SEEDB**: Efficient Data-Driven Visualization**  
**Recommendations to Support Visual Analytics**

**Names:** Nitin Kishore Sai Samala (30168428)

Dung Thai (30522685)

# Brief Introduction

Visualizing high dimensional datasets can be laborious. Given a new dataset like the **Census Data** or a new question about an existing dataset, an analyst builds various visualizations to get a feel for the data, to find anomalies and outliers, and to identify patterns that might merit further investigation. A visual recommendation like **SeeDB** intelligently explores the space of visualizations, evaluates promising visualizations for trends, and recommends those it deems most “useful” or “interesting”.

# Relevant Terms

* Database D with a snowflake schema
* Dimension Attributes A, that we would like to group-by in our visualizations
* Measure attributes M, that we would like to aggregate in our visualizations
* F, the set of potential aggregate functions over the measure attribute
* Query Q, to explore a subset of data

We assume that we can group D along any of the dimension attributes A and we can aggregate any of the measure attributes M and the resulting two-column table can be easily visualized via standard visualization mechanisms, such as bar charts or trend lines. The goal of SEEDB is to recommend visualizations of Q that have high utility. The class of queries Q posed over D that we support encompasses a general class of queries that select a horizontal fragment of the fact table and one or more-dimension tables. We can view this as a simple selection query over the result of joining all the tables involved in the snowflake schema

# Methodology

Q can select any subset of records from the Census table resulting in . Visualization can be translated into an aggregate over group-by query on the underlying data and is represented as a function represented by where .

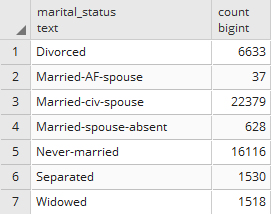
represents the results of grouping the data in D by a, then aggregating the m values using f;

represents a similar visualization applied to the data in which is compared to a reference dataset (called ) to see if it has high utility. The reference dataset may be defined as the entire underlying dataset (D), the complement of or data selected by an arbitrary query,

The result of the view queries view queries are summaries with two columns, namely a and f(m). To ensure that all aggregate summaries have the same scale, we normalize each summary into a probability distribution (i.e. the values of f(m) sum to 1).

Utility of is the distance between the probability distributions for target view () and the reference view ()

In this project



We grouped **[Married-AF-spouse, Married-civ-spouse, Married-spouse-absent, Separated]** as married group and **[Never-married, Divorced, Widowed]** as the unmarried group which is our reference data

We do an exhaustive search over the combinations of (group by attribute 'a', measure 'f', aggregate function f) and find the combinations that maximize the distance between the distribution of the target group (married people) and the distribution of the reference group (unmarried people). We intelligently merge and batch queries, reducing the number of queries issued to the database minimizing the number of scans of the data since the queries only differ in attributes used for grouping and aggregation. Regarding aggregation we stick to using only **avg, sum, min, max and count.**

# Measurement attributes

1. Age
2. Fnlwgt
3. Hours per week
4. Capital\_gain
5. Capital\_loss

­

# Dimension attributes

1. workclass
2. education (education\_num is the encoding for this)
3. occupation
4. relationship
5. race
6. sex
7. native\_country
8. economic\_indicator

We also applied the following optimizations

**Combine Multiple Aggregates:**

Here we rewrite aggregate views with same group-by attribute as

requiring just execution of two queries

**Confidence Interval-Based Pruning:**

During each phase, we keep an estimate of the mean utility for every aggregate view and a confidence interval around that mean. If the upper bound of the utility of view is less than the lower bound of the utility of k or more views, then is discarded