Optimizing Business Intelligence Dashboards: A Multi-Stage Framework and Unified Performance Metric

Dev Dani, Hayato Yamana
Waseda University
Tokyo, Japan
dev@toki.waseda.jp, yamana@yama.info.waseda.ac.jp

***Abstract*—Slow dashboard response can interrupt the filtering, comparison, and investigation required in executive analysis. Existing optimization techniques address individual data, query, or visualization costs but do not provide a common way to evaluate a BI visual across data refresh, initial rendering, and repeated interaction. This paper presents a cumulative, per-visual optimization framework supported by a workload performance score. The framework applies three cumulative stages (source representation, semantic model, and visual-specific aggregation or sampling), rejects candidates that change the required analytical output, and selects the highest-scoring valid stage for each visual. A controlled Power BI case study uses 8,549,307 New York City taxi records with three trials per configuration and a five-configuration Stage 2 ablation. Refresh time decreased from 236.07 s for the CSV baseline to 49.99 s after Stage 3, with narrow source projection producing the largest isolated reduction. Stage 3 produced the highest median workload score for four visual groups; the KPI group fell within the Stage 2 and Stage 3 trial-range overlap. For the dense scatter plot, Stage 3 sampling reduced the median worst interaction time by 68.12% and increased the workload score by 130.19% relative to Stage 2. Sensitivity analysis showed that the selected mapping changes with the assumed interaction and refresh frequencies, supporting workload-specific, per-visual evaluation.**

***Index Terms*—business intelligence, dashboard performance, interaction latency, Power BI, performance evaluation, visual analytics**

I. Introduction

Business intelligence (BI) dashboards provide a direct connection between organizational data and the people responsible for acting on it. Executives use them to monitor key performance indicators, compare business units, investigate unexpected changes, and review the information behind a summary value. Their usefulness therefore depends on more than presenting the correct numbers. The information must also remain accessible while a decision is being formed. Yigitbasioglu and Velcu [1] describe dashboards as tools that can improve decision making by supporting human perception and reducing information overload. Sarikaya et al. [2] also show that dashboards are used across a wide range of decision-making settings and have become a common point of access to large data sources.

This role makes slow visual response a practical problem in executive workloads. A dashboard is rarely viewed only once as a static report. Users apply filters, change time periods, move from summaries to detailed views, and compare several measures before reaching a conclusion. When each action is followed by buffering or a delayed visual, this sequence is interrupted. The effect of latency is not limited to inconvenience. Liu and Heer [3] found that an additional delay of 500 milliseconds reduced user activity, dataset coverage, and the rate at which participants formed observations and hypotheses during exploratory visual analysis. Lowe and Matthee [4] similarly identify interactivity, scalability, and fast retrieval as central requirements for big data visualization tools. These findings suggest that slow visuals can influence both the pace and depth of analysis, which weakens the value of a dashboard as a decision-support tool.

Improving this experience is difficult because dashboard performance is produced by several connected operations. Data must be stored, prepared, loaded into a model, queried, aggregated, and rendered before a visual can respond. Research has proposed valuable techniques for individual parts of this process. Park et al. [5], for example, propose visualization-aware sampling to reduce the amount of processed data while attempting to preserve important visual structure. Database and query optimization address other sources of delay. However, evaluating one operation in isolation does not show whether the combined refresh, rendering, and interaction workload has improved. A short rendering time may hide a costly data-loading process, while a faster initial load may provide little benefit if filters remain slow during repeated use. What remains insufficiently addressed is a practical methodology that evaluates these costs together and helps dashboard users decide how optimization techniques should be applied as a coordinated process.

The contribution of this research does not lie in proposing new storage, modeling, aggregation, or sampling techniques. These are established approaches. The methodological contribution lies in treating them as cumulative candidates that are measured under one workload and selected separately for each visual after analytical validity has been checked. This changes the output of optimization from one assumed final dashboard configuration into a mapping between visuals and their selected valid stages.

The methodology has two connected components. Its primary component is a staged evaluation and selection pro-

cess that organizes established techniques into representation, model, and visual-specific layers. The supporting workload performance score combines amortized refresh cost, initial rendering, and repeated interaction so that these timing components are assessed together rather than in isolation. Source row count is recorded as scale context, but the present fixed-scale case study does not establish cross-scale comparison. Together, these components are intended to identify meaningful performance improvements rather than optimize one timing value or assume that the most aggressive stage is universally preferable. The practical objective is to produce faster and less disruptive visual interaction for executives working with large dashboards. The present study evaluates the methodology in Microsoft Power BI. Evaluation in other BI tools is left for future work. The current paper therefore does not assume that every implementation detail will transfer unchanged across platforms.

II. Related Work

Dashboard-performance research spans dashboard use, interactive workload evaluation, and the data systems beneath analytical interfaces. This section focuses on how those areas define responsiveness and where their evaluation boundaries differ from the present work.

A. Dashboard Use and Interaction Latency

Yigitbasioglu and Velcu [1] describe dashboards as performance-management tools whose value depends on matching presentation and interaction to user needs. Sarikaya et al. [2] show that dashboards cover analytical and operational displays with different purposes and interaction levels. Performance evaluation must therefore distinguish occasional monitoring from repeated filtering and exploration.

Lowe and Matthee [4] identify interactivity, fast retrieval, scalability, and data reduction as central requirements for big data visualization. Liu and Heer [3] demonstrate that latency reduces exploration and the formation of observations and hypotheses. These studies establish the analytical importance of responsiveness, but they do not combine refresh, initial-rendering, and interaction costs within one BI report workload.

B. Decision-Support and Interactive Workload Evaluation

TPC-H [6] combines query power and throughput at a controlled scale. TPC-DS [7] broadens the workload to query response, multiuser throughput, and maintenance, while TPC-DI [8] evaluates warehouse-loading throughput. These benchmarks provide reproducible system evaluations, but their boundary does not include visual rendering, dashboard interaction, or per-visual optimization selection.

Battle et al. [9] call for visualization benchmarks with scalable data, realistic tasks, interface actions, and interpretable measures. IDEBench [10] evaluates incremental exploration through generated data and query sequences. SIMBA [11] generates goal-driven dashboard interactions for DBMS comparison. Together, these approaches show that interactive workloads should represent related actions rather than isolated queries.

VegaPlus [12] coordinates Vega and DBMS execution by rewriting transformations and selecting interaction-aware plans. It is the closest integrated pipeline, but its output is an execution plan. The present work instead evaluates practitioner-applied changes to an existing BI report and returns an analytically valid stage for each visual.

TABLE I. Comparison with Related Evaluation Approaches

Approach	Evaluated workload	Evaluation output
TPC-H/DS [6], [7]	Query response, throughput, maintenance, and controlled scale	Standardized system result
TPC-DI [8]	Data transformation and warehouse loading	Integration throughput
IDEBench [10]	Incremental exploration queries and result quality	Query-engine comparison
SIMBA [11]	Goal-driven dashboard query sequences	DBMS comparison
VegaPlus [12]	Vega and DBMS dataflow execution	Interaction-aware plan
This work	Refresh, visual load, action sequence, and analytical validation	Stage retained per visual

Table I separates standardized system benchmarks, interactive engine comparisons, execution-plan selection, and the present report-level evaluation. The proposed contribution is the combination of cumulative practitioner-applied stages, a workload score, analytical validation, and per-visual stopping. Its boundary begins with source refresh and continues through initial rendering and predefined user actions. It does not evaluate gateway traffic, network delivery, or concurrent enterprise use.

C. Data Organization and Query Acceleration

Abadi et al. [13] show that column stores reduce analytical cost through restricted data access, compression, and specialized execution. Gray et al. [14] organize reusable aggregates across dimensions to avoid repeated scans. These principles support the source-representation and pre-aggregation stages evaluated here.

imMens [15] uses binned aggregation and data tiles, while Nanocubes [16] uses a hierarchical structure for large spatiotemporal aggregates. Both demonstrate the value of organizing data around visual queries, although their specialized architectures are not directly applicable to ordinary self-service BI reports.

D. Approximate and Progressive Processing

BlinkDB [17] trades accuracy for response time through sampled SQL results. Park et al. [5] instead select samples using visual fidelity, since aggregate accuracy alone may not preserve shapes or outliers. Stolper et al. [18] expose intermediate results during computation. These methods improve responsiveness by reducing data or changing when results appear, but they also require semantic or visual validation.

The present framework combines that validation with refresh, rendering, and interaction measurements to select a stage for each visual.

III. Problem Formulation and Evaluation Metric

A. Problem Definition

Let D denote a BI dashboard constructed from a source dataset containing S rows, and let v denote one of its visuals. The dashboard is evaluated at a baseline and after each optimization stage k. The problem is to determine whether a stage improves responsiveness under the same analytical workload. The source scope, requested calculation, filters, and meaning of the displayed values must remain consistent across stages. A faster visual is not considered an improvement if it answers a simpler question or omits information required by the user.

Performance is evaluated primarily at the visual level because dashboard visuals differ in their queries and rendering behavior. Dashboard-level analysis provides a secondary summary of the most restrictive response times.

B. Workload Performance Score

Dashboard delay spans data preparation, initial rendering, and repeated user actions; thus one timing value can misrepresent the cost experienced during use. Summarizing several performance components through an interpretable measure has precedent in other domains. Apdex [19] reports transactional response through a common index, while Mok et al. [20] relate three buffering measures to a streaming-media quality estimate. The proposed score applies this general idea to the timing components of a BI workload while recording the represented row count:

$$\mathrm{Score}_{v,k} = \frac{\log_{10}(S)}{\frac{L_k}{N} + V_{v,k} + \alpha I_{v,k}}. \tag{1}$$

Here, $S > 1$ is the integer number of source rows represented by the task. It remains constant because every stage represents the same workload, even when an internal table is reduced. L_k is the dataset load or refresh time at stage k. $N > 0$ is the expected number of sessions between refreshes, and L_k/N therefore amortizes the refresh cost. $V_{v,k}$ is the initial load time of visual v, and $I_{v,k}$ is its response time for a predefined filter or slicer interaction. The nonnegative parameter α is the expected number of these interactions per session. All times are nonnegative and are converted from milliseconds to seconds before calculation. The denominator must be positive.

For dashboard-level reporting, let $V_{\max,k} = \max_v V_{v,k}$ and $I_{\max,k} = \max_v I_{v,k}$. The dashboard bottleneck score is

$$\mathrm{Score}_{D,k} = \frac{\log_{10}(S)}{\frac{L_k}{N} + V_{\max,k} + \alpha I_{\max,k}}. \tag{2}$$

This is a conservative bottleneck summary rather than an estimate of the average dashboard experience. The per-visual score remains the primary basis for identifying how individual visuals respond to optimization. Relative improvement from baseline is reported as $\Delta_{v,k} = (\mathrm{Score}_{v,k}/\mathrm{Score}_{v,0} - 1) \times 100\%$.

C. Baseline Comparison Measures

Two simpler measures are retained for comparison. The first is the weighted workload latency represented by the denominator of the proposed score:

$$C_{v,k} = \frac{L_k}{N} + V_{v,k} + \alpha I_{v,k}. \tag{3}$$

The second is the largest single latency component:

$$M_{v,k} = \max\left(\frac{L_k}{N}, V_{v,k}, I_{v,k}\right). \tag{4}$$

Smaller values indicate lower latency for both baselines. $C_{v,k}$ represents the combined workload but does not include dataset scale. $M_{v,k}$ identifies the most restrictive individual delay, but it does not accumulate repeated interactions. Dashboard-level versions replace $V_{v,k}$ and $I_{v,k}$ with $V_{\max,k}$ and $I_{\max,k}$.

D. Interpretation and Comparison Conditions

A higher score indicates lower expected delay for the same analytical workload. With a positive denominator, decreasing any timing component increases the score. Since $\mathrm{Score}_{v,k} = \log_{10}(S)/C_{v,k}$, the proposed score and weighted latency produce identical stage rankings when S is fixed. The scale term therefore supplies no additional ranking information in the present single-dataset experiment. Base 10 expresses row count by orders of magnitude, but changing the logarithm base would only rescale every score by the same positive constant.

The intended distinction appears when a matched analytical workload is evaluated across controlled data scales. For example, a workload cost of $C = 20$ s produces a score of 0.300 for $S = 10^6$ and 0.350 for $S = 10^7$. The latency is identical, but the latter score records that the same workload cost was maintained while representing an additional order of magnitude. If the 10^7-row workload instead requires $C = 40$ s, its score falls to 0.175. The metric therefore expresses scale-relative workload efficiency rather than response time alone. This interpretation is appropriate only when the schema, analytical task, timing definitions, and interaction procedure are controlled. Row count by itself does not make unrelated dashboards or datasets comparable. This example illustrates the intended mechanics of the scale term; its empirical validation across controlled dataset sizes is left for future work.

The score is a comparative index without a universal acceptance threshold. Comparisons require consistent timing definitions, task, dataset scope, execution environment, and interaction procedure. The workload parameters N and α must be selected from the target usage scenario and reported with the experimental conditions. Finally, the score measures responsiveness and not correctness. Each optimized stage must

preserve the baseline values and analytical meaning. A stage that fails this validation is excluded regardless of its score.

IV. Proposed Optimization Framework

The framework applies cumulative changes from the source representation to visual-specific processing, as summarized in Fig. 1. Stage 0 preserves the original CSV source, model, measures, and visuals as the baseline. It establishes the timing and output values used to evaluate later candidates. Stages 1 and 2 apply broadly useful data and model changes, while Stage 3 selects smaller representations for visuals that can use them. The analytical task and predefined interaction workload are retained after each stage to keep the candidates comparable.

The sequence follows a progression from changes with broad scope to changes with narrow scope. Abadi et al. [13] describe source-level columnar techniques that affect analytical access across the workload. Kimball and Ross [21] describe dimensional organization shared by report queries. Gray et al. [14] and Park et al. [5] describe aggregation and sampling techniques whose suitability depends on query grain or visual purpose. Source representation therefore precedes shared model design, which precedes visual-specific reduction. The individual implementations may differ across tools, but this broad-to-specific ordering provides a reusable analytical optimization sequence rather than a sequence created only for the present dashboard.

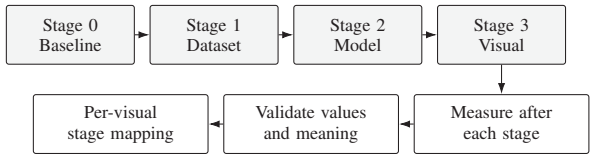


Fig. 1. Cumulative optimization and per-visual stage selection

A. Stage 1: Dataset Representation

The first stage changes the source representation without altering dashboard logic. The baseline CSV data is converted to Parquet, data types are assigned explicitly, and columns that do not support a relationship, measure, filter, tooltip, or displayed field are removed. Explicit typing avoids repeated interpretation of text-based values and allows the storage engine to handle dates and numeric fields consistently. Column pruning also prevents attributes that are irrelevant to the dashboard from being read and compressed. These changes follow the broader advantages of column-oriented storage for analytical workloads described by Abadi et al. [13].

Automatic date hierarchies, including Power BI's Auto Date/Time feature, are disabled when an explicit date dimension is available. This prevents hidden date structures from increasing model size or introducing alternate filter paths. The resulting Parquet source is treated as another representation of the same analytical data rather than a revised dataset. Row counts, key fields, and the inputs used by baseline measures are therefore checked before the candidate proceeds to Stage 2.

B. Stage 2: Model and Calculation Design

Following the dimensional-modeling principles of Kimball and Ross [21], the second stage reorganizes the model as a star schema that separates trip-level facts from descriptive dimensions. The implemented model retains `taxi` as the trip-grain fact table and adds four dimensions for date, payment type, pickup location, and pickup hour. Each dimension has an active one-to-many relationship to the fact table with single-direction filtering. This structure gives report filters a predictable path and avoids using descriptive fact-table columns directly in the visuals.

The date and hour keys are prepared in Power Query from the pickup timestamp because their values are deterministic and do not need to be recalculated during interaction. An explicit Q1 2025 date dimension replaces automatic date hierarchies, and Power BI's Auto Date/Time feature remains disabled. Data Analysis Expressions (DAX), Power BI's formula language, are checked for repeated work and unnecessary row iteration. The implemented report retains four reusable benchmark measures, and the average-fare measure is stored as a numeric value with currency formatting rather than as formatted text. Microsoft [22] recommends Boolean filter arguments when their semantics permit. For example, a card-payment measure can pass `FactTrips[PaymentType]=''Card''` directly to `CALCULATE` instead of constructing a row-by-row `FILTER` table. Complex conditions retain `FILTER` when required. The slicers and visual axes are then rebound to dimension fields while the analytical layout is preserved.

The recorded Stage 2 build also narrows the Parquet projection to the six source columns required by the benchmark and corrects the pickup-date derivation to use the pickup timestamp consistently. These changes are necessary for the valid implemented model, but they mean the primary Stage 2 comparison represents the complete cumulative configuration. A five-configuration ablation therefore begins with a corrected flat wide control and adds narrow projection, prepared date and hour fields, the star schema, and the revised DAX measure in sequence. This design separates the incremental effects within the implemented order instead of assigning the complete Stage 2 improvement to the star schema or DAX alone.

C. Stage 3: Pre-Aggregation and Visual-Specific Reduction

The third stage creates smaller inputs matched to the fields and granularity required by each visual. Pre-aggregated tables are prepared at frequently requested levels, such as hourly or daily intervals and commonly filtered categories. A visual is directed to the smallest table that contains its dimensions, measures, and supported interactions. For example, a daily time-series chart can use a daily summary, while a detail table that exposes individual trips must retain the original grain. In Power BI, this routing can use the Manage aggregations feature when supported by the model's storage mode. This follows the data-cube principle described by Gray et al. [14].

Reduction is selected by visual purpose. KPI cards and exact totals use complete aggregates. Time-series charts and

heatmaps can use summaries at the displayed resolution, while dense scatter plots may use a sample when additional marks add cost without distinguishable structure. Sampling is accepted only when the required pattern and meaning are preserved, following the visualization-aware evaluation of Park et al. [5]. Stage 3 candidates coexist with the full model rather than replacing it globally. A visual may therefore retain an earlier valid stage instead of using every Stage 3 technique.

D. Evaluation-Guided Selection

After every stage, the same refresh, initial-load, and interaction workload is executed and the per-visual score is recalculated. Functional validation is performed before a score can be used for selection. KPI values, displayed categories, filter responses, and the meaning of each visual must remain consistent with the baseline. A candidate that is faster but fails this check is rejected.

Let $\mathcal{V}$ be the visual set and $\mathcal{K} = \{0, 1, 2, 3\}$ the candidate stages. For each pair (v, k), where $v \in \mathcal{V}$ and $k \in \mathcal{K}$, the framework takes the measured inputs $(S, L_k, V_{v,k}, I_{v,k}, N, \alpha)$ and a Boolean validation variable $A_{v,k} \in \{0, 1\}$. The value is 1 when stage k preserves the required output and analytical purpose of visual v, and 0 otherwise:

$$A_{v,k} = \begin{cases} 1, & \text{analytically valid,} \\ 0, & \text{otherwise,} \end{cases} \tag{5}$$

The admissible set for each visual is

$$\mathcal{K}_v = \{k \in \mathcal{K} : A_{v,k} = 1\}. \tag{6}$$

Selection is then defined by

$$k_v^* = \arg\max_{k \in \mathcal{K}_v} \mathrm{Score}_{v,k}. \tag{7}$$

If two candidates are equal at four decimal places, the lower stage is retained to avoid adding later-stage complexity without a measured benefit. The framework returns the mapping $\mathcal{M}(D) = \{(v, k_v^*) : v \in \mathcal{V}\}$. This mapping is the main output. The dashboard bottleneck score reports the slowest visual response under each stage, but stage selection remains based on the per-visual scores.

A pre-aggregated Stage 3 table may add refresh work while reducing filter latency, which the L_k/N term evaluates over repeated sessions. Sampling may reduce $V_{v,k}$ for a dense scatter plot, but only if its visual structure remains meaningful. Equations 6 and 7 therefore separate semantic acceptance from performance selection. They also allow one dashboard to retain candidates from different stages instead of assuming that the last or most aggressive stage is always preferable.

V. Experimental Methodology

A. Evaluation Objectives and Experimental Design

The evaluation is designed to test the two components of the proposed method. It examines whether cumulative optimization produces different valid stage selections across visuals and whether the proposed score distinguishes workload effects that are not represented by a single maximum-latency value. Four research questions organize the experiment. RQ1 evaluates cumulative performance and per-visual stage selection. RQ2 identifies the contributions within Stage 2. RQ3 compares the proposed score with weighted workload latency C and maximum component latency M. RQ4 tests whether the selected mapping changes with the workload assumptions N and α.

The primary comparison uses four cumulative configurations corresponding to the framework in Section IV. Stage 0 is the original CSV baseline. Stage 1 changes the source representation to Parquet and applies explicit typing and column pruning. Stage 2 adds the dimensional model, prepared fields, and revised measures. Stage 3 adds visual-specific aggregates and sampling. The source records, analytical questions, visual encodings, and five interaction states are retained while the configuration is varied. Three Performance Analyzer traces were collected for each configuration. This comparison determines whether one final configuration is suitable for every visual or whether different visuals retain different valid stages under the same workload.

A separate five-configuration ablation evaluates the cumulative increments within Stage 2 using the same three-run protocol. Its objective is to identify which source, field-preparation, dimensional-modeling, and measure changes account for the Stage 2 result and whether their effects differ across visuals. The primary unit of analysis is a configuration–visual pair. A dashboard-level analysis is also produced to represent the delay experienced when several visuals respond to the same action.

B. Dataset and Benchmark Dashboard

The benchmark uses the Yellow Taxi Trip Records published by the New York City Taxi and Limousine Commission [23]. The frozen experimental dataset contains the January through March 2025 records after a single cleaning process, giving 8,549,307 trips. Cleaning is completed before any optimization is applied, and every stage is derived from the same retained records. The source row count S therefore remains fixed across stages even when an internal aggregate or sample is used to answer a visual query.

The cleaning script processes 500,000-row chunks and parses pickup and drop-off timestamps before filtering. It retains records with a 2025 pickup, a nonnull passenger count above zero, trip distance in $(0, 100]$ miles, fare amount in $[0, 1000]$, total amount in $(0, 1500]$, and nonnegative tip amount. Pickup must precede drop-off, and trip duration must be in $(0, 360]$ minutes. These rules are applied once to create the frozen CSV and Parquet inputs rather than being repeated inside the report files.

The selected fields support both temporal and categorical analysis. They include pickup time, payment type, trip distance, total amount, and pickup location. The report contains three slicers for pickup date, payment type, and trip distance. Its analytical visuals are grouped into four KPI cards, a revenue and trip-count time series, a trip-distance scatter

plot, a pickup-location bar chart, and a day-by-hour matrix. The KPI cards are treated as one visual group because they answer a shared summary task. In addition to the complete dashboard, each group is placed on a separate report page with the same applicable slicers. This produces five visual-level evaluation units and one dashboard-level unit without changing the underlying analytical task.

The four baseline measures are defined as trip-row count, sum of total amount, average fare amount, and average trip distance. Their DAX expressions are `COUNTROWS(taxi)`, `SUM(taxi[total_amount])`, `AVERAGE(taxi[fare_amount])`, and `AVERAGE(taxi[trip_distance])`. Later measures may change their execution path, but the requested quantities remain the same.

C. Environment and Experimental Conditions

All measurements were collected using 64-bit Microsoft Power BI Desktop version 2.157.879.0 (August 2026) on Windows 11 Pro, version 25H2, using the same workstation. The system uses an AMD Ryzen 7 3700X processor at 3.6 GHz, 32 GB of 3200 MHz memory, an NVIDIA GeForce RTX 2080 Super graphics card, and a 512 GB M.2 SSD. The same operating-system configuration, Power BI Desktop installation, display settings, and storage mode were used throughout data collection.

Four report files represent the primary cumulative conditions described in Section IV. Stage 0 loads the baseline CSV source and retains the original model, measures, and visuals. Stage 1 replaces the source representation with Parquet, assigns explicit data types, removes unused columns, and disables automatic date structures. Stage 2 retains the trip-grain fact table, adds four dimensions, prepares date and hour keys, revises the benchmark measures, and rebinds report fields to the dimensions. Stage 3 retains the Stage 2 model and adds exact aggregates and a sampled input for visual-specific routing. The analytical questions and visual layout remain fixed. Model fields and measure implementations change only when required by the optimization.

Five additional report files form the Stage 2 ablation. Configuration A is a corrected flat wide control that retains the original wide source projection. Configuration B imports only the six source columns required by the benchmark. Configuration C creates pickup date and pickup hour in Power Query rather than through row-level DAX. Configuration D adds the four dimensions, relationships, and dimension-based report bindings. Configuration E adds the final numeric average-fare measure and formatting. Each configuration includes all changes from the preceding configuration. The comparison therefore estimates the contribution of each increment conditional on the earlier changes, while A-to-E represents the complete Stage 2 pipeline.

D. Stage 3 Construction

Stage 3 retains the complete 8,549,307-row fact table for unsupported filters and detail-level queries. DuckDB 1.5.5 generated five frozen Parquet inputs from the same source. The daily aggregate contains 361 rows at pickup-date and payment-type grain. The pickup-location aggregate contains 44,687 rows, and the day-hour aggregate contains 8,593 rows. A separate trip-distance dimension contains 5,319 distinct values. These exact structures serve the KPI cards, time series, pickup-location bar chart, and day-hour matrix at the grain required by each visual.

The dense scatter plot uses a deterministic hash sample containing 436,365 rows, or 5.1041% of the fact table. The base sampling rate is 5%. Rows with an absolute trip distance of at least 30 or an absolute total amount of at least 175 are always retained to preserve the defined extreme observations. The scatter keeps the same axes and pickup-time series used in the preceding stage. A generation manifest records the source and output SHA-256 hashes, file sizes, row counts, sampling rule, and reconciliation totals.

The existing date, payment, location, and hour dimensions are connected to the new aggregate tables through active one-to-many relationships with single-direction filtering. A shared trip-distance dimension replaces the direct fact-table field in the slicers. Stage 3 measures use an aggregate only when its grain supports the active filter context. If a trip-distance filter or another unsupported dimension is active, the measure falls back to the complete fact table. The sum of trip counts in each exact aggregate was reconciled to 8,549,307 before evaluation. Revenue, fare, and distance totals in the daily aggregate were also reconciled within a floating-point tolerance below 0.01.

E. Timing Protocol

Dataset load time L_k is measured from the start of a full data refresh until Power BI reports completion. These measurements are recorded separately from the visual traces and are treated as configuration-level inputs. Microsoft Power BI Performance Analyzer [24] records visual duration in milliseconds and captures reloads caused by slicer actions. It is used to collect the visual and interaction measurements. The retained traces use Performance Analyzer JSON schema version 1.1.0.

For each of the nine configurations, three traces were recorded. Recording begins on a neutral control page before the benchmark page is opened. The first benchmark-page lifecycle supplies $V_{v,k}$ for that trial. A Performance Analyzer refresh is then recorded as a warm observation and excluded from $V_{v,k}$. The five scored interactions follow in the same order and use the same values in all 27 traces. Each action is completed before the next begins.

TABLE II. FIXED INTERACTION SEQUENCE USED IN EVERY TRACE

Step	User action
1	Set the trip-distance maximum to 50 miles
2	Reduce the trip-distance maximum to 25 miles
3	Select Credit card
4	Set the pickup-date endpoint to February 2, 2025
5	Select Friday at pickup hour 1 in the heat map

The DAX queries in the traces confirm all five selections. The sequence and selected values are identical across configurations. Line-chart highlighting and clearing transitions are excluded because they are outside the fixed five-action workload.

The retained Performance Analyzer traces include action times, lifecycle events, visual identifiers, and generated DAX queries. These fields allow every reported visual value to be linked to a configuration, trial, and action without making the file-naming convention part of the methodology.

F. Score Calculation and Validation

Recorded milliseconds are converted to seconds before the score in Section III is calculated. Within each trial, $V_{v,k}$ is the initial lifecycle duration and $I_{v,k}$ is the worst response of visual v among the five actions. A score is calculated for every configuration–visual–trial combination using the configuration-level refresh input. The three trial values are summarized by their median, and the observed minimum and maximum are retained as a measure of runtime variation. Dashboard-level reporting first takes the slowest visual-load and interaction values within each trial, $V_{\max,k}$ and $I_{\max,k}$, before calculating the trial score.

The term $\alpha I_{v,k}$ is not the observed sum of the five action times. Since $I_{v,k}$ is the longest response among them, the primary setting $\alpha = 5$ treats all five interactions as if each incurred that response. It therefore gives a conservative upper bound for the interaction portion of the recorded five-action session. This model emphasizes repeated exposure to the slowest interaction and permits the assumed interaction count to be varied without changing the measured timing input. The score should consequently be interpreted as a worst-interaction workload comparison rather than an exact reconstruction of total session latency.

The primary workload uses $\alpha = 5$ because one benchmark session contains the five predefined interactions in Table II. It uses $N = 10$ to represent the evaluated operating scenario in which one dataset refresh supports ten dashboard sessions. These values are workload assumptions rather than universal defaults. They are applied consistently to every configuration, and the sensitivity analysis recalculates all three trials before taking the median under alternative settings.

Performance was evaluated together with analytical consistency. For the exact visuals, outputs from the optimized structures were compared with the baseline under the benchmark filter states. The KPI, line-chart, bar-chart, and matrix outputs matched the baseline across those states. For the sampled scatter plot, the full and reduced views were compared in terms of association, dominant density regions, and relevant outliers. The sampled view retained these features. These checks confirmed that the selected stage for each visual preserved its analytical purpose.

VI. Results

Initial rendering and interaction values are medians of three trials, with the observed minimum and maximum reported as ranges.

A. RQ1: Cumulative Performance and Per-Visual Selection

1) Refresh effect: The two Stage 0 full-refresh observations were 221.69 s and 250.45 s, giving a median of 236.07 s. The recorded full-refresh times for Stages 1, 2, and 3 were 141.84 s, 51.72 s, and 49.99 s. Stage 1 reduced the baseline median by 39.92%, while Stage 2 reduced the Stage 1 value by a further 63.54%. Stage 3 was 3.34% lower than Stage 2 and 78.82% lower than the baseline median. Of the 186.08 s reduction between the baseline median and Stage 3, 184.35 s, or 99.07%, had already been achieved by Stage 2.

2) Initial rendering: Table III reports the initial lifecycle duration for each visual group. The KPI value is the slowest lifecycle among the four cards within each trial.

TABLE III. Initial Visual-Load Time: Median and Three-Run Range

Visual group	S0 (s)	S1 (s)	S2 (s)	S3 (s)
KPI cards	2.120	1.996	1.815	2.117
range	1.851–2.292	1.972–2.444	1.815–2.112	2.011–2.158
Revenue trend	2.185	2.047	1.896	2.179
range	1.886–2.375	2.030–2.489	1.880–2.193	2.096–2.228
Scatter plot	7.131	7.495	7.023	3.528
range	6.595–8.613	6.953–8.382	6.658–7.116	3.407–3.560
Pickup zones bar	2.183	2.045	1.895	2.172
range	1.885–2.373	1.870–2.489	1.879–2.192	2.095–2.227
Day-hour matrix	2.017	2.003	1.850	2.133
range	1.626–2.130	1.993–2.439	1.616–2.148	2.058–2.182

Stage 2 produced the lowest median initial duration for the four exact-output visual groups. Their Stage 3 medians were 14.62% to 16.64% higher, but the Stage 2 and Stage 3 ranges overlapped for every group. These measurements therefore indicate a tendency toward higher initial rendering time at Stage 3 rather than a stable degradation. The scatter plot followed the opposite pattern. Its initial duration fell from 7.023 s at Stage 2 to 3.528 s at Stage 3, a reduction of 49.77%. Its Stage 3 range was also narrow at 3.407–3.560 s. The sampled representation therefore reduced the first-render cost of the dense scatter plot under the measured conditions.

3) Interaction response: Within each trial, the interaction value for a visual is the longest response among the five fixed actions. Table IV reports the median of these three trial-level maxima. The matrix does not produce a separate lifecycle when it is the source of the heat-map selection. Its trial maximum is therefore taken from the other four actions.

TABLE IV. Worst Interaction Time: Median and Three-Run Range

Visual group	S0 (s)	S1 (s)	S2 (s)	S3 (s)
KPI cards	1.586	1.692	0.518	0.499
range	1.519–1.854	1.603–1.751	0.514–0.528	0.496–0.504
Revenue trend	1.399	1.512	0.468	0.428
range	0.492–1.607	1.460–1.583	0.451–0.472	0.427–0.434
Scatter plot	5.625	6.368	5.464	1.742
range	5.459–6.960	6.232–6.573	5.448–5.478	1.734–1.748
Pickup zones bar	1.399	1.510	0.468	0.427
range	0.480–1.607	1.459–1.582	0.451–0.471	0.418–0.427
Day-hour matrix	0.481	0.520	0.414	0.387
range	0.459–0.518	0.447–1.533	0.412–0.423	0.380–0.389

Stage 3 reduced the median worst interaction for every visual relative to Stage 2. The reductions were 3.67% for

the KPI group, 8.55% for the revenue trend, 8.76% for the pickup-location bar, and 6.52% for the matrix. The scatter plot showed the largest change. Its median fell from 5.464 s to 1.742 s, a reduction of 68.12%, and the three Stage 3 trials remained within 0.014 s of one another. This result is consistent with sampling as the main contributor to the reduction of the dominant interaction bottleneck.

4) Selection outcome: Scores were calculated for every trial using $S = 8{,}549{,}307$, $N = 10$, and $\alpha = 5$, then summarized by the median. Table V reports the resulting values.

TABLE V. MEDIAN WORKLOAD PERFORMANCE SCORES FOR STAGES 0 TO 3

Visual group	S0	S1	S2	S3
KPI cards	0.2076	0.2816	0.7238	0.7212
Revenue trend	0.2134	0.2871	0.7352	0.7415
Scatter plot	0.1178	0.1290	0.1751	0.4031
Pickup zones bar	0.2134	0.2872	0.7357	0.7485
Day-hour matrix	0.2473	0.3606	0.7587	0.7675
Dashboard	0.1178	0.1290	0.1751	0.4031

Stage 3 increased the median scatter score by 130.19% relative to Stage 2. It also increased the trend, bar, and matrix scores by 0.86%, 1.74%, and 1.16%. The KPI score decreased by 0.35%, from 0.7238 to 0.7212, because its slower initial rendering narrowly outweighed its refresh and interaction gains. The Stage 2 KPI score ranged from 0.6985 to 0.7253, while the Stage 3 range was 0.7163–0.7304. The overlapping ranges show that the primary KPI difference is small relative to the observed runtime variation.

The dashboard score follows the scatter plot because that visual supplied both dashboard maxima in every stage. It increased from 0.1178 at baseline to 0.4031 at Stage 3, a gain of 242.29%. This bottleneck improvement does not determine every per-visual choice. Under the median rule, Stage 2 is retained for the KPI group and Stage 3 is selected for the other four groups.

TABLE VI. VALIDATED PER-VISUAL SELECTIONS UNDER THE PRIMARY WORKLOAD

Visual group	Selected stage	Median score	Range
KPI cards	Stage 2	0.7238	0.6985–0.7253
Revenue trend	Stage 3	0.7415	0.7400–0.7510
Scatter plot	Stage 3	0.4031	0.4007–0.4050
Pickup zones bar	Stage 3	0.7485	0.7405–0.7511
Day-hour matrix	Stage 3	0.7675	0.7604–0.7700

The exact aggregate inputs reconciled to the full 8,549,307-row source before refresh. Report outputs from the exact Stage 3 structures matched the preceding valid configuration under the benchmark states. The sampled scatter preserved its association, dominant density regions, and relevant outliers. The timing selections in Table VI therefore satisfy the semantic acceptance rule. RQ1 shows that the final cumulative configuration is not selected universally because the KPI group retains Stage 2 while the other groups select Stage 3.

B. RQ2: Contributions Within Stage 2

The Stage 2 ablation measures five cumulative configurations with the same dataset, report layout, interaction sequence, and three-run timing procedure. Configuration A is the corrected flat wide reference. B adds the six-column source projection, C prepares pickup date and hour in Power Query, D adds the star schema and dimension bindings, and E applies the final numeric DAX measure and formatting.

TABLE VII. STAGE 2 ABLATION TIMING AND DASHBOARD SCORE

Code	Added configuration	L (s)	$V_{\max}$ (s)	$I_{\max}$ (s)	Score$_D$
A	Corrected flat wide	127.25	6.695	5.393	0.1489
B	Narrow projection	57.92	6.590	5.474	0.1744
C	Prepared fields	57.29	6.719	5.471	0.1742
D	Star schema	52.23	7.224	5.571	0.1724
E	Revised DAX	52.42	6.676	5.535	0.1752

TABLE VIII. MEDIAN PER-VISUAL SCORES IN THE STAGE 2 ABLATION

Visual group	A	B	C	D	E
KPI cards	0.3137	0.4659	0.4645	0.7027	0.7097
Revenue trend	0.3255	0.4875	0.6991	0.7042	0.7189
Scatter plot	0.1489	0.1744	0.1742	0.1724	0.1752
Pickup zones bar	0.3255	0.4875	0.7037	0.7314	0.7375
Day-hour matrix	0.4162	0.7121	0.7113	0.7258	0.7364

The A-to-B projection change reduced refresh time by 54.48% and increased the median dashboard score by 17.06%. It was the largest isolated refresh contribution. Preparing the date and hour fields in C changed refresh by only 1.09%. Its median trend and bar scores increased by 43.40% and 44.34%, although their three-run ranges overlapped those of B. The effect should therefore be interpreted as a measured tendency rather than a stable effect estimate.

Adding the star schema in D reduced refresh by 8.83% relative to C. It increased the median KPI, trend, bar, and matrix scores, with the largest change occurring for the KPI group. The scatter score decreased by 1.03%, and the dashboard bottleneck score also decreased by 1.03%. The E measure revision increased the visual medians by 0.84% to 2.08% while refresh increased by 0.19 s. The score ranges of B through E overlap for the scatter-driven dashboard summary. RQ2 therefore identifies source projection as the largest refresh contribution and shows that the later increments have visual-dependent effects. It does not support treating every small difference as stable across repeated execution.

C. RQ3: Insight Beyond Simpler Baselines

The selected stage for a visual is the analytically valid stage with the highest median score under these primary workload settings. Table IX compares the proposed score with weighted workload latency C and maximum component latency M under the primary settings. Each entry reports the selected stage followed by its median value.

RQ3 demonstrates two distinctions. First, maximum latency selects Stage 3 for the KPI group because its largest component falls from 5.172 s to 4.999 s. The combined workload

TABLE IX. SELECTED STAGE UNDER THREE PERFORMANCE MEASURES

Visual group	Min. C (s)	Min. M (s)	Max. score
KPI cards	S2 (9.577)	S3 (4.999)	S2 (0.7238)
Revenue trend	S3 (9.348)	S3 (4.999)	S3 (0.7415)
Scatter plot	S3 (17.197)	S3 (4.999)	S3 (0.4031)
Pickup zones bar	S3 (9.261)	S3 (4.999)	S3 (0.7485)
Day-hour matrix	S3 (9.032)	S3 (4.999)	S3 (0.7675)
Dashboard	S3 (17.197)	S3 (4.999)	S3 (0.4031)

cost rises slightly from 9.577 s to 9.611 s after initial rendering and five interactions are accumulated. Consequently, C and the proposed score retain Stage 2. Maximum latency alone therefore does not represent the complete cost of the stated workload.

Second, the proposed score and C produce the same selection because S is fixed. The score provides no additional ranking information in this experiment, and all observed stage selections are determined by the workload denominator. The current comparison demonstrates the behavior of the workload denominator and its difference from a maximum-latency summary. It does not validate the scale term or support a cross-scale comparison claim.

D. RQ4: Sensitivity to Workload Assumptions

The scores were recalculated from all three trials with $\alpha \in \{0, 1, 3, 5, 10, 20\}$ while holding $S = 8{,}549{,}307$ and $N = 10$. Table X shows that the primary mapping depends on the expected interaction count.

TABLE X. SELECTED STAGE UNDER ALTERNATIVE VALUES OF α

Visual group	0	1	3	5	10	20
KPI cards	S2	S2	S2	S2	S3	S3
Revenue trend	S2	S2	S3	S3	S3	S3
Scatter plot	S3	S3	S3	S3	S3	S3
Pickup zones bar	S2	S2	S3	S3	S3	S3
Day-hour matrix	S2	S2	S3	S3	S3	S3
Dashboard	S3	S3	S3	S3	S3	S3

When α is 0 or 1, Stage 2 is selected for the four exact-output groups because initial rendering receives more relative weight. At $\alpha = 3$, the interaction gains move the trend, bar, and matrix to Stage 3. The KPI group changes at $\alpha = 10$. The scatter and dashboard remain at Stage 3 throughout because their reductions are large in both initial rendering and interaction.

The number of sessions between refreshes was evaluated separately with $N \in \{1, 5, 10, 20, 50\}$ while $\alpha = 5$. Table XI shows the complementary effect of refresh amortization.

Stage 3 is selected for all groups when the complete refresh cost is assigned to each session at $N = 1$. As refresh is distributed across more sessions, the slower Stage 3 initial rendering becomes more influential for the exact-output groups. The KPI changes at $N = 10$, the trend at $N = 20$, and the matrix at $N = 50$. The bar remains at Stage 3 across the evaluated grid because its interaction improvement offsets the initial penalty. These results show that N and α are operational assumptions rather than harmless constants. The framework produces a workload-specific mapping and should be recalculated when expected use changes.

TABLE XI. SELECTED STAGE UNDER ALTERNATIVE VALUES OF N

Visual group	1	5	10	20	50
KPI cards	S3	S3	S2	S2	S2
Revenue trend	S3	S3	S3	S2	S2
Scatter plot	S3	S3	S3	S3	S3
Pickup zones bar	S3	S3	S3	S3	S3
Day-hour matrix	S3	S3	S3	S3	S2
Dashboard	S3	S3	S3	S3	S3

VII. DISCUSSION

The four evaluation questions show which parts of the proposed method are supported by the experiment. RQ1 and RQ2 evaluate the staged framework and its per-visual output. RQ3 and RQ4 evaluate how the score represents a workload and how its selection changes with that workload. The evidence does not establish one universally superior configuration. It demonstrates why cumulative changes must be measured, validated, and selected at the visual level.

A. Evidence for the Framework

RQ1 supports per-visual stopping. Stage 3 was selected for the trend, scatter, bar, and matrix, while the KPI group retained Stage 2 under the primary workload. Because the KPI difference was only 0.35% and its trial ranges overlapped, the mapping records the defined median rule rather than a stable advantage. More importantly, the result shows that automatically retaining the final cumulative stage would hide a measured tradeoff for one visual group.

RQ2 supports the staged organization by separating effects that a single Stage 2 label would conceal. Narrow source projection produced the largest refresh reduction. Prepared fields, the star schema, and the DAX revision had smaller or visual-dependent effects. The framework does not claim these techniques as new. Its value is the common procedure used to attribute their measured effects and prevent a broad label such as model optimization from receiving credit for every change.

Stage 3 further demonstrates why analytical validation and visual purpose are part of the framework. The deterministic sample reduced the scatter plot's initial rendering by 49.77% and its worst interaction by 68.12%, while increasing its score by 130.19% relative to Stage 2 and preserving the required visual features. The exact-output groups could not use that sample. Their slower first rendering may result from aggregate or fallback paths through the expanded model, although this explanation remains a hypothesis because the Stage 3 components were not separately ablated.

B. Evidence for the Workload Score

RQ3 shows why maximum latency and accumulated workload cost answer different questions. Maximum latency selects

Stage 3 for the KPI group because its largest individual component is lower. Weighted workload latency and the proposed score retain Stage 2 because their calculation includes refresh, initial rendering, and five interactions. At the fixed value of S, the score and C produce the same ranking. The current experiment therefore supports the workload denominator and its distinction from a maximum-latency summary, but it makes no claim that the scale term improves cross-scale performance comparison.

RQ4 shows that the framework responds to its declared usage assumptions. Increasing α moves the exact-output groups toward Stage 3 as interaction cost receives more weight. Increasing N reduces the per-session refresh contribution and allows initial rendering to influence more selections. A dashboard opened once after each refresh therefore represents a different workload from one opened repeatedly and filtered many times. The output is a mapping for a specified usage pattern, not a permanent ranking independent of deployment conditions.

Timing selection remains conditional on analytical acceptance. Exact outputs reconcile to the complete source, and the sampled scatter retains the association, density regions, and relevant outliers required by the task. A faster candidate that fails these checks is excluded before selection. This combination of cumulative measurement, workload definition, semantic validation, and per-visual stopping is the main evidence for the proposed method's usefulness in the evaluated dashboard.

VIII. Threats to Validity

Each initial-render and interaction condition was measured in three Performance Analyzer traces. Medians and observed ranges reduce dependence on one execution, but three trials provide limited information about the shape of runtime variation. Background activity, caching, storage state, thermal conditions, and Power BI scheduling may still affect the measurements. This is particularly important for the KPI comparison, where the Stage 2 and Stage 3 medians differ by only 0.35% and their ranges overlap.

Full-refresh inputs were recorded separately from the Performance Analyzer traces. Repeated full-refresh checks did not change the reported configuration ordering or summary values at the retained precision. However, refresh variability is characterized less fully than visual and interaction variability because the refresh measurements were not exported as part of the Performance Analyzer traces.

The Stage 2 ablation is cumulative and order-dependent. It identifies the increment observed when each change is added after the preceding configuration, but it is not a full factorial experiment. An effect may differ if the same change is introduced earlier or without another component. Some three-run ranges also overlap, especially among configurations B through E. Small score differences should therefore not be interpreted as precise causal effects.

Construct validity depends on how well the proposed score represents use. The sensitivity analysis confirms that N and α can change the selected stage. The primary mapping therefore represents one assumed workload rather than a universal quality rating. The worst interaction time gives a conservative view, but it does not describe every delay across a session. Performance Analyzer measures visual lifecycle duration and not perceived frustration, decision quality, or interpretation time. User studies are needed to connect the timing score to executive analytical behavior.

The row count is fixed across all reported configurations. The experiment therefore validates neither the scale term nor cross-scale performance comparison. It only confirms that the term leaves the within-dataset ordering unchanged. Matched experiments at several controlled row counts would be required before making a scale-relative efficiency claim.

External validity is limited by one Power BI dashboard, one workstation, one quarter of taxi data, and five visual groups. Other datasets, relationships, measures, hardware, storage modes, and BI engines may respond differently. The framework and score are tool-independent at a conceptual level, but the implemented changes include Power BI-specific features. Cross-platform experiments are required before claiming that the observed effects generalize to other BI tools.

Stage 3 semantic validation was conducted for the fixed benchmark filters and defined visual comparisons. It does not establish equivalence for untested filter combinations, changed visual encodings, or other dashboards. The selected mapping should therefore be interpreted within the evaluated workload.

IX. Conclusion

This research presented a per-visual staged framework for evaluating established BI dashboard optimizations under a common workload. It treats source, model, and visual-specific changes as cumulative candidates, verifies their analytical validity, and retains the highest-scoring valid stage for each visual. The Stage 2 ablation further separates the measured contributions within the pipeline rather than treating model optimization as one undivided change.

The results show that the selected stage depends on both the visual and the assumed workload. Under the primary settings, Stage 3 was selected for the revenue trend, scatter plot, pickup-location bar, and day-hour matrix, while the KPI group retained Stage 2. As one representative example, Stage 3 sampling reduced the scatter plot's median worst interaction time from 5.464 s to 1.742 s, or 68.12%, while preserving the defined visual features. The sensitivity analysis changed the mapping when N and α changed, showing that refresh and interaction frequency must be stated as part of the evaluation.

The scale term of the proposed score has not yet been empirically validated because the row count was fixed in this experiment. At fixed S, the score and weighted workload latency C produce the same ranking. Controlled cross-scale experiments are therefore required to test the intended scale-relative interpretation. Overall, the case study supports cumulative measurement, semantic validation, and workload-specific per-visual selection, while broader dataset and platform evaluation remains future work.

REFERENCES

[1] O. M. Yigitbasioglu and O. Velcu, "A review of dashboards in performance management: Implications for design and research," *International Journal of Accounting Information Systems*, vol. 13, no. 1, pp. 41–59, 2012.

[2] A. Sarikaya, M. Correll, L. Bartram, M. Tory, and D. Fisher, "What do we talk about when we talk about dashboards?" *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 1, pp. 682–692, 2019.

[3] Z. Liu and J. Heer, "The effects of interactive latency on exploratory visual analysis," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2122–2131, 2014.

[4] J. Lowe and M. Matthee, "Requirements of data visualisation tools to analyse big data: A structured literature review," in *Responsible Design, Implementation and Use of Information and Communication Technology*, ser. Lecture Notes in Computer Science, vol. 12066. Springer, 2020, pp. 469–480.

[5] Y. Park, M. Cafarella, and B. Mozafari, "Visualization-aware sampling for very large databases," in *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*, 2016, pp. 755–766.

[6] Transaction Processing Performance Council, "TPC Benchmark H Standard Specification," Transaction Processing Performance Council, Tech. Rep. Version 3.0.1, 2022. [Online]. Available: https://www.tpc.org/tpc_documents_current_versions/pdf/tpc-h_v3.0.1.pdf

[7] ——, "TPC Benchmark DS Standard Specification," Transaction Processing Performance Council, Tech. Rep. Version 4.0.0, 2024. [Online]. Available: https://www.tpc.org/TPC_Documents_Current_Versions/pdf/TPC-DS_v4.0.0.pdf

[8] ——, "TPC Benchmark DI Standard Specification," Transaction Processing Performance Council, Tech. Rep. Revision 1.1.0, 2014. [Online]. Available: https://www.tpc.org/TPC_Documents_Current_Versions/pdf/TPC-DI_v1.1.0.pdf

[9] L. Battle, R. Chang, J. Heer, and M. Stonebraker, "Position statement: The case for a visualization performance benchmark," in *Proceedings of the 2nd Workshop on Data Systems for Interactive Analysis*, 2017. [Online]. Available: https://homes.cs.washington.edu/~leibatt/static/papers/dsia2017cameraready_08_23_2017_11_28.pdf

[10] P. Eichmann, E. Zgraggen, C. Binnig, and T. Kraska, "IDEBench: A benchmark for interactive data exploration," in *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, 2020, pp. 1555–1569.

[11] J. Purich, A. Wise, and L. Battle, "An adaptive benchmark for modeling user exploration of large datasets," *Proceedings of the ACM on Management of Data*, vol. 3, no. 1, pp. 8:1–8:24, 2025.

[12] J. Yang, H. K. Joo, S. Yerramreddy, D. Moritz, and L. Battle, "Optimizing dataflow systems for scalable interactive visualization," *Proceedings of the ACM on Management of Data*, vol. 2, no. 1, pp. 21:1–21:25, 2024.

[13] D. J. Abadi, S. R. Madden, and N. Hachem, "Column-stores vs. row-stores: How different are they really?" in *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, 2008, pp. 967–980.

[14] J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh, "Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals," *Data Mining and Knowledge Discovery*, vol. 1, no. 1, pp. 29–53, 1997.

[15] Z. Liu, B. Jiang, and J. Heer, "immens: Real-time visual querying of big data," *Computer Graphics Forum*, vol. 32, no. 3, pp. 421–430, 2013.

[16] L. Lins, J. T. Klosowski, and C. Scheidegger, "Nanocubes for real-time exploration of spatiotemporal datasets," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2456–2465, 2013.

[17] S. Agarwal, B. Mozafari, A. Panda, H. Milner, S. Madden, and I. Stoica, "Blinkdb: Queries with bounded errors and bounded response times on very large data," in *Proceedings of the 8th ACM European Conference on Computer Systems*, 2013, pp. 29–42.

[18] C. D. Stolper, A. Perer, and D. Gotz, "Progressive visual analytics: User-driven visual exploration of in-progress analytics," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 1653–1662, 2014.

[19] Apdex Alliance, "Application performance index: Apdex technical specification," Apdex Alliance, Inc., Tech. Rep. Version 1.1, January 2007.

[20] R. K. P. Mok, E. W. W. Chan, and R. K. C. Chang, "Measuring the quality of experience of HTTP video streaming," in *12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011) and Workshops*, 2011, pp. 485–492.

[21] R. Kimball and M. Ross, *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling*, 3rd ed. John Wiley & Sons, 2013.

[22] Microsoft, "Avoid using FILTER as a filter argument in DAX," Microsoft Learn, 2022, accessed July 19, 2026. [Online]. Available: https://learn.microsoft.com/en-us/dax/best-practices/dax-avoid-avoid-filter-as-filter-argument

[23] New York City Taxi and Limousine Commission, "TLC trip record data," NYC.gov, 2025, accessed July 19, 2026. [Online]. Available: https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page

[24] Microsoft, "Use performance analyzer to examine report performance," Microsoft Learn, 2025, accessed July 19, 2026. [Online]. Available: https://learn.microsoft.com/en-us/power-bi/create-reports/performance-analyzer