Data warehouse enhancement: A semantic cube model approach

Shi-Ming Huang a, Tung-Hsiang Chou b, *, Jia-Lang Seng c

a Department of Accounting and Information Technology, National Chung-Cheng University, Chia-Yi, Taiwan

b Department of Management Information Systems, National Cheng-Chi University, No. 64, Sec. 2, ZhiNan Rd., Wenshan District, Taipei City 11605, Taipei, Taiwan

c Visiting Fulbright Senior Scholar, Stanford University, and Professor of Information Systems, Graduate School of Accountancy, College of Commerce, National Chengchi University, Wenshan District 116, Taipei, Taiwan

Received 16 April 2004; received in revised form 9 December 2006; accepted 13 December 2006

Abstract

Many data warehouse systems have been developed recently, yet data warehouse practice is not sufficiently sophisticated for practical usage. Most data warehouse systems have some limitations in terms of flexibility, efficiency, and scalability. In particular, the sizes of these data warehouses are forever growing and becoming overloaded with data, a scenario that leads to difficulties in data maintenance and data analysis. This research focuses on data-information integration between data cubes. This research might contribute to the resolution of two concerns: the problem of redundancy and the problem of data cubes’ independent information. This work presents a semantic cube model, which extends object-oriented technology to data warehouses and which enables users to design the generalization relationship between different cubes. In this regard, this work’s objectives are to improve the performance of query integrity and to reduce data duplication in data warehouse. To deal with the handling of increasing data volume in data warehouses, we discovered important inter-relationships that hold among data cubes, that facilitate information integration, and that prevent the loss of data semantics.

© 2006 Elsevier Inc. All rights reserved.

Keywords: Data warehouse; Data cube; Data mart; Object-oriented technology; Semantic cube model

1. Introduction

The function of a data warehouse is to effectively integrate operational databases into an environment that facilitates strategic use of data that, in turn, improves the productivity of a decision-maker through consolidation, conversion, transformation, and integration of operational data. In designing a data warehouse, one would first integrate into one another various data sources from an enterprise’s heterogeneous databases. Whenever data from multiple sources has to be consolidated, developers must analyze the structure and the content of the source before defining the rules for merging. To execute these rules, researchers in the field must develop a satisfactory process. The integration should ensure results that exhibit data consistency across...
the entire enterprise. Ideally, end user should be able to access data from the data warehouse without knowing
either where data resides or the form in which it is stored.

The architecture of data warehouses falls into three categories [3,14,25]:

1. Virtual view approach: The repository of the data warehouse contains only the data schema, and the local
databases store the physical data. This approach uses query pre-processing and query shipping to answer
queries that queries make against the integrated view. The disadvantage of this architecture is its poor
performance.

2. Materialized view approach: The repository of the data warehouse contains the data schema and the phys-
ical data. It collects all relevant information in the data warehouse. The disadvantage of this architecture is
the difficulty it poses in the managing and the maintaining of a huge data bank.

3. Datamart approach: This approach extracts data from a primary data warehouse for Datamart applica-
tion. The extraction has a special purpose. Hence, the repository of the data warehouse contains the subject
of information, which is in Datamart format. The disadvantage of the architecture is that a data warehouse
contains only limited knowledge. This approach cannot manage huge amounts of data.

Most current data warehouse systems, such as the Microsoft OLAP server, the Oracle OLAP server, Sybase
IQ, and Business Objects, use the third architecture as their data warehouse approach. And most of these com-
ercial software tools use data cubes to represent Datamart, and the use has a special purpose. The data type
of the data cube is a multidimensional matrix that lets users explore and analyze a collection of data from
many different perspectives and usually from several dimensions at once.

Use of the Datamart approach for the implementation of an enterprise’s data warehouse will develop many
cubes, as seen in Fig. 1, and each cube is an independent data aggregation. Because a less semantic relationship
has been defined between each of the data cubes, they become isolated bits of information. Users retrieve the
knowledge from one single angle and not from a global view; therefore, problems like data duplication, incon-
sistency, and query integrity could occur.

For instance, in Fig. 2, both Cube C1 ((A, B), (m)) and Cube C2 ((B, C), (m)) are cubes. The C1 cube has
two dimensions (A and B) and has aggregate measure m. The C2 cube also has two dimensions (B and C) and
has aggregate measure m. The new cube C3 has three dimensions (A, B, and C) and has aggregate measure m.

![Diagram of a data warehouse environment](image-url)

Fig. 1. The environment of a data warehouse.
The data of Cube C3 comprises Cube C1 and Cube C2. In this regard, the problems of maintaining both duplication of data and consistency of data could become serious. This research considers connectivity by identifying the relationship between data cubes so as to reduce duplication of data in data warehouses.

Currently, there is much relevant research that discusses the maintenance of data cubes in data warehouses [13,16]. This research stresses both data consistency and reduction of the data duplication of a single data cube. But none of the research discusses the influence that data cubes have on one another. In the past, many researchers thought that the relational model was insufficient. Hence, some researchers increased the semantic data model to account for the insufficiency in the relational model [11,33]. We think that integrating a semantic cube model into data warehouses is a feasible solution that could also resolve the problems of data cube duplication and data cube independent information.

This research describes a semantic cube model that extends object-oriented technology for data warehouses. The model enables users to design the generalization relationship between different cubes, and the objectives are (1) to improve the performance of query integrity and (2) to reduce data duplication.

2. Related research

Although much research has been done on data warehouse design, work on specific methods for the improvement of data warehouse efficiency is scarce in the literature. This research here improves data warehouse efficiency by identifying the semantic relationships that exist between data warehouse data cubes.

2.1. Data cube type and data cube relationship

A multidimensional database or data cube consists of (1) a huge amount of attributes in the fact table and (2) a relatively small set of dimensions with respect to which the data is analyzed [15]. For example, a car cube may store sales amounts, sales models, and so forth. Many research papers have discussed the implementation of data cubes in data warehouses [5,15,18].

In the 1970s, Codd developed a relational model to organize data into databases [9], but it supported only text, the numeric data types that are insufficient for complex applications. The object-oriented (OO) model emerged and reflected researchers’ attempts to solve the problem in the relational model [2,19,26]. At present, most data warehouse systems use a relational data model that conveniently transforms relational databases into data warehouses. During the query process, it is difficult to represent some information in a relational data model, especially in terms of abstract semantics. The semantics of the OO model are much richer than the semantics of the relational model [17]. Most research has suggested that the OO model is appropriate for the development of a data warehouse [3,11,15]. The OO techniques are therefore widely used in data...
warehouses [11], and can provide all kinds of OO characteristics and object features in data warehouses to reduce developing time as well as to increase productivity, especially in the object inherited [6,7].

Hence, this research here uses the OO technique to implement a cube model. To generate the semantics of data warehouses is significant, and with the use of semantics, this research could then develop a more efficient data warehouse and could also reuse the sub-cube to implement—with the generalization method—the new cube.

2.2. The classification of abstract semantic relationships

Several data modeling approaches were reported in the early 1960s. They are the concept data model, the classical data model, and the semantic data model. The conceptual data model retrieves descriptions of both objects and behaviors in the real world. The designer uses this data model to build schema in reality. The classical data model describes three popular data models used in database management systems: the hierarchical, the network, and the relational data models. In order to supplement the insufficiency in classical and conceptual data models, some researchers increase the semantics in a data model [17,22,31].

In general, the semantics of databases fall into two categories. One is user-defined semantics, enabling the user to define the relationship between the data and the entity. For example, users could define the relationship between the two entities in the E–R diagram. The other category is abstraction semantics; it is close to the syntax of the real world and thus accounts for the generalization, the aggregation, and the categorization. And this research presenting a new method is based on this semantic data model [1].

1. Generalization abstraction: It defines a subset relationship between the elements of two objects. For example, a PERSON is a generalization of a MAN and a WOMAN.
2. Aggregation abstraction: An aggregation abstraction defines a new object from a set of objects that represents its complete components. For instance, a CAR is obtained from the BODY and WHEELS.
3. Categorization abstraction: This is used for defining a concept as a real-world object. The concept month is composed of January, February, . . . , and December.

Table 1 summarizes the comparison of the semantic data model and discusses the relevant research on data warehouses.

<table>
<thead>
<tr>
<th>Relevant research</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrating databases [8,20,24]</td>
<td>• Resolve semantic heterogeneity</td>
</tr>
<tr>
<td></td>
<td>• Present a model that allows for uniform specification of database requests and for an application program</td>
</tr>
<tr>
<td>Integrating semantics into databases [23,32]</td>
<td>• Automatically implement the semantics of database updates</td>
</tr>
<tr>
<td>Architecture of data warehouses [6,7,25]</td>
<td>• Support the semantics and the object-oriented modeling of software environments</td>
</tr>
<tr>
<td></td>
<td>• Define the architecture of data warehouses</td>
</tr>
<tr>
<td></td>
<td>• Use O–O technology to consider the model of data warehouses</td>
</tr>
<tr>
<td></td>
<td>• Discuss access methods in data warehouse systems</td>
</tr>
<tr>
<td>Object-oriented analysis [2,17,19,26]</td>
<td>• Define a core conceptual OO database model</td>
</tr>
<tr>
<td></td>
<td>• OO design with application</td>
</tr>
<tr>
<td></td>
<td>• OO modeling and design</td>
</tr>
<tr>
<td></td>
<td>• OO software engineering</td>
</tr>
<tr>
<td>Data cube operations [10,12,13,16,21,28,30]</td>
<td>• Use greedy algorithms to implement an optimal data cube model</td>
</tr>
<tr>
<td></td>
<td>• Use greedy algorithms to automate the selection of summary tables and indexes</td>
</tr>
<tr>
<td></td>
<td>• Propose a greedy method to select material views and, in turn, to recompute the size of the stored views</td>
</tr>
<tr>
<td></td>
<td>• Use SQL statements to implement data cubes and to provide several functions of data cube operations in data warehouses</td>
</tr>
<tr>
<td>Applications of data warehouse systems [4,27,34]</td>
<td>• Implement effective commerce-strategy analyses of data warehouse projects</td>
</tr>
<tr>
<td></td>
<td>• Develop applications to integrate the data warehouses and DSS into the real environment</td>
</tr>
<tr>
<td></td>
<td>• Apply a multi-agent approach to brain modeling and neurofuzzy control in the data warehouse environment</td>
</tr>
</tbody>
</table>
3. The semantic cube model

This section presents a novel semantic cube model that serves data warehouses. This research based the model on OO methodology. The model enables users to create, between different cubes, a number of abstract-relationship types such as generalization, aggregation, and categorization. Fig. 3 below shows the syntax of the semantic cube model and describes the declaration of the semantic cube model. Regarding this model, this research has to define several meanings attributable to data cubes. First is the dimension declaration; second is the measure declaration. Level and constraint are the optional definitions in the end. Fig. 4 depicts the various notations of the semantic cube model.

In the developmental stage, this research integrates into a data warehouse some of the object-oriented characteristics (such as the “Generalization,” the “Aggregation,” and the “Categorization”) of data cubes. This research explains each characteristic of the object-oriented cube model as follows:

3.1. Data cube with generalization semantics

Generalization: It defines a new class (also called subclass) that is inherited by other classes (also called superclass), and the subclass may have all kinds of properties identical to those of the superclass. The relationship between subclass and superclass is called generalization; we implement this new data cube with the generalization of the semantic cube model.

For example, there is a car-sales database, and the user created a data cube named ‘Car’. The other user created a data cube named ‘Electronic_Car’. In this case, if the user wants to create another new data cube and wants to relate to the data cube of ‘car’, the new data cube will duplicate the original data. Hence, we start to consider a solution to solve the problem that was called the “Semantic Cube Model”. We inherit those data from the ‘Car’ cube directly, and this step not only saves table-calculating time but also frees up storage capacity. This research derived the definition of generalization semantics from the OO concept, and the definition of generalization semantics is as below, in Definition 1:

**Definition 1 (Define data cube with generalization).** In order to implement a more flexible building methodology for data cubes in the object-based data warehouses, this research uses a generalization of OO technology in the semantic cube model and makes the data cubes that exhibit object-oriented behavior in the

<table>
<thead>
<tr>
<th>Semantic Cube Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUBE new_cube_name [ UNMATCH</td>
</tr>
<tr>
<td>[ DIMENSIONS</td>
</tr>
<tr>
<td>{ new_dimension_name</td>
</tr>
<tr>
<td>[SYSTEM_TYPE</td>
</tr>
<tr>
<td>] MEASURE</td>
</tr>
<tr>
<td>{ (aggregation function, measure aggregate) }</td>
</tr>
<tr>
<td>[ LEVEL</td>
</tr>
<tr>
<td>{ dimension_name (new_dimension_name1, new_dimension_name2…) }</td>
</tr>
<tr>
<td>[ Constraint</td>
</tr>
<tr>
<td>{ [Condition equation] }</td>
</tr>
<tr>
<td>]</td>
</tr>
</tbody>
</table>

Fig. 3. The syntax of the semantic cube model.
model. Through “inherit” technology, a designer can use all the superclass characteristics in the subclass. Therefore, this research could use the generalization in relation to the data cube. The type of generalization is illustrated as “IS_A” in Fig. 3.

According to this definition, this research could use both the generalization function and the semantic cube model to implement the ‘Car’ data cube and the ‘Electronic_Car’ data cube, as Fig. 5a and b illustrate.

Electronic_Car will inherit all relative data from car with the constraint “Car.kind = ‘Electronic Engine’”. The fact data of ‘Electronic_Car’ would be as it is depicted in Fig. 5c. And Fig. 5d represents a possible relationship with ‘Car’. In short, a generalization is an abstraction that enables a class of individual objects to be thought of generically as a single-name object. Generalization is perhaps the most important mechanism we have for conceptualizing the real world, and this mechanism helps us to reuse objects and to reduce duplicates in a system.

3.2. Data cube with aggregation semantics

Aggregation is another abstractive semantic of the object-oriented methodology. It describes a relationship between objects. For example, a car object could be aggregated by an engine object and a body object, since a

<table>
<thead>
<tr>
<th>Symbol (syntax)</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>[]</td>
<td>The context between braces will occur zero times or once</td>
</tr>
<tr>
<td>()</td>
<td>The context between parentheses represents once or more than once</td>
</tr>
<tr>
<td></td>
<td>The vertical line expresses the ‘or’ operation</td>
</tr>
<tr>
<td>IS_A(Cube)</td>
<td>It defines the new cube that is inherited by ‘Cube’, and it could be one cube in the ‘Cube’; this research used ‘,’ to partition each cube</td>
</tr>
<tr>
<td>A_PART_OF(cube_name)</td>
<td>It expresses the relationship between attributes and “Cube”; attribute is one of a cube’s dimensions</td>
</tr>
<tr>
<td>A_MEMBER_OF(cube_name)</td>
<td>It defines the attribute categorized by ‘cube_names’ and must comprise more than one cube in the ‘cube_names’, this research used ‘,’ to partition each cube</td>
</tr>
<tr>
<td>UNMATCH</td>
<td>It defines what information users not want to calculate in the data warehouse of the data cube</td>
</tr>
<tr>
<td>SYSTEM_TYPE</td>
<td>The system provides a default-data type such as integer and character</td>
</tr>
<tr>
<td>MEASURE</td>
<td>It defines what information users want to measure in the data warehouse of the data cube</td>
</tr>
<tr>
<td>Level</td>
<td>Defines the hierarchy levels of dimensions and writes upper level in front of parentheses; the lower level would be between two parentheses</td>
</tr>
<tr>
<td>Constraint</td>
<td>The function shows the limitation between cubes with a condition equation; the condition equation is a conditional (Boolean) expression that indicates the data cubes to be retrieved by the query</td>
</tr>
</tbody>
</table>

Fig. 4. The notation of the syntax of the semantic cube model.
car is composed of only an engine and a body. The car can be regarded as a higher-level object. In making such an abstraction, one may ignore many details of the relationship. In order to formalize the aggregation operation, this research used Definition 2 to explain it more clearly.

**Definition 2** (Define data cube with aggregation). Generalization represents a vertical relation and also a horizontal relation that is called aggregation. So we use the semantic cube model of Fig. 3 to represent aggregation. In Fig. 6a, it is easy to explain that the engine and the body integrate into the car. Maybe users would find it very difficult to discover the relationship between the subcubes if they wanted to know something about the car. So, this research would just create a cube having the name ‘Car’ and would increase some information about the car cube such as the car-maintenance costs of the engine and the body. And this research defines the car cube according to aggregation semantics. The semantics of this example are illustrated in Fig. 6b.

This research expresses this situation in a special way. In this study’s OO system, aggregation permitted a class (C) that included other subclasses (C₁, C₂, C₃,..., Cₙ), and this research used the “A_PART_OF” to represent the operation of aggregation. According to this principle, this research could define the subclasses (C₁, C₂, C₃,..., Cₙ) as parts of class (C). Hence, it could be represented by
In Fig. 6a, the acquisition of more detailed information about ‘Car’ would require a merging of both data cube ‘Engine’ and data cube ‘Body’, because engine and body constitute a car. Therefore, to discover more information about ‘Car’ with its engine and its body, one could refer to the following formula:

\[ C_{\text{car}} \supseteq (C_{\text{engine}} + C_{\text{body}}) \]

Through the aggregation technique, this research easily present a cube combined with the other cube. Hence, this research could use the aggregation function to combine other data cubes, and could use these data cubes to make another data cube that represents the object (or subject) in the data warehouse.

3.3. Data cube with categorization semantics

Categorization is another OO characteristic: it brought out subclasses inherited from different classes, and this is the “categorization” relationship in the systems. For example, a whale is one kind of animal; its properties belong to mammals and halobios, so that whales could inherit properties from mammals and halobios. By this method, we could identify the other type of semantic “A_MEMBER_OF” in our research. In Fig. 7, we define the Amphibious_Car cube by categorization semantics. In a literal sense, ‘Amphibious’ refers to water and to road vehicles. And we assume that there are two cubes (car and ship) in the data warehouse; hence, we conclude that this cube is inherited from car and ship.

**Definition 3 (Define Data Cube with Categorization).** It is permitted that an object could use multi-references to build its relationship with semantics. This method allows the object to have more relationships with other
objects. This research could express this matter differently. The OO system permits superclasses \( (P_1, P_2) \) to inherit a subclass \( (C) \). According to this principle, subclass \( C \) has multiple inheritances, because it will substitute for all superclasses. Hence, the new data cube could be represented by a new subclass \( C_{\text{new\_class\_name}} = \text{superclass} (P_1 + P_2 + \ldots + P_n) \). The collective data cube \( P_n \) is based on condition equations. In Fig. 7, we wanted to know the data cube of ‘Amphibious_car’; this research combines only the data cube of ‘Car’ and the data cube of ‘Ship’ because Amphibious_car will have the same properties as do ‘Car’ and ‘Ship’. Therefore, in object-oriented design, it is easy to find a new child with ‘Amphibious_Car’. The formula could be

\[
C_{\text{Amphibious\_car}} = \text{superclass}(P_{\text{car}} + P_{\text{ship}}).
\]

In this case, users could discover the ‘Amphibious_Car’, inherited by ‘Car’ and ‘Ship’. This method would help us to identify a new data cube, but users must define the multi-inherited problem. In this example, ‘Amphibious_Car’ inherited by ‘Car’ and ‘Ship’, but both of them have the same attribute name: ‘Type’. Hence, users need to identify the inheritance rule while the redundancy event occurs. In this case, this research assume that the inherit directory was the data cube ‘Car’, which this research defined in the presence of categorization semantics.

Fig. 7. The example of data cube with categorization semantics.

Fig. 8. The meta-data of SCM.
3.4. The meta-data of SCM

After presenting an overview of the semantic cube model and of data warehousing, this research discusses the meta-data of SCM. Meta-data refers to data about data. It provides important information for data cubes and records the format of our SCM. Fig. 8 concerns the meta-data of SCM, which also follow the star schema concept.

This research introduces the components of SCM meta-data schema here. The meta-data are extended by the star schema, and the star schema falls into two categories. One is the fact table, such as Cube_Table: it records all data cubes in the data warehouse environment. The other is the dimension table, such as Semantic_Cube, Fact_Table, Measurement, Dimension, and Dimension_Data. Each data cube fills the Fact_Table and Measurement with Fact_Data. In the Dimension and Dimension_Data, these tables record all the dimensions of each data cube. The Semantic_Cube has all the relationships of each data cube. The semantic data cube model is composed of these data tables and semantics.

4. Semantic structured query language

After applying the semantic cube model to the data cubes, this research can put semantic data cubes into the data warehouse system. The data warehouse system includes all kinds of information. When users request information from the system, the application program retrieves the information from the storage. Hence, the system must have an interface that presents to the user the result of the user’s request. And users could change their view of the data of the data cube. Each data cube possesses dimension-related information and measure-related information; also, each data cube represents a view of the active fact.

In order to present our semantic cube model in the data warehouse, we develop a Semantic Structured Query Language (SSQL). The Syntax of SSQL is described in Fig. 9. The SSQL is based on MDX (Multi-Dimensional eXtension) syntax and operation. MDX is a syntax that supports the definition and the manipulation of multidimensional schema. MDX is similar to the Structured Query Language (SQL) syntax, but MDX is not an extension of the SQL language. MDX uses multi-dimensional logic to present the data of schema [29]. And this research added these MDX concepts to our semantic query function. Therefore, each data cube possesses its properties, and this model could make any query and could conduct any analysis with this SSQL. The form of the basic query of a data warehouse system is outlined below:

(a) Roll-up operator decreases the detail of the measure by aggregating it along the level of dimension. Fig. 10a illustrates the SCM query syntax with SSQL.
(b) Drill-down operator will aggregate the data cube that is joined with other data cube that has more detailed information than original data cube. This operator increases the more detail of the measure in the data cube. Fig. 10b illustrates the SSQL syntax of query.
(c) Slice operator refers to the special dimension information in the data cube. This operator enables data cubes to cut one dimension for the analysis of a data cube’s information. Fig. 10c illustrates the query of SSQL.

[SELECT [DIM_ACTION (OBJECT) | CUBE_ACTION (Cube)] ON X-AXIS [DIM_ACTION (OBJECT) | CUBE_ACTION (Cube)] ON Y-AXIS
DIM_ACTION = [ROLLUP | DRILLDOWN | SLICE | DICE]
CUBE_ACTION = [ZOOMIN | ZOOMOUT | EXPAND | NARROWDOWN | FOCUSON]
OBJECT = [ DIMENSION | DIMENSION.Data ]
FROM CUBE
[WHERE condition expression]]

Fig. 9. Semantic structured query language.
Fig. 10. The examples of SCM query syntax with SSQL.

<table>
<thead>
<tr>
<th>Expression</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{SELECT ROLLUP (TIME.month) ON X-AXIS} )</td>
<td>(a) Roll-up expression of SSQL.</td>
</tr>
<tr>
<td>( \text{FROM SALES} )</td>
<td></td>
</tr>
<tr>
<td>( \text{WHERE TIME.Year=2005} )</td>
<td></td>
</tr>
<tr>
<td>( \text{SELECT DRILLDOWN (TIME.YEAR.1995) ON X-AXIS} )</td>
<td>(b) The drill-down expression of SSQL.</td>
</tr>
<tr>
<td>( \text{FROM SALES} )</td>
<td></td>
</tr>
<tr>
<td>( \text{SELECT SLICE (KIND) ON X-AXIS} )</td>
<td>(c) The slice expression of SSQL.</td>
</tr>
<tr>
<td>( \text{FROM SALES} )</td>
<td></td>
</tr>
<tr>
<td>( \text{SELECT DICE (TIME.YEAR) ON X-AXIS} )</td>
<td>(d) The dice expression of SSQL.</td>
</tr>
<tr>
<td>( \text{DICE (KIND) ON Y-AXIS} )</td>
<td></td>
</tr>
<tr>
<td>( \text{FROM SALES} )</td>
<td></td>
</tr>
<tr>
<td>( \text{WHERE TIME.YEAR=’1995’} )</td>
<td></td>
</tr>
<tr>
<td>( \text{SELECT ZOOMOUT (SALES) ON X-AXIS} )</td>
<td>(e) The zoom-in expression of SSQL.</td>
</tr>
<tr>
<td>( \text{FROM CUBES} )</td>
<td></td>
</tr>
<tr>
<td>( \text{SELECT ZOOMOUT (SALES) ON X-AXIS} )</td>
<td>(f) The zoom-out expression of SSQL.</td>
</tr>
<tr>
<td>( \text{FROM CUBES} )</td>
<td></td>
</tr>
<tr>
<td>( \text{SELECT EXPAND (ENGINE) ON X-AXIS} )</td>
<td>(g) The EXPAND expression of SSQL.</td>
</tr>
<tr>
<td>( \text{FROM ENGINE} )</td>
<td></td>
</tr>
<tr>
<td>( \text{SELECT NARROWDOWN (AUTOSALESCUBE) ON X-AXIS} )</td>
<td>(h) The narrow-down expression of SSQL.</td>
</tr>
<tr>
<td>( \text{FROM AUTOSALESCUBE} )</td>
<td></td>
</tr>
<tr>
<td>( \text{SELECT FOCUSON (Company) ON X-AXIS} )</td>
<td>(i) The focus-on expression of SSQL.</td>
</tr>
<tr>
<td>( \text{FROM Company} )</td>
<td></td>
</tr>
</tbody>
</table>
(d) Dice operator combines the different dimension information in the data cube. The dice operator also enables the semantic cube model to aggregate more dimensions for the analysis of a data cube’s information. Fig. 10d illustrates the query of SSQL.

(e) For this research, it invents several new queries to extend the original query of the data warehouse system. The new queries are described below:

(f) Zoom-in operator refers to the semantics of inheritance and of categorization in this SCM. If a user used this model to implement—with inherited and categorized semantics—a data cube in a data warehouse system, the system would have the zoom-in function, which would enable the user to browse more details in the data cube. Fig. 10e illustrates the query of SSQL.

(g) Zoom-out operator stands in contrast to the zoom-in operator. The former describes the semantics of inheritance and of categorization in our SCM. A user would not view detailed information, and used the zoom-out function to browse a summary of the data cube. Fig. 10f illustrates the query of SSQL.

(h) Expand operator describes the semantics of aggregation in our SCM. While viewing the relevant data cube information, a user would execute the expand function to browse other data cubes’ data. Fig. 10g illustrates the query of SSQL.

(i) Narrow-down operator stands in contrast to the expand operator. The former operator describes the semantics of aggregation in our SCM. For a user who wants to return the original data cube information to him/herself, the data cube executes the narrow-down function to browse other data cubes’ data. Fig. 10h illustrates the query of SSQL.

(j) Focus-on operator concerns the dimension to a data cube that is in the upper level. This operator also describes OO semantics in our SCM. A user who wants to view the relevant data cube information would execute the focus-on function to browse other data cubes’ dimensions. Fig. 10i illustrates the query of SSQL.

5. System architecture

We have developed a system architecture in order to accomplish two goals: first, we want to decrease the duplication of data in the data warehouse; second, we want to improve the performance of query integrity. By enhancing the data warehouse, this research could design a physical system on the basis of the semantic cube model. In this research, the data warehouse may have multi-level problems: for example, customer ‘A’ might request a data cube that has a product P in a special time period. But if customer ‘B’ wants to analyze more detailed information about the product P (such as category-related information), ‘B’ needs to implement another data cube and the duplicate will occur. Hence, semantic cube model will solve this problem, improve system performance, and reduce data records.

And because the developed useful data warehouse system should be able to merge tightly with the Internet, this research used three-tier architecture to design the system. This architecture comprises client, web server, and application server and is illustrated in Fig. 11.
In the following paragraph, we illustrate the operations between server and client:

1. The operation of the server
   The architecture of the server consists of two parts: (1) the web server and (2) the application server. The application server operates four components: the database, the semantic cube model generator, the meta-data system, and the data warehouse system. The web server is divided into two parts: a web server (i.e., IIS or PWS) and our data cube browser.
   The DW designer collects all kinds of data from heterogeneous databases and then extracts the useful data to make the data cube. The designer may generate a data cube and add semantics (i.e., SCM) to both the database and the data warehouse system. The meta-data system will record the data-cube information, and the data warehouse will store the data cube’s row of data. The function of the data cube browser is to display all information about each data cube, and the web server (i.e., IIS or PWS) will receive requests from the client.
2. The user operation at client
After accomplishing the SCM implementation, a user may retrieve a data cube through a web server by using the data cube browser. Users may use any kind of platform to connect to the web server through the Internet. The web server will wait for the client’s request to browse data cubes from the data cube browser.

6. System implementation

This research built a physical system that verifies its hypothesis. This research built the system by using Visual Basic, an MS-SQL server, and IBM Visual Age. Fig. 3 depicts the metadata of SCM’s, which are built into the MS-SQL server. The interface of the SCM is shown in Fig. 12.

There are several steps in the establishment of the semantic cube model. First, SCM needs to define the new cube’s information such as cube name, cube owner, cube date, descript, relationship between the data cubes with OO semantics, and constraints. Then the system defines how many dimensions will appear in the new data cube. The Step1 of Fig. 12 illustrates the setup operation, and The Step2 of Fig. 12 defines the property of each dimension. The Step3 of Fig. 12 defines the measure of the new data cube, and the Step4 of Fig. 12 defines the level of each dimension. After these setup operations, the semantic cube model can start to build—on the basis of these settings—the new data cube.

For the user who accomplishes these five steps, the semantic data cube would automatically be added to the data warehouse. The semantics not only provide a flexible solution but also reduce system resources.

After the system integrated the data cube into the data warehouse, the client user could query the data cube by using the front–end interface, a scenario that Fig. 13 illustrates. The browser of SCM also allows for many kinds of queries in it, such as roll-up, drill-down, slice, dice, and zoom-in, which are described in Section 4.

7. System evaluation: a case study of a clothing Company

Billy Kids Company produces many kinds of leisure clothes. It was established in the British Virgin Islands, IBS in 2001 and distributes popular clothing in Taiwan. The company produces many kinds of clothing such
as T-shirts and blouses. The shops are located in the central and the southern regions of Taiwan. This research could create a sales data cube in our data warehouse system. Fig. 14 illustrates this query interface in the browser.

If the head of the case study company wanted to analyze the southern region, the DW designer would need to use only our SCM to generate the related data cube. Fig. 12 illustrates the interface of the SCM system. After five steps, the south data cube would be implemented in the data warehouse system. Then, the client could use the browser to query the south-sales information on the web.

In this case, this research used 12,288 database records as our source data (2 years * 12 months * 2 areas * 8 branch stores * 2 kinds * 16 products) and built two different data cubes. One is called sales, and the records of that cube were huge. The data cube possessed 53,703 data records (3 * 13 * 9 * 3 * 17). Then, this research built a second cube for the southern region, but we did not generate any record. Because the southern_region’s record set was included in sales, we used SCM to build a new data cube. But if we used another application program such as an MS-OLAP server to realize the same objective, we would increase the data warehouse records by 16,848 (3 * 13 * 9 * 3 * 16). This finding stems from the fact that the data cube lacked a semantic function. In the end, this research evaluates the difference between SCM and the MS-OLAP server, and Fig. 15 outlines this evaluation.

### 8. Concluding remarks and future work

In this paper, we have described and discussed a theoretically sound approach to advance the development of semantics in the data warehouse design. In the method, the semantic cube model (SCM) is created to present an innovative approach to generate an active and agile data warehouse system. The semantics of generalization, aggregation and categorization are modeled in a systematic manner. A set of experiments have been conducted in a prototype system to illustrate the feasibility and validity of the method. The prototype system was built to implement the method for the experiment.

Fig. 15 illustrates a quantitative and comparative study of the system performance between Microsoft OLAP server and the SCM approach. Results show the SCM approach could reduce the storage space by more than 3500 records. Fig. 16 illustrates a functional test of multi-dimensional operations between MS

<table>
<thead>
<tr>
<th>Source Data</th>
<th>12,288 records=2 years * 12 months * 2 areas * 8 branch stores * 2 kinds * 16 products</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sales data cube</td>
<td>53,703 records = 3<em>13</em>3<em>9</em>3*17</td>
</tr>
<tr>
<td>Semantic Cube Model</td>
<td>MS-OLAP Server</td>
</tr>
<tr>
<td><strong>The records of a new data cube for the southern region</strong></td>
<td><strong>NONE</strong></td>
</tr>
</tbody>
</table>

Fig. 15. The comparison of SCM with MS-OLAP server.

Fig. 16. Comparison of query functionality.
OLAP server, Gray 1997, and SCM. Results show our method could produce better and larger scalability across the different datasets. SCM gives the multi-query, memory-resident, atomic capability in the experiment. Both storage requirements and system performance are improved in an order of magnitude in the tests.

In the near future, we will continue to extend and expand on the SCM model. We will integrate the active rule mechanism into the model formulation and query processing. Inference engine works with the semantic model will provide a high-end active data warehouse system. We will test the method in a heavier run environment and with more sophisticated data cubes. A user-friendly GUI will be introduced in the browser functionality. We will test the method further with the industry standard such as TPC benchmarks on the query optimization and storage efficiency.

References