Map Cube: A Visualization Tool for Spatial Data Warehouses

S. Shekhar, C. T. Lu, X. Tan, S. Chawla
Computer Science Department, University of Minnesota
200 Union Street SE, Minneapolis, MN-55455
[*shekhar, ctlu, xtan, chawla, vatsavai*]@cs.umn.edu TEL:(612) 6248307 FAX:(612)6250572
http://www.cs.umn.edu/Research/shashi-group

January 17, 2002

Abstract

Data warehouses(DW) are becoming essential tools in decision making and data analysis. A data cube operator is used to generate the union of a set of alpha-numeric summary tables corresponding to a given aggregation hierarchy. Spatial data warehouses prefer browsing aggregated data in terms of albums of maps rather than alpha-numeric summary tables. It is quite tedious to convert the output of a data cube operator to an album of maps using current tools. We extend the concept of a data cube to the spatial domain by proposing "map cube," an operator which takes a base map, associated data tables, cartographic preferences, and generates an album of maps. A map cube organizes the album of generated maps using the given aggregation hierarchy to support browsing via roll-up, drill-down, and other operators on aggregation hierarchy. We use 1990 census data to illustrate the notion of map cube, and discuss research issues raised by the map cube operator.

Keywords: Map Cube, Data Cube, Spatial Data Warehouse, Maps, GIS

1 Introduction

A data warehouse(DW) (Inm93; IH94; Wid95; CD97; IWG97; KRRT98) is a repository of subject-oriented, integrated, and non-volatile information, aimed at supporting knowledge workers(executives, managers, analysts) to make better and faster decisions. Data warehouses contain large amounts of information, which is collected from a variety of independent sources and is often maintained separately from the operational databases. Traditionally, operational databases are optimized for on-line transaction processing (OLTP), where consistency and recoverability are critical. Transactions are typically small and access a small number of individual records based on the primary key. Operational databases maintain current state information. In contrast, data warehouses maintain historical, summarized, and consolidated information, and are designed for on-line analytical processing (OLAP) (CCS93; Cod95). The data in the warehouse are often modeled as a multidimensional space to facilitate the query engines for OLAP, where queries typically aggregate data across many dimensions in order to detect trends and anomalies (MQM97). There is a set of numeric measures that are the subjects of analysis in a multidimensional data model. Each of the numeric measures is determined by a set of dimensions. In a census data warehouse, for example, the measure is population, and the dimensions of interest are age group, ethnicity, income type, time (year), and location(census tract). Given N dimensions, the measures can be aggregated in 2^N different ways, The SQL aggregate functions and the group-by operator only produce one out of 2^N aggregates at a time. A data cube (GBLP95) is an aggregate operator which computes all 2^N aggregates in one shot.

Spatial data warehouses contain geographic data, e.g., satellite images, aerial photography (HSK98; Mic00), in addition to non-spatial data. Examples of spatial data-warehouses include the US Census data-set (Fer99; USC00a), Earth Observation System archives of satellite imagery (USG98), Sequoia 2000 (SFD93), and highway traffic measurement archives. The research in spatial data warehouses has focused on case-studies (ESR98; Mic00) and on the per-dimension concept hierarchy (HSK98). A major difference between conventional and spatial data warehouses lies in the visualization of the results. Conventional data warehouse OLAP results are often shown as summary tables or spread sheets of text and numbers, whereas in the case of spatial data warehouse the results may be albums of maps. It is not trivial to convert the alpha-numeric output of a data cube on spatial data warehouses into an organized collection of maps. Another issue concerns the aggregate operators on geometry data types(e.g., point, line, polygon). Neither existing databases nor the emerging standard for geographic data, OGIS (OGI99), has addressed this issue. In this paper we present the *map cube*, an operator based on the conventional data cube but extended for spatial data warehouses. With the *map cube* operator, we visualize the data cube in spatial domain via an album of maps. For some spatial applications, e.g., census data, which require aggregation on different dimensions and comparison between different categories in each attribute, the map cube operator can be very useful.

A map cube is an operator which takes a base map, associated data tables, aggregation hierarchy, and cartographic preferences to produce an album of maps. This album of maps is organized using the given aggregation hierarchy. The goal is to support exploration of the map collection via roll-up, drill-down, and other operations on the aggregation hierarchy. We also provide a set of aggregate operators for geometric data types and classify them using a well-known classification scheme for aggregate functions. Existing mapping tools, e.g., Quick

Thematic Maps in American FactFinder (USC00b), allow users to view geographic patterns for the areas of interest. For instance, a map could show the average household income for Minneapolis by census tract. The map might display tracts with a high household income in shades of dark blue, and tracts with a low household income in shades of lighter blue. In contrast, the proposed map cube operator generates a set of maps for different categories in the chosen dimensions, thus providing an efficient tool for pattern analysis and cross-dimension comparison.

The rest of the paper is organized as follows. Section 2 discusses some basic concepts of Data Warehouses and Geographic Information System. In section 3, the definition and operation of the map cube are introduced with an example. Section 4 demonstrates an application of the map cube. The research issues related to the map cube are discussed in Section 5. Finally, Section 6 includes a summary and a discussion of future research directions.

2 Basic Concepts

2.1 An Example of Geographic Information System

A *geographic information system (GIS)* (Wor95; Chr96) is a computer-based information system consisting of an integrated set of programs and procedures which enable the capture, modeling, manipulation, retrieval, analysis and presentation of geographically referenced data. We would define common concepts in GIS via the entities in Figure 1. The purpose is provide a context for map cube and to show how it may relate to other concepts in GIS. Figure 1 is not designed to capture all the concepts in all popular GIS. A map is often organized as a collection of vector and raster layers as shown in Figure 1. Each map has its own visual representation, including graphics, layout, legend, and title.

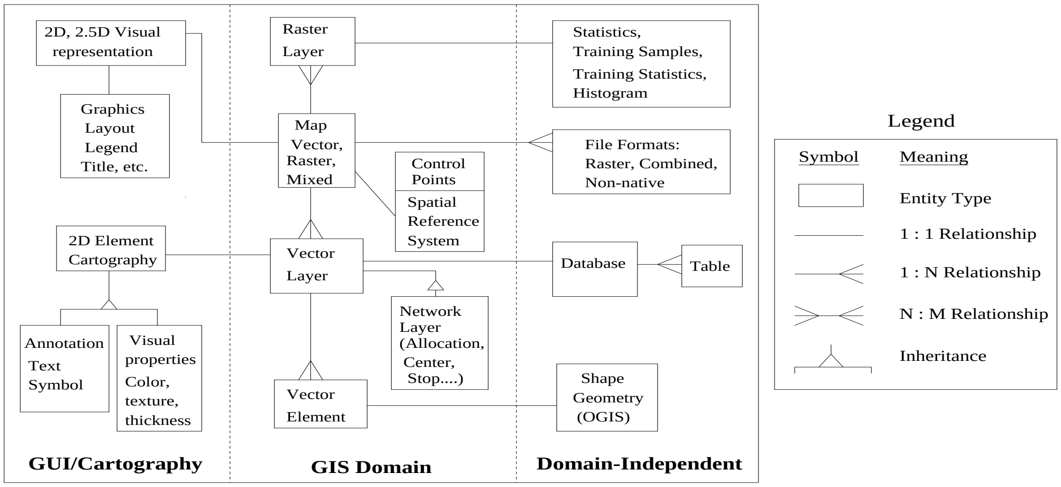


Figure 1: Concepts in Geographic Information Systems

In a raster layer representation, the space is commonly divided into rectangular grid cells, often called pixels. The locations of a geographic object or conditions can be defined by the row and column of the pixels they occupy. The area that each pixel represents may characterize the spatial resolution available. The raster layer is a collection of pixels and may represent raw images collected directly from satellites, aerial photography, etc. The raster layer may also represent interpreted images showing a classification of areas.

Information associated with the raster layer representation includes statistics, training samples, training statistics, and histograms for the classified images, among other things. Mean, standard deviation, variance, minimum value, maximum value, variance-covariance matrix, and correlation matrix are some examples of statistical information. Training samples and training statistics are used by supervised classification, which is performed when the analyst have the knowledge about the scene, or have identity and location information about the land cover types, e.g. forest, agriculture crops, water, urban, etc. Training samples are collected through a combination of fieldwork, existing maps, or interpretation of high resolution imagery. Base on these training samples, multivariate training statistics are computed from each band for each class of interest. Each pixel is then classified into one of the training classes according to classification decision rules(e.g. minimum distance to means, maximum likelihood etc.).

Vector layers are collections of vector elements. The shape of a vector element may be zero dimensional(point), one dimensional(curves, lines) or two dimensional(surface, polygons). For example, in a vector layer representation, an object type *house* may have attributes referencing further object types: polygon, person name, address, and date. The polygon for the house is stored as a "Vector Element" in Figure 1. A vector layer may have its own cartographic preferences to display the elements. These cartographic preferences include text, symbol, and some visual properties such as color, texture, and thickness.

The elements and attributes in the vector layers may be associated with non-spatial attributes managed by a database system which consists of many tables. In Figure 4, for example, the vector layer Base-Map has its corresponding table, Election-Base, which has the attributes of State Name, GPP, LPP, Boundary, and Delegates. The Boundary is a foreign key pointing to another table which describes the geometric boundary of each polygon.

A network layer is a special case of a vector layer in Figure 1. It is composed of a finite collection of points, the line-segments connecting the points, the location of the points, and the attributes of the points and line-segments. For example, a network layer for transportation applications may store road intersection points and the road segments connecting the intersections.

Maps are also associated with reference systems and control points. A spatial reference system is a coordinate system attached to the surface of the earth which allows users to locate the map element on the surface of the earth. Different map layers can be geo-registered to a common spatial reference system, thus producing a composite overlay for analysis and display. Control points are common to the base map and the slave map being prepared. The exact locations of the control points are well defined. Examples include intersection of roads and railways or other land marks or monuments. The control points are used to geo-register newly acquired maps to the well-defined base map at different scales.

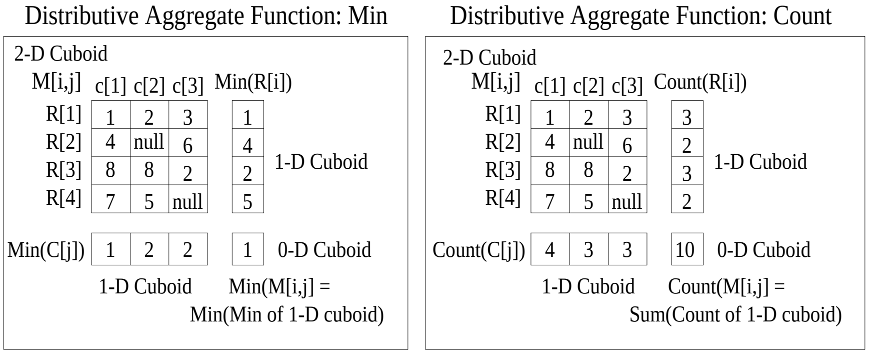
2.2 Aggregate Functions

A data cube consists of a lattice of cuboids, each of which represents a certain level of hierarchy. Aggregate functions compute statistics for a given set of values within each cuboid. Examples of aggregate functions include sum, average, and centroid. Aggregate functions can be grouped into three categories, namely, distributive, algebraic, and holistic (GBLP95). We define these functions in this section and provide some examples from the GIS domain. Table 1 shows all of these aggregation functions for different data types.

	Aggregation Function		
Data Type	Distributive Function	Algebraic Function	Holistic Function
Set of numbers	Count, Min, Max, Sum	Average, Standard Deviation, MaxN, MinN()	Median, MostFrequent, Rank
Set of points, lines, polygons	Minimal Orthogonal Bounding Box, Geometric Union, Geometric Intersection	Centroid, Center of mass, Center of gravity	Nearest neighbor index, Equi-partition

Table 1: Aggregation Operations

- **Distributive**: An aggregate function F is called distributive if there exists a function G such that the value of F for an N-dimensional cuboid can be computed by applying a G function to the value of F in $(N+1)$-dimensional cuboid. For example, when $N = 1$, $F(M_{ij}) = G(F(C_j)) = G(F(R_i))$, where M_{ij} represents the elements of a 2-dimensional matrix, C_j denotes each column of the matrix, and R_i denotes each row of the matrix. Consider the aggregate function Min and Count as shown in Figure 2. In the first example, $F = Min$, then $G = Min$, since $Min(M_{ij}) = Min(Min(C_j)) = Min(Min(R_i))$. In the second example, $F = Count$, $G = Sum$, since $Count(M_{ij}) = Sum(Count(C_j)) = Sum(Count(R_i))$. Other distributive aggregate functions include Max and Sum. Note that "null" valued elements are ignored in computing aggregate functions.



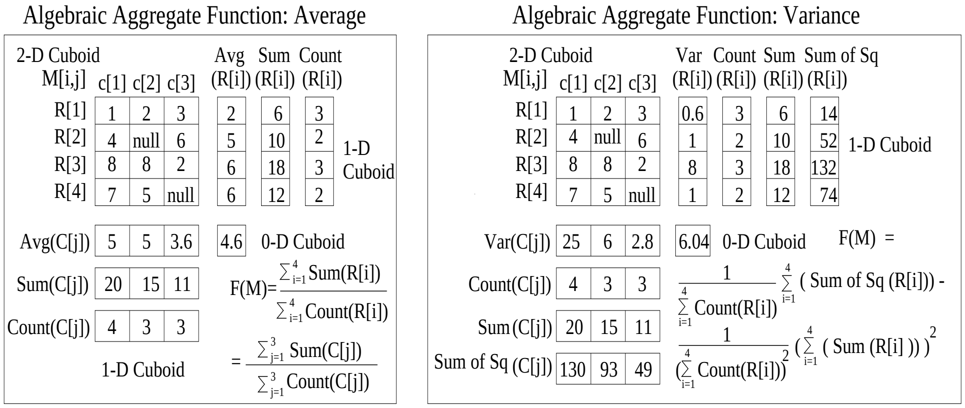
M[i,j]	c[1]	c[2]	c[3]	Min(R[i])
R[1]	1	2	3	1
R[2]	4	null	6	4
R[3]	8	8	2	2
R[4]	7	5	null	5
Min(C[j])	1	2	2	1

M[i,j]	c[1]	c[2]	c[3]	Count(R[i])
R[1]	1	2	3	3
R[2]	4	null	6	2
R[3]	8	8	2	3
R[4]	7	5	null	2
Count(C[j])	4	3	3	10

Figure 2: Computation of distributive aggregate function

Distributive GIS aggregate operations include Minimal Orthogonal Bounding Box, Geometric Union, and Geometric Intersection. The geometric union is a binary operation that takes two sets of geometric areas and return the set of regions that are covered by at least one of the original areas. For all of these aggregations, the operator aggregates the computed regions of the subset, and then computes the final result.

- **Algebraic**: An aggregate function F is algebraic if F of an N-dimensional cuboid can be computed using a fixed number of aggregates of the $(N+1)$-dimensional cuboid. Average, Variance, Standard Deviation, MaxN, MinN are all algebraic. In Figure 3, for example, the computations of Average and Variance for the matrix M are shown. The average of elements in the two dimensional matrix M can be computed from Sum and Count values of the 1-D sub-cubes(e.g., rows or columns). The Variance can be derived from, Count, Sum(i.e. $\sum_i X_i$), and Sum of Sq(i.e. $\sum_i X_i^2$), of rows or columns. Similar techniques apply to other algebraic functions.



Algebraic Aggregate Function: Average
2-D Cuboid
1-D Cuboid
0-D Cuboid
1-D Cuboid
Algebraic Aggregate Function: Variance
2-D Cuboid
1-D Cuboid
0-D Cuboid

M[i,j]	c[1]	c[2]	c[3]	Avg (R[i])	Sum (R[i])	Count (R[i])
R[1]	1	2	3	2	6	3
R[2]	4	null	6	5	10	2
R[3]	8	8	2	6	18	3
R[4]	7	5	null	6	12	2
Avg(C[j])	5	5	3.6	4.6		
Sum(C[j])	20	15	11			
Count(C[j])	4	3	3			

$$F(M)=\frac{\sum_{i=1}^{4}\mathrm{Sum}(R[i])}{\sum_{i=1}^{4}\mathrm{Count}(R[i])}=\frac{\sum_{j=1}^{3}\mathrm{Sum}(C[j])}{\sum_{j=1}^{3}\mathrm{Count}(C[j])}$$

M[i,j]	c[1]	c[2]	c[3]	Var (R[i])	Count (R[i])	Sum (R[i])	Sum of Sq (R[i])
R[1]	1	2	3	0.6	3	6	14
R[2]	4	null	6	1	2	10	52
R[3]	8	8	2	8	3	18	132
R[4]	7	5	null	1	2	12	74
Var(C[j])	25	6	2.8	6.04			
Count(C[j])	4	3	3				
Sum(C[j])	20	15	11				
Sum of Sq (C[j])	130	93	49				

$$F(M) = \frac{1}{\sum_{i=1}^{4}\mathrm{Count}(R[i])}\sum_{i=1}^{4}(\text{Sum of Sq }(R[i])) - \frac{1}{(\sum_{i=1}^{4}\mathrm{Count}(R[i]))^{2}}\left(\sum_{i=1}^{4}(\mathrm{Sum}(R[i]))\right)^{2}$$

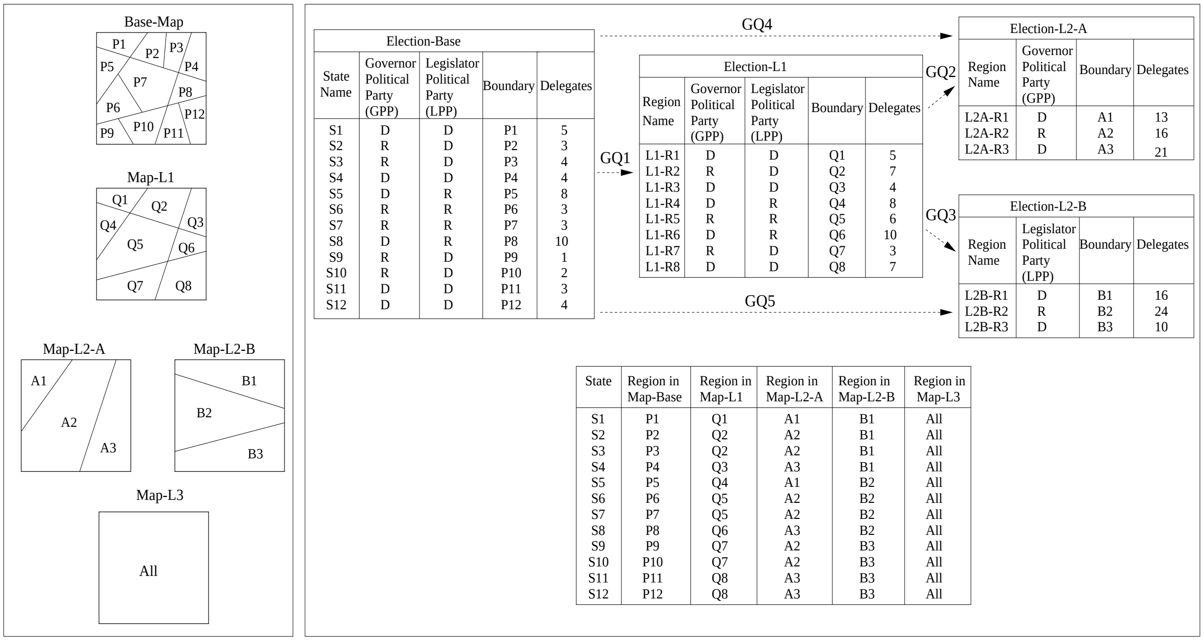
Figure 3: Computation of algebraic aggregate function

An algebraic aggregate operation in GIS is Center. The center of n geometric points $\vec{V}^i = (V_x^i, V_y^i)$ is defined as $\hat{Center} = \frac{1}{n}\sum \vec{V}_i$, $C_x = \frac{\sum V_x}{n}$, $C_y = \frac{\sum V_y}{n}$. Both the center and the count are required to compute the result for the next layer. The center of mass and the center of gravity are other examples of algebraic aggregate functions.

- **Holistic**: An aggregate function F is called holistic if the value of F for an N-dimensional cuboid cannot be computed from a constant number of aggregates of the (N+1)-dimensional cuboid. To compute the value of F at each level, we need to access the base data. Examples of holistic function include Median, MostFrequent, and Rank.

  Holistic GIS aggregate operations include equi-partition and nearest-neighbor index. Equi-partition of a set of points yields a line L such that there are the same number of point objects on each side of L. Nearest-neighbor index measures the degree of clustering of objects in a spatial field. If a spatial field has the property that like values tend to cluster together, then the field exhibits a high nearest-neighbor index. When new data are added, many of the tuples in the nearest neighbor relationship may change. Therefore, the nearest-neighbor index is holistic. The line of Equi-partition could be changed with any new added points. To compute the equi-partition or nearest neighbor-index in all levels of dimensions, we need the base data.

The computation of aggregate functions has graduated difficulty. The distributive function can be computed from the next lower level dimension values. The algebraic function can be computed from a set of aggregates of the next lower level data. The holistic function needs the base data to compute the result in all levels of dimension.



Election-Base

State Name	Governor Political Party (GPP)	Legislator Political Party (LPP)	Boundary	Delegates
S1	D	D	P1	5
S2	R	D	P2	3
S3	R	D	P3	4
S4	D	D	P4	4
S5	D	R	P5	8
S6	R	R	P6	3
S7	R	R	P7	3
S8	D	R	P8	10
S9	R	D	P9	1
S10	R	D	P10	2
S11	D	D	P11	3
S12	D	D	P12	4

Election-L1

Region Name	Governor Political Party (GPP)	Legislator Political Party (LPP)	Boundary	Delegates
L1-R1	D	D	Q1	5
L1-R2	R	D	Q2	7
L1-R3	D	D	Q3	4
L1-R4	D	R	Q4	8
L1-R5	R	R	Q5	6
L1-R6	D	R	Q6	10
L1-R7	R	D	Q7	3
L1-R8	D	D	Q8	7

Election-L2-A

Region Name	Governor Political Party (GPP)	Boundary	Delegates
L2A-R1	D	A1	13
L2A-R2	R	A2	16
L2A-R3	D	A3	21

Election-L2-B

Region Name	Legislator Political Party (LPP)	Boundary	Delegates
L2B-R1	D	B1	16
L2B-R2	R	B2	24
L2B-R3	D	B3	10

State	Region in Map-Base	Region in Map-L1	Region in Map-L2-A	Region in Map-L2-B	Region in Map-L3
S1	P1	Q1	A1	B1	All
S2	P2	Q2	A2	B1	All
S3	P3	Q2	A2	B1	All
S4	P4	Q3	A3	B1	All
S5	P5	Q4	A1	B2	All
S6	P6	Q5	A2	B2	All
S7	P7	Q5	A2	B2	All
S8	P8	Q6	A3	B2	All
S9	P9	Q7	A2	B3	All
S10	P10	Q7	A2	B3	All
S11	P11	Q8	A3	B3	All
S12	P12	Q8	A3	B3	All

Map Interpretation

Computation by Group-By Queries

Figure 4: An example of GIS aggregate function, geometric-union

An example of geometric aggregation

GQ1	**SELECT** GPP, LPP, Geometric-Union-by-Continuous-Polygon(Boundary)
	FROM Election-Base **GROUP BY** GPP, LPP

GQ2	**SELECT** GPP, Geometric-Union-by-Continuous-Polygon(Boundary)
	FROM Election-L1 **GROUP BY** GPP

GQ3	**SELECT** LPP, Geometric-Union-by-Continuous-Polygon(Boundary)
	FROM Election-L1 **GROUP BY** LPP

GQ4	**SELECT** GPP, Geometric-Union-by-Continuous-Polygon(Boundary)
	FROM Election-Base **GROUP BY** GPP

GQ5	**SELECT** LPP, Geometric-Union-by-Continuous-Polygon(Boundary)
	FROM Election-Base **GROUP BY** LPP

Table 2: SQL queries for map reclassification

Figure 4 shows the results from the aggregation operation of geometric union. The base table, Election-Base, has the following attributes: State Name, Governor Political Party(GPP), majority Legislator Political Party(LPP), Boundary, and Delegates. The Boundary is a foreign key pointing to another table which describes the geometric polygon representing the state boundary polygon. From the base table and its corresponding map, we issue the queries GQ1, GQ2, GQ3, GQ4, and GQ5 as listed in Table 2. If the neighboring polygons have the same value in the attributes of the GROUP BY clause, the Geometric-Union-by-Continuous-Polygon operator merges them into one large polygon. These queries have the same effect as the GIS reclassify map operation. For example, the query GQ1 generates the same result as Base-Map reclassified by attributes GPP and LPP. The map interpretation of these queries is shown in the left portion of Figure 4. The boundaries of regions $Q1, Q2, \ldots, Q8$ are derived from the geometric union of smaller regions $P1, P2, \ldots, P12$ in the Base-map. For example, $Q2$ represents the geometric union of $P2$ and $P3$ since they have same value for grouping attribute set <LPP,GPP>. Similarly, regions $A1, A2, A3$ in Map-L2-A and regions $B1, B2, B3$ in Map-L2-B are derived from the geometric-union of smaller regions in Map-L1(or Base-map). For example, region $A1$ in Map-L2-A is the geometric union of region $Q1$ and $Q4$ from Map-L1.

2.3 Aggregation hierarchy

The CUBE operator (GBLP95) generalizes the histogram, cross-tabulation, roll-up, drill-down, and sub-total constructs. It is the N-dimensional generalization of simple aggregate functions. Figure 5 shows the concept for aggregations up to 3-dimensions. The dimensions are Year, Company, and Region. The measure is sales. The 0D data cube is a point which shows the total summary. There are three 1-D data cubes: Group-by Region, Group-by Company, and Group-by Year. The three 2-D data cubes are cross tabs, which are a combination of these three dimensions. The 3D data cube is a cube with three intersecting 2D cross tabs. Figure 6 shows the tabular forms of the total elements in a 3D data cube after a CUBE operation. Creating a data cube requires generating a power set of the aggregation columns.

A tabular view of the individual sub-space data-cubes of Figure 5 is shown in Figure 7. The union of all the tables in Figure 7 yields the resulting table from the data cube operator. The 0-dimensional sub-space cube labeled “Aggregate” in Figure 5 is represented by Table “SALES-L2” in Figure 7. The one-dimensional sub-space cube labeled “By Company” in Figure 5 is represented by Table “SALES-L1-C” in Figure 7. The two-dimensional cube labeled “By Company & Year” is represented by Table “SALES-L0-C” in Figure 7. Readers can establish the correspondence between the remaining sub-space cubes and tables.

The cube operator can be modeled by a family of SQL queries using *GROUP BY* operators and aggregation functions. Each arrow in Figure 7 is represented by a SQL query. In Table 3, we provide the corresponding queries for the five arrows labeled $Q1, Q2, \ldots, Q5$ in Figure 7. For example, query $Q1$ in Table 3 aggregates “Sales” by “Year” and “Region,” and generates Table “SALES-L0-A” in Figure 7.

The *GROUP BY* clause specifies the grouping attributes which should also appear in the *SELECT* clause, so that the value resulting from applying each function to a group of tuples appears along with the value of the grouping attribute(s).

Q1	**SELECT** ’ALL’, Year, Region, **SUM**(Sales) **FROM** SALES-Base **GROUP BY** Year,Region
Q2	**SELECT** ’ALL’, ’ALL’, Region **SUM**(Sales) **FROM** SALES-L0-A **GROUP BY** Region
Q3	**SELECT** ’ALL’, ’ALL’, ’ALL’ **SUM**(Sales) **FROM** SALES-L1-A
Q4	**SELECT** ’ALL’, ’ALL’, Region, **SUM**(Sales) **FROM** SALES-Base **GROUP BY** Region
Q5	**SELECT** ’ALL’, ’ALL’, ’ALL’, **SUM**(Sales) **FROM** SALES-Base

Table 3: Table of **GROUP BY** queries

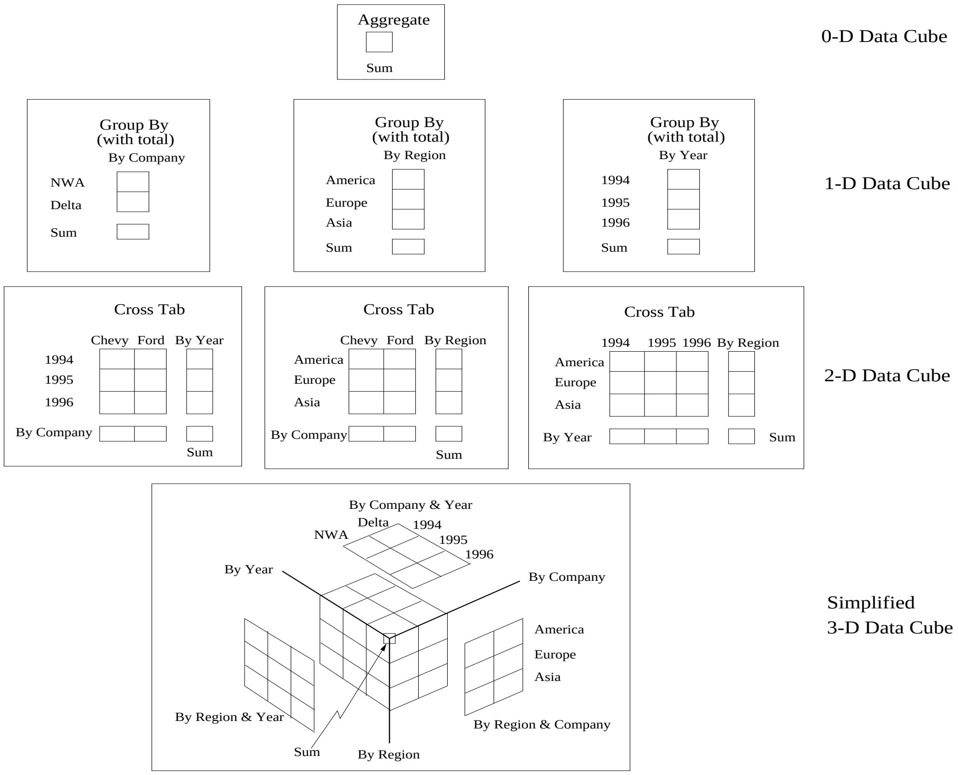


Figure 5: The 0-D, 1-D, 2-D, and 3-D data cubes
(GBLP95)

2.4 What is an Aggregation Hierarchy used for?

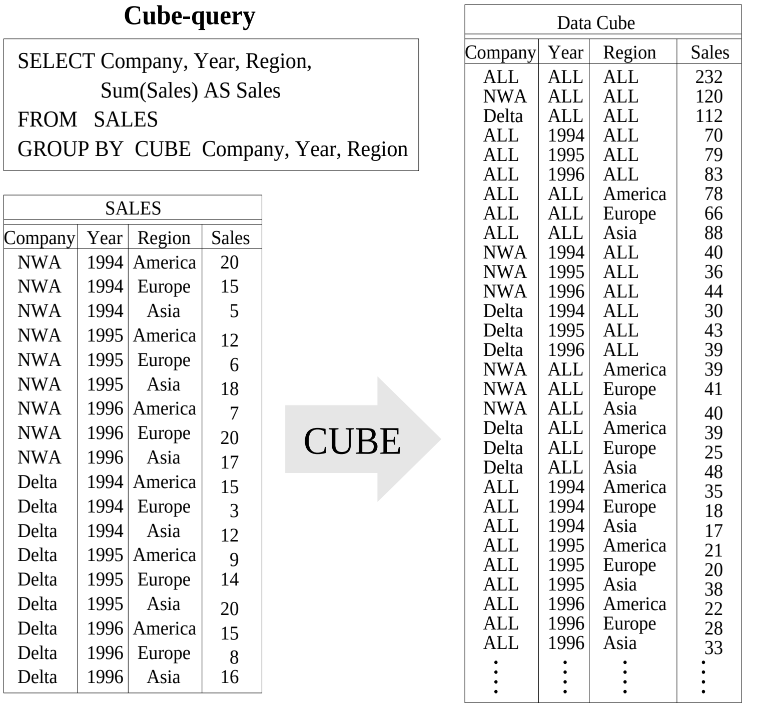
To support OLAP, the data cube provides the following operators : roll-up, drill-down, slice and dice, and pivot. We now define these operators.

- Roll-up: increasing the level of abstraction. This operator generalizes one or more dimensions and aggregates the corresponding measures. For example, Table SALES-L0-A in Figure 7 is the roll-up of Table SALES-Base on the Company dimension.

- Drill-down: decreasing the level of abstraction or increasing detail. It specializes in one or a few dimensions and presents low-level aggregations. For example, Table SALES-L0-A in Figure 7 is the drill-down of Table SALES-L1-A on the Year dimension.

- Slice and Dice: selection and projection. Slicing into one dimension is very much like drilling one level down into that dimension, but the number of entries displayed is limited to that specified in the slice command. A dice operation is like a slice on more than one dimension. On a 2-dimensional display, for example, dicing means slicing on both the row and column dimensions.

  Table 4 shows the result of slicing into the value of "America" on the Year dimension from the Table SALES-L2 in Figure 7.



Cube-query

```
SELECT Company, Year, Region,
          Sum(Sales) AS Sales
FROM   SALES
GROUP BY  CUBE  Company, Year, Region
```

SALES

Company	Year	Region	Sales
NWA	1994	America	20
NWA	1994	Europe	15
NWA	1994	Asia	5
NWA	1995	America	12
NWA	1995	Europe	6
NWA	1995	Asia	18
NWA	1996	America	7
NWA	1996	Europe	20
NWA	1996	Asia	17
Delta	1994	America	15
Delta	1994	Europe	3
Delta	1994	Asia	12
Delta	1995	America	9
Delta	1995	Europe	14
Delta	1995	Asia	20
Delta	1996	America	15
Delta	1996	Europe	8
Delta	1996	Asia	16

Base table

Data Cube

Company	Year	Region	Sales
ALL	ALL	ALL	232
NWA	ALL	ALL	120
Delta	ALL	ALL	112
ALL	1994	ALL	70
ALL	1995	ALL	79
ALL	1996	ALL	83
ALL	ALL	America	78
ALL	ALL	Europe	66
ALL	ALL	Asia	88
NWA	1994	ALL	40
NWA	1995	ALL	36
NWA	1996	ALL	44
Delta	1994	ALL	30
Delta	1995	ALL	43
Delta	1996	ALL	39
NWA	ALL	America	39
NWA	ALL	Europe	41
NWA	ALL	Asia	40
Delta	ALL	America	39
Delta	ALL	Europe	25
Delta	ALL	Asia	48
ALL	1994	America	35
ALL	1994	Europe	18
ALL	1994	Asia	17
ALL	1995	America	21
ALL	1995	Europe	20
ALL	1995	Asia	38
ALL	1996	America	22
ALL	1996	Europe	28
ALL	1996	Asia	33
⋮	⋮	⋮	⋮

Resulting table from Cube operator
(aka data cube)

Figure 6: An example of data cube
(GBLP95)

Company	Year	Region	Sales
ALL	ALL	America	78

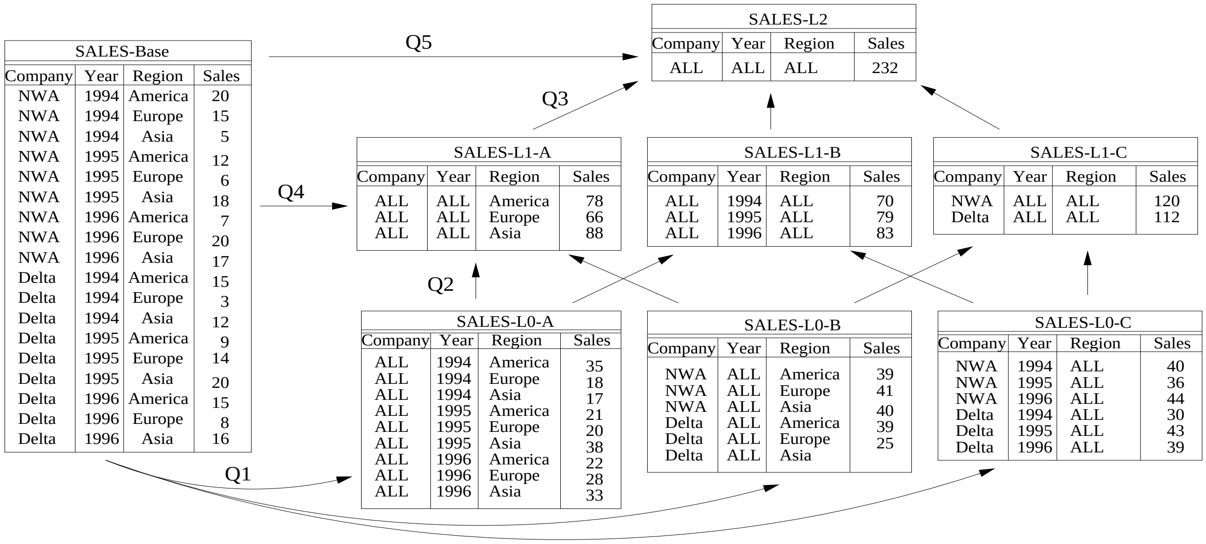
Table 4: Slice on the value America of the Region dimension

Table 5 shows the result of dicing into the value of "1994" on the Year dimension and the value of "America" on the Region dimension from Table SALES-L2 in Figure 7.

- Pivoting: re-orienting the multidimensional view of data. It presents the measures in different cross-tabular layouts

3 Map Cube

In this section, we define the map cube operator, which provides an album of maps to browse results of aggregations. We use a simple example to show the distinction between a data cube and a map cube. We also provide the grammar and the translation rules for the map cube operator.



SALES-Base

Company	Year	Region	Sales
NWA	1994	America	20
NWA	1994	Europe	15
NWA	1994	Asia	5
NWA	1995	America	12
NWA	1995	Europe	6
NWA	1995	Asia	18
NWA	1996	America	7
NWA	1996	Europe	20
NWA	1996	Asia	17
Delta	1994	America	15
Delta	1994	Europe	3
Delta	1994	Asia	12
Delta	1995	America	9
Delta	1995	Europe	14
Delta	1995	Asia	20
Delta	1996	America	15
Delta	1996	Europe	8
Delta	1996	Asia	16

SALES-L2

Company	Year	Region	Sales
ALL	ALL	ALL	232

SALES-L1-A

Company	Year	Region	Sales
ALL	ALL	America	78
ALL	ALL	Europe	66
ALL	ALL	Asia	88

SALES-L1-B

Company	Year	Region	Sales
ALL	1994	ALL	70
ALL	1995	ALL	79
ALL	1996	ALL	83

SALES-L1-C

Company	Year	Region	Sales
NWA	ALL	ALL	120
Delta	ALL	ALL	112

SALES-L0-A

Company	Year	Region	Sales
ALL	1994	America	35
ALL	1994	Europe	18
ALL	1994	Asia	17
ALL	1995	America	21
ALL	1995	Europe	20
ALL	1995	Asia	38
ALL	1996	America	22
ALL	1996	Europe	28
ALL	1996	Asia	33

SALES-L0-B

Company	Year	Region	Sales
NWA	ALL	America	39
NWA	ALL	Europe	41
NWA	ALL	Asia	40
Delta	ALL	America	39
Delta	ALL	Europe	25
Delta	ALL	Asia	

SALES-L0-C

Company	Year	Region	Sales
NWA	1994	ALL	40
NWA	1995	ALL	36
NWA	1996	ALL	44
Delta	1994	ALL	30
Delta	1995	ALL	43
Delta	1996	ALL	39

Figure 7: An example of group-by

Company	Year	Region	Sales
ALL	1994	America	35

Table 5: Dice on value 1994 of Year dimension and value America of Region dimension

3.1 Definition

We extend the concept of data cube to the spatial domain by proposing the "map cube" operator as shown in Figure 8. A map cube is defined as an operator which takes the input parameters, i.e., Base map, Base table, Cartographic preferences, and generates an album of maps for analysis and comparison. It is built from the requirements of a spatial data warehouse, that is, to aggregate data across many dimensions looking for trends or unusual pattern related to spatial attributes. The basis of a map cube is the hierarchy lattice, either a dimension power-set* hierarchy, or a concept hierarchy, or a mixture of both. In this paper, we focus on the dimension power-set hierarchy. Figure 9 shows an example of a dimension power-set hierarchy. This example has three attributes: **Maker(M)**, **Type(T)**, and **Dealer(D)**. There are eight possible groupings of all the attributes. Each node in the lattice corresponds to one group-by. Figure 10 shows a three-dimension concept hierarchy. The car makers can be classified as American, European, or Japanese. American car makers include Ford, GM, and Chrysler. The car type dimension can be classified as Sedan and Pickup Trucks. The Sedan can go down one detail level and be classified as Small, Medium, Large, and Luxury.

Let the number of dimensions be m, each dimension be A_i, i = 1,2, ... ,m, and A_m be the geographic location dimension, then we have (m-1) different levels of lattice for the dimension

*A power-set is a set of all possible subsets. There are 2^N elements in a power-set of a set of N items

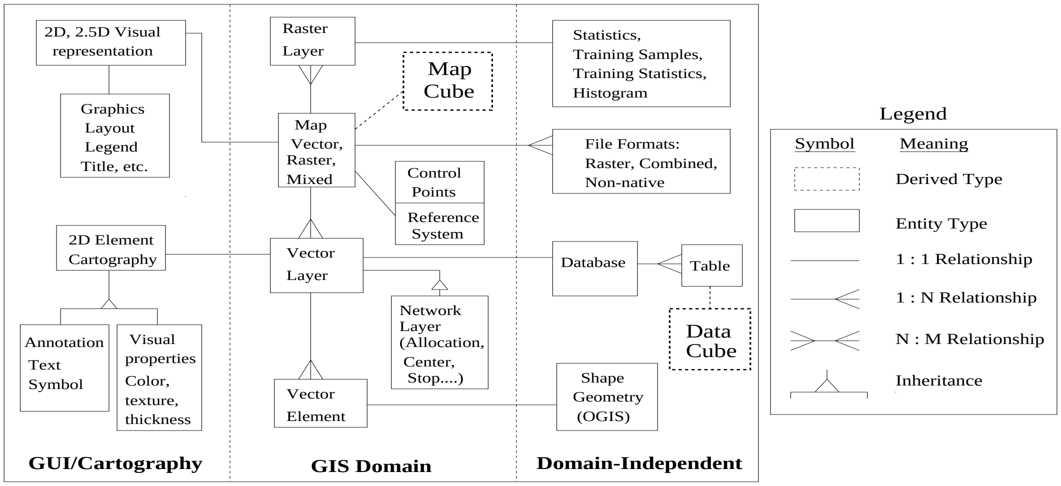


Figure 8: Concepts in GIS with data cube & map-cube

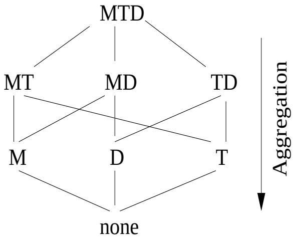


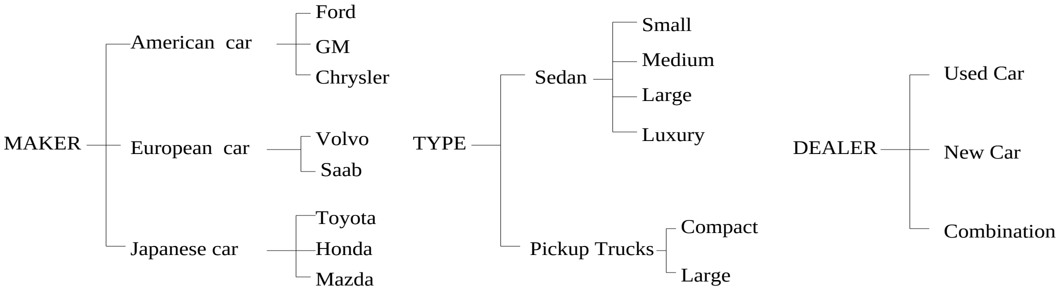
Figure 9: The Dimension Power-Set Hierarchy

power-set hierarchy. Let the level with only one dimension A_i, i = 1,2, ... , m-1, be the 1st level, the level with two dimensions, A_{ij}, where i≠j, i=1,2,...m-1, j=1,2,...m-1, be the 2nd level, and the level with the complete set of dimensions $A_1A_2A_3, \ldots, A_{m-1}$ be the (m-1)th level. The total number of cuboids is $\sum_{i=1}^{m-1}\binom{m-1}{i}$. Let the cardinality of each dimension A_i be c_i, then for each cuboid, such as, $A_iA_j...A_k$, we have an album of $c_i \times c_j \times ... \times c_k$ maps.

In other words, a map cube is a data cube with cartographic visualization of each dimension to generate an album of related maps for a dimension power-set hierarchy or a concept hierarchy or a mixed hierarchy. A map cube adds more capability to traditional GIS where maps are often independent (Bed99). The data-cube capability of roll-up, drill-down, slicing and dicing gets combined with the map view. This can benefit analysis and decision making based on spatial data warehouses.

3.2 An example of a map cube

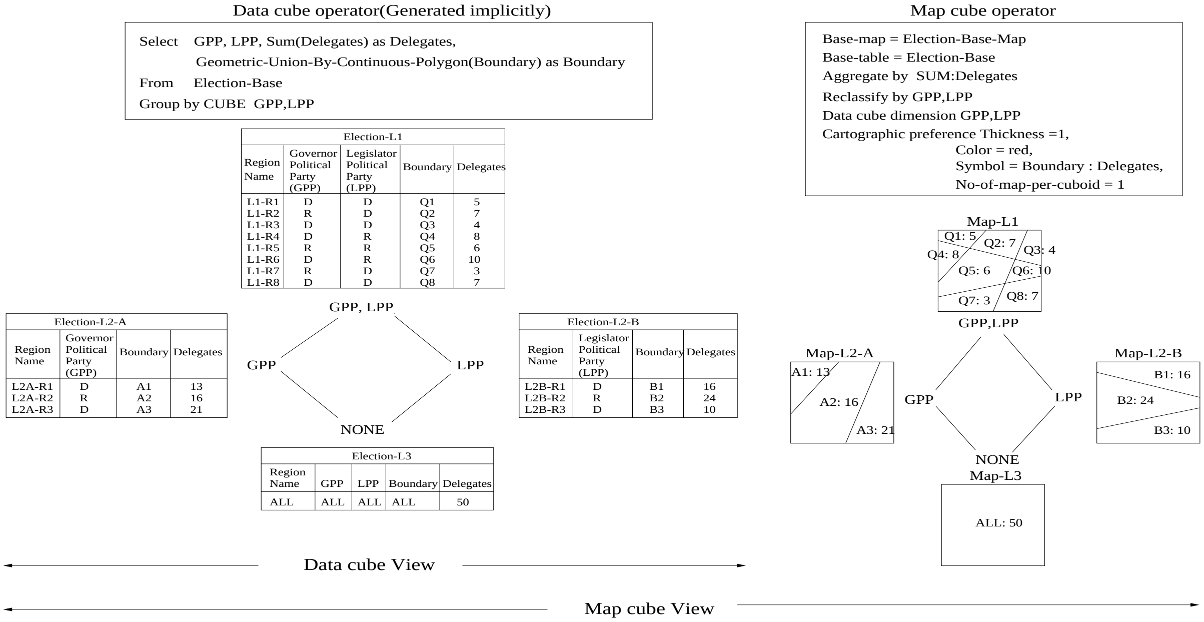
Figure 11 demonstrates the comparison between a map cube and a data cube. The operators for the data cube and the map cube are also provided. The left portion of the figure is the effect of data cube operation. There are 2 attributes in the Group by Cube clause, so the data cube will generate 4 tables. The map cube operator not only generates the tables, but also produces the associated map for each table. In the map cube operator, the attributes for "Reclassify by"



(a) The MAKER hierarchy (b) The TYPE hierarchy (c) The DEALER hierarchy

Figure 10: The Concept Hierarchies of (a)MAKER and (b)VEHICLE (c)DEALER

and "Data cube dimension" are the same, and the parameter, No-of-map-per-cuboid, in the Cartographic preference is set to 1, so there is only one map associated with each reclassified table. In other situations, however, each reclassified table is a cuboid, and will generate more than one map after the map cube operation, depending on the number of attributes in the "Data cube dimension." The queries for the data cube and the map cube are also provided at the top of the figure.



Data cube operator(Generated implicitly)

```
Select   GPP, LPP, Sum(Delegates) as Delegates,
          Geometric-Union-By-Continuous-Polygon(Boundary) as Boundary
From     Election-Base
Group by CUBE  GPP,LPP
```

Map cube operator

```
Base-map = Election-Base-Map
Base-table = Election-Base
Aggregate by  SUM:Delegates
Reclassify by GPP,LPP
Data cube dimension GPP,LPP
Cartographic preference Thickness =1,
                        Color = red,
                        Symbol = Boundary : Delegates,
                        No-of-map-per-cuboid = 1
```

Election-L1

Region Name	Governor Political Party (GPP)	Legislator Political Party (LPP)	Boundary	Delegates
L1-R1	D	D	Q1	5
L1-R2	R	D	Q2	7
L1-R3	D	D	Q3	4
L1-R4	D	R	Q4	8
L1-R5	R	R	Q5	6
L1-R6	D	R	Q6	10
L1-R7	R	D	Q7	3
L1-R8	D	D	Q8	7

Election-L2-A

Region Name	Governor Political Party (GPP)	Boundary	Delegates
L2A-R1	D	A1	13
L2A-R2	R	A2	16
L2A-R3	D	A3	21

Election-L2-B

Region Name	Legislator Political Party (LPP)	Boundary	Delegates
L2B-R1	D	B1	16
L2B-R2	R	B2	24
L2B-R3	D	B3	10

Election-L3

Region Name	GPP	LPP	Boundary	Delegates
ALL	ALL	ALL	ALL	50

Figure 11: Map cube view

3.3 Steps in Generating a Map Cube

To generate the map cube, first we issue the map-cube query, as shown in Figure 12. This query is decomposed into the DB engine and the geometry engine. The relational DB engine processes the 2^n group-by SQL queries and generates 2^n tables, where n is the number of attributes in the

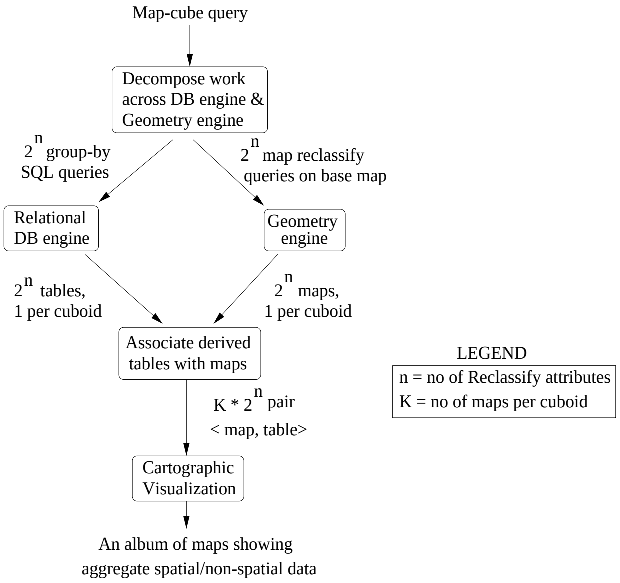

Map-cube query
Decompose work across DB engine & Geometry engine
2^n group-by SQL queries
2^n map reclassify queries on base map
Relational DB engine
Geometry engine
2^n tables, 1 per cuboid
2^n maps, 1 per cuboid
Associate derived tables with maps
K * 2^n pair < map, table>
Cartographic Visualization
An album of maps showing aggregate spatial/non-spatial data
LEGEND
n = no of Reclassify attributes
K = no of maps per cuboid

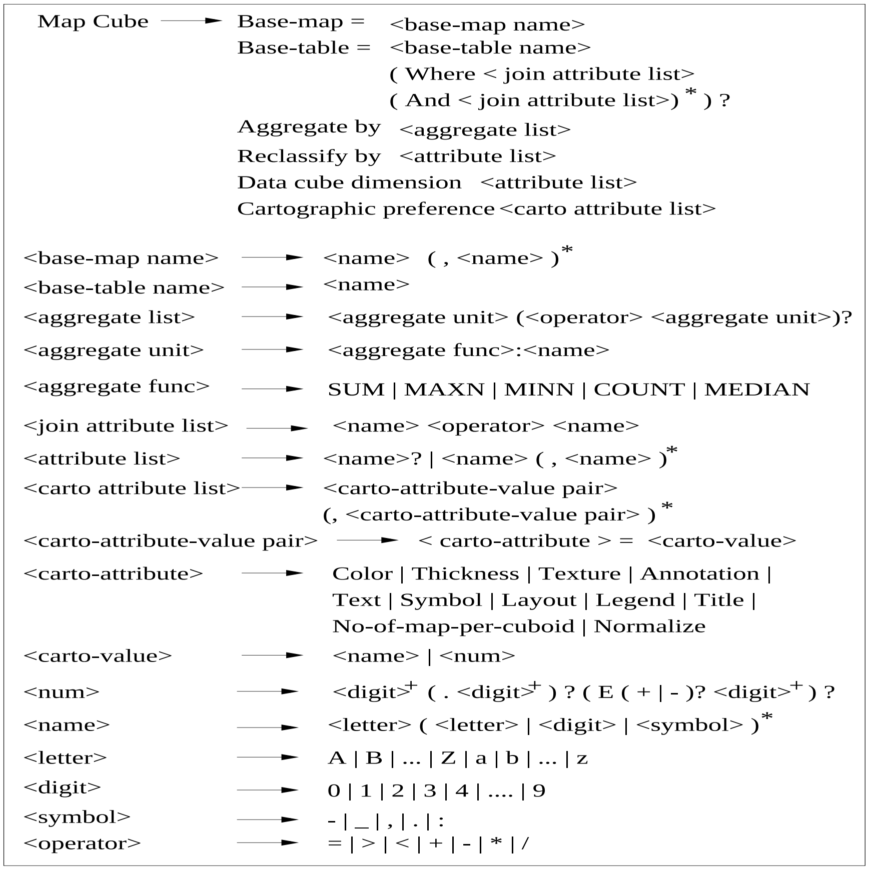
Figure 12: Steps in Generating a Map Cube

"Reclassify by" clause. The geometry engine processes the map reclassify queries on the base map and generates 2^n maps. These maps and tables are in one-one correspondence with each other. Finally, the cartographic preferences are dealt with in the Cartographic Visualization Step, and an album of maps is plot.

3.4 The Grammar for the Map Cube Operator

We have used "yacc" (LMB92) like syntax to describe the syntax of a map cube via the grammar and translation rules, as listed in Figure 13. Words in angular brackets, e.g. <*carto-value*> , denote non-terminating elements of the language. For example, the <*carto-value*> will be translated into either <*name*> or <*num*>. The vertical bar (|) means "or," and the parentheses are used to group subexpressions. The star (*) denotes zero or more instances of the expression, and the plus (+) denotes one or more instances of the expression. The unary postfix operator (?) means zero or one instance of the expression. These translations will continue until we reach a terminating elements. For example, <*letter*> will eventually be translated into one character.

An example of this grammar is the map cube operator used in figure 11. In this map cube operator, the following translations are done: <*base-map name*> into Election-Base-Map, <*base-table name*> into Election-Base, <*aggregate list*> into {SUM:Delegates}, <*attribute list*> of both "Reclassify by" and "Data cube dimension" into {GPP,LPP}, and <*carto attribute list*> into {Thickness=1,Color=red,Symbol=Boundary:Delegates,No-of-map-per-cuboid=1}.



```
Map Cube ──► Base-map =   <base-map name>
             Base-table = <base-table name>
                          ( Where < join attribute list>
                          ( And < join attribute list>) * ) ?
             Aggregate by  <aggregate list>
             Reclassify by  <attribute list>
             Data cube dimension   <attribute list>
             Cartographic preference <carto attribute list>

<base-map name>        ──►  <name>  ( , <name> )*
<base-table name>      ──►  <name>
<aggregate list>       ──►  <aggregate unit> (<operator> <aggregate unit>)?
<aggregate unit>       ──►  <aggregate func>:<name>
<aggregate func>       ──►  SUM | MAXN | MINN | COUNT | MEDIAN
<join attribute list>  ──►  <name> <operator> <name>
<attribute list>       ──►  <name>? | <name> ( , <name> )*
<carto attribute list>──►   <carto-attribute-value pair>
                            (, <carto-attribute-value pair> ) *
<carto-attribute-value pair>  ──►  < carto-attribute > =  <carto-value>
<carto-attribute>      ──►  Color | Thickness | Texture | Annotation |
                            Text | Symbol | Layout | Legend | Title |
                            No-of-map-per-cuboid | Normalize
<carto-value>          ──►  <name> | <num>
<num>                  ──►  <digit>+ ( . <digit>+ ) ? ( E ( + | - )? <digit>+ ) ?
<name>                 ──►  <letter> ( <letter> | <digit> | <symbol> )*
<letter>               ──►  A | B | ... | Z | a | b | ... | z
<digit>                ──►  0 | 1 | 2 | 3 | 4 | .... | 9
<symbol>               ──►  - | _ | , | . | :
<operator>             ──►  = | > | < | + | - | * | /
```

Figure 13: The grammar for the map cube operator

4 A Case Study - Analyzing Census Data

4.1 Application Domain

The 1990 Census is the most detailed tabulation of Americans demographic data. It contains detailed data on population, race and ethnicity, age and sex, education, employment, income, poverty, housing, and many other topics for each of several different levels of geography:

- The United States and major regions of the country
- Each state and metropolitan area
- All 3,000+ counties in the United States
- Municipalities, census tracts, and block groups

In this case study, we summarize data for the seven-county Twin Cities metropolitan area: Hennepin, Ramsey, Anoka, Carver, Dakota, Scott, and Washington Counties. We use census tracts as the unit of analysis, since census tracts provide better geographic detail than city-level data. Census tracts contain an average of 4,000 people. In cities like Minneapolis or Saint Paul, a census tract is often ten-to-twenty city blocks in size. Minneapolis contains 126 census tracts and Saint Paul, 82.

4.2 Map cube definition

4.2.1 Base Table

In this case study, there are four dimensions and one measure. The four dimensions are county location, age group, income type and race category. The measure is population. The location dimension is embedded in the map, and we should consider the other three dimensions in the power-set hierarchy(see Figure 14). We now provide a detailed explanation for each dimension:

- Age Group: There are seven groups – under 25 years, 25 to 34 years, 35 to 44 years, 45 to 54 years, 55 to 64 years, 65 to 74 years, 75 year and over;
- Income type: There are nine types – less than \$5,000, \$5,000 to \$9,999, \$10,000 to \$14,999, \$15,000 to \$24,999, \$25,000 to \$34,999, \$35,000 to \$49,999, \$50,000 to \$74,999, \$75,000 to \$99,999, \$100,000 or more;
- Race category: There are six categories – (white), (black), (American Indian, Eskimo, or Aleut), (Asian or pacific islander), (other);

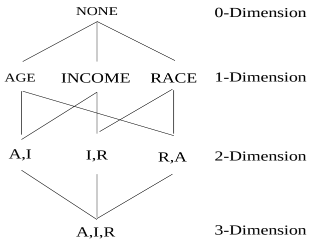


Figure 14: Census data dimension power-set hierarchy

Table 6 and table 7 are two examples of the fundamental tables for constructing the map cube. Table 6 describes the spatial attributes and spatial concept hierarchy for each census track. The Boundary attribute contains the latitude and longitude of each point in the census track boundary. Table 7 contains the non-spatial attributes (Age Group, Income Type, Race category) and measure (number of people) for each census track.

CT-B			
Census-Track-ID	Boundary	County	State
10123	(,), (,),...	Dakota	Minnesota
10186	(,), (,),...	Hennepin	Minnesota

Table 6: Base Table CT-B for the Census Data

Population(Householder)				
Census-Track-ID	Age	Income	Race	No-of-people(House Holder)
10123	35-44	$5,000 to $9,999	White	40
10123	35-44	$10,000 to $14,999	White	25

Table 7: Base Table Population for the Census Data

4.2.2 Map cube 0-D SQL query

In this dimension, there is one map corresponding to householder population, which is the summary of all the age groups, income types, and race categories. Householder population density is defined as the the number of householders divided by the size of the region. The householder population density data are listed in table 8. In this table, householder population density is converted to the range of 0 to 1, which is the gray scale range. The query is to generate the map and reclassify the original map from a census block unit to be a county level block.

The query for the 0-D map cube:

Base-map=CT-B-Map
Base-table=CT-B,Population
Where CT-B.Census-Track-ID = Population.Census-Track-ID
Aggregate by Sum:No-of-people / Sum:Area(CT-B.boundary)
Reclassify by CT-B.County
Data cube dimension
Cartographic preference Color=grayscale,No-of-map-per-cuboid=one per category

The query for the 0-D data cube:

Select Geometric-union(CT-B.Boundary), Sum(No-of-people)/Sum(Area(CT-B.boundary))
From CT-B,Population
Where CT-B.Census-Track-ID = Population.Census-Track-ID
Group by CT-B.County

Table 8 and Figure 15 show the map cube and the associated data cube generated from the above map cube query. This 0-dimensional map cube allows users to observe the relative householder population densities of the seven counties within the Twin Cities metropolitan area. The data used based on 1990 census data. The maps are for illustrative purposes only since data quality was not evaluated. Located in the center area, Ramsey county has the highest householder population density with the dark region, while Carver county, located in the southwest of the map, has the lowest householder population density, as we can easily observe from the map.

County Name	Householder population Density (Householders / Sq km)	Gray Scale
Anoka	75.29	0.156
Carver	18.03	0.037
Dakota	67.39	0.140
Hennepin	293.0	0.610
Ramsey	480.3	1.000
Scott	21.07	0.043
Washington	48.88	0.101

Table 8: Householder population density in the seven counties of the Twin Cities region

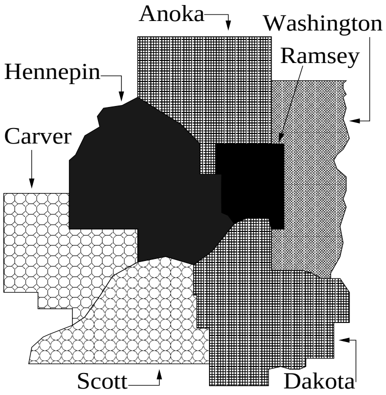


Figure 15: House-holder population density in the seven counties of the Twin Cities region

4.2.3 Map cube 1-D SQL query

In this dimension, there are seven maps, one for each age group. This query reclassifies the base map from a census block unit to a county level block, and generates a corresponding map for each age group.

The query for the 1-D map cube:

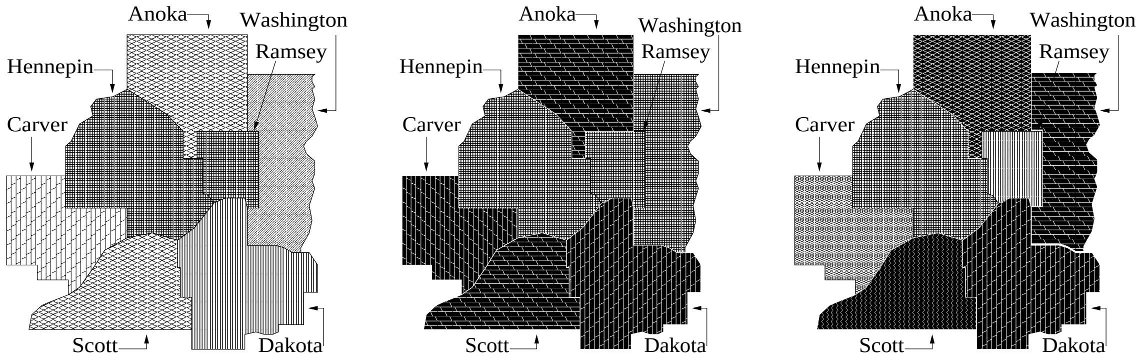
Base-map=CT-B-Map
Base-table=CT-B,Population
Where CT-B.Census-Track-ID = Population.Census-Track-ID
Aggregate by Sum:No-of-people
Reclassify by CT-B.County
Data cube dimension Age
Cartographic preference Color=grayscale,No-of-map-per-cuboid=one per category,
Normalize=CT-B.County(Population)

The query for the 1-D data cube:

Select Geometric-union(CT-B.Boundary), Sum(No-of-people)
From CT-B,Population
Where CT-B.Census-Track-ID = Population.Census-Track-ID
Group by CT-B.County, Age

County Name	Age <25	Age 25-34	Age 35-44	Age 45-54	Age 55-64	Age 65-74	Age >75	Total
Anoka	0.046	0.278	0.275	0.183	0.111	0.069	0.035	1
Carver	0.041	0.280	0.253	0.158	0.112	0.079	0.074	1
Dakota	0.052	0.300	0.273	0.163	0.101	0.066	0.043	1
Hennepin	0.063	0.265	0.235	0.142	0.112	0.099	0.080	1
Ramsey	0.064	0.252	0.223	0.135	0.114	0.108	0.092	1
Scott	0.038	0.295	0.270	0.165	0.097	0.073	0.059	1
Washington	0.027	0.254	0.285	0.198	0.117	0.070	0.046	1

Table 9: Fraction of people in the same age group within each county, normalized within each county



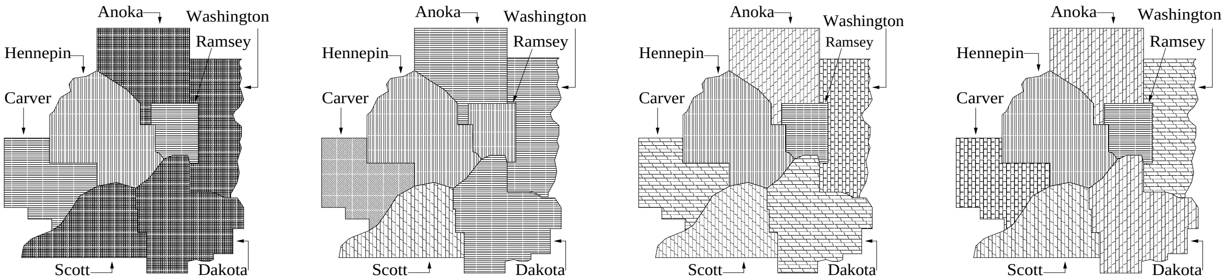
(a) Fraction of age group <25

(b) Fraction of age group 25-34

(c) Fraction of age group 35-44

Figure 16: Fraction of people in the same age group within each county

Table 9 and Figures 16 and 17 show the map cube and the associated data cube generated from the above map cube query. This 1-dimensional map cube allows users to observe the fraction of the age group within each county. Note that householder population is normalized within each county, which is specified by the cartographic preferences. A few interesting trends can be observed from the map cube. A comparison of the age-groups shows that most counties have more middle-age householders, i.e., from age groups 25-34, 35-44, 45-54, than other age groups. Comparing counties shows that the central counties, Ramsey and Hennepin, have a higher fraction of seniors as well as children.



(a) Fraction of age group 45-54

(b) Fraction of age group 55-64

(c) Fraction of age group 65-74

(d) Fraction of age group >75

Figure 17: Fraction of people in the same age group within each county

4.2.4 Map cube 2-D SQL query

In this dimension, there are sixty-three maps corresponding to the combination of the two dimensions (seven age groups and nine income types). This query reclassifies the base map from a census block unit to a county level block, and generates a corresponding map for each combination of age group and income level.

The query for the 2-D map cube:

Base-map=CT-B-Map
Base-table=CT-B,Population
Where CT-B.Census-Track-ID = Population.Census-Track-ID
Aggregate by Sum:No-of-people
Reclassify by CT-B.County
Data cube dimension Age, Income
Cartographic preference Color=grayscale,No-of-map-per-cuboid=one per category,
Normalize=CT-B.County(Population)

The query for the 2-D data cube:

Select Geometric-union(CT-B.Boundary), Sum(No-of-people)
From CT-B,Population
Where CT-B.Census-Track-ID = Population.Census-Track-ID
Group by CT-B.County, Age, Income

Table 10 and Figures 18 and 19 show the map cube and the associated data cube generated from the above map cube query. This 2-dimensional map cube allows users to observe the fraction of age group and income level within each county, where the householder population is normalized within each county. In the figures, we only display the maps for the less than 25 age group. Within this age group, there are nine maps associated with these nine income levels. As can be seen, few householders aged less than 25 have incomes more than $75,000.

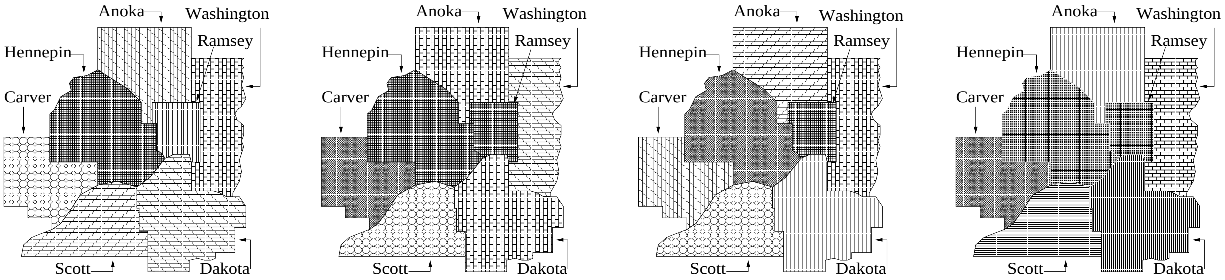
County	Age	less	$5,000	$10,000	$15,000	$25,000	$35,000	$50,000	$75,000	$100,000	County
		than	-	-	-	-	-	-	-	or	
Name	Group	$5,000	$9,999	$14,999	$24,999	$34,999	$49,999	$74,999	$99,999	more	Total
Anoka	<25	0.002	0.004	0.003	0.012	0.010	0.008	0.003	0.000	0.000	1
	25-34	0.004	0.009	0.009	0.036	0.054	0.096	0.056	0.007	0.003	
	35-44	0.002	0.004	0.006	0.026	0.045	0.080	0.080	0.019	0.009	
	45-54	0.001	0.001	0.003	0.013	0.023	0.046	0.061	0.021	0.009	
	55-64	0.002	0.004	0.004	0.014	0.018	0.027	0.025	0.009	0.004	
	65-74	0.002	0.008	0.010	0.017	0.012	0.009	0.005	0.000	0.000	
	>75	0.003	0.010	0.006	0.007	0.003	0.002	0.000	3.613	0.004	
Carver	<25	0.000	0.007	0.001	0.009	0.008	0.008	0.003	0.000	0.000	1
	25-34	0.002	0.007	0.009	0.034	0.057	0.085	0.060	0.014	0.007	
	35-44	0.002	0.003	0.007	0.018	0.042	0.069	0.067	0.024	0.014	
	45-54	0.001	0.002	0.003	0.017	0.021	0.028	0.042	0.021	0.017	
	55-64	0.002	0.004	0.006	0.013	0.017	0.022	0.027	0.007	0.009	
	65-74	0.004	0.010	0.013	0.019	0.011	0.009	0.007	0.001	0.001	
	>75	0.008	0.022	0.012	0.016	0.008	0.003	0.001	0.000	0.005	
Dakota	<25	0.001	0.005	0.006	0.012	0.010	0.010	0.004	0.000	0.000	1
	25-34	0.003	0.008	0.007	0.035	0.058	0.093	0.073	0.014	0.005	
	35-44	0.002	0.004	0.005	0.022	0.038	0.069	0.083	0.028	0.018	
	45-54	0.001	0.001	0.003	0.011	0.016	0.032	0.056	0.023	0.016	
	55-64	0.002	0.002	0.003	0.014	0.015	0.018	0.024	0.009	0.009	
	65-74	0.002	0.006	0.008	0.016	0.012	0.009	0.005	0.001	0.001	
	>75	0.004	0.010	0.007	0.010	0.004	0.003	0.001	0.000	0.003	
Hennepin	<25	0.005	0.010	0.008	0.016	0.011	0.008	0.002	0.000	0.000	1
	25-34	0.007	0.015	0.014	0.042	0.047	0.067	0.050	0.011	0.006	
	35-44	0.004	0.008	0.009	0.027	0.034	0.053	0.056	0.021	0.020	
	45-54	0.003	0.004	0.004	0.013	0.016	0.027	0.036	0.017	0.018	
	55-64	0.004	0.005	0.005	0.015	0.015	0.022	0.023	0.009	0.011	
	65-74	0.003	0.011	0.012	0.023	0.018	0.014	0.009	0.002	0.003	
	>75	0.005	0.020	0.013	0.018	0.008	0.006	0.004	0.000	0.004	
Ramsey	<25	0.005	0.011	0.009	0.016	0.010	0.008	0.002	0.000	0.000	1
	25-34	0.007	0.016	0.016	0.051	0.049	0.058	0.040	0.007	0.003	
	35-44	0.005	0.010	0.009	0.029	0.035	0.053	0.053	0.015	0.011	
	45-54	0.003	0.004	0.005	0.013	0.018	0.028	0.033	0.016	0.013	
	55-64	0.004	0.007	0.007	0.016	0.017	0.023	0.022	0.009	0.007	
	65-74	0.005	0.014	0.014	0.026	0.017	0.017	0.009	0.002	0.002	
	>75	0.006	0.026	0.016	0.019	0.010	0.006	0.004	0.001	0.003	
Scott	<25	0.001	0.002	0.001	0.010	0.006	0.011	0.003	0.000	0.000	1
	25-34	0.002	0.006	0.008	0.031	0.054	0.103	0.067	0.013	0.006	
	35-44	0.002	0.004	0.004	0.022	0.042	0.072	0.083	0.022	0.014	
	45-54	0.002	0.002	0.004	0.012	0.017	0.041	0.050	0.017	0.015	
	55-64	0.001	0.001	0.004	0.015	0.019	0.023	0.018	0.008	0.003	
	65-74	0.002	0.011	0.010	0.020	0.010	0.008	0.005	0.000	0.002	
	>75	0.007	0.021	0.009	0.013	0.004	0.002	0.000	0	0.004	
Washington	<25	0.001	0.004	0.002	0.005	0.005	0.005	0.001	0.000	0.000	1
	25-34	0.002	0.008	0.007	0.029	0.048	0.077	0.061	0.012	0.005	
	35-44	0.002	0.004	0.006	0.018	0.033	0.074	0.093	0.028	0.021	
	45-54	0.001	0.002	0.004	0.011	0.018	0.041	0.064	0.031	0.022	
	55-64	0.002	0.003	0.003	0.014	0.015	0.024	0.026	0.014	0.010	
	65-74	0.001	0.011	0.008	0.016	0.012	0.009	0.007	0.001	0.001	
	>75	0.003	0.011	0.009	0.009	0.004	0.004	0.001	0.000	0.004	

Table 10: Fraction of age group and income level within each county, normalized within each county

Therefore, in Figures 19(d) and 19(e), we see nearly empty maps.

5 Research Issues

In this section we elaborate on the research issues involving constructing spatial data warehouses, and in particular map cubes. The map cube inherits ideas from three different domains, namely, data warehouse, visualization and GIS (Chr96; KO96; Tuf97). A data warehouse is defined as "a subject-oriented, integrated, time-varying, non-volatile collection of data that is used primarily in organizational decision making." A spatial data warehouse is an extension to support spatial databases. Visualization is essentially a mapping process from computer representations to perceptual representations, choosing encoding techniques to maximize human understanding and communication. The primary objective of data visualization is to gain insight into an information space by mapping data onto graphical primitives (SI94). A map



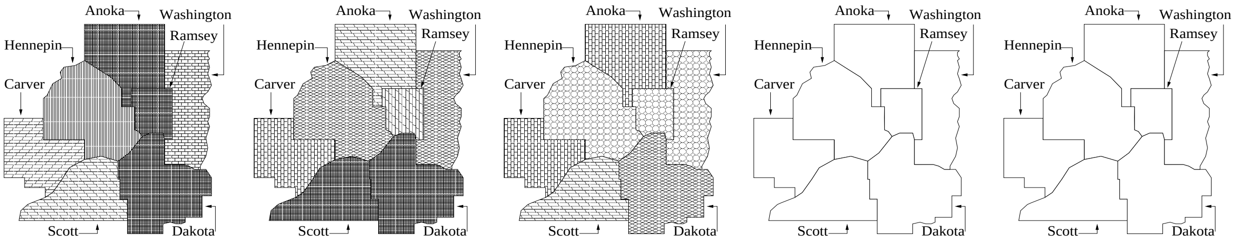
(a) Fraction of age group <25 and income level <$5,000

(b) Fraction of age group <25 and income level $5,000-$9,999

(c) Fraction of age group <25 and income level $10,000-$14,999

(d) Fraction of age group <25 and income level $15,000-$24,999

Figure 18: Fraction of the age group <25, with different income levels



(a) Fraction of age group <25 and income level $25,000-$34,999

(b) Fraction of age group <25 and income level $35,000-$49,999

(c) Fraction of age group <25 and income level $50,000-$74,999

(d) Fraction of age group <25 and income level $75,000-$99,999

(e) Fraction of age group <25 and income level > $100,000

Figure 19: Fraction of the age group <25, with different income levels

is a graphic representation of spatial relationships and spatial forms. Cartography is the art, science and technology of making maps (Mey73). A GIS is best defined as a system which uses a spatial database to provide answers to queries of geographical nature (Goo85). The three-way interaction model is shown in figure 20. The tertiary intersection is our topic "map cube."

5.1 Data Warehouse

5.1.1 Distinction between dimensions and attributes

To build a data cube from the original dataset, not all the attributes in the table can be represented as dimensions in the cube. Here, we define the concept of full functional dependency. A functional dependency $X \rightarrow Y$ is a full functional dependency if the removal of any attribute A from X means that the dependency does not hold any more. For example, in figure 21, we have three sets of full functional dependency, {Model,Year,Region}→{Profits}, {Model,Year,Region}→{# of Customers}, and {Model,Year,Region}→{Sales}. The attributes

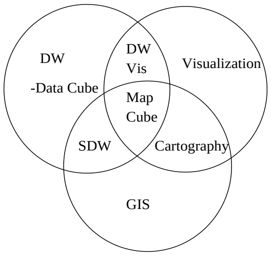


Figure 20: The relationship of the Map Cube with three parent domains

Model, Year, and Region can be used as the dimensions in the aggregate of Profits, # of customers, and Sales.

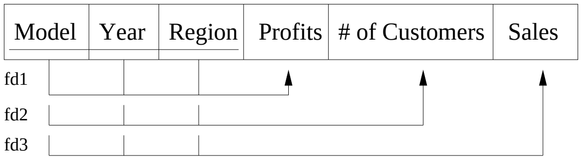


Figure 21: Example of functional dependency

5.1.2 Are all nodes in dimension hierarchy meaningful in map cubes?

For geographic data, the nodes in a higher dimension may not be meaningful. The reason is two-fold. First, the sample size is low in a higher dimension, so the statistical significance is decreased, such as the mean and median functions. Secondly, there may be privacy issues, as in the case of census data, where only block or census track data can be revealed, but not personal information. However, this constraint could be violated in high dimensions of the map cube. We illustrate this with the following example.

County	Census Track	Age	Income	Race	No of people
Dakota	24	35 - 44	$75,000 to $99,999	White	2
Dakota	24	35 - 44	$75,000 to $99,999	Black	1
Dakota	24	35 - 44	$75,000 to $99,999	Asian	2
Dakota	24	35 - 44	$100,000 or more	White	2
Dakota	24	35 - 44	$100,000 or more	Black	2
Dakota	24	35 - 44	$100,000 or more	Asian	1

Table 11: 3D data cube

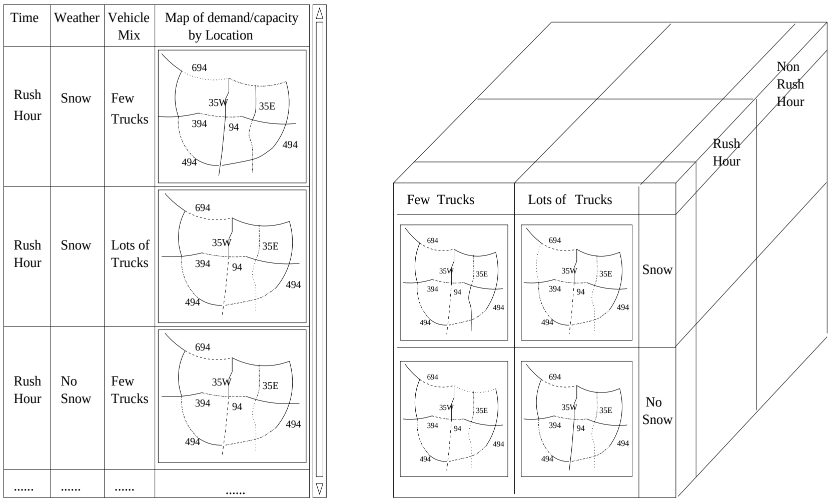
Table 11 shows the data cube of a census track, from which we can easily infer the income of a specific person if there is only one person in a certain category, and which would violate the personal privacy. Table 12 shows the aggregate data of a 2D data cube, which satisfies the personal privacy constraint. So while exploring census data, the answer to the question of whether a map cube should generate a full detail cube is clearly no as it violates the privacy constraint.

County	Census Track	Age	Income	No of people
Dakota	24	35 - 44	$75,000 to $99,999	5
Dakota	24	35 - 44	$100,000 or more	5

Table 12: 2D data cube

This brings on another question: “To what extent can users explore a map cube?” While this is subject specific, the map cube should provide constraints based exploration mechanisms.

5.1.3 Given a cuboid in the dimension hierarchy, is the tabular representation adequate?



Time	Weather	Vehicle Mix	Map of demand/capacity by Location
Rush Hour	Snow	Few Trucks	
Rush Hour	Snow	Lots of Trucks	
Rush Hour	No Snow	Few Trucks	
......	......	......	......

(a) The tabular view of the data

(b) The cube view of the data

Figure 22: Tabular view and cube view

Figure 22 shows the tabular and cube views of some traffic data. The tabular view provides a scroll bar for browsing the whole combination of the categories, including the ALL function in each dimension. The data can be sorted on each dimension, and the users can perform the Select, Project, and Join operations. In contrast, the cube view of the traffic data provides additional operations, such as slice or dice. Given the dimension lattice, the user will browse the subcube by using the slice or dice operation. Which view is more suitable for representing the map cube and more convenient for users to browse the requested information, the tabular or cube view?

5.1.4 How can the Map Cube be efficiently implemented?

A map cube is an extension of a conventional data cube, which utilizes cartographic techniques for efficient exploration of spatial data. The conventional data cube operations are designed to work with the data stored in relational DBMS, which deals with well-defined data types, but not with geographic data types. Therefore, new algorithms are needed when dealing with spatial data in order to provide a tabular or pivot view of a map cube. Furthermore, a map cube and an ordinary data cube require different implementation algorithms.

Selection of Groupbys for Precomputation

Each cell of a map cube is a view consisting of an aggregation of interest. The values of many of these cells are dependent on the values of other cells in the map cube. A query optimization technique is used to materialize some or all of these cells rather than computing them from raw data cubes. The important objective is to develop techniques for optimizing the space-time tradeoff when implementing a lattice of views, which is known to be an NP-complete problem. Harinarayan et al. investigated the issue of which cells(views) to materialize when it is too expensive to materialize all views (HRU96). Gupta proposed a framework for the selection of views to optimize the query response time within polynomial time (Gup97).

The analysis of whether a cuboid should be selected for materalization in a spatial data cube is similar to that in a non-spatial one. Spatial computation, such as region merging, map overlaying and spatial join, could be expensive, especially when a large number of spatial objects are involved. There is an additional factor to be considered for a spatial data cube: the cost of on-line computation of a particular spatial cuboid. Han et al. proposed some techniques for selective materalization of spatial computation results (HSK98). They assumed that a set of cuboids have been selected for materalization using an extended cuboid-selection algorithm similar to the one in (HRU96), and examined how to determine which sets of mergeable spatial objects should be precomputed. Two algorithms, *pointer intersection* and *object connection*, are worked out for the selection of precomputed objects.

5.1.5 Is the location one dimension or two or more?

Figure 23 shows how to aggregate the data on a three dimensional Earth space. Initially, the cuboid has three dimensions < Latitude,Longitude,Altitude>. Then the data is aggregated into two dimensions. At the bottom level, all data are summarized into one point.

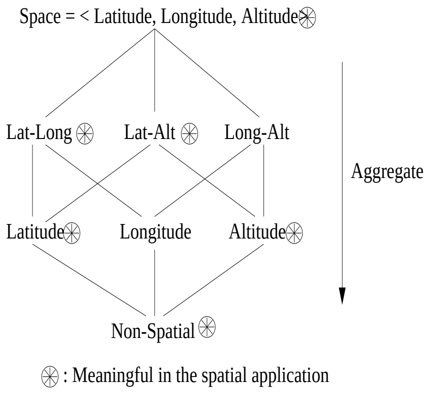

Space = < Latitude, Longitude, Altitude⊛
Lat-Long ⊛
Lat-Alt ⊛
Long-Alt
Aggregate
Latitude⊛
Longitude
Altitude⊛
Non-Spatial⊛
⊛ : Meaningful in the spatial application

Figure 23: The space lattice

The cuboids in the lattice that are meaningful for certain applications, e.g., weather analysis, are marked. For example, groupby Altitude is important for analyzing weather patterns in each Elevation. However, it makes no sense to group by Longitude. In designing a map cube, we can either allow users to explore all subspaces(cuboids) of the three dimensional geographic space, or we may restrict subspaces(cuboids) to meaningful ones.

Choice	Allow all sub-space	Restrict to useful subspace
Computation Effort	High	Low, since useless subspaces are not computed
Semantic Integrity	Low, since some maps generated in map-cube may be violating geographic constraints	High, all maps are meaningful

Table 13: Decision to allow all sub-spaces to be computed or not

5.2 Cartographic Visualization

As discussed in the previous sections, a map is the core output of a map cube. A map cube can be defined as a graphic depiction of all or part of a geographic realm in which real-world features have been replaced with symbols in their correct spatial location at a reduced scale. The process of creating graphic symbols to represent feature attribute values is part of what we call symbolization. The symbolization process is carried out after applying classification, simplification and exaggeration processes to the database selected for mapping. Hence we have two separate components of the map design process. The first encodes information, and is an abstract process related to generalization. The second involves conceptual constraints and focuses on graphic representation. Both components have a theoretical basis in the field of semiotics. Semiotics addresses the relations among symbols, signs, and meanings, and forms an appropriate basis for cartographic symbolization. Geographic features have a dimensionality ranging from 0 to 3. A point feature is dimensionless (zero dimension) while a line feature has one dimension. An area feature has two dimensions and a volume feature has three dimensions.

5.2.1 Issues involved in the visualization of geographic features

Geographic features conceived as points (e.g., lamp posts, buildings or even cities depending on scale) are portrayed by point symbols. On a nominal or ordinal scale, points can be depicted by symbols with differentiating visual variables (like shape, hue(color) and orientation). Quantitative point attribute data can be symbolized by using primary visual variable size. Geographic features conceived as lines (e.g., rivers, roads, ...) can be portrayed by line symbols. Nominally scaled line features can be depicted by primary visual variables (hue and shape) and quantitative line features are symbolized using visual variables of size and hue. Features conceived as areas (like thematic maps, or remote sensing images) are depicted by secondary visual variables (texture, arrangement, and orientation). Multi-channel images are shown as either black and white (with varying gray shades), or RGB (by assigning different channels to each of red, blue and green color guns). Now several questions arise:

- Is the classification and symbolization process sufficient to visualize complex query maps resulting from aggregation of multi-dimensions?.
- How can a glyph family be designed in a way such that the members can be related in a manner similar to the data cube dimension hierarchy?
- What is the effect of a data cube on map legend design and other cartographic issues?
- How do we visualize trends across spatial as well as non-spatial dimensions?

In a map cube, not only do users compare different measures in all regions within one map, but they may also be interested in measuring the variations across different maps. It is not easy to show both the trends within a map and across different maps at the same level.

For example, Table 14 shows the population ratio for each age group in each county. The views that users may be interested in are the variation across different counties within that same age group, and the variation across different age groups in the same county. The most suitable cartographic representation is a map of counties, and within each county, a pie-chart for different age groups.

County-name	under 25	25 to 34	35 to 44	45 to 54	55 to 64	65 to 74	75 and over
Anoka	0.045	0.277	0.274	0.183	0.110	0.068	0.039
Carver	0.041	0.279	0.251	0.157	0.111	0.078	0.079
Dakota	0.051	0.299	0.272	0.162	0.101	0.066	0.045
Hennepin	0.063	0.264	0.235	0.142	0.112	0.098	0.082
Ramsey	0.064	0.253	0.225	0.136	0.115	0.109	0.095
Scott	0.037	0.294	0.269	0.164	0.096	0.073	0.063
Washington	0.027	0.254	0.284	0.197	0.116	0.070	0.049

Table 14: Fraction of people in the same age group within each county

5.3 Geographic Information System

5.3.1 Which applications can benefit from a map cube?

In our previous examples, we have shown the usefulness of the map cube in analyzing traffic and census data. In summary, a map cube can be applied to any spatial application which requires comparisons between different attributes and requires some aggregation functions within each attribute.

5.3.2 How to integrate new aggregate functions into a map cube?

Aggregate operations are domain specific. Thus, the aggregate operations in a conventional data cube, which can be realized by a groupby, may not be directly applicable to spatial databases. For example, the arithmetic addition of two images may not give a meaningful result, whereas the same operation on an attribute (say, salary) in a database will be valid. Therefore, it is necessary to find and integrate suitable operators for spatial databases. In this section we discuss some of these issues.

Applied Spatial Statistics: Applied spatial statistics deals with those methods that are applicable to data that are spatially correlated. Some of the methods which are interesting to spatial data warehouses include moving window statistics (e.g., calculating mean, variance etc,. within a small moving window), spatial clustering, cross-correlation, cross-covariance, spatial autocorrelation and spatial sampling methods. Examples where these methods are needed include calculating the number of neighbors within a distance D, and grouping crimes by the nearest police station.

Local operation: A local operation acts upon one or more spatial field functions of the field-model to produce a new field. The value of the new field at a location is dependent only on the value of the original field at that location. Examples include pointwise sums, differences in maximums, minimums, or means.

Focal operation: For a focal operation, the attribute value derived at a location x may depend not only on the attributes of the input spatial field function at x, but also on the attributes of these functions in the neighborhood $n(x)$ of x. Examples include slope, aspect, and weighted average of neighborhood.

Zonal operation: A zonal operation aggregates values of a field over each of a set of zones in the framework. Examples include sum, mean, or maximum of field value in each zone.

Geometric Union, Geometric Intersection: Aggregation in a spatial dimension includes geometric union and intersection. For example, at a lower dimension we may have a state boundary map and in the next dimension we may have country maps which are derived from state boundary maps by applying a geometric union operation(a spatial aggregate function).

NDVI: In the case of satellite images, the normalized density vegetation index, obtained by rationing the difference between near infra-red and red to their sum. This rationed image highlights (or distinguishes) vegetative regions from non vegetative regions.

$$NDVI = \frac{NearInfraredData - VisibleData}{NearInfraredData + VisibleData} \tag{1}$$

The higher values of the NDVI reveal pixels dominated by high proportions of green biomass.

OGIS: Much work has been done over the last decade on the design of spatial Abstract Data Types(ADTs) and their embedding in a query language. Recently, the OGIS (OGI99) consortium has proposed a specification for incorporating 2D geospatial ADTs in SQL. The present OGIS standard is not sufficient to construct complex spatial data warehouses, which include many other data types and operations not covered by OGIS.

5.3.3 Is a scale operation the same as a concept hierarchy?

A scale operation changes the size of an object in an embedding plane without changing the shape, position or orientation of the object. For example, the scaling of $'a'$ in the x-direction and $'b'$ in the y-direction is affected by the rule:$(x, y) \rightarrow (ax, by)$. To build a concept hierarchy by performing a scale operation, we plot a fixed size grid in the embedding plane. Initially, all the map objects are within one grid, which is the top level in the hierarchy. As we scale up the map, the map will extend to the surrounding grids, by which we construct the lower level in the hierarchy. The effect of scaling up/down can also be accomplished by changing the size of the grid in the embedding plane.

6 Conclusion and Future Work

The cube operator generalizes and unifies several common and popular concepts: aggregation, group-by, histogram, roll-up, drill-down, and cross-tab. Maps are the core of the Geographic Information System. In this paper, we extended the concept of data cube to spatial domain via the proposed "map cube" operator, which is built from the requirements of spatial data warehouses. We defined the "map cube" operator, provided grammar and translation rules, and used census data as an application of the map cube. Since the map cube inherits ideas from three different domains, namely, data warehouse, visualization and GIS, we also discussed the research issues involved with these domains. In the future, we would like to explore the implementation and application of the map cube in spatial data warehouses.

7 Acknowledgment

This work is sponsored in part by the Army High Performance Computing Research Center under the auspices of the Department of the Army, Army Research Laboratory cooperative agreement number DAAH04-95-2-0003/contract number DAAH04-95-C-0008, the content of which does not necessarily reflect the position or the policy of the government, and no official endorsement should be inferred. This work was also supported in part by NSF grant #9631539.

References

Y. Bedard. Visual modeling of spatial databases towards spatial extensions and uml. In *Geomatica*, 1999.

E.F. Codd, S.B. Codd, and C.T. Salley. *Providing OLAP (On-line Analytical Processing) to User-Analysts: An IT Mandate.* Arbor Software Corporation. In URL: http://www.arborsoft.com/essbase/wht_ppr/coddToc.html, 1993.

S. Chaudhuri and U. Dayal. An overview of data warehousing and olap technology. *SIGMOD Record*, 26(1):65–74, 1997.

N. Chrisman. *Exploring Gepgraphic Information Systems.* John Wiley and Sons, 1996.

E.F. Codd. Twelve rules for on-line analytic processing. In *Computerworld, April 1995*, 1995.

ESRI. White paper: Spatial data warehousing for hospital organizations. In *URL: http://www.esri.com/library/whitepapers/addl_lit.html*, 1998.

P. Ferguson. Census 2000 behinds the scenes. In *Intelligent Enterprise*, October 1999.

J. Gray, A. Bosworth, A. Layman, and H. Pirahesh. Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-total. In *Proceedings of the Twelfth IEEE International Conference on Data Engineering*, pages 152–159, 1995.

M.F. Goodchild. Geographic information systems in undergraduate geography: A contemporary dilemma. *The Operational Geographer*, 8:34–38, 1985.

H. Gupta. Selection of views to materialize in a data warehouse. In *International Conference on Database Theory, Delphi, Greece*, January 1997.

V. Harinarayan, A. Rajaraman, and J. Ullman. Implementing data cubes efficiently. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, Montreal, Quebec, Canada, June 4-6, 1996*, pages 205–216. ACM Press, 1996.

J. Han, N. Stefanovic, and K. Koperski. Selective materialization: An efficient method for spatial data cube construction. In *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'98)*, pages 144–158, 1998.

W.H. Inmon and R.D. Hackathorn. *Using the Data Warehouse.* New York, NY: John Wiley & Sons, 1994.

W.H. Inmon. *Building the Data Warehouse.* New York, NY: John Wiley & Sons, 1993.

W.H. Inmon, J.D. Welch, and K.L. Glassey. *Managing the Data Warehouse.* New York, NY: John Wiley & Sons, 1997.

M.J. Kraak and F. Ormeling. *Cartographer: Visualization of Spatial Data.* Longman, 1996.

R. Kimball, L. Reeves, M. Ross, and W. Thornthwaite. *The Data Warehouse Lifecycle Toolkit.* Wiley, 1998.

J.R. Levine, T. Mason, and D. Brown. *Lex and Yacc.* O'Reilly And Associates, 1992.

E. Meyen. Multilingual dictionary of technical terms in cartography. In *International Cartographic Association, Commission II*, page 573. Franz Steiner Verlag, 1973.

Microsoft. Terraserver: A spatial data warehouse. In *URL: http://terraserver.microsoft.com/*, 2000.

I.S. Mumick, D. Quass, and B.S. Mumick. Maintenance of data cubes and summary tables in a warehouse. In *SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data, May 13-15, 1997, Tucson, Arizona, USA*, pages 100–111. ACM Press, 1997.

OGIS. Open GIS Consortium: Open GIS Simple Features Specification for SQL (Revision 1.1). In *URL: http://www.opengis.org/techno/specs.htm*, 1999.

M. Stonebraker, J. Frew, and J. Dozier. The sequoia 2000 project. In *Proceedings of the Third International Symposium on Large Spatial Databases*, 1993.

H. Senay and E. Ignatius. A knowledge-based system for visualization design. *IEEE Computergraphics & Applications*, 14(6):36, 1994.

E. Tufte. *Envisioning Information.* Graphic Press, 1997.

USCB. U.S. Census Bureau. In *URL: http://www.census.gov*, 2000.

USCB. U.S. Census Bureau: American FactFinder. In *URL: http://factfinder.census.gov*, 2000.

USGS. National satellite land remote sensing data archive. In *URL: http://edc.usgs.gov/programs/nslrsda/overview.html*, 1998.

J. Widom. Research problems in data warehousing. In *CIKM '95, Proceedings of the 1995 International Conference on Information and Knowledge Management, November 28 - December 2, 1995, Baltimore, Maryland, USA*, pages 25–30. ACM Press, 1995.

M.F. Worboys. *GIS: A Computing Perspective.* Taylor & Francis, 1995.