to display the member property information when the user hovers over the particular member. In some Analysis Tools each attribute even can have its own hierarchy.

Y. Key Point 25:

One may create member properties (attributes) for fields being used to create measures. For example the QuantityPerUnit field can be used as a measure, but also as a member property of the Product. Depending on how the OLAP client tool displays measures and attributes, it may be appropriate to make this information available in more than one way. Before one spends a lot of time building fields to be used in Actions, it be ensured that the OLAP client tool supports the kind of Actions to be created.

X. THE FINISHED OLAP STRUCTURES

Here we apply our concepts listed above to present proposed finished OLAP structures based on the two data samples, Food Mart and GTD. While Food Mart Sample is available with Microsoft Analysis Services, the GTD database description from an OLAP perspective [12] may be referred before analyzing the proposed cubes in the following section. The following conventions are being followed:-

❖ *Dimension Names and Hierarchy Names are listed together, separated by a dot:*
DimensionName.HierarchyName

❖ *The Level Names are bullet-listed with single indentation under each Dimension and Hierarchy.*

❖ *The Member Properties (Attributes) are italicized in each level.* When a level (such as CompanyName) is used in several hierarchies, the member properties for the level are only listed once.

XI. PROPOSED CUBES BASED ON FOOD MART

➢ *Typical measures (for an Orders Cube (Food Mart)):*
UnitPrice, Quantity, DiscountPercent, DiscountAmount, ExtendedPriceBeforeDiscount, ExtendedPriceAfterDiscount, Freight, Order Count, Order Detail Count, Order Detail Shipped, Percent Shipped, AVGOrderedRequiredDiff, AVGOrderedShippedDiff, AVGRequiredShippedDiff

➢ *Typical measures (for a Product Inventory (Food Mart)):*
QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder, ReorderLevel, ValueInStock, ValueOnOrder

➢ *Customer.Identity (Food Mart):*

CompanyName (*CustomerID, ContactName, Address, City, Region, PostalCode, Country, Phone, Fax*)

➢ *Customer.Demographics (Food Mart)*

CustomerDesc (*CustomerTypeID, CompanyName*)

➢ *Customer.Geography (Food Mart)*

Cou
ntry
Regi
on
City
CompanyName

➢ *Employee.Identity (Food Mart)*

EmployeeName (*EmployeeID, LastName, FirstName, Title, TitleOfCourtesy, BirthDate, HireDate, Address, City, Region, PostalCode, Country, HomePhone, Extension, PhotoPath, Notes, Reports To*)

➢ *Employee.Geography (Food Mart)*

Cou
ntry
Regi
on
City
EmployeeName

➢ *Employee.Supervisor (Food Mart)*

ParentChild (EmployeeID and Reports To)

➢ *Order (Food Mart)*

OrderRange [Artificially created field on OrderID ranges] (*OrderID, OrderDate, RequiredDate, ShippedDate, ShipVia, Freight, ShipName, ShipAddress, ShipCity, ShipRegion, ShipPostalCode, ShipCountry*)
TimeShipped *(Time levels derived from date field)*
TimeOrdered *(Time levels derived from date field)*

TimeRequired *(Time levels derived from date field)*

- *ShipTo.Geography (Food Mart)*

  ShipCoun
  try
  ShipRegi
  on
  ShipCity
  ShipName *(ShipAddress, ShipPostalCode)*

- *Shipper*

  CompanyName (ShipperID, Phone)

- *Products.Category (Food Mart)*

  CategoryName *(CategoryID, Description, Picture)*
  ProductName *(ProductID, Supplier, QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder, ReorderLevel, Discontinued)*

- *Products.Supplier (Food Mart)*

  CompanyName *(SupplierID, ContactName, ContactTitle, Address, City, Region, PostalCode, Country, Phone, Fax, HomePage,)*
  ProductName *(ProductID, CategoryName, QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder, ReorderLevel, Discontinued)*

- *Supplier.Geography (Food Mart)*

  Coun
  try
  Regi
  on
  City
  CompanyName

- *DiscountPercent (Food Mart)*

  Discount Range Percent
  Percent Discount

XII. PROPOSED CUBES BASED ON GTD

Typical measures for Cubes based on GTD: nkill, nkillus, nkillter, nwound, nwoundus, nwoundte, propvalue, ransomamt, nhostkid, nhostkidus.

- *Attacks.Success (GTD)*

  Success
  Attacktype

- *Perpetrator.Identity (GTD)*

  gname
  gsubname

- *Incident.Location (GTD)*

  Region_
  Txt
  Country_
  txt
  Provstate
  City
  Incident *(Summary, Crit1, Crit2, Crit3, doubter, alternate_txt, multiple, conflict)*

- *Incident.AttackDistributionsCountryWise (GTD)*

  Attacktyp
  e
  Region_
  Txt
  Country_
  txt
  Incident

- *Incident.WeaponsDistributionsCountryWise (GTD)*

  Region_
  Txt
  Country_
  txt
  Weaptype *(Weapsubtype , Weapdetail)*

- *Attacks.ByPerpetrators (GTD)*

  Gpname
  Gpsubna
  me
  Attacktyp
  e

Attacks.ByPerpetrators (GTD)

Gpname

Gpsubname
Attacktype

Attacks.ByWeapons (GTD)

Weaptype

Weapsubtype Weapdetail Attacktype

Targtype Success

Attacks.ByCityWeaponsTargetsPerpetrators (GTD)

Gpname Targtype Weaptype

Region_txt Country_txt City
Attacktype

These are some of the cubes over. We conclude that depending on the problem at hand many more cubes are possible.

REFERENCES

1. E.F. Codd, S.B. Codd, and C.T. Salley, "Providing OLAP On-Line Analytical Processing) to User-Analysts: An IT Mandate," http://www.hyperion.com/solutions/whitepapers.cfm (Nov. 2001).
2. R.Karthikeyan,Dr.T.Geetha "Honeypots for Network Security", International journal for Research & Development in Technology.Volume 7.Issue 2 ,Jan 2017,Page No.:62-66 ISSN:2349-3585
3. R.Karthikeyan,"A Survey on Position Based Routing in Mobile Adhoc Networks" in the international journal of P2P Network Trends and Technology, Volume 3 Issue 7 2013, ISSN:2249-2615
4. T.B. Pedersen, C.S. Jensen, and C.E. Dyreson, "A Foundation for Capturing and Querying Complex Multidimensional Data," Information Systems, vol. 26, no. 5, 2001, pp. 383-423.
5. R.Karthikeyan,"A Survey on Sensor Networks" in the International Journal for Research & Development in Technology Volume 7, Issue 1, Jan 2017, Page No:71-77
6. R.Karthikeyan,Dr.T.Geetha "Web Based Honeypots Network",in the International journal for Research & Development in Technology.Volume 7.Issue 2 ,Jan 2017,Page No.:67-73 ISSN:2349-3585.
7. R.Karthikeyan,Dr.T.Geetha,"A Simple Transmit Diversity Technique for Wireless Communication",in the International journal for Engineering and
14. Techniques, Volume 3 Issue 1, Jan - Feb 2017, ISSN:2395-1303,PP No.:115-120.
15. E. Thomsen, OLAP Solutions: Building Multidimensional Information Systems, John Wiley & Sons, 1997.
16. H-J. Lenz and A. Shoshani, "Summarizability in OLAP and Statistical Data Bases," Proc. 9th Int'l Conf. Scientific and Statistical Database Management, IEEE CS Press, Los Alamitos, Calif., 1997, pp. 39-48.
17. R.Karthikeyan,Dr.T.Geetha"Application Optimization in Mobile Cloud Computing" in the international journal of Engineering and Techniques, Volume 3 Issue 1, Jan - Feb 2017, ISSN:2395-1303,PP No.:121-125.
18. T. Zurek and M. Sinnwell, "Data Warehousing Has More Colours Than Just Black and White," Proc. 25th Int'l Conf. Very Large Databases, Morgan Kaufmann,San Mateo, Calif., 1999, pp. 726-729.
19. UK National Health Service, Read Codes Version 3, Sept. 1999, http://www.cams.co.uk/readcode.htm (current Nov. 2001).

Techniques. Volume 3. Issue 1, Feb 2017, Page No.:56-61 ISSN:2395-1303.

8. C.Ganesh,B.Sathyabhama,Dr.T.Geetha " Fast Frequent Pattern Mining using Vertical Data Format for Knowledge Discovery "International Journal of Engineering Research in Management & Technology. Vol.5,Issue-5,Pages:141-149.
9. R.Karthikeyan,Dr.T.Geetha "Strategy of Trible – E on Solving Trojan Defense in Cyber Crime Cases", International journal for Research & Development in Technology.Volume 7.Issue 1 ,Jan 2017,Page No.:167-171
10. http://www.cubeslice.com/CubeSlice_Performance _Demo.exe
11. R.Karthikeyan,Dr.T.Geetha"Advanced Honey Pot Architecture for Network Threats Quantification" in the international journal of Engineering and Techniques, Volume 3 Issue 2, March 2017, ISSN:2395-1303, PP No.:92-96.
12. 13 .R. Kimball, The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses,John Wiley & Sons, 1996. K.Ramya and K.Pavithradevi "Effective Wireless Communication",International journal of Advanced Research, Vol 4(12), pp.1599-1562 dec 2016.
13. R.Karthikeyan,Dr.T.Geetha "FLIP-OFDM for Optical Wireless Communications" in the international journal of Engineering and
20. T.B. Pedersen, C.S. Jensen, and C.E. Dyreson, "Extending Practical Pre-Aggregation in On-Line Analytical Processing," Proc. 25th Int'l Conf. Very Large Databases,Morgan Kaufmann, San Mateo, Calif., 1999, pp. 663-674.
21. http://cubeslice.com/session2.htm
22. R.Karthikeyan,"A Survey on Position Based Routing in Mobile Adhoc Networks" in the international journal of P2P Network Trends and Technology, Volume 3 Issue 7 March 2013, ISSN:2249-2615.
23. R.Karthikeyan,Dr.T.Geetha"Estimating Driving Behavior by a smart phone" in the international journal of Engineering and Techniques, Volume 3 Issue 2, March 2017, ISSN:2395-1303,PP No.:84-91.
24. Karanjit Singh and Dr Shuchita Bhasin , "Modification of GTD from Flat File Format to OLAP for Data Mining " International Journal of Innovative Technology and Creative Engineering" (ISSN-2045-8711) Vol-1 No.4 Apr 2011 http://issuu.com/ijitce/docs/ijitce_apr