
# SDEcho: Efficient Explanation of Aggregated Sequence Difference

Fei Ye
Fudan University
21110240013@m.fudan.edu.cn

Zikang Liu
Fudan University
18301020004@fudan.edu.cn

Xi Zhang
Fudan University
xizhang21@m.fudan.edu.cn

Yinan Jing*
Fudan University
jingyn@fudan.edu.cn

Zhenying He*
Fudan University
zhenying@fudan.edu.cn

Yuxin Che
Fudan University
yxche23@m.fudan.edu.cn

Haoran Xiong
Fudan University
hrxiong20@fudan.edu.cn

Kai Zhang
Fudan University
zhangk@fudan.edu.cn

X. Sean Wang*
Fudan University
xywangCS@fudan.edu.cn

## ABSTRACT

Understanding the reasons behind differences between aggregated sequences derived from SQL queries is crucial for data scientists. However, existing methods often suffer from being labor-intensive, lacking scalability, providing only approximate solutions, and inadequately supporting sequence difference explanations. In response, we introduce SDEcho, a novel framework designed to automate the explanation searching for sequence differences in high-dimensional and high-volume datasets. SDEcho utilizes advanced pruning techniques, considering pattern, order, and dimension perspectives, as well as their interactions, to prune the entire explanation space while maintaining explanations accurate and concise. This hybrid pruning approach significantly accelerates the explanation searching process, making SDEcho a valuable tool for data analysis tasks. Extensive experiments on synthetic and real-world datasets, along with a case study, demonstrate that SDEcho outperforms existing methods in terms of both effectiveness and efficiency.

**PVLDB Reference Format:**
Fei Ye, Zikang Liu, Xi Zhang, Yinan Jing, Zhenying He, Yuxin Che, Haoran Xiong, Kai Zhang, and X. Sean Wang. SDEcho: Efficient Explanation of Aggregated Sequence Difference. PVLDB, 18(3): 784 - 797, 2024.
doi:10.14778/3712221.3712242

**PVLDB Artifact Availability:**
The source code, data, and/or other artifacts have been made available at https://github.com/Sherhom/SDEcho.

## 1 INTRODUCTION

Data scientists often need to run SQL queries with group-by and aggregate functions to obtain aggregated sequences, compare different aggregated sequences to gain insights, and further their research. Besides involving simple interactive visualizations, understanding the reasons behind sequence differences co-occurred in the same frame, i.e., "why" questions, is gradually gaining more attention in the database community from academia[11, 15, 25, 35, 45] and industry[3, 4, 6, 7]. Researchers need to manually execute a series of cumbersome queries involving GROUP BY, UNION, and CUBE to identify commonalities among data record groups corresponding to the aggregated sequences with differences.

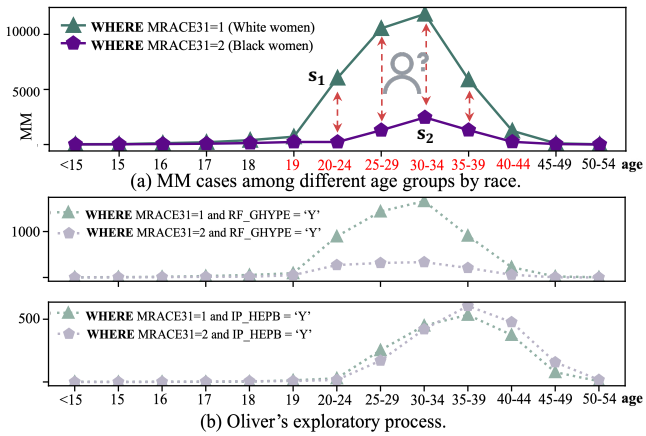


**Figure 1: A visualization of the natality22_MM query results.**

Example 1. *Research on differences in incidence count across racial and ethnic groups is very popular[21, 28, 31]. Researcher Oliver seeks to explore the reasons behind racial disparities in Maternal Morbidity (MM). He leverages the Natality dataset provided by the CDC[1], a comprehensive resource covering factors like Hepatitis B (IP_HEPB) and Gestational Hypertension (RF_GHYPE) across multiple dimensions. One subset is illustrated in Figure 2. Initially, Oliver runs the following query statements with group-by clauses to analyze the number of MM cases among different racial groups across various age groups in 2022:*

```
SELECT MAGER14, count(*) AS MM
FROM natality22_MM
WHERE [condition] GROUP BY MAGER14
```

*As shown in Figure 1(a), he visualizes the query output as two sequences ($s_1$ and $s_2$), ordering the result set from different racial groups*

*Corresponding authors

Proceedings of the VLDB Endowment, Vol. 18, No. 3 ISSN 2150-8097.
doi:10.14778/3712221.3712242

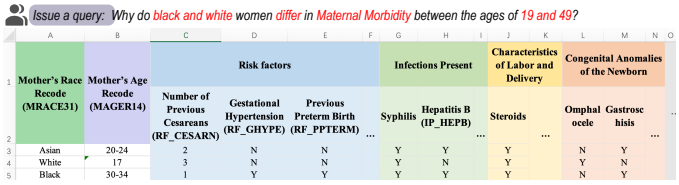


**Figure 2: A subset of the Natality dataset natality22_MM.**

*by aligning point-to-point along the "age" dimension. Notably, the disparity in the number of MM cases between White and Black women aged 19-44 caught Oliver's attention. To investigate sequence differences, Oliver manually executes SQL queries to examine dimensions potentially explaining them. As shown in Figure 1(b), he identified Gestational Hypertension (RF_GHYPE) as similar to the difference to be explained. Oliver needs to further verify whether RF_GHYPE = 'Y' contributes to the sequence difference to determine whether it can be used as an explanation.*

To address the above issues, one popular option is the intervention-based method[25, 42, 45]. It works by re-running queries after removing subsets of the dataset. If the sequence difference is close to zero after this removal, the removed subset is identified as an explanation for sequence differences. Following the literature [11, 15, 18, 45, 53], the explanation in our work is not intended to imply causation. Instead, it identifies the data slices that contribute the most to the overall differences between the two sequences. It's important to emphasize that finding the root cause of "why" questions is generally only feasible when combining human interpretation with analytical tools. Among intervention-based methods, there are two types of explanation settings: fine-grained (a set of tuples) and coarse-grained (a predicate or a pattern)[35, 39], where the predicate can provide a comprehensible explanation and identify common attributes of input tuples that lead to unexpected results. Implementation of this method can be achieved by extending DBMS with custom user-defined functions (UDFs) or by utilizing explanation engines[11, 15, 38, 45], which have gained popularity in the database community in recent years.

Unfortunately, in our scenarios, the aforementioned methods face the following dilemmas: **(1) Human-labored.** Although it is possible to perform intervention-based explanation searching using the interface languages provided by databases or certain data analysis tools[6, 7], it is challenging for humans to translate these into corresponding SQL, DAX[6, 23], etc. Moreover, most explanation engines are limited to single-dimension or three-dimensional combinations, making enumerating potential combinations cumbersome. Ordinary analysts need to repeatedly enumerate explanations with up to three attributes within a restricted search space. Even when analyzed by domain experts, different experts might take different directions. For example, pediatricians explore the explanation of the sequence differences in Figure 1(a) from the perspective of congenital anomalies in newborns, and there are as many as 12 attributes related to this field. **(2) Poor in scalability.** The limitation on the number of dimensions for explanation search is due to the current coarse-grained explanation engines struggling to cope with high-dimensional and massive data scenarios, thus making it difficult to meet users' needs for explanations during the data analysis process. **(3) Loss by approx.** Explanations require pinpointing concise and precise tuple sets[11, 41, 53], which means highlighting the largest sequence difference with the fewest number of tuples, enabling users to decide on subsequent analysis steps based on explanations. The impact of a small number of tuples with numerical anomalies on the results can be significant. Predicates corresponding to this subset typically involve numerous attributes. However, a limited explanation search space or approximate solutions may result in overlooking these explanations. Such omissions due to approximation could potentially mislead data analysts in their next steps of exploration, thereby increasing their workload. **(4) Sequence diff unsupported.** Existing explanation engines focus on explaining a single aggregate value[27], differences between two given relations[10, 11, 35, 45], or searching for a single series' key performance indicators (KPIs)[15], which do not support the sequence difference explanation scenario in this work, which involves accumulating point-by-point differences between sequences.

To avoid the above dilemmas, and as a first approach toward explaining aggregated sequence differences, we introduce SDEcho, a framework designed for searching explanations in **(i) sequence difference scenarios**. Given a pair of user-specified aggregated sequence fragments co-occurred in the same frame, SDEcho measures point-wise differences between sequences and automatically searches for top-$k$ explanations *online*, eliminating the need for users to explore potential dimension combinations manually. We describe a novel technique tailored for **(ii) high-dimensional, high-volume data scenarios** to optimize the speed of explanation search while ensuring **(iii) results accuracy**. By projecting sequence pairs into a vector space, SDEcho leverages the additivity of projection distances for non-overlapping explanations and the relationship between explanation measure and sequence distance to prune the candidate explanation space efficiently. Our method can support most of the popular aggregation functions. Through extensive experiments, we demonstrate that SDEcho significantly accelerates explanation searching compared to existing solutions[11, 15, 35]. We also developed a benchmark and introduced an explanation confidence score to compare the quality of explanations produced by different methods. Finally, for **(iv) ease of use**, we have implemented SDEcho's functionalities as operators in Postgres. This allows users to achieve the same functionalities as SDEcho without the need to write lengthy data manipulation statements compared to existing solutions for out-database execution.

In summary, we make the following contributions:

- We formulate the problem of explaining aggregated sequence differences with point-wise sequence comparison and generate concise explanations tailored for this scenario.
- We propose SDEcho, an automated explanation search framework that prunes the candidate explanation space at the pattern, order, and dimension levels. By integrating multiple pruning methods, we develop a hybrid approach to enhance performance.
- We constructed a benchmark for explaining aggregated sequence differences and designed an explanation confidence score to evaluate the quality of the explanations. Experiments on real-world workloads and datasets demonstrate that SDEcho significantly accelerates the explanation search process while maintaining quality compared to existing methods.

## 2 PROBLEM FORMULATION

We first introduce some fundamental concepts regarding our two-sequences difference explanation, followed by our explanation measure to evaluate the explanation quality. Subsequently, we will delve into our explanation searching optimization problem.

Definition 1 (Aggregated sequence $s$). *An aggregated sequence $s = Q(R)$ is the result set obtained by executing the following query $Q$ with group-by:*

`SELECT` $\mathcal{F}(M)$, $G$ `FROM` [$R$] `WHERE [con] GROUP BY` $G$,

*where $R$ is a relation of the database instance $D$ with attributes $A = \{a_1, a_2, ..., a_n\}$. $A$ includes the group-by attribute $G$ and the measure attribute $M$. $\mathcal{F}()$ is the aggregate function (such as* `COUNT()`, `MIN()`, `MAX()`, *or* `SUM()`*).*

When performing visual analysis, users typically provide an order to display this set as a sequence. For example, users can determine the sequence order using an `ORDER BY` clause or by specifying their own order. After that, this aggregated result set is converted into a sequence.

In our work, the aggregated sequences $s_1$ and $s_2$ for comparison are derived from $Q_1(R_1)$ and $Q_2(R_2)$, respectively. Only $G$ and $M$ must be shared between $Q_1(R_1)$ and $Q_2(R_2)$, and the value domains of $G$ must be consistent. $R_1$ and $R_2$ can refer to the same fact table, as shown in Figure 1(a) with the query on the table "natality22_MM."

Given two aggregated sequences $s_1$ and $s_2$ of length $n$, we use the distance function $\text{dist}(s_1, s_2)$ to measure how similar two sequences are. In our work, we focus on the point-wise differences between two aggregated sequences and apply the Euclidean distance to capture these differences. Euclidean distance is straightforward and widely known for its simplicity and intuitive geometric interpretation. It is commonly used[13, 22] in exploratory analysis as it quickly provides a baseline for measuring point-wise differences, which aligns well with the goals of SDEcho, helping both technical and non-technical users interpret sequence differences.

$$\text{dist}\,(s_1, s_2) = |s_1 - s_2| = \sqrt{\sum_{i}^{n} (s_1[i] - s_2[i])^2}. \tag{1}$$

The smaller the distance, the more similar the sequences are.

Example 2. *In Figure 1(a), the two aggregated sequences represent the number of MM cases for black women and white women at different ages. According to equation (1), the Euclidean distance is calculated by comparing black women and white women age by age, which is the difference between the two sequences.*

Definition 2 (Explanation $e$). *Given aggregated sequences $s_1$ and $s_2$, the explanation $e$ of order $\beta$ for the distance between them is defined as a conjunction of $\beta$ predicates (a.k.a., a pattern):*

$$\left(a_1 = v_1 \wedge a_2 = v_2 \wedge \ldots \wedge a_\beta = v_\beta\right),$$

*where $a_i$ is in the set of explain-by attributes $\mathcal{A}$ ( $\mathcal{A} \subseteq A$, $\beta \leq |\mathcal{A}|$).*

It is important to note that $\mathcal{A}$ should exclude the measure and group-by attributes $M$ and $G$ as specified in the SQL statements, i.e., $\mathcal{A} \subseteq A - G - M$. We borrow this definition from prior work on explanation engines[10, 11, 45]. In our work, it's worth noting that explanations $e_1$ and $e_2$ being non-overlapping means that their corresponding data intersections in $D$ are empty, i.e., $\sigma_{e_1}(D) \cap \sigma_{e_2}(D) = \emptyset$ where $\sigma$ is the selection operation. Furthermore, to handle numerical attributes, we adopt the binning method used in [17, 19] for discretization.

Based on the above definitions, our goal is to identify the explanation that best accounts for the difference between two sequences. Using counterfactuals, we mask or remove tuples corresponding to the explanation and measure the resulting sequence distance. A reduction in distance indicates that the explanation is significant for explaining the difference between the two sequences.

After providing an explanation, the analyst must examine the removed tuples to identify the root cause of the sequence difference. Large tuple sets complicate this process, and removing all tuples, while maximizing distance change, makes the explanation meaningless. Thus, explanations with fewer tuples should be prioritized when achieving the same sequence distance reduction. To facilitate this, we introduce a penalty factor $\psi_e$ that accounts for tuple count, encouraging concise explanations.

Definition 3 (Penalty factor $\psi_e$). *Given an explanation $e$, penalty factor $\psi_e$ is defined as the normalized number of tuples corresponding to $e$ that need to be removed to explain the difference between two aggregated sequences $s_1$ and $s_2$:*

$$\psi_e(R_1, R_2) = \frac{|\sigma_e(R_1)|}{|R_1|} + \frac{|\sigma_e(R_2)|}{|R_2|} + 1. \tag{2}$$

Definition 4 (Explanation Measure $\gamma$). *Given two aggregated sequences $s_1$ and $s_2$, they are obtained by running $Q_1(R_1)$ and $Q_2(R_2)$, respectively. Then, we define the measure $\gamma$ of an explanation $e$ as*

$$\begin{aligned}\gamma\,(e, s_1, s_2) &= \frac{\text{dist}\left(Q_1^{\neg e}, Q_2^{\neg e}\right)}{\text{dist}(s_1, s_2)} * \psi_e\,(R_1, R_2) \\ &= \frac{\text{dist}\,(Q_1\,(R_1 - \sigma_e(R_1))\,, Q_2\,(R_2 - \sigma_e(R_2)))}{\text{dist}\,(Q_1\,(R_1)\,, Q_2\,(R_2))} \\ &\quad * \left(1 + \frac{|\sigma_e(R_1)|}{|R_1|} + \frac{|\sigma_e(R_2)|}{|R_2|}\right),\end{aligned} \tag{3}$$

*where* $\text{dist}\left(Q_1^{\neg e}, Q_2^{\neg e}\right)$ *is the distance between the two sequences after removing tuples corresponding to the explanation $e$, i.e., the effect of the explanation $e$ on $Q_1$ and $Q_2$.*

In addition, we use the original distance between $Q_1(R_1)$ and $Q_2(R_2)$ for normalization to offset the effects of the explanation. Obviously, the lower the explanation measure of explanation $e$, the better the explanation.

Based on our explanation measure, our explanations possess the minimality property[11], providing valuable insights without overwhelming users with redundant information.

Definition 5 (Top-$k$ Explanation Problem). *Given two aggregated sequences $s_1$ and $s_2$, find the top-k explanations $\mathcal{E}^*$ — from all possible explanations $E$ — that satisfy the following condition:*

$$\mathcal{E}^* = \underset{\substack{\mathcal{E} \subset E \\ |\mathcal{E}| = k}}{\arg\min} \sum_{e \in \mathcal{E}} \gamma(e, s_1, s_2). \tag{4}$$

Since computing the explanation measure $\gamma$ by enumerating possible explanations incurs large computing costs, we seek a solution to avoid enumerating and computing each one individually.

## 3 SDECHO: EXPLANATION SEARCHING

### 3.1 A Basic Approach and its Challenge

We first scan the instance $D$ and enumerate all possible explanations, i.e., all patterns. For each explanation, we compute its explanation measure $\gamma$ using Equation 3. Then, we rank and find the optimal top-$k$ explanations $\mathcal{E}^*$ for a minimum explanation measure. Assuming the average cardinality of each attribute is $c$, and the number of tuples in the table is $r$, then the complexity of the above solution is $O\left(2^{|\mathcal{A}|} \cdot c^{|\mathcal{A}|} \cdot r\right)$. We can clearly see that generating the top-$k$ explanation set $\mathcal{E}^*$ is very time-consuming for three reasons. (i) There are exponentially many, $2^{|\mathcal{A}|}$, combinations of attributes. (ii) For each attribute combination, there are $c^{|\mathcal{A}|}$ possible explanations, i.e., patterns. (iii) Calculating the explanation measure for each explanation requires scanning through the table once. Hence, the complexity is proportional to the number of tuples $r$, which is $O(r)$.

To solve this problem, we propose SDEcho, whose architecture is illustrated in Figure 3. Users can query the database using SQL and obtain two aggregated sequences, $s_1$ and $s_2$, through visual analytical tools, which serve as inputs to SDEcho for explanation searching of the differences between the sequences. SDEcho consists of three modules—PatternPrune, OrderPrune, and DimPrune—that perform pruning to accelerate the generation of the top-$k$ explanations $\mathcal{E}^*$ based on different aspects of complexities: the number of patterns, the order of patterns, and the number of dimensions. Additionally, PatternPrune shares intermediate results computed from partial data with the other modules, which enhances the pruning effectiveness of both OrderPrune and DimPrune. The interaction among these three modules forms a Hybrid Pruning Algorithm (as shown in Algorithm 1).

### 3.2 Aggregation Function Analysis

We divide the aggregate function into **incrementally removable** and **independent** aggregates based on their properties[11, 55]. An aggregate is incrementally removable if the updated result, excluding a subset $s$ from inputs $R$, can be computed using only $s$. For example, `SUM` is incrementally removable as `SUM(D-s) = SUM(D) - SUM(s)`, with `SUM(D)` cached. Aggregates like `SUM()` and `COUNT()` and arithmetic expressions based on them are incrementally removable, whereas `MIN()` and `MAX()` are independent, meaning input tuples influence the result independently. Next, we first introduce the optimization of explanation searching for incrementally removable aggregates, followed by a discussion on independent aggregates.

### 3.3 PatternPrune: Pattern-level Pruning

Independently calculating the explanation measure for each explanation requires repeated table scans, which is time-consuming. The intuition behind our pattern-level pruning is that, when considering only the distance component of our explanation measure, we can project the Euclidean distance between two sequences in multi-dimensional space onto one-dimensional space. By utilizing the additivity of projection distances in one-dimensional space for non-overlapping explanations, we can avoid the independent computation of the distance measure for each explanation. This allows us to prune the explanation space based solely on the relationship between the distance component and the explanation measure.

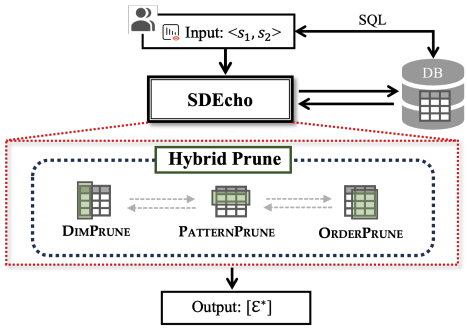


**Figure 3: The overall architecture of SDEcho**

Therefore, in Section 3.3.1, we first analyze the relationship between the distance scores corresponding to non-overlapping explanations, and then in Section 3.3.2, we propose Theorem 1 to perform pruning by leveraging the relationship between the projection distance and the actual explanation measure.

*3.3.1 Bottom-up Computation.* As shown in Equation 3, our explanation measure includes a distance score and a penalty factor, reflecting the explanation's contribution to sequence differences and its intrinsic properties. For incrementally removable aggregates, to reduce computational costs from numerous explanations, we aim to derive the combined explanation's distance score directly from existing ones.

The intuition behind our method is that the Euclidean distance between sequences under different explanations lacks additivity, requiring independent computation for each sequence and explanation. However, point-to-point value additivity exists under non-overlapping explanations. Thus, before calculating the Euclidean distance, we treat sequences as vectors and analyze additivity in the context of vector sums. We first define the explanation vector and use Lemma 1 to prove that explanation vectors of non-overlapping explanations can be accumulated to obtain the vector of their combined explanation.

Definition 6 (Explanation vector $\boldsymbol{d_e}$). *Given an explanation $e$, the explanation vector $\boldsymbol{d_e}$ is defined as the distance difference vector under the explanation $e$ of two aggregated sequences, i.e., $\boldsymbol{d_e} = Q_1(\sigma_e(R_1))$ - $Q_2(\sigma_e(R_2))$.*

For example, in Figure 1(b), if the explanation $e$ is {RF_GHYPE = 'Y'}, $\boldsymbol{d_e}$ is derived from the point-to-point differences between the two sequences under the conditions "MRACE31 = 1 and RF_GHYPE = 'Y'" and "MRACE31=2 and RF_GHYPE='Y'". The vector has 13 points, each representing the difference in the number of MM cases between the two sequences for that specific age group.

Lemma 1 (Additivity of explanation vectors). *If the explanation $e_i$ and $e_j$ are non-overlapping, and $\sigma_{e_0}(D) = \sigma_{e_i}(D) \cup \sigma_{e_j}(D)$, then the explanation vector $\boldsymbol{d_{e_0}} = \boldsymbol{d_{e_i}} + \boldsymbol{d_{e_j}}$.*

Proof. Given an explanation $e_i$, sequences of Query $Q_1$ on $\sigma_{e_i}(R_1)$ and Query $Q_2$ on $\sigma_{e_i}(R_2)$ are denoted as vectors $\boldsymbol{v_{i1}}$ and $\boldsymbol{v_{i2}}$. Since $e_i$ and $e_j$ are non-overlapping and $\sigma_{e_0}(D) = \sigma_{e_i}(D) \cup \sigma_{e_j}(D)$, the vector $\boldsymbol{v_{01}}$ from executing the aggregate on explanation $e_0$ in relation $R_1$ is the sum of the vectors from executing the aggregates

for $e_i$ and $e_j$ in $R_1$, i.e., $\boldsymbol{v_{01}} = \boldsymbol{v_{i1}} + \boldsymbol{v_{j1}}$, where $\boldsymbol{v_{ij}}$ is the vector from executing the aggregate for $e_i$ on relation $R_j$. Therefore,

$$\begin{aligned} \boldsymbol{d_{e_0}} &= \boldsymbol{v_{01}} - \boldsymbol{v_{02}} = (\boldsymbol{v_{i1}} + \boldsymbol{v_{j1}}) - (\boldsymbol{v_{i2}} + \boldsymbol{v_{j2}}) \\ &= (\boldsymbol{v_{i1}} - \boldsymbol{v_{i2}}) + (\boldsymbol{v_{j1}} - \boldsymbol{v_{j2}}) = \boldsymbol{d_{e_i}} + \boldsymbol{d_{e_j}}. \end{aligned} \tag{5}$$

Based on Lemma 1, the explanation vector corresponding to the $i$-order explanations can be obtained by accumulating the explanation vectors of non-overlapping $(i+1)$-order explanations. For example, in Figure 4(a), the attribute IP_HEPB has two values, 'Y' and 'N'. The 2-order explanations $e_2$ and $e_3$ correspond to patterns {RF_GHYPE = 'Y' ∧ IP_HEPB='Y'} and {RF_GHYPE = 'Y' ∧ IP_HEPB='N'} respectively. Since $e_2$ and $e_3$ are non-overlapping, the 1-order explanation $e_1$ ({RF_GHYPE = 'Y'}) corresponds to the explanation vector $\boldsymbol{d_{e_1}}$, which can be obtained by adding $\boldsymbol{d_{e_2}}$ and $\boldsymbol{d_{e_3}}$ as shown in Figure 4(b), i.e., $\boldsymbol{d_{e_1}} = \boldsymbol{d_{e_2}} + \boldsymbol{d_{e_3}}$.

*3.3.2 Pruning using Projection Distance.* Given an explanation $e$, the explanation vector obtained by aggregating the tuples corresponding to $e$ is denoted as $\boldsymbol{d_e}$, and the vector for the tuples not corresponding to $e$ is $\boldsymbol{d_{\neg e}}$. Since the tuples for both vectors do not overlap and their union covers all tuples in $D$, based on Lemma 1, the relationship is $\boldsymbol{d} = \boldsymbol{d_{\neg e}} + \boldsymbol{d_e}$, where $\boldsymbol{d}$ represents the total aggregate vector for $D$, i.e., $\boldsymbol{d}$ = $Q_1(R_1)$ - $Q_2(R_2)$.

Since the norms of explanation vectors under Euclidean distance satisfy the triangle inequality—i.e., $|\boldsymbol{d_e} + \boldsymbol{d_{\neg e}}| \leq |\boldsymbol{d_e}| + |\boldsymbol{d_{\neg e}}|$—it is necessary to recalculate the distance score, $\frac{\text{dist}(Q_1^{\neg e}, Q_2^{\neg e})}{\text{dist}(s_1, s_2)}$, for each explanation $e$. This either requires retaining the original vectors, increasing time and space complexity, or recalculating the distance. Additionally, since the explanation measure uses the norm of $\boldsymbol{d_{\neg e}}$, the original vectors must still be kept. To address this, we convert the Euclidean distance calculation to a projection distance, leveraging additivity—where $\boldsymbol{d} = \boldsymbol{d_e} + \boldsymbol{d_{\neg e}}$ and the projection of $\boldsymbol{d_e}$ onto $\boldsymbol{d}$ plus the projection of $\boldsymbol{d_{\neg e}}$ onto $\boldsymbol{d}$ equals $|\boldsymbol{d}| = d_e^p + d_{\neg e}^p$. By applying Theorem 1, we can compute the projection distance efficiently and use it to prune explanations during the search.

Theorem 1. *For an explanation $e$, the lower bound of its $\gamma$ is $\frac{|\boldsymbol{d}| - d_e^p}{\text{dist}(s_1, s_2)}$, where $d_e^p$ is the projection distance of $\boldsymbol{d_e}$ onto $\boldsymbol{d}$.*

Proof. Given a pair of aggregated sequences, considering the lower bound of the penalty factor $\psi_e(R_1, R_2)$ is 1, and the distance $\text{dist}(s_1, s_2)$ remains constant, we scale the score function as follows:

$$\begin{aligned} \gamma(e, s_1, s_2) &= \frac{\text{dist}\left(Q_1^{\neg e}, Q_2^{\neg e}\right)}{\text{dist}(s_1, s_2)} * \psi_e(R_1, R_2) \\ &\geq \frac{\text{dist}(Q_1^{\neg e}, Q_2^{\neg e})}{\text{dist}(s_1, s_2)} \\ &= \frac{\text{dist}(Q_1 - Q_1^e, Q_2 - Q_2^e)}{\text{dist}(s_1, s_2)}. \end{aligned} \tag{6}$$

Since $\text{dist}(Q_1^{\neg e}, Q_2^{\neg e}) = |\boldsymbol{d_{\neg e}}|$, we convert the lower bound into a vector for calculation:

$$\gamma(e, s_1, s_2) \geq \frac{|\boldsymbol{d_{\neg e}}|}{\text{dist}(s_1, s_2)} \geq \frac{|\boldsymbol{d}| - d_e^p}{\text{dist}(s_1, s_2)} = \frac{d - d_e^p}{\text{dist}(s_1, s_2)}, \tag{7}$$

where $d$ is the original distance between $s_1$ and $s_2$, i.e., $d$ = $\text{dist}(Q_1(R_1), Q_2(R_2))$.

Based on the additivity of projection distance and the lower bound of explanation measures, our pattern-level pruning strategy calculates the projection distance for the highest-order explanations. Using the cube operation, we then compute projection distances for all explanations and sort them in descending order. By maintaining a min-heap of size $k$, we store the current top-$k$ explanation measures and apply the lower bound (Equation 7) to efficiently prune the sorted explanations, ultimately obtaining the top-$k$ explanations.

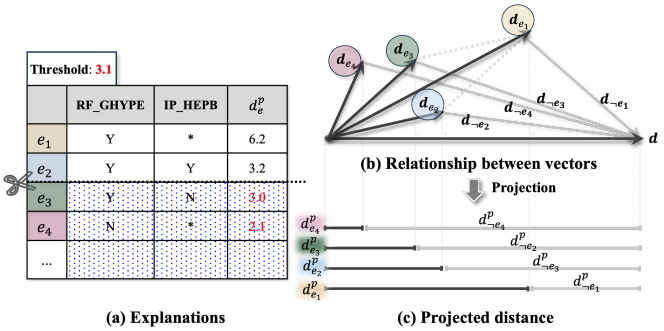


**Figure 4: Pattern-level pruning**

Example 3. *As shown in Figure 4(a), given the explain-by attributes $\mathcal{A}$ = {RF_GHYPE, IP_HEPB}, we first calculate the projection distance for the highest-order explanations (second-order). For example, $e_2$ = {RF_GHYPE = 'Y' ∧ IP_HEPB = 'Y' } with $d_{e_2}^p$ = 3.2. Using the additivity of projection distance, we calculate and sort the projection distances of all explanations via cube operations. For instance, the projection distance of $e_1$ is $d_{e_1}^p = d_{e_2}^p + d_{e_3}^p$. Based on the smallest score in the current min-heap, we calculate the projection distance threshold as 3.1 using Equation 7. Therefore, only $e_1$ and $e_2$ remain, while explanations like $e_3$ and $e_4$ are pruned.*

Our projection distance-based pruning has a strong geometric significance: taking $e_3$ in Figure 4(b) as an example, the module of the desired $\boldsymbol{d_{\neg e_3}}$ is related to its projection distance $d_{\neg e_3}^p$, while the module of the original sequence distance vector $\boldsymbol{d}$ remains constant. Since $d_{\neg e_3}^p = |\boldsymbol{d}| - d_{e_3}^p$, $|\boldsymbol{d_{\neg e_3}}|$ and $|\boldsymbol{d_{e_3}}|$ are negatively correlated. This means that the larger the projection distance $d_e^p$ of an explanation $e$, the smaller its final distance score may be, increasing the likelihood of it being among the top-$k$ explanations, as shown in Figure 4(c).

**Takeaway.** Our method leverages the additive property of projection distances to simplify calculations, achieving fast computation by scanning the table only once. Additionally, we employ a lower-bound pruning technique that terminates the traversal early when the projection distance falls below a specified threshold, thereby enhancing overall efficiency.

## 3.4 OrderPrune: Order-level Pruning

As discussed in Section 3.1, the large number of attribute combinations is a bottleneck in our pipeline. Order-level pruning is designed to address this by determining the maximum order within the top-$k$ explanations, helping SDEcho avoid computing projection distances for all explanations starting from the highest-order ones.

**High-level Idea.** Our intuition behind Order-level pruning is that the deletion of a small number of tuples is unlikely to cause

significant changes in the sequence difference. Taking COUNT() as an example, if an explanation $e$ only encompasses 3 tuples, i.e., $|\sigma_e(D)| = 3$, the change in the sequence distance affected by this explanation cannot exceed 3. Obviously, the higher the order of the explanation, the fewer tuples will meet the explanation predicate. Excessively high-order explanations can only affect a handful of tuples and are insufficient to make the distance exceed the threshold of the top-$k$ explanations.

Theorem 2. *Given an explanation $e_i$, the lower bound for the distance reduction under it is $|\boldsymbol{d}|-\sum_{i=1}^{n}(Q_1(|\sigma_{e_i}R_1|)[i]+Q_2(|\sigma_{e_i}R_2|)[i])$, where $n$ is the sequence length, $|\sigma_{e_i}R_1|$ and $|\sigma_{e_i}R_2|$ are the absolute values of aggregated attributes in $\sigma_{e_i}R_1$ and $\sigma_{e_i}R_2$.*

Proof. As discussed in Section 3.3, the explanation vector satisfies $|\boldsymbol{d_{\neg e_i}}| = |\boldsymbol{d} - \boldsymbol{d_{e_i}}| \geq |\boldsymbol{d}| - |\boldsymbol{d_{e_i}}|$, based on Lemma 1, the upper bound of $|\boldsymbol{d_{e_i}}|$ is:

$$\begin{aligned} |\boldsymbol{d_{e_i}}| &= |\boldsymbol{v_{i1}} - \boldsymbol{v_{i2}}| \leq |\boldsymbol{v_{i1}}| + |\boldsymbol{v_{i2}}| \\ &= \sqrt{\sum_{i=1}^{n} Q_1(\sigma_{e_i}R_1)[i]^2} + \sqrt{\sum_{i=1}^{n} Q_2(\sigma_{e_i}R_2)[i]^2} \\ &\leq \sum_{i=1}^{n} |Q_1(\sigma_{e_i}R_1)[i]| + \sum_{i=1}^{n} |Q_2(\sigma_{e_i}R_2)[i]| \\ &\leq \sum_{i=1}^{n} Q_1(|\sigma_{e_i}R_1|)[i] + \sum_{i=1}^{n} Q_2(|\sigma_{e_i}R_2|)[i]. \end{aligned} \tag{8}$$

Therefore, the lower bound of $|\boldsymbol{d_{\neg e}}|$ is:

$$\begin{aligned} |\boldsymbol{d_{\neg e_i}}| &\geq |\boldsymbol{d}| - |\boldsymbol{d_{e_i}}| \\ &\geq |\boldsymbol{d}| - \sum_{i=1}^{n} (Q_1(|\sigma_{e_i}R_1|)[i] + Q_2(|\sigma_{e_i}R_2|)[i]). \end{aligned} \tag{9}$$

The lower bound decreases as the number of tuples affected by explanation $e$ increases. When the affected tuples are few, the potential for distance reduction is limited. Therefore, based on Theorem 1 and Theorem 2, we use the distance reduction of the current top-$k$ explanations as a pruning threshold for explanations of a certain order. If the lower bound of an explanation meets the threshold, higher-order explanations under this explanation can be pruned; otherwise, we continue drilling down to higher-order explanations until the threshold is met.

**Practical Consideration.** We use the explanation measures of the top-$k$ explanations from first-order explanations as the threshold, starting order pruning from the third order, for the following reasons: (1) According to [11], most practical explanations are essentially third-order, so higher-order explanations can be pruned. (2) The number of first-order explanations is much smaller than second- and third-order ones, reducing computational costs. (3) The initial threshold will be updated as more explanations are computed, so high accuracy is not needed at the start. (4) Dimension pruning in subsequent steps relies on statistical information from first-order explanations, as detailed in Section 3.5.

Example 4. *As shown in Figure 5(a), the explain-by attribute set $\mathcal{A}$ is {RF_GHYPE, IP_HEPB, RF_PPTERM, Steroids }. Instead of calculating the explanation distance scores starting from the fourth order, we first compute the top-k explanations of all first-order explanations. According to Theorem 1, we take the maximum score $\gamma$ from the top-k explanations as the threshold (threshold = 2.7). Then, based on Theorem 2, we calculate the distance score for all third-order explanations. For example, when calculating the explanation $e_i$ = {RF_GHYPE = 'Y' ∧ IP_HEPB = 'N' ∧ RF_PPTERM = 'N' }, if $|d_{\neg e_i}| > 2.5$ (less than the threshold), it cannot be pruned. Therefore, it is necessary to calculate from the top down to determine whether the fourth-order explanations based on $e_i$, {RF_GHYPE = 'Y' ∧ IP_HEPB = 'N' ∧ RF_PPTERM = 'N' ∧ Steroids = 'Y'} and {RF_GHYPE = 'Y' ∧ IP_HEPB = 'N' ∧ RF_PPTERM = 'N' ∧ Steroids = 'N' }, can be pruned.*

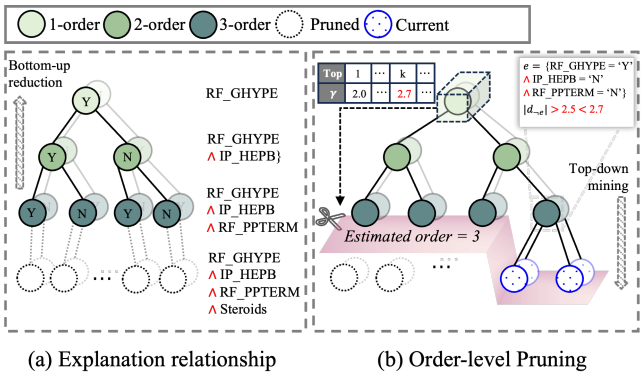


**Figure 5: Order-level pruning**

**Takeaway.** Combining pattern-level pruning, we implemented an explanation search method that integrates bottom-up computation with top-down searching. Order-level pruning uses attribute cardinality to determine the maximum order $O$, starting calculations from the $O$-th order bottom-up. For explanations of order higher than $O$, based on Theorem 2, if their maximum potential impact on sequence differences exceeds a threshold, we apply top-down searching approach until it falls below the threshold, ensuring no top-$k$ explanations are missed and preserving pruning precision.

## 3.5 DimPrune: Dimension-level Pruning

As discussed in Section 3.1, a large number of explain-by attributes $|\mathcal{A}|$ slows down explanation searching. In this subsection, we utilize the cardinality of dimensions and functional dependencies to perform pruning at the dimension granularity.

**(1) Leveraging Cardinality.** We sort dimensions in ascending order by cardinality, as explanations with larger cardinalities affect fewer tuples and may have less impact on distance reduction, per Theorem 2. To avoid an excessive number of patterns from considering all dimensions together, we treat high-cardinality dimensions separately. Instead of pruning all explanations at a certain order using the lower bound from Theorem 2, we prune all first-order explanations under a certain explain-by attribute. If all explanations under a dimension meet the criteria, subsequent searches can disregard that dimension, similar to order-level pruning.

**(2) Leveraging Functional Dependencies.** Analyzing metadata for functional dependencies can effectively prevent explanations from becoming too rich and can be used for dimension pruning. When there are many dimensions, we can follow the approach described in [11] to prune using functional dependencies. Given dimensions $a_i$ and $a_j$, if $a_i$ functionally determines $a_j$, meaning

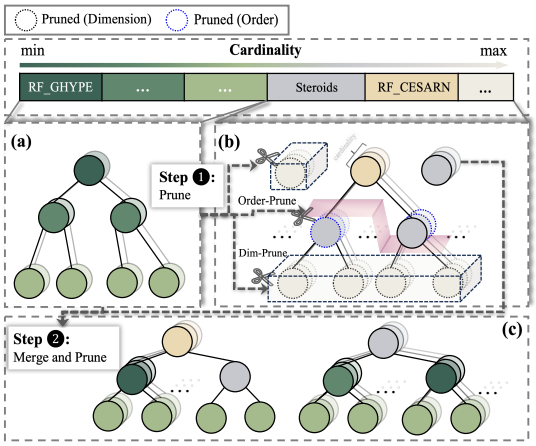


**Figure 6: Hybrid pruning**

**Algorithm 1:** Top-$k$ Explanation Searching

**Input:** sequences $s_1$ and $s_2$, $Q$, $E = [\,]$, $\mathcal{A}$
**Output:** Top-$k$ explanations $\mathcal{E}^*$

1 sort($\mathcal{A}$) ; // by Cardinality
2 $[\mathcal{A}_i] \leftarrow$ partition($\mathcal{A}$)
3 $\gamma^* \leftarrow 0$, $E \leftarrow []$
4 **for** $\mathcal{A}_i$ in $[\mathcal{A}_i]$ **do**
5  **if** $i = 0$ **then**
6  $E_0 \leftarrow$ FindExplanations($\beta = 1$, $\mathcal{A}_1$)
7  $\mathcal{E}_0 \leftarrow$ PatternPrune($E$, $\gamma^*$)
8  **else**
9  $\mathcal{A}_i \leftarrow$ DimPrune($\mathcal{A}_i$, $\gamma^*$)
10  $E_i \leftarrow$ OrderPrune($\gamma^*$, FindExplanations($\mathcal{A}_i$))
11  Update($\gamma^*$)
12  $E$.append($E_i$)
13 $\mathcal{E}^* \leftarrow$ PatternPrune($E$)
14 **return** $\mathcal{E}^*$

that tuples containing $x \in a_i$ also necessarily contain a particular attribute $y \in a_j$, this is denoted as $a_i \rightarrow a_j$. When $a_i \rightarrow a_j$, we can ignore explanations that include both $a_i$ and $a_j$. Furthermore, when $a_i \rightarrow a_j$ and $a_j \rightarrow a_i$, there is no need to consider explanations that include $a_i$.

## 3.6 Putting Things Together: Hybrid Pruning

So far, we have proposed three pruning methods mentioned above, targeting pattern, order, and dimension pruning respectively. To further optimize, we adjust the pruning order based on the characteristics of each pruning method to perform hybrid pruning.

As discussed in Section 3.4, dimension-level and order-level pruning use Theorem 2 to compute the lower bound of distance reduction, denoted as $|d_{\neg e}|$. Then, using Theorem 1, the threshold computed on a subset of the data is compared with this lower bound to determine whether pruning can be performed. The smaller the provided threshold, the more aggressive the pruning. Our high-level idea is to use Pattern-level pruning on a small set of explanations to generate a tighter threshold, guiding order-level and dimension-level pruning for better results. Finally, we consider all explanations based on the remaining dimensions and unpruned explanations to return the global top-$k$ explanations.

Our hybrid pruning process is illustrated in Figure 6, with the algorithm details provided in Algorithm 1. Due to Theorem 2, which relates the number of tuples corresponding to an explanation and $|d_{\neg e}|$, we first sort and partition the explain-by attributes according to their cardinality (line 1 to 2). We generate explanations from the dimension set with smaller cardinalities and perform pattern-level pruning to return the local top-$k$ explanations (line 5 to 7). This process tends to yield greater distance reduction, as discussed in Section 3.5 regarding cardinality, while also ensuring that the number of subsets handled by the pattern-level pruning is minimized. For example, in Figure 6(a), "RF_GHYPE" has a cardinality of 2, with only "Y" and "N" as values. Thus, the number of explanations generated by this type of attribute is small, allowing us to quickly obtain the top-$k$ explanations using pattern-level pruning.

Next, we use the local top-$k$ explanations to perform dimension-level pruning and order-level pruning on the attributes with larger cardinalities (line 9 to 10). We then combine the unpruned explanations with those generated from the smaller cardinality dimensions (line 12). Using pattern-level pruning again (line 13), we obtain the global top-$k$ explanations. For example, in Figure 6(b), after dimension and order pruning, only the explanations composed of "steroids" and "RF_CESARN" remain. We combine this set of explanations with the explanations from Figure 6(a) to form the explanation set $E$, as shown in Figure 6(c). We then perform pattern-level pruning on $E$ to obtain the final top-$k$ explanations $\mathcal{E}^*$.

Since unpruned explanations from both high-cardinality and low-cardinality attributes are combined, SDEcho ensures no potential global top-$k$ explanations are missed, pruning only those not included in the final top-$k$ results. Additionally, because the threshold is continuously updated, dimension partitioning does not directly impact pruning effectiveness. This partition-and-merge approach also facilitates the parallel expansion of SDEcho.

**Time Complexity Analysis.** Aligning with the notation used in the analysis of the naive method in Section 3.1, we begin by partitioning dimensions based on their cardinalities. For a subset of $a$ dimensions ($a < |\mathcal{A}|$), we calculate the projection distance of highest-order explanations in $O(r)$. Then, using a cube operation, we compute the projection distances of all explanations with a complexity of $O(2^a \cdot c^a)$. Based on Theorem 1, we prune to obtain the top-$k$ explanations costing $O(p \cdot r)$, where $p$ is the number of explanations requiring the exact explanation measure. Next, within the remaining $|\mathcal{A}| - a$ dimensions, we perform dimension pruning at $O((|\mathcal{A}|-a)r+(|\mathcal{A}|-a)c)$, reducing the number of dimensions to $A'$ ($A' < |\mathcal{A}| - a$). Assuming an estimated order $M$, order pruning then takes $O(A'^M \cdot c^{A'})$. The remaining $N$ explanations ($N \ll A'^M \cdot c^{A'}$) are combined with the previously pruned explanations, and the global top-$k$ explanations are obtained using the above steps with a cost of $O(2^a \cdot c^a \cdot N + p \cdot r)$. The overall time complexity of hybrid pruning is $O(2^a \cdot c^a \cdot N + ((p+|\mathcal{A}|-a)r + (|\mathcal{A}|-a)c + A'^M \cdot c^{A'}))$, effectively reducing exponential table scans to polynomial levels. Experiments validate that our pattern pruning using projection distance is highly effective, requiring only a small number of explanations $p$ to be computed with the exact explanation measure. Additionally,

columns with higher cardinality $c$ are more likely to be pruned in the dimension pruning step, further reducing $A'$. Ideally, according to practical considerations in [11], under $M = 3$, the complexity can be optimized to $O(2^a \cdot c^a \cdot N + ((p+|\mathcal{A}|-a)r + (|\mathcal{A}|-a)c + A'^3 \cdot c^{A'}))$.

### 3.7 Optimization for Independent Aggregates

Although independent aggregates lack the "removable" feature of incrementally removable aggregates and the aggregate result is determined by only a few tuples, making their complexity much smaller compared to incrementally removable aggregates, we still provide a computation strategy for independent aggregates such as `min` and `max` to quickly obtain the top-$k$ explanations without repeatedly scanning the table. We first identify all tuples on $R_1$ and $R_2$ that take the output value and project these tuples onto $\mathcal{A}$ as the highest-order explanations. These highest-order explanations can be used to enumerate all explanation candidates to calculate scores and output. The magnitude of the highest-order explanations is relatively low, making the computational complexity acceptable. Given explanations $e_1$ and $e_2$, with $\sigma_{e_1}(D) \subseteq \sigma_{e_2}(D)$, thus $(D - \sigma_{e_1}(D)) = (D - \sigma_{e_2}(D) \cup (\sigma_{e_2}(D) - \sigma_{e_1}(D)))$, we can utilize the following equation for further computational optimization:

$$\mathcal{F}(D - \sigma_{e_1}(D)) = \mathcal{F}(\mathcal{F}(D - \sigma_{e_2}(D)), \mathcal{F}(\sigma_{e_2}(D) - \sigma_{e_1}(D))) \quad (10)$$

By leveraging the inclusion relationships between explanations, we can utilize the results on known explanations and the differences between explanations to compute the independent aggregate results quickly. This approach eliminates the need to traverse the entire table, significantly reducing computational overhead.

### 3.8 Discussion

The design of SDEcho aims to provide top-$k$ explanations of sequence differences based on intervention-based methods, allowing users to delve deeper into the insights provided by its outputs. SDEcho can serve as a general component for computing pattern-based explanations, supporting explanation searches, and potentially extending to various customizations. For instance, SDEcho can be extended to support predicates with other operations, such as ">" or "<," simply by modifying the cube operations within the pattern-level pruning. Additionally, users can introduce user-specified diversity metrics (e.g., Simpson index and Shannon entropy)[19, 24, 56] to derive diverse top-$k$ explanations from the output of $2k$ explanations, with minimal time overhead through submodular optimization. Furthermore, users can leverage causal Directed Acyclic Graphs (DAGs) to incorporate knowledge and conduct rapid explanation searches on dimensions with causal relationships, thereby making explanations more logical.

## 4 EXPERIMENTS

We evaluate SDEcho by conducting comprehensive experiments to answer the following research questions:

- **RQ1:** Can SDEcho accurately yield explanations for the differences between aggregated sequences?
- **RQ2:** How does SDEcho scale with increasing data volume and dimension compared to existing work?
- **RQ3:** How can SDEcho facilitate end users in data analysis?

### 4.1 Datasets

The experiments are conducted on the following three widely used datasets[11, 45, 50].

**TPC-H.** We use *dbgen* and *qgen* from the TPC-H benchmark to generate instances and queries at scales {0.001, 0.005, 0.01, 0.05, 0.1}. This dataset benchmarks decision support systems with complex, business-oriented queries. For our experiments, we selected group-by queries and modified the where conditions to generate paired aggregation sequences, enabling us to evaluate SDEcho's scalability.

**CMS.** The Center for Medicare Studies (CMS) dataset[2] contains 1.1 million records (1GB) detailing payments from pharmaceutical and biotech companies to doctors. We focus on its temporal, financial, and demographic dimensions to generate paired aggregation sequences through queries, such as comparing monthly payments from 2021 and 2022. This helps assess our method's response speed under high dimensionality and data volume.

**Natality.** The 2022 U.S. Natality dataset from the National Center for Health Statistics (NCHS)[5] contains 3.6 million anonymized entries (4.6GB) with 233 attributes. These cover maternal details (e.g., race, smoking, medical conditions), delivery characteristics, and newborn health. We generated 100 pairs of aggregated sequences for our experiments, such as comparing MM cases across age ranges and races, as discussed in Section 1.

### 4.2 Benchmark

*Confidence in explanation.* To more fairly evaluate whether our explanation measure is better suited for aggregated sequence difference explanation scenarios compared to those used in other works[11, 15, 45, 53, 55], we conduct a quantitative analysis of the confidence of the explanations. To our best knowledge, there is no off-the-shelf benchmark for explaining the aggregated sequence difference. The lack of a gold standard for explanations poses significant challenges in this field of research[25]. To address this issue, we constructed a benchmark tailored to explain scenarios involving differences in aggregated sequences. In our benchmark, the ground truth consists of the tuples that cause the sequence differences. Since this paper focuses on pattern-based explanations, the ground truth is the pattern and its corresponding tuples.

Our high-level idea involves inserting tuples at the pattern level into two co-occurring sequences that initially have a distance of zero within the same frame. If these sequences exhibit a difference after the insertion, the inserted pattern (i.e., the tuples corresponding to that pattern) serves as the ground truth for explaining the differences between the two sequences. Considering that differences in real-world sequences may accumulate from the insertion of multiple patterns, we assess the contribution of each pattern by examining the distance between the sequences after each pattern is inserted independently. We treat the set of inserted patterns and their respective contributions as the ground truth. We constructed our benchmark using the TPC-H dataset and utilized the data generation tool *dbgen* to generate the tuple sets corresponding to the inserted patterns. We denote the inserted pattern and its contribution as $p_i$ and $\Delta_{p_i}$. Based on the sequence pairs generated through the above process and their corresponding inserted patterns, we consider how to compare our top-$k$ explanations for these two sequences with the inserted patterns.

As most explanation engines[11, 15, 35] focus on top-$k$ explanations, we draw inspiration from the information retrieval community. We refer to a commonly used measure for ranking quality in information retrieval, discounted cumulative gain (DCG)[29], to assess the similarity between our top-$k$ explanations and the corresponding tuples set of the ground truth, using $k$ as a weighting factor. Given the inserted patterns $P$ ($P = p_1, p_2, ..., p_m$) and the top-$k$ explanations $\mathcal{E}^*$ generated by the explanation engine, we employ the Jaccard index $\mathcal{J}$ to measure the similarity between the sets of tuples corresponding to different patterns. Our **explanation confidence (EC) score** is as follows:

$$EC(P, \mathcal{E}^*) = \sum_{i=1}^{m} \sum_{j=1}^{k} \frac{|\Delta_{p_i}| \mathcal{J}(e_j, p_i)}{\log_2(j+1)}. \tag{11}$$

To simplify the experiment, we ignore the interactions among inserted patterns, counting each independently inserted pattern's contribution when the aggregate sequence difference is 0. In this scenario, the rank 1 explanation measure still shows stronger explanatory ability than other measures. To fairly compare the explanatory abilities of different measures, we replace the Jaccard index with multiple measures such as **Precision** and **F1-score**. We evaluate the quality of explanations from various engines using these measures.

## 4.3 Baselines

We compare SDEcho against two baseline sets: one evaluates explanation confidence using different measures, and the other compares explanation search efficiency with explanation engines. Additionally, we compare SDEcho with a class of methods that first perform dimension reduction before explanation searching. We refer to this class of methods as the two-step method.

*Explanation Measures.* We organize commonly used explanation measures from existing work as baselines, including difference measures for analyzing disparities between two relations such as **Absolute Change**[15], **Risk Ratio**[11], **Diagnosis Cost**[53], and **Influence**[55]. Additionally, we include measures for explaining aggregate results of a single relation, such as **Aggravation**[45] and **Intervention**[45]. To create a single sequence, we subtract one aggregated sequence from another and apply these measures with parameter settings from the original papers.

*Explanation Engines.* **(1) BOE**[35] is an explanation engine that employs Bayesian optimization to search for explanations across user-defined explanation measures. **(2) DIFF**[11] is a relational aggregation operator that introduces logical and physical optimizations for explanation searches in high-dimensional, high-volume data. **(3) TSE**[15] identifies key performance indicators for a single aggregated time series. **(4)** Additionally, we implemented an equivalent explanation search effect as a baseline using SQL in **Postgres**. To ensure a fair comparison, all baselines will use the same explanation measure as SDEcho.

*Two-step methods.* Two-step methods first reduce dimensions and then search for explanations within the resulting dimensions. We surveyed two dimension reduction techniques: **(1) XInsight**[38], a multi-step explanation engine that learns causal graphs from data with faithfulness violations, using the output from its second module, Xtranslator, as the basis for explanation searching; and **(2)**

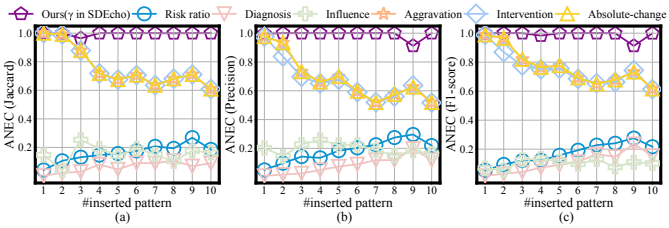


**Figure 7: Explanation confidence comparison with explanation measures baselines under different #inserted patterns.**

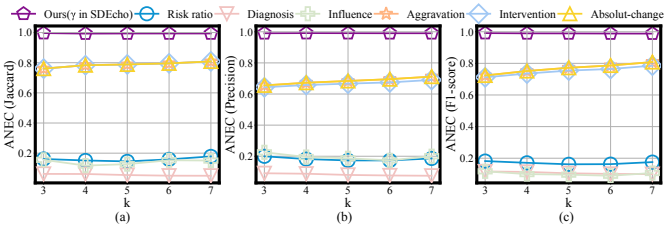


**Figure 8: Explanation confidence comparison with explanation measures baselines under different k.**

**PCA**[54], a standard dimension reduction method that transforms data into a new coordinate system aligned with maximum variance. We implemented Sparse PCA using scikit-learn[43].

All experiments are conducted on a computer running Ubuntu 18.04 64-bit, with an Intel(R) Xeon(R) E5-2620 v2 @ 2.10 GHz * 24 CPU, 64 GB RAM, and a 2 TB disk.

## 4.4 Quality of Top-$k$ Explanations (RQ1)

To evaluate the explanation confidence of SDEcho, we compare our measure $\gamma$ with baseline measures using the benchmark in Section 4.2. We analyze SDEcho's explanation confidence under varying #inserted patterns and assess the impact of $k$ on the quality of the top-$k$ explanation lists produced by nearly all explanation engines.

*4.4.1 Explanation Confidence on Varying #inserted Patterns and $k$.* For each query pair and its corresponding ground truth, we compute EC for each measure based on Equation(11), normalizing the results based on the highest confidence across all measures. The average normalized explanation confidence (ANEC) under varying #inserted patterns and $k$ is shown in Figure 7 and Figure 8. Our explanation measure consistently achieves the highest scores across different pattern similarity measures.

Explanation measures that only consider the impact of patterns on sequence difference, such as Intervention, Aggravation, and Absolute change, show a decline in explanation confidence relative to our explanation measure as #inserted patterns increase. This is because, as more patterns are inserted, the scenarios more closely mimic real-world situations that cause sequence differences. These measures do not account for the number of tuples associated with each explanation, which can lead to top-$k$ explanations that are redundant and contain many tuples, including those related to the ground truth, thus reducing explanation confidence. Measures like Risk ratio, Diagnosis-cost, and Influence are designed to explain

differences between two relations. To explain sequence differences, we aim to maximize the sum of explanation measures through point-by-point comparisons within the sequence pair. Our explanation measure accounts for both the impact of patterns on sequence differences and the number of tuples within those patterns. As a result, it provides concise explanations and demonstrates stable, robust explanatory power compared to other measures.

**Answer to RQ1:** SDEcho consistently demonstrates higher explanation confidence than other methods. It maintains superior performance even as #inserted pattern and $k$ vary. This stability highlights SDEcho's effectiveness in providing accurate explanations for differences between aggregated sequences.

## 4.5 Efficiency Evaluation of SDEcho (RQ2)

We evaluate SDEcho's performance by measuring query workload times across three datasets and analyzing the effects of the number of dimensions (#Dim) and explanation size ($k$) on search time. Additionally, we perform ablation studies on SDEcho's components and compare it with two-step methods. All experiments are subject to a three-hour time-out, marked by a dashed line.

*4.5.1 End-to-end Time Cost.* Since Postgres and TSE perform exact computations under our explanation measure (e.g., Postgres enumerates all possible explanations and outputs the top-$k$), while DIFF and BOE provide approximate solutions (e.g., DIFF finds up to third-order explanations through hard coding), Postgres and TSE can require significant time, with some query pairs taking over 3 hours to solve. Therefore, we extract a small-scale query workload from the three datasets for comparison with Postgres and TSE.

**Time Cost on Varying #Dim.** As shown in Figure 9(a), 9(b), and 9(c), we collected explanation search times across different datasets with varying #Dim (the number of explain-by attributes), setting $k$ = 5 for all methods. As #Dim increases, search times for all baselines significantly exceed those of SDEcho. Specifically, at #Dim = 6, our method offers a 30× and 600× speedup over TSE and Postgres, respectively. Even against approximate methods, SDEcho achieves a 10× and 60× speedup over DIFF and BOE, respectively, while maintaining exact computations at #Dim = 8. Moreover, when #Dim exceeds 10, most baselines time out, while SDEcho continues to operate efficiently.

**Time Cost on Varying $k$.** As shown in Figure 9(e), except for BOE, the change of $k$ does not have a significant impact on the performance of most methods (including SDEcho). This is because these methods primarily maintain a min-heap of size $k$, resulting in minimal overhead as $k$ ranges from 1 to 100. In contrast, for BOE, the sampling convergence time fluctuates with different output explanation sizes. As $k$ increases, the Tree-structured Parzen Estimator (TPE) in BOE requires more samples and iterations to ensure the accuracy of the top-$k$ list.

*4.5.2 Effectiveness of SDEcho Components.* **Time Cost on Varying #Dim.** As illustrated in Figure 10(a), we conducted an ablation study on SDEcho. Pattern-level pruning demonstrates the most substantial optimization. When #Dim $\leq$ 10, the explanation search time with only pattern-level pruning is comparable to that of SDEcho, underscoring its critical role. When #Dim exceeds 10, dimension-level pruning can eliminate dimensions with large cardinality, while order-level pruning removes higher-order explanations with minimal impact on sequence differences, enhancing search efficiency in high-dimensional data spaces.

*4.5.3 Scalability.* We generated TPC-H datasets of varying sizes using *dbgen* and evaluated the time costs of different explanation engines. As shown in Figure 9(d), SDEcho outperforms other engines, achieving efficiency improvements of ×23 and ×55 over approximate methods like DIFF and BOE at scale 0.1, and nearly ×10 over exact methods like Postgres and TSE. As shown in Figure 10(b), pattern-level pruning plays the most significant role in pruning.

*4.5.4 Comparison with Two-step Methods.* To investigate the relationship between two-step methods and SDEcho, we collected the time costs for both stages of the two-step methods: the dimension reduction phase and the explanation searching phase. Subsequently, we compared the explanation confidence of the top-$k$ explanations generated by these methods using our benchmark.

PCA's principal components are often difficult to interpret due to non-zero weights across all features. To enhance interpretability, we used sparse PCA, which retains high variance while ensuring each component includes only a few significant features. We performed dimension reduction on 11 dimensions using sparse PCA. Figure 11(a) shows the components extracted by sparse PCA. Although it is more sparse than PCA components, the explanation composed of the original dimensions is still difficult to map to the principal components. For example, when "components" is set to 3, cmpt_2 and cmpt_3 are both composed of the dimension set {partkey, shipdate, commitdate, receiptdate}. Since the original data has changed, the predicate combination composed of the principal components is difficult to understand and is not suitable for our explanation task.

As shown in Figure 11(b), we compared the efficiency and explanation confidence of the XInsight-based two-step method and SDEcho on TPC-H across various scales. With increasing scale, XInsight's training time rises significantly, surpassing SDEcho's explanation searching time. At scale = 0.05, XInsight times out. In contrast, SDEcho consistently incurs lower time costs while achieving higher explanation confidence for top-$k$ explanations. This is because the two-step method performs explanation searches within a limited explanation space after dimension reduction.

**Answer to RQ2:** Our experiments demonstrate that SDEcho significantly outperforms existing methods in efficiency and scalability. It achieves substantial speedups as #Dim increases, maintaining high efficiency even as data volume scales. Compared to two-step methods, SDEcho provides accurate and rapid explanations online. Our ablation studies highlight the importance of pattern-level pruning, especially in high-dimensional contexts, reinforcing SDEcho's advantages for large-scale data analysis.

## 4.6 Empirical Studies (RQ3)

This section presents a user study and case study utilizing the Natality dataset. We invited external experts to evaluate SDEcho's utility and provide a case study demonstrating how SDEcho-derived answers to "why" questions can relate to actual research outcomes.

*4.6.1 Case Study.* We used SDEcho to address Oliver's problem discussed in Section 1. Figure 12 demonstrates the usage of SDEcho, showcasing a portion of its outputs. By revealing how SDEcho's

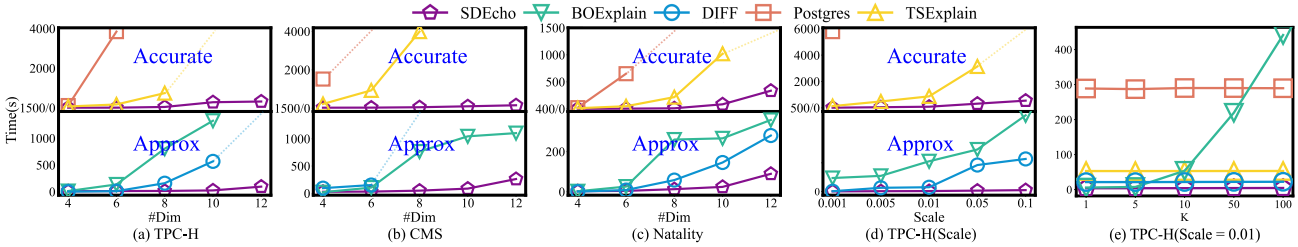


**Figure 9: Efficiency comparison with explanation engine baselines across various datasets.**

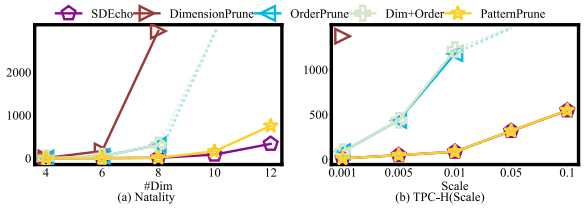


**Figure 10: Ablation study of SDEcho.**

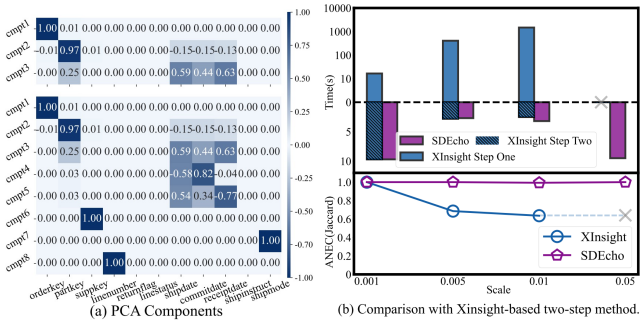


**Figure 11: Comparison with two-step methods. (×: time-out)**

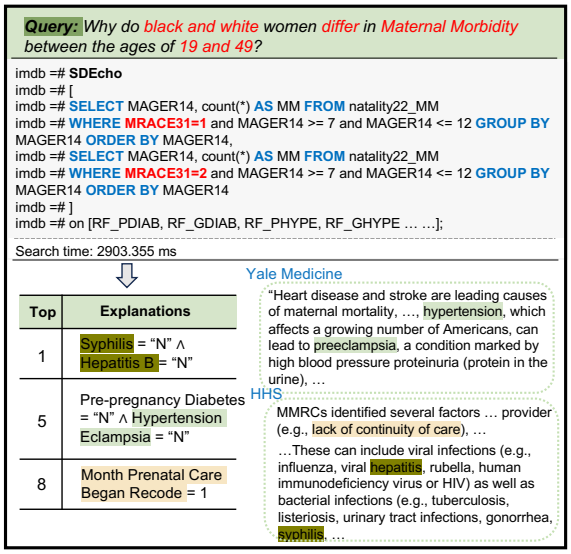


**Figure 12: A real-world case of SDEcho**

output aligns with conclusions from relevant real-world studies, we illustrate that SDEcho can assist users in conducting sequence comparison analysis.

As discussed in Section 1, due to the high dimensionality of the natality22_MM dataset, the user randomly selected several dimension sets, such as Risk Factors, Infections Present, etc., encompassing a total of 18 dimensions like F_PDIAB and RF_GDIAB. Users only need to use the SDEcho operator in Postgres, specifying the queries corresponding to the two sequences to be compared and the explain-by attributes. SDEcho returns the top 10 explanations in just 2.9s. We found that some of SDEcho's outputs are similar to research reports on this issue published by institutions like Yale Medicine[9] and the U.S. Department of Health & Human Services[8]. SDEcho enables users to perform aggregated sequence comparison analysis efficiently, requiring no prior knowledge.

*4.6.2 User Study.* We conducted a user study for the Natality dataset to investigate (S1) whether SDEcho provides meaningful explanations and (S2) whether SDEcho reduces the amount of time a user needs to find explanations.

**Participants.** There were 20 participants[1]—6 with medical backgrounds and 14 from non-medical fields. All participants had prior SQL experience, enabling them to complete the tasks. The six with medical backgrounds formed the "expert" group, while the remaining 14 were randomly assigned to the "treatment" or "control" groups, with 7 participants in each.

**Tasks.** We first introduced all participants to the Natality schema and explained the meaning of each field. We prepared six pairs of aggregated sequences derived from SQL, along with corresponding "why" questions. Participants were shown the visualized results and raw data, then asked to find and submit the top-5 explanations for each question within 20 minutes.

Only the "treatment" group received the top-10 shuffled explanations generated by SDEcho beforehand and were informed of their potential inaccuracy, emphasizing the need for verification through their own queries.

After completing the tasks, the "expert" group was provided with the top-5 explanations generated by SDEcho for each question. To

[1]To address potential ethical concerns, we ensured that all participants were fully informed about the scope and purpose of the study and obtained their explicit consent for the use of the collected data in our ongoing research. Additionally, we avoided using any privacy-sensitive devices, such as cameras, and anonymized all data to protect participants' privacy.

**Table 1: User study results. Note: "Rel." = relevance rating, "Und." = understandability rating, "Con." = conciseness rating, "Avg." = average rating, Time (in minutes).**

| | | Q1 | Q2 | Q3 | Q4 | Q5 | Q6 |
|---|---|---|---|---|---|---|---|
| SDEcho | Rel. | 4.1±0.6 | 4.3±0.5 | 4.1±0.6 | 3.9±0.5 | 4.4±0.3 | 4.2±0.5 |
| | Und. | 3.9±0.4 | 3.8±0.4 | 3.9±0.6 | 3.6±0.5 | 3.9±0.6 | 4.1±0.7 |
| | Con. | 4.3±0.3 | 4.2±0.4 | 4.0±0.4 | 3.9±0.4 | 4.1±0.3 | 4.2±0.4 |
| Treatment | Avg | 3.8±0.6 | 3.8±0.8 | 3.4±0.6 | 3.7±0.5 | 3.8±0.6 | 3.9±0.7 |
| Control | Avg | 3.0±0.5 | 2.7±0.7 | 2.0±0.6 | 1.8±0.8 | 2.4±0.5 | 2.7±0.5 |
| Treatment | Time | 5.1±3.5 | 7.6±2.1 | 9.3±2.0 | 8.9±1.1 | 6.5±2.6 | 5.9±2.8 |
| Control | Time | 17.3±2.7 | 16.4±2.5 | 17.2±2.7 | 17.2±2.8 | 18.0±2.0 | 16.7±3.1 |

answer S1, following [34, 37, 38, 42], we asked each participant in "expert" group to rate SDEcho's output on three criteria: "relevance", "understandability", and "conciseness" using scale from 1 (strongly disagree) to 5 (strongly agree), along with outputs from the treatment and control groups. To answer S2, we recorded the actual submission times for both the treatment and control groups.

**Results and Analysis.** All results are shown in Table 1, where each cell presents the data in the form of (mean ± standard deviation). Overall, the responses were positive: the expert group's average rating across all questions was 4.05±0.47, indicating that the experts agreed that the explanations provided by SDEcho were helpful for data analysis and could guide further research. However, we observed that the "Understandability" rating was slightly lower compared to the other criteria. This suggests that some explanations were not intuitively understandable through patterns alone and seemed counter-intuitive. In future work, we plan to incorporate causal relationships to better assist users in understanding the explanations[46, 57].

The treatment group scored significantly higher than the control group and took less time, indicating that SDEcho's explanations helped users without domain expertise in data analysis. In contrast, control group members spent more time manually testing dimension combinations and selecting patterns for validation.

**Answer to RQ3:** SDEcho generates plausible explanations consistent with expert knowledge and reduces the time users spend searching for explanations, assisting them in data analysis to address the "why" questions of aggregation sequence pairs.

## 5 RELATED WORK

**SQL Explanation Engine.** In recent years, explanation engines have garnered widespread attention in the data management community. From the perspective of the objects to be explained, existing works can be categorized into explanations for one aggregated value [27, 30], differences between relations[10, 11, 34, 35, 38, 40, 42, 44, 45, 47, 48, 53, 55], and time-series[15, 32, 33, 51]. However, as discussed earlier, contrast-based explanation engines cannot support scenarios involving sequence differences. Moreover, explanation engines like [15], which focus on key performance indicators in time-series models, typically analyze changes between adjacent data points within a single sequence. This limits their ability to explain differences between two sequences, even when provided with the difference sequence. In addition, causal-based explanations[46, 57] have gained popularity in the data management community due to their potential to reveal causal relationships and distinguish between correlations and cause-effect relationships. However, unlike these causal methods, our data-centric approach can provide real-time explanations without relying on background knowledge encoded in a causal DAG. Thus, our work not only supports explanations for sequence differences but also emphasizes accelerating explanation searches in high-volume, high-dimensional data sources while ensuring accurate outputs.

**Visual Analytics.** There has been work on visual analysis where comparing subsets or groups of tuples to find the relevant ones is a common theme. Among these, [20, 49, 52] perform subset comparisons in the form of middleware, resulting in significant data movement overhead, which leads to latency and poor scalability. PowerBI[6] and Tableau[7] support SQL query interfaces, reducing data transfer, but users currently need to write complex SQL queries or generate all possible visualizations and manually compare them. [48], as an in-database execution operator, allows users to customize distance metrics between trends to compare trends or trend sets. However, compared to SDEcho, it does not support the explanation of sequence differences. Additionally, visualization tools such as DeepEye [36] and Table2Charts [58] focus on discovering relationships between dimensions and selecting appropriate chart types for visualization, which is orthogonal to our objective of providing targeted explanations for sequence differences. Unlike these visualization tools, SDEcho employs a deletion-based method, which does not require assessing the similarity of observed sequences and patterns from a shape perspective.

**Provenance.** Relational query provenance[14] identifies the input that contributes to the query result. For non-aggregate queries[16, 26], why-provenance returns a set of input tuples responsible for a given output tuple, while how-provenance encodes how a query combines input tuples to produce an answer. For aggregate queries [12], symbolic expressions based on semiring extensions are used to represent how the aggregate result is computed. Unlike the fine-grained explanation approaches mentioned above, which output a set of tuples, most existing explanation engines, including our work, provide pattern-based explanations[18], represented by selection predicates. These patterns are concise summaries of the corresponding tuples. Unlike data provenance work[42], our explanations are derived from the tuples in the provenance.

## 6 CONCLUSION AND FUTURE WORK

This paper presented a novel framework, SDEcho, for explaining differences in aggregate sequences. SDEcho employs pruning from pattern, order, and dimension perspectives, offering a hybrid optimization approach that ensures concise and accurate explanations with theoretical guarantees. Through comprehensive experimental evaluation, we demonstrated that SDEcho outperforms existing methods significantly in terms of both effectiveness and efficiency. Several directions are open for future work, including extending support for more sequence distance metrics, such as those used for comparing probability distributions.

## ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China (Grants No. 62272106 and No. 62072113).

# REFERENCES


[1] 2024. *CDC.* https://www.cdc.gov/nchs Last accessed: 2024-12-16.

[2] 2024. *CMS.* https://www.cms.gov/priorities/key-initiatives/open-payments/data Last accessed: 2024-5-1.

[3] 2024. *Google Trend.* https://trends.google.com/trends/explore?q=%2Fg%2F11j0_8y5xw,%2Fg%2F11c75ypgws Last accessed: 2024-12-16.

[4] 2024. *Imply.* https://docs.imply.io/latest/explain/ Last accessed: 2024-12-16.

[5] 2024. *Natality Dataset.* https://www.cdc.gov/nchs/data_access/VitalStatsOnline.htm#Births Last accessed: 2024-12-16.

[6] 2024. *Power BI.* https://learn.microsoft.com/en-us/power-bi/visuals/power-bi-visualization-influencers?tabs=powerbi-desktop Last accessed: 2024-12-16.

[7] 2024. *Tableau.* https://help.tableau.com/current/pro/desktop/en-us/explain_data_basics.htm Last accessed: 2024-12-16.

[8] 2024. *U.S. Department of Health & Human Services.* https://www.hhs.gov/sites/default/files/call-to-action-maternal-health.pdf Last accessed: 2024-12-16.

[9] 2024. *Yale Medicine.* https://www.yalemedicine.org/news/maternal-mortality-on-the-rise Last accessed: 2024-12-16.

[10] Firas Abuzaid, Peter Bailis, Jialin Ding, Edward Gan, Samuel Madden, Deepak Narayanan, Kexin Rong, and Sahaana Suri. 2018. Macrobase: Prioritizing attention in fast data. *ACM Transactions on Database Systems (TODS)* 43, 4 (2018), 1–45.

[11] Firas Abuzaid, Peter Kraft, Sahaana Suri, Edward Gan, Eric Xu, Atul Shenoy, Asvin Ananthanarayan, John Sheu, Erik Meijer, Xi Wu, et al. 2018. Diff: a relational interface for large-scale data explanation. *Proceedings of the VLDB Endowment* 12, 4 (2018), 419–432.

[12] Yael Amsterdamer, Daniel Deutch, and Val Tannen. 2011. Provenance for aggregate queries. In *Proceedings of the thirtieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. 153–164.

[13] Christopher M Bishop and Nasser M Nasrabadi. 2006. *Pattern recognition and machine learning*. Springer.

[14] Peter Buneman, Sanjeev Khanna, and Wang-Chiew Tan. 2002. On propagation of deletions and annotations through views. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. 150–158.

[15] Yiru Chen and Silu Huang. 2023. TSExplain: Explaining Aggregated Time Series by Surfacing Evolving Contributors. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*. 708–720.

[16] James Cheney, Laura Chiticariu, Wang-Chiew Tan, et al. 2009. Provenance in databases: Why, how, and where. *Foundations and Trends® in Databases* 1, 4 (2009), 379–474.

[17] Roni Copul, Nave Frost, Tova Milo, and Kathy Razmadze. 2024. TabEE: Tabular Embeddings Explanations. *Proceedings of the ACM on Management of Data* 2, 1 (2024), 1–26.

[18] Vargha Dadvar, Lukasz Golab, and Divesh Srivastava. 2022. Exploring Data Using Patterns: A Survey. *Information Systems* 108, C (2022), 11.

[19] Daniel Deutch, Amir Gilad, Tova Milo, Amit Mualem, and Amit Somech. 2022. FEDEX: An Explainability Framework for Data Exploration Steps. *Proceedings of the VLDB Endowment* 15, 13 (2022), 3854–3868.

[20] Rui Ding, Shi Han, Yong Xu, Haidong Zhang, and Dongmei Zhang. 2019. Quicklinsights: Quick and automatic discovery of insights from multi-dimensional data. In *Proceedings of the 2019 international conference on management of data*. 317–332.

[21] Deepa Dongarwar, Danyal Tahseen, Liye Wang, Muktar H Aliyu, and Hamisu M Salihu. 2021. Temporal trends in preterm birth phenotypes by plurality: Black–White disparity over half a century. *Journal of Perinatology* 41, 2 (2021), 204–211.

[22] Richard O Duda, Peter E Hart, et al. 2006. *Pattern classification*. John Wiley & Sons.

[23] Yuankai Fan, Tonghui Ren, Can Huang, Beini Zheng, Yinan Jing, Zhenying He, Jinbao Li, and Jianxin Li. 2024. A confidence-based knowledge integration framework for cross-domain table question answering. *Knowledge-Based Systems* 306 (2024), 112718.

[24] Liqiang Geng and Howard J Hamilton. 2006. Interestingness measures for data mining: A survey. *ACM Computing Surveys (CSUR)* 38, 3 (2006), 9–es.

[25] Boris Glavic, Alexandra Meliou, and Sudeepa Roy. 2021. Trends in explanations: Understanding and debugging data-driven systems. *Foundations and Trends® in Databases* 11, 3 (2021).

[26] Melanie Herschel, Ralf Diestelkämper, and Houssem Ben Lahmar. 2017. A survey on provenance: What for? What form? What from? *The VLDB Journal* 26, 6 (2017), 881–906.

[27] Zezhou Huang and Eugene Wu. 2022. Reptile: Aggregation-Level Explanations for Hierarchical Data. In *Proceedings of the 2022 International Conference on Management of Data*. 399–413.

[28] John H Huber, Mengmeng Ji, Yi-Hsuan Shih, Mei Wang, Graham Colditz, and Su-Hsin Chang. 2023. Disentangling age, gender, and racial/ethnic disparities in multiple myeloma burden: a modeling study. *Nature communications* 14, 1 (2023), 5768.

[29] Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (TOIS)* 20, 4 (2002), 422–446.

[30] Manas Joglekar, Hector Garcia-Molina, and Aditya Parameswaran. 2015. Smart drill-down: A new data exploration operator. In *Proceedings of the VLDB Endowment International Conference on Very Large Data Bases*, Vol. 8. 1928.

[31] Nicole Blair Johnson, Locola D Hayes, Kathryn Brown, Elizabeth C Hoo, Kathleen A Ethier, Centers for Disease Control, Prevention (CDC), et al. 2014. CDC National Health Report: leading causes of morbidity and mortality and associated behavioral risk and protective factors–United States, 2005-2013. *MMWR suppl* 63, 4 (2014), 3–27.

[32] Jokin Labaien, Ekhi Zugasti, and Xabier De Carlos. 2020. Contrastive explanations for a deep learning model on time-series data. In *International Conference on Big Data Analytics and Knowledge Discovery*. 235–244.

[33] Kin Kwan Leung, Clayton Rooke, Jonathan Smith, Saba Zuberi, and Maksims Volkovs. 2023. Temporal Dependencies in Feature Importance for Time Series Predictions. arXiv:2107.14317

[34] Chenjie Li, Zhengjie Miao, Qitian Zeng, Boris Glavic, and Sudeepa Roy. 2021. Putting things into context: Rich explanations for query answers using join graphs. In *Proceedings of the 2021 International Conference on Management of Data*. 1051–1063.

[35] Brandon Lockhart, Jinglin Peng, Weiyuan Wu, Jiannan Wang, and Eugene Wu. 2021. Explaining Inference Queries with Bayesian Optimization. *Proceedings of the VLDB Endowment* 14, 11 (2021), 2576–2585.

[36] Yuyu Luo, Xuedi Qin, Nan Tang, and Guoliang Li. 2018. DeepEye: Towards Automatic Data Visualization. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. 101–112.

[37] Pingchuan Ma, Rui Ding, Shuai Wang, Shi Han, and Dongmei Zhang. 2023. InsightPilot: An LLM-empowered automated data exploration system. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. 346–352.

[38] Pingchuan Ma, Rui Ding, Shuai Wang, Shi Han, and Dongmei Zhang. 2023. XInsight: EXplainable Data Analysis Through The Lens of Causality. *Proceedings of the ACM on Management of Data* 1, 2 (2023), 27.

[39] Alexandra Meliou, Sudeepa Roy, and Dan Suciu. 2014. Causality and explanations in databases. *Proceedings of the VLDB Endowment* 7, 13 (2014), 1715–1716.

[40] Zhengjie Miao, Andrew Lee, and Sudeepa Roy. 2019. LensXPlain: Visualizing and explaining contributing subsets for aggregate query answers. *Proceedings of the VLDB Endowment* 12, 12 (2019), 1898–1901.

[41] Zhengjie Miao, Sudeepa Roy, and Jun Yang. 2019. Explaining wrong queries using small examples. In *Proceedings of the 2019 International Conference on Management of Data*. 503–520.

[42] Zhengjie Miao, Qitian Zeng, Boris Glavic, and Sudeepa Roy. 2019. Going beyond provenance: Explaining query answers with pattern-based counterbalances. In *Proceedings of the 2019 International Conference on Management of Data*. 485–502.

[43] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research* 12 (2011), 2825–2830.

[44] Sudeepa Roy, Laurel Orr, and Dan Suciu. 2015. Explaining query answers with explanation-ready databases. *Proceedings of the VLDB Endowment* 9, 4 (2015), 348–359.

[45] Sudeepa Roy and Dan Suciu. 2014. A formal approach to finding explanations for database queries. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*. 1579–1590.

[46] Babak Salimi, Johannes Gehrke, and Dan Suciu. 2018. Bias in OLAP queries: Detection, explanation, and removal. In *Proceedings of the 2018 International Conference on Management of Data*. 1021–1035.

[47] Sunita Sarawagi. 2001. idiff: Informative summarization of differences in multidimensional aggregates. *Data Mining and Knowledge Discovery* 5 (2001), 255–276.

[48] Tarique Siddiqui, Surajit Chaudhuri, and Vivek Narasayya. 2021. COMPARE: Accelerating Groupwise Comparison in Relational Databases for Data Analytics. *Proceedings of the VLDB Endowment* 14, 11 (2021), 2419–2431.

[49] Tarique Siddiqui, Albert Kim, John Lee, Karrie Karahalios, and Aditya Parameswaran. 2016. Effortless data exploration with zenvisage: an expressive and interactive visual analytics system. *Proceedings of the VLDB Endowment* 10, 4 (2016), 457–468.

[50] Yuchao Tao, Xi He, Ashwin Machanavajjhala, and Sudeepa Roy. 2020. Computing local sensitivities of counting queries with joins. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 479–494.

[51] Sana Tonekaboni, Shalmali Joshi, Kieran Campbell, David K Duvenaud, and Anna Goldenberg. 2020. What went wrong and when? Instance-wise feature importance for time-series black-box models. *Advances in Neural Information Processing Systems* 33 (2020), 799–809.

[52] Manasi Vartak, Sajjadur Rahman, Samuel Madden, Aditya Parameswaran, and Neoklis Polyzotis. 2015. SeeDB: efficient data-driven visualization recommendations to support visual analytics. *Proceedings of the VLDB Endowment* 8, 13 (2015), 2182–2193.

[53] Xiaolan Wang, Xin Luna Dong, and Alexandra Meliou. 2015. Data x-ray: A diagnostic tool for data errors. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*. 1231–1245.

[54] Svante Wold, Kim Esbensen, and Paul Geladi. 1987. Principal component analysis. *Chemometrics and intelligent laboratory systems* 2, 1-3 (1987), 37–52.
[55] Eugene Wu and Samuel Madden. 2013. Scorpion: Explaining Away Outliers in Aggregate Queries. *Proceedings of the VLDB Endowment* 6, 8 (2013), 553–564.
[56] Brit Youngmann, Sihem Amer-Yahia, and Aurelien Personnaz. 2022. Guided exploration of data summaries. *Proceedings of the VLDB Endowment* 15 (2022), 1798–1807.
[57] Brit Youngmann, Michael Cafarella, Amir Gilad, and Sudeepa Roy. 2024. Summarized Causal Explanations For Aggregate Views. *Proceedings of the ACM on Management of Data* 2, 1 (2024), 1–27.
[58] Mengyu Zhou, Qingtao Li, Xinyi He, Yuejiang Li, Yibo Liu, Wei Ji, Shi Han, Yining Chen, Daxin Jiang, and Dongmei Zhang. 2021. Table2Charts: recommending charts by learning shared table representations. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 2389–2399.