Data Cube: A Relational Aggregation Operator
Generalizing Group-By, Cross-Tab, and Sub-Totals

J. Gray, al., Microsoft Research
F. Pellow, al., IBM Research

Presented by Jim Cao
Discussion led by Otto

Topic Outline

- Visualization and dimension reduction
- The relational representation of N-dimensional data
- What is CUBE
- Why we need ALL value
- Summary

Data analysis applications
- Looking for anomalies or unusual patterns
- Four steps to aggregate data across many dimensions
  - formulating a query that extracts relevant data from a database
  - extracting the aggregated data from the database into a table
  - visualizing the results in a graphical way, and
  - analyzing the results and formulating a new query
- Represent the dataset as an N-dimensional space

“Dimensionality Reduction”

Analyse car sales
- Focus on the role of model, year and color of the cars
- Ignore the differences between two sales along the dimensions of date of sale or car dealership
- As a result, extensive constructs are used, such as cross-tabulation, subtotals, roll-up and drill-down

One Dimensional Aggregation

Example: Car sales for year 1994 and 1995 showed in table_1:

<table>
<thead>
<tr>
<th>Model</th>
<th>Sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chevy</td>
<td>290</td>
</tr>
<tr>
<td>Ford</td>
<td>220</td>
</tr>
</tbody>
</table>

If we need to know the sales for model, we can easily query it by:

SELECT sales FROM table_1 GROUP BY model

Three Dimensional Aggregation

If we need more dimensional generalization of these operators

<table>
<thead>
<tr>
<th>Model</th>
<th>Year</th>
<th>Color</th>
<th>Sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chevy</td>
<td>1994</td>
<td>black</td>
<td>50</td>
</tr>
<tr>
<td>Chevy</td>
<td>1995</td>
<td>black</td>
<td>85</td>
</tr>
<tr>
<td>Chevy</td>
<td>1994</td>
<td>white</td>
<td>40</td>
</tr>
<tr>
<td>Chevy</td>
<td>1995</td>
<td>white</td>
<td>115</td>
</tr>
</tbody>
</table>
Three Dimensional Aggregation (con.t)

If we need to query the sales by model, by year, and by color, then how we can do it?

Typically, we can make a report as showed by Table_2a:

<table>
<thead>
<tr>
<th>Model</th>
<th>Year</th>
<th>Color</th>
<th>Sales by Model by Year by Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chevy</td>
<td>1994</td>
<td>Black</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>White</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>1995</td>
<td>Black</td>
<td>85</td>
</tr>
<tr>
<td></td>
<td></td>
<td>White</td>
<td>135</td>
</tr>
</tbody>
</table>

For Table_2a:
- Concepts: going up the levels is called **rolling-up** the data.
- Going down is called **drilling-down** into the data.
- In this table, sales are rolled up by using totals and subtotals.
- Data is aggregated by Model, then by Year, then by Color.
- The report shows data aggregated at three levels, that is, at Model level, Year level, and Color level.
- Data aggregated at each distinct level produces a sub-total.

Three Dimensional Aggregation (con.t)

What problems with Table_2a approach?
- Table_2a suggests creating 2\(N\) aggregation columns for a roll-up of \(N\) elements. That is, there are **six columns** in table_2a
- Also, the representation of Table_2a is **not relational**, because the empty cells (presumably NULL values), **cannot form a key**

A pivot table in Excel

The approach by using a pivot table in Excel is showed by Table_2c:

<table>
<thead>
<tr>
<th>Year</th>
<th>Model</th>
<th>Color</th>
<th>1994</th>
<th>1995</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Chevy</td>
<td>Black</td>
<td>20</td>
<td>85</td>
<td>105</td>
</tr>
<tr>
<td></td>
<td></td>
<td>White</td>
<td>40</td>
<td>75</td>
<td>115</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Total</td>
<td>140</td>
<td>160</td>
<td>290</td>
</tr>
</tbody>
</table>

What problems with pivot table approach?
- The pivot operator typically aggregating cells based on values in the cells.
- Pivot creates columns based on subsets of column values-this is a much larger set!
- If one pivots on two columns containing \(N\) and \(M\) values, the resulting pivot table has \(N \times M\) values, that’s, so many columns and such obtuse column names!

ALL value approach

One more approach by adding an ALL value is available
- Do not extend the result table to have many new columns
- Avoid the exponential growth of columns by overloading column values
- The dummy value “ALL” has been added to fill in the super-aggregation items

ALL value approach (con.t)

Table_3a: Sales summary

<table>
<thead>
<tr>
<th>Model</th>
<th>Year</th>
<th>Color</th>
<th>Unites</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chevy</td>
<td>1994</td>
<td>Black</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>White</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>1995</td>
<td>Black</td>
<td>85</td>
</tr>
<tr>
<td></td>
<td></td>
<td>White</td>
<td>135</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ALL</td>
<td>290</td>
</tr>
</tbody>
</table>

For Table_3a:
- This is a 3-dimensional roll-up
- It have three unions
- The fact is that aggregating over \(N\) dimensions requires \(N\) such UNIONS!
ALL value approach (con.t)

Since table-3a is a relation, it could be built using SQL, like this statement:

```sql
SELECT Model, Year, Color, SUM(Sales) AS sales
FROM Sales
WHERE Model in ['Ford', 'Chevy']
AND Year BETWEEN 1994 AND 1995
GROUP BY CUBE Model, Year, Color;
```

The CUBE operator

- The CUBE operator is the N-dimensional generalization of simple aggregate functions.
- The N-1 lower-dimensional aggregates appear as points, lines, planes, cubes.
- The data cube operator builds a table containing all these aggregate values.
- The 0D data cube is a point.
- The 1D data cube is a line with a point.
- The 2D data cube is a cross tabulation, a plane, two lines, and a point.
- The 3D data cube is a cube with three intersecting 2D cross tabs.

The introduction of ALL creates substantial complexity

- ALL becomes a new keyword denoting the set value
- ALL [NOT] ALLOWED is added to the column definition syntax and to the column attributes in the system catalogs
- If ALL presents a set then the other values of that domain must be treated as singleton sets in order to have uniform operators on the domain
- However, it is impossible to express results of CUBE as a single relation in the current framework of SQL without ALL value!
- Therefore, the ALL value is needed.
Summary

• The cube operator generalizes and unifies several common and popular concepts: such as aggregates, group by, histograms, roll-ups and drill-downs and, cross tabs.
• The cube operator is based on a relational representation of aggregate data using the ALL value to denote the set over which each aggregation is computed.
• The data cube is easy to compute for a wide class of functions
• SQL’s basic set of five aggregate functions needs careful extension to include

Discussion Questions

• The authors state that “Veteran SQL implementers will be terrified of the ALL value --- like NULL, it will create many special cases.”
  What are some of the special cases that you can imagine are created by NULL?
  What cases can you imagine being created by ALL?
  Do think ALL is a bigger or a lesser concern than NULL?
• How many applications can you imagine using Data Cubes?
• Does this strike you as a big or a small change to SQL? What about to the mentality of relational databases?