



# Polaris: A System for Query, Analysis and Visualization of Multidimensional Databases

---

*Authors: Chris Stolte, Diane Tang, and Pat Hanrahan*

*Role play by: Shreena Bhati (Author 1) | Mayank Sethi (Author 2)*

# *Problem Overview*

Corporations often need to perform analysis on data to:

- *discover structure*
- *find patterns*
- *derive causal relationships*
- *explore to uncover hidden meaning*

**But**, this data behind such analysis is rapidly becoming **large and multidimensional**.

**As a result**, such large data sets require **dense graphical representation** instead of spreadsheets and charts.

And, these visualizations need to be in **real-time** when running test cycles because **exploration involves experimentation**.

# The challenge behind analysis on data

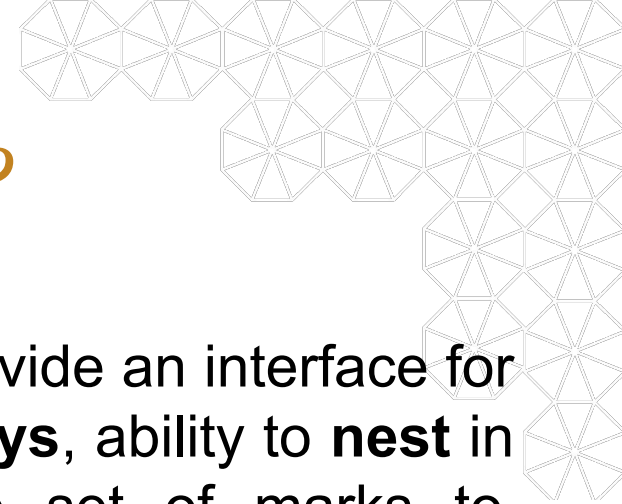
- How to design an interface for **multidimensional databases** fulfilling such **demanding user needs**.
- How to cater to an **unpredictable exploratory path**.
- How to accommodate **what and how to view data in real-time**.

The screenshot displays a web application interface. On the left, an 'Organization Chart Demo' shows a hierarchical tree structure with nodes for various roles and names, such as Eric Joplin (Chief Executive Officer), Gary Roberts (Senior Account Executive), and David Kiery (Chief Financial Officer). On the right, a 'Properties' panel for Eric Joplin lists contact information: DEPT: Executive Unit, EMAIL: ejoplin@yfiles.com, PHONE: 555-0100, FAX: 555-0011, and STATUS: Available. Below the chart and properties, a table lists employee data with columns for ID, Name, Email, Phone, and Status. The table contains 37 rows of data, including roles like 'Chief Executive Officer', 'Senior Executive Assistant', 'Quality Assurance Technician', and 'Tool Designer'.

ID	Column1	Column2	Column3	Column4	Column5
1	Chief Executive Officer	Eric Joplin	ejoplin@yfiles.com	555-0100	555-0101
2	Senior Executive Assistant	Alexander Burns	aburns@yfiles.com	555-0102	555-0103
3	Quality Assurance Technician	Janis Lovelace	jlovelac@yfiles.com	555-0104	555-0105
4	Quality Assurance Technician	Gary Elliott	gelliott@yfiles.com	555-0106	555-0107
5	Quality Assurance Technician	Martin Connell	mconnell@yfiles.com	555-0108	555-0109
6	Quality Assurance Technician	Valerie Burnett	vburnett@yfiles.com	555-0110	555-0111
7	Quality Assurance Technician	Edward Monge	emonge@yfiles.com	555-0112	555-0113
8	Quality Assurance Technician	Suzanne Moran	smoran@yfiles.com	555-0114	555-0115
9	Quality Assurance Technician	Valerie Burnett	vburnett@yfiles.com	555-0116	555-0117
10	Quality Assurance Technician	Edward Monge	emonge@yfiles.com	555-0118	555-0119
11	Quality Assurance Technician	Suzanne Moran	smoran@yfiles.com	555-0120	555-0121
12	Quality Assurance Technician	Valerie Burnett	vburnett@yfiles.com	555-0122	555-0123
13	Quality Assurance Technician	Edward Monge	emonge@yfiles.com	555-0124	555-0125
14	Quality Assurance Technician	Suzanne Moran	smoran@yfiles.com	555-0126	555-0127
15	Quality Assurance Technician	Valerie Burnett	vburnett@yfiles.com	555-0128	555-0129
16	Quality Assurance Technician	Edward Monge	emonge@yfiles.com	555-0130	555-0131
17	Quality Assurance Technician	Suzanne Moran	smoran@yfiles.com	555-0132	555-0133
18	Quality Assurance Technician	Valerie Burnett	vburnett@yfiles.com	555-0134	555-0135
19	Quality Assurance Technician	Edward Monge	emonge@yfiles.com	555-0136	555-0137
20	Quality Assurance Technician	Suzanne Moran	smoran@yfiles.com	555-0138	555-0139
21	Quality Assurance Technician	Valerie Burnett	vburnett@yfiles.com	555-0140	555-0141
22	Quality Assurance Technician	Edward Monge	emonge@yfiles.com	555-0142	555-0143
23	Quality Assurance Technician	Suzanne Moran	smoran@yfiles.com	555-0144	555-0145
24	Quality Assurance Technician	Valerie Burnett	vburnett@yfiles.com	555-0146	555-0147
25	Quality Assurance Technician	Edward Monge	emonge@yfiles.com	555-0148	555-0149
26	Quality Assurance Technician	Suzanne Moran	smoran@yfiles.com	555-0150	555-0151
27	Quality Assurance Technician	Valerie Burnett	vburnett@yfiles.com	555-0152	555-0153
28	Quality Assurance Technician	Edward Monge	emonge@yfiles.com	555-0154	555-0155
29	Quality Assurance Technician	Suzanne Moran	smoran@yfiles.com	555-0156	555-0157
30	Quality Assurance Technician	Valerie Burnett	vburnett@yfiles.com	555-0158	555-0159
31	Quality Assurance Technician	Edward Monge	emonge@yfiles.com	555-0160	555-0161
32	Quality Assurance Technician	Suzanne Moran	smoran@yfiles.com	555-0162	555-0163
33	Quality Assurance Technician	Valerie Burnett	vburnett@yfiles.com	555-0164	555-0165
34	Quality Assurance Technician	Edward Monge	emonge@yfiles.com	555-0166	555-0167
35	Quality Assurance Technician	Suzanne Moran	smoran@yfiles.com	555-0168	555-0169
36	Quality Assurance Technician	Valerie Burnett	vburnett@yfiles.com	555-0170	555-0171
37	Quality Assurance Technician	Edward Monge	emonge@yfiles.com	555-0172	555-0173

# Current approaches

Tool/Approach	Description and flaws
<b>Table Based Displays</b>	
<b>Pivot Table</b>	→ Pivot Tables allow analysts to explore different projections of large multidimensional datasets by interactively specifying assignments of fields to the table axes, but are <b>limited to text-based displays</b> .
<b>Table Lens &amp; Focus</b>	→ These visualization systems provide table displays that present data in a relational table view, using simple graphics in the cells to communicate <b>only quantitative values</b> .
<b>Exploration Tools</b>	
<b>VQE, Visage, DEVis</b>	→ Directly support interactive database exploration by visual queries. → None leverages <b>table-based organizations of their visualizations</b> .
<b>XmdvTool, Spotfire, XGobi</b>	→ Has a set of predefined visualizations like scatterplot. → Approach is <b>much more limiting</b> than providing the user with a set of building blocks that can be used to interactively construct.
<b>VisDB</b>	→ Displays as <b>many data items</b> as possible to provide feedback. Even displays tuples that do not meet the query and indicates their distance from the query criteria using spatial encodings and color. → <b>No ability to dive deeper into data or roll up</b> as user develops sense of the structure.

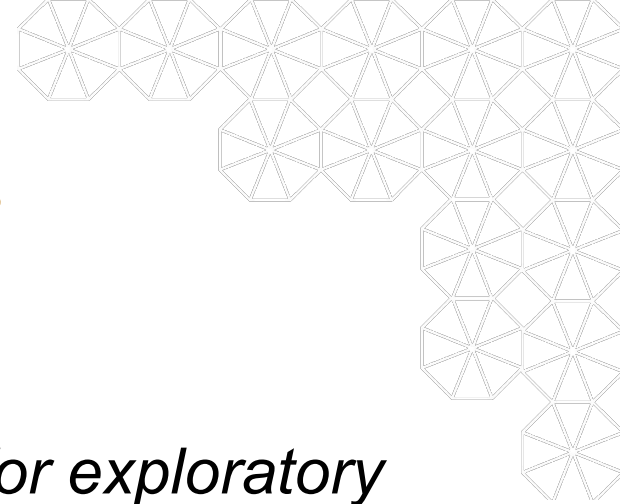


## *Why the current solutions fail?*

Current solutions have limited capacity to provide an interface for incrementally generating **table-based displays**, ability to **nest** in various dimensions, and **visually encode** set of marks to encode a graphic.

And we need table based display because they enable:

1. **Displaying multivariate dimensions:** explicitly encoded in the structure of the table.
2. **Performing comparative analysis:** easily compared, exposing patterns and trends across dimensions of the data.
3. **Familiarity:** Table-based displays have an extensive history.



# *Segmented Research Objective*

*Developing a human interface or tool for exploratory data analysis which formalises constructing table views, graphs and building data transformation on large multidimensional datasets.*

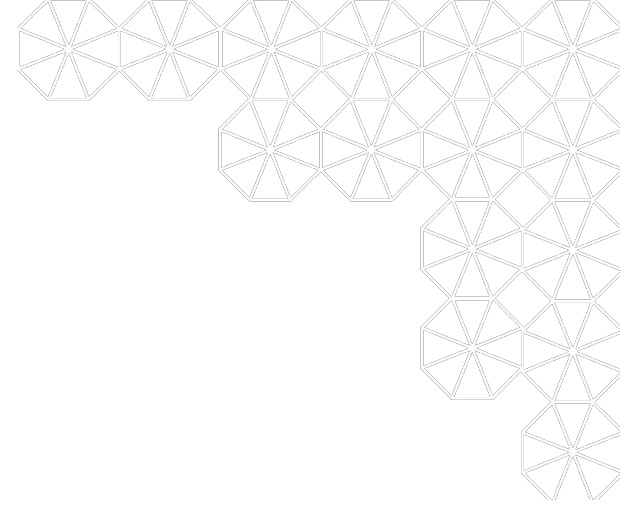
***We call this tool as Polaris.***

# Technical aspects of the problem

- Firstly, data in any relational database is in table format.
- Usually, data fields are characterized as - **nominal, ordinal, quantitative or interval**.
- Also, such data can be partitioned based on **dimensions or measures**, like independent and dependent variables.

So, to effectively support analysis, we need to have:

- **Data-dense displays**: to create visualisations that display many dimensions at once.
- **Multiple display types**: generate displays suited to discovering correlations, finding patterns and uncovering structure.
- **Exploratory interface**: rapidly change what and how users are viewing data.



# Research Process



# Behind the technical requirements

In Polaris, we define each table which consists of rows, columns, and layers.

- Each table axis may contain multiple nested dimensions.
- Each table entry, or pane, contains a set of records that are visually encoded as a set of marks to create a graphic.

**Database Schema:** The user drags fields from the database schema to shelves to define the visual specification.

**Layer Tabs:** Each layer has its own tab; different transformations and mappings can be specified for each layer.

**Axis Shelves:** The fields placed here determine the structure of the table and the types of graphs in each table pane.

**Context Menu:** The context menu provides access to the data transformation and interaction capabilities of Polaris such as sorting, filtering, and aggregation.

**Layer Shelf:** The fields placed here determine how records are partitioned into layers.

**Grouping and Sorting Shelves:** The fields placed here determine how records are grouped and sorted within the table panes.

**Mark Pulldown:** Relations in each pane are mapped to marks of the selected type.

**Retinal Property Shelves:** The fields placed here determine how data is encoded in the retinal properties of the marks.

**Legends:** Legends enable the user to see and modify the mappings from data to retinal properties.

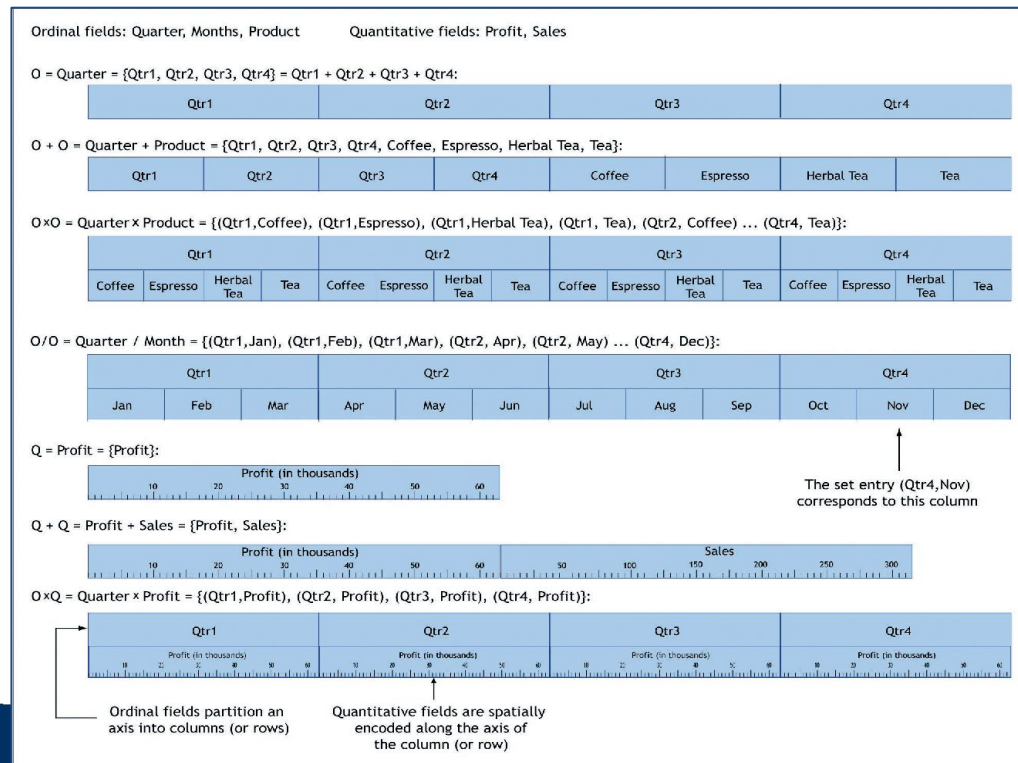
# *Specifying data requirements visually*

The primary interaction with the tool leads to defining -

- *Mapping of data sources to layers to combine into single visualization.*
- *Number of rows, columns and layers and their relative order.*
- *Selection of records from database*
- *Partitioning of records*
- *Computation of statistical properties, aggregates and other derived fields.*
- *Type of graphic displayed*
- *Mapping of data fields to retinal properties of marks in graphics.*

# Working on generating Graphics

1. Visual specifications consists of 3 components-
  - Table configuration using defined Table Algebra:
    - This includes axes along x and y and a dimension along axis z (layers).
    - This algebra type supports concatenation, Cross, Nest(Like quarter/month) within set of tuples along a dimension.



# Continued...

## *Working on generating Graphics*

### 2. Types of Graphics:

After the table specification, graphic space is structured into 3 families:

- Ordinal-Ordinal (Eg. Plot for attributes)
  - Task is to understand patterns and trends in some function.
- Ordinal-Quantitative (Eg: Bar Chart, dot plot and Gantt chart)
  - Task is to understand or compare the properties of some set of function
- Quantitative-Quantitative (Eg. ScatterPlot)
  - Discover causal relationships between two quantitative variables







# Continued...

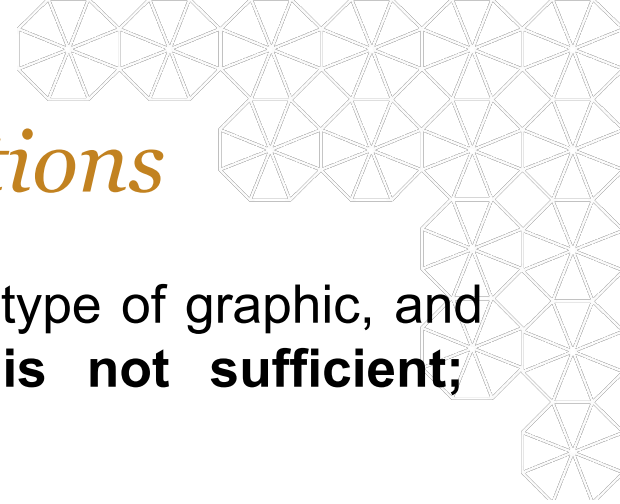
## *Working on generating Graphics*

### 3. Visual Mappings

It encodes fields of the records into visual or retinal properties of the selected mark.

The visual properties in Polaris are based on Bertin's retinal variables: **shape**, **size**, **orientation**, **color** (value and hue), and **texture** (not supported in the current version of Polaris).

property	marks	ordinal/nominal mapping	quantitative mapping
shape	glyph	○ □ + △ S U	
size	rectangle, circle, glyph, text		
orientation	rectangle, line, text		
color	rectangle, circle, line, glyph, y-bar, x-bar, text, gantt bar		



# *Performing Data Transformations*

Ability to rapidly change the table configuration, type of graphic, and visual encodings is necessary. However, **it is not sufficient; additional interactivity is needed.**

So,

- Users must be able to **sort, filter, and transform data** to uncover meaningful relationships.
- They must be able to form **ad hoc groups and partitions.**

Thus, technically, we must enable **deriving additional fields, sorting and filtering, brushing and tooltips, and undo and redo.**

# *Understanding Data Transformations*

## **1. Deriving Additional Fields**

This enables ability for:

- *Simple Aggregation:*
  - apply summation, average, minimum, and maximum to quantitative field.
- *Counting of Ordinal Dimensions:*
  - counting of distinct values for an ordinal field.
- **Partitioning**
  - Useful for encoding additional categorizations
- **Ad hoc Grouping**
  - allows the users to add their own domain knowledge to the analysis
- **Threshold Aggregation**
  - allows user to specify threshold values below which data is uninteresting.

# *Understanding Data Transformations*

## **2. Sorting and Filtering**

**Sorting** allows the user to uncover hidden patterns and trends and to group together interesting values by changing the order of values.

**Filtering** allows the user to choose which values to display so that he can focus on and devote more screen space

## **3. Brushing and Tooltips**

**Brushing** allows users to choose a set of interesting points by drawing a rubberband around them.

**Tooltips** allow user to view more details about point.

## **4. Undo and Redo**

**Users** can use the Back and Forward buttons on the top toolbar to either return to a previous visual specification or to move forward again.

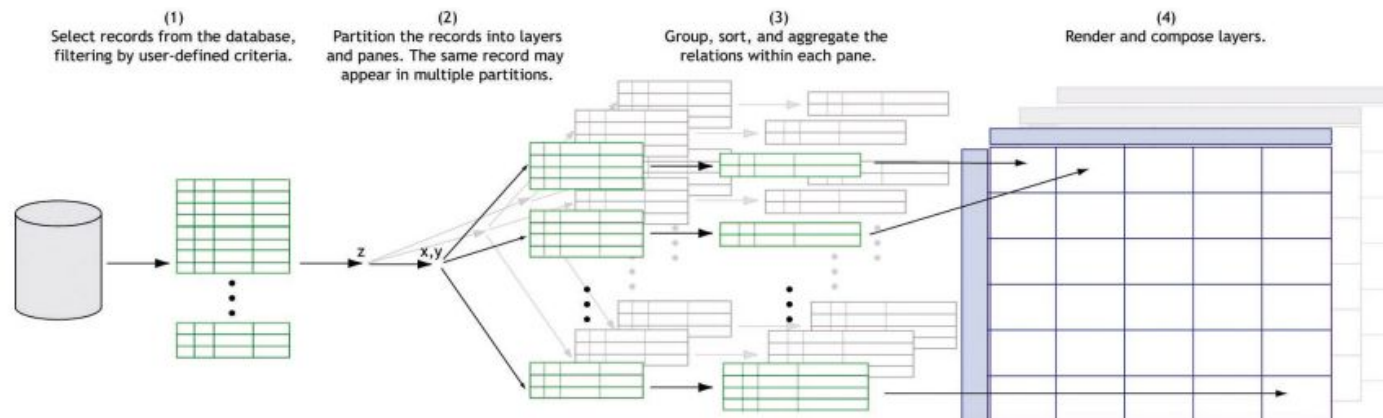


# Generating Database Queries for retrieval

After visual specification, retrieving data from DB requires queries, which may be done considering aspects like:

- Selecting the Records
- Partitioning the Records into Panes
- Transforming Records within the Panes

```
SELECT {dim},{aggregates}  
GROUP BY {G}  
HAVING {filters}  
ORDER BY {S}.
```

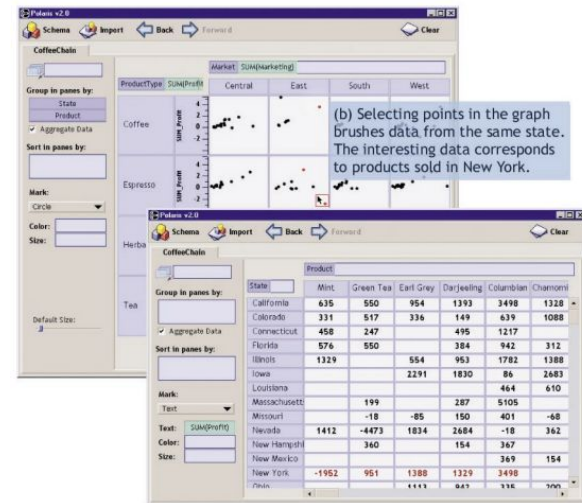
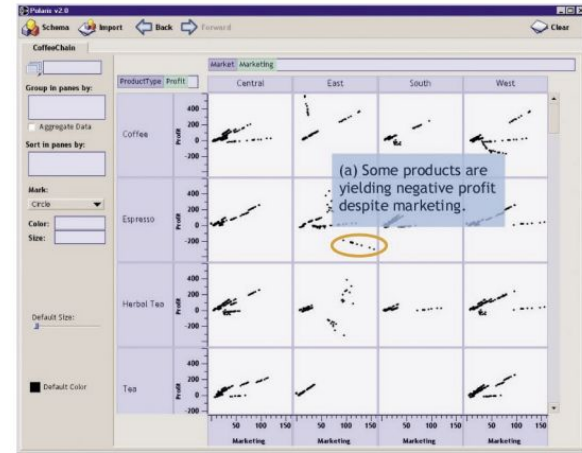


# Use case - Commercial Database Analysis

The chief financial officer (CFO) of a national coffee store chain has just been told to cut expenses. To get an initial understanding of the situation, the CFO creates a table of scatterplots showing the relationship between marketing costs and profit categorized by product type and market (Fig. 6a). After studying the graphics, the CFO notices an interesting trend: Certain products have high marketing costs with little or no return in the form of profit.

To further investigate, the CFO creates two linked displays: a table of scatterplots showing profit and marketing costs for each state and a text table itemizing profit by product and state (Fig. 6b). The two views are linked by the state field: If records are selected in either display, then all records with the same state value as the selected records are highlighted. The CFO is able to use these linked views to determine that, in New York, several products are offering very little return despite high expenditures.

The CFO then creates a third display (Fig. 6c): a set of bar charts showing profit, sales, and marketing for each product sold in New York, broken down by month. In this view, the CFO can clearly see that Caffé Mocha's profit margin does not justify its marketing expenses. With this data, the CFO can change the company's marketing and sales strategies in this state.

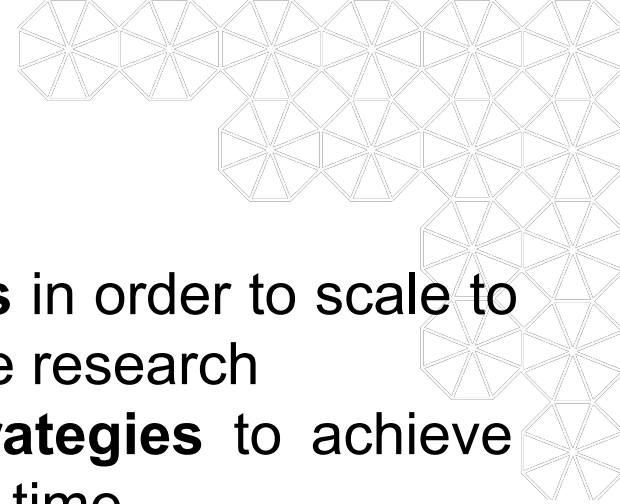


# *Evaluation & Why our approach is better?*

<b>Polaris</b>	<b>Wilkinson's System</b>
Focus on tool for multidimensional relational databases using relational algebra	Intentionally uses data model that is not relational, citing shortcomings in relational model's support for statistical analysis
Our concatenation operation just performs partitioning	Blend operator performs both set union and partition the axes.

Overall, whether the system is better than Wilkinson's is hard to judge completely, and will require more experience using systems to solve practical problems.

However, one major advantage is it **leverages existing database systems** and is therefore very **easy to implement**.

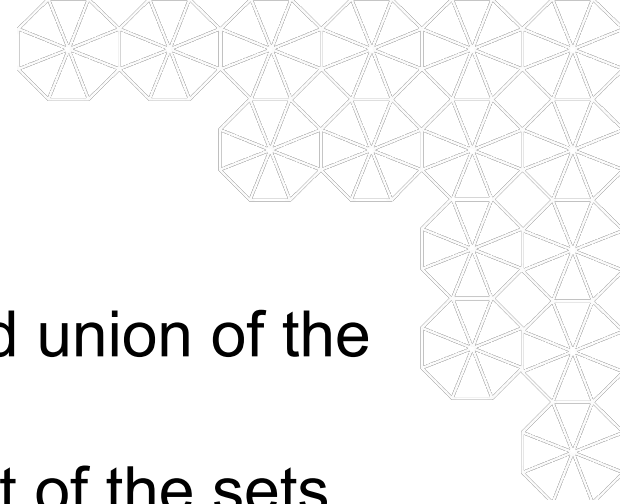


## *Future Work*

- Exploring **database performance issues** in order to scale to much larger data sets as part of our future research
- Exploring **prefetching and caching strategies** to achieve real-time interactivity and faster response time.
- Leverage **direct correspondence of graphical marks to tuples** in relational databases to generate database tables from a selected set of graphical marks.



# Appendix



# Table Algebra

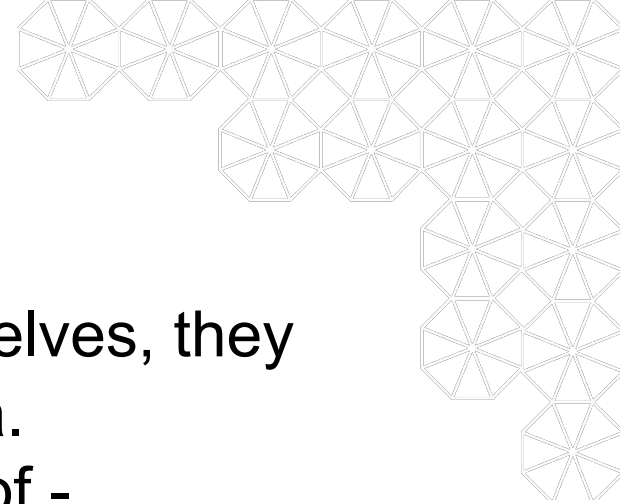
- Concatenation(+): performs an ordered union of the sets.
- Cross(x): performs a Cartesian product of the sets.
- Nest(/): similar to cross operator, but it only creates set entries for which there exist records with those domain values.

$$\begin{aligned} A \times B &= \{a_1, \dots, a_n\} \times \{b_1, \dots, b_m\} \\ &= \{a_1 b_1, \dots, a_1 b_m, \\ &\quad a_2 b_1, \dots, a_2 b_m, \dots, \\ &\quad a_n b_1, \dots, a_n b_m\} \\ A \times P &= \{a_1, \dots, a_n\} \times P \\ &= \{a_1 P, \dots, a_n P\} \end{aligned}$$



# *Types of Graphics*

- allows user to flexibility construct graphics by specifying individual components of graphics
- by default, dimensions: are independent variables
- and measures: dependent variables.
- Types:
  - Ordinal-Ordinal
    - characteristic members: table of numbers or of marks encoding attributes of source record.
    - axis variables are independent of each other
    - task is to understand patterns and trends in some function.



# *Table Algebra*

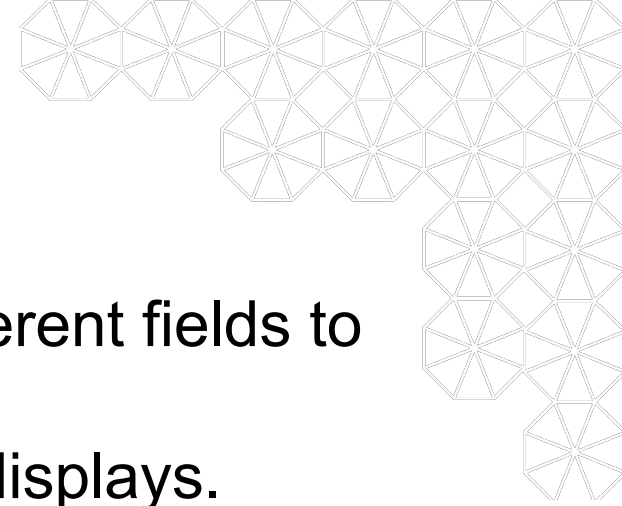
- when user places fields on the axis shelves, they implicitly create expressions in algebra.
- complete table configuration consists of -
  - two expressions define configuration of  $x$  and  $y$  axes of the table - row and column
  - define  $z$ -axis of table - layers
- operands: ordinal and quantitative fields
- ordinal fields: partition into rows and columns
- quantitative fields: spatially encoded as axes within panes
- operators: concatenation(+), cross( $x$ ) and nest(/)





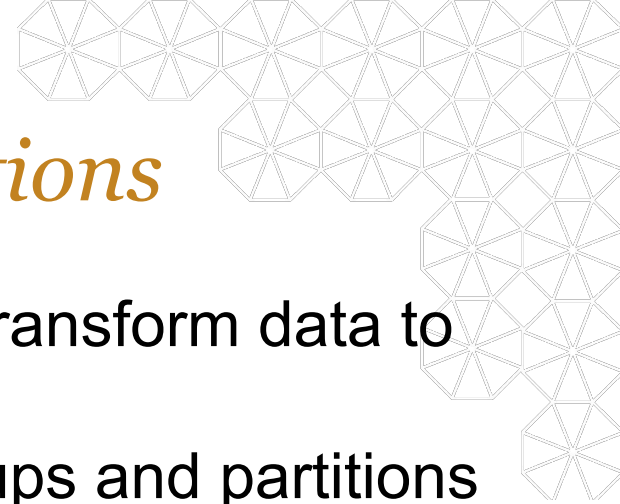
# *Types of Graphics*

- Types (contd.):
  - Ordinal-Quantitative
    - characteristic members: bar chart (clustered or stacked), dot plot and Gantt chart.
    - quantitative variable is dependent on ordinal variable.
    - task is to understand or compare the properties of some set of function.
  - Quantitative-Quantitative
    - used to understand distribution of data as a function of one or both quantitative variables.
    - discover causal relationships between two quantitative variables.



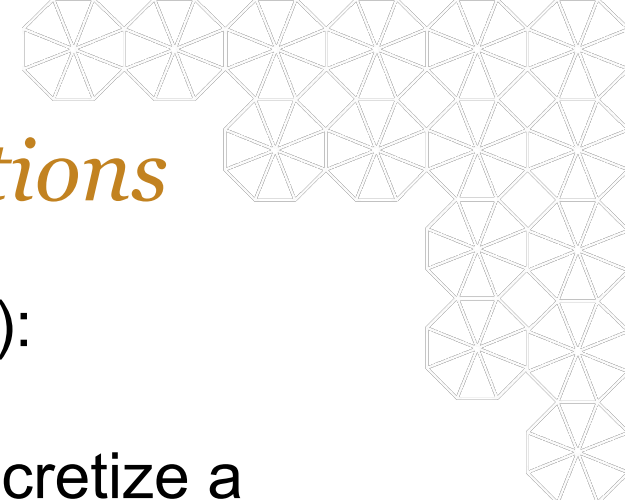
## *Visual Mappings*

- allowing users to explicitly encode different fields to retinal properties of display
- enhances data density and variety of displays.
- Types:
  - Shape: uses set of shapes recommended by Cleveland for encoding ordinal data and additional shapes.
  - Size: use to encode either ordinal or quantitative.
  - Orientation: for ordinal, the orientation needs to vary by at least  $30^\circ$  between categories. For quantitative, the orientation varies linearly with domain of the field.
  - Color: for ordinal, use predefined palette. For quantitative, vary only one psychological variable such as hue or value



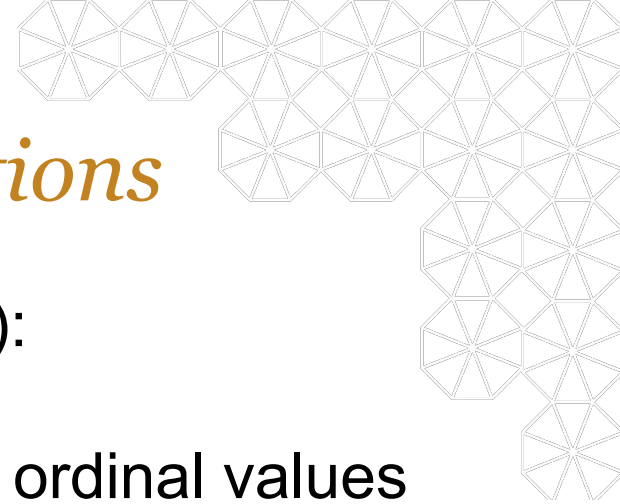
# *Performing Data Transformations*

- Users must be able to sort, filter, and transform data to uncover meaningful relationships.
- They must be able to form ad hoc groups and partitions
- Types of interaction techniques:
  - Deriving Additional Fields
    - Simple Aggregation: apply summation, average, minimum, and maximum to quantitative field.
    - Counting of Ordinal Dimensions: counting of distinct values for an ordinal field; applying CNT(count) operator changes the field type to quantitative and thus changing table config. and graph type in each pane.



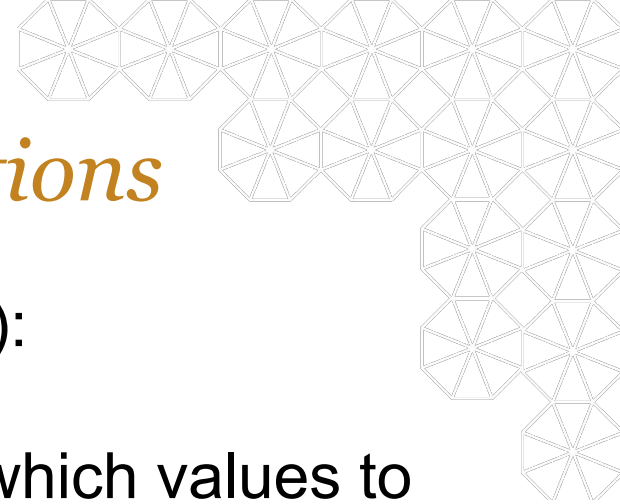
# *Performing Data Transformations*

- Types of interaction techniques (contd):
  - Deriving Additional Fields
    - Discrete Partitioning: used to discretize a continuous domain.
      - Methods: Binning and Partitioning.
      - Binning: specify a regular bin size in which to aggregate data; does not change the graph type as resulting field is also quantitative. Useful for creating histograms.
      - Partitioning: specify size and name of bin; partition of quantitative field will give ordinal. Useful for encoding additional categorizations



# *Performing Data Transformations*

- Types of interaction techniques (contd):
  - Deriving Additional Fields
    - Ad-hoc grouping: group different ordinal values
      - this is powerful as it allows the users to add their own domain knowledge to the analysis.
      - derives an ordinal field from ordinal field
      - graph type does not change
    - Threshold Aggregation: derived from ordinal and quantitative field.
      - if quantitative field  $<$  threshold value for any values of ordinal field, those values are aggregated to form “Other” category.
      - allows user to specify threshold values below which data is uninteresting.



# *Performing Data Transformations*

- Types of interaction techniques (contd):
  - Sorting and Filtering
    - Filtering: allows user to choose which values to display
      - Ordinal fields: listbox with all possible values to check and uncheck to display.
      - Quantitative fields: a dynamic query slider.
      - Filter changes the interpretation of the field in the algebra.
  - Brushing and Tooltips
  - Undo and Redo