Explaining aggregate database query answers using alternative attribute values

Final report

Akbota Anuarbek
akbota.anuarbek@duke.edu

1 Introduction

Current technologies permit collection and storage of very large amounts of data across different fields. This incredibly big amounts of data are in turn analyzed by users having different backgrounds for various purposes. However, in all of those cases users face the same challenge of efficiently finding an explanation for some particular patterns observed in the data. As a result, the topic of finding explanations for database query answers is gaining an increasing consideration of researchers in the database field recently.

Recent research papers on the topic focus on visualizing aggregate query answers using some user interfaces and then explaining unexpected patterns in the answers by conjunction of predicates [4, 5]. Such conjunction of predicates are found either based on *aggravation*, restricting the database only to the tuples that further aggravate the situation, or *intervention*, removal of the records that change the answer to the query significantly. For example, after observing the answer for a query the user asks a question *"Why the number of SIGMOD publications is extremely high during the period from 2000 to 2010?"* Then the conjunction of predicates suggested as an example could be [**institution** = 'Duke' $\wedge$ **year**= '2004']. Based on this explanation the number of SIGMOD publications is high because researchers at Duke published intensively during the year 2004.

An alternative way of finding explanations to aggregate query answers can be finding conjunction of predicates that are constructed by either changing values of one or a few attributes or completely removing some attribute specified in the *where* clause of the query. Suppose after receiving the answer for an aggregate query the user asks *"why the person named John Brown from MIT published a very few number of papers in Database in 2007?"*. A possible explanation to such unexpected query answer could be *"because John published more papers in Machine learning in 2007"*. Another valid explanation could be *"because John in general published less in 2007 in all areas"*. Such approach of explaining unusual query answers is different from previously researched methods because it considers records that are disjoint from the records selected by the original query answer, whereas the previous research focused on finding outliers among the selected records.

This paper will first describe similar research in the area, then proceed by formally defining the problem and giving some expamples. It further introduces the algorithm for performing query answer explanation, as well as discussing its implementation and qualitatively evaluating its performance on the real world dataset. Finally, it will conclude by giving ideas for possible future research.

2 Related Work

Most of the previous research on the topic concentrated on a single table and queries not containing joins [2, 3, 1, 6]. However, most informative and interesting queries on databases are obtained by integrating several tables. Hereby, S. Roy and D. Suciu [4] propose a way of efficiently explaining complex queries with joins by choosing explanations based on their unique minimal intervention. For that, degrees of all candidate explanations are computed and top k explanations with the highest degree values are returned to the user. S. Roy, L. Orr, and D. Suciu [5] present a framework that allows to explain query answers in more sophisticated way by preparing the database offline.

The approach considered in this paper also supports queries on several tables. However, the

method of finding conjuncture of predicates is related to tuples that are disjoint from the tuples provided as an answer to the query, which is different from methods discussed in [4, 5].

3 Problem Definition

3.1 Formal definition

Let D be a database with relations $R_1, ..., R_n$, where relations can be joined by foreign key-primay key relationships. Let $\mathcal{R} \subset \{R_1, ..., R_n\}$ be some subset of relations in the database D. Also, suppose $\{a_0, a_1, a_2, ..., a_r\}$ is some subset of attributes from relations in $\mathcal{R}$. Here a_0 is an attribute corresponding to some entity, usually a primary key, and our aggregate query of interested is going to be about this entity. In our earlier example this corresponds to a person named John, where we were interested in the number of his publications. a_1 corresponds to a time attribute such as datetime, month, year, etc and finally each of a_i, $i \geq 2$ corresponds to some other attributes.

Now let Q^* be an answer to a an SQL query of the following form:

SELECT $agg(u)$
FROM $\mathcal{R}$
WHERE {join predicates} *AND* $a_0 = v_0$ *AND* ... *AND* $a_r = v_r$

where v_i is a value from $dom(a_i)$, u corresponds either to a single attribute or a tuple, $agg()$ corresponds to an aggregate function like count(), sum(), min(), max(), or average(), and {*join predicates*} are f.k.-p.k. equalities that join different relations.

Then we can define a user question in a similar way as defined in [4].

Definition 3.1. *A* ***user question*** *is a triple* (Q^*, dir, a_0)*, where (i)* Q^* *is an answer to a query, and (ii)* $dir = \{high, low\}$ *is a direction, such that the user wants to know why the value of* Q *corresponding to an entity* $a_0 = v_0$ *is too high if* $dir = high$ *or too low if* $dir = low$*.*

Given a user question (Q^*, dir, a_0), the system returns a list of candidate explanations, where each explanation is denoted by ϕ.

Definition 3.2. *An* ***explanation*** ϕ *is a conjunction of predicates on attributes* $\phi = \wedge_{i=1,...,r'}\phi_i = \wedge_{i=1,...,r'}[a_i = \tilde{v}_i]$*, where* $\tilde{v}_i \in dom(a_i)$, $\forall i$*, and* $\{a_1, ..., a'_r\}$ *is the subset of attributes in the where clause of the original query, such that for* $r - 1$ *of the attributes* $\tilde{v}_i = v_i$ *and for one of the attributes* $v_i \neq \tilde{v}_i$ *or that attribute is completely removed.*

Now for each query Q^* and an explanation ϕ let's define an *explanatory query* Q by changing the value of or completely removing one of the attributes a_i, $i = 1, ..., r$ from Q^*. This will correspond to the following SQL query:

SELECT $agg(u)$
FROM $\mathcal{R}$
WHERE {*join predicates*} AND $a_0 = v_0$ AND $a_1 = \tilde{v}_1$ AND...AND $a_r = \tilde{v}_r$

where $\tilde{v}_i \in dom(a_i)$, $\forall i$ and there exists one $\hat{i}$ such that $v_{\hat{i}} \neq \tilde{v}_{\hat{i}}$ or such that attribute a_i is removed.

Finally before we move on to explaining our explanation criteria, for each aggregate query Q define an *average query across time* $\bar{Q}_t$ by removing the time attribute in Q and dividing the obtained answer by the number of periods considered. This can be calculated as the following SQL query:

WITH CountsByPeriod AS
(
SELECT a_1*,* $count(*)$ *AS cnt*
FROM $\mathcal{R}$
WHERE {join predicates} *AND* $a_0 = v_0$ *AND* $a_2 = \tilde{v}_2$... *AND* $a_r = \tilde{v}_r$
GROUP BY a_1
)
SELECT avg(cnt)
FROM CountsByPeriod

Now we are ready to introduce our explanation criteria.

Definition 3.3. *Given a database* D *and a user question* (Q^*, dir, a_0)*, the degree of* ***inverse*** *candidate explanation* ϕ *is defined as*

$$\mu_{inv}(D, Q^*, dir, a_0) = \begin{cases} \frac{Q(D)}{\bar{Q}_t(D)} & \text{if dir=low} \\ -\frac{Q(D)}{\bar{Q}_t(D)} & \text{if dir=high} \end{cases}$$

Definition 3.4. *Given a database* D *and a user question* (Q^*, dir, a_0)*, the degree of* ***direct*** *candidate explanation* ϕ *is defined as*

$$\mu_{dir}(D, Q^*, dir, a_0) = \begin{cases} -\frac{Q(D)}{\bar{Q}_t(D)} & \text{if dir=low} \\ \frac{Q(D)}{\bar{Q}_t(D)} & \text{if dir=high} \end{cases}$$

In other words, given an answer to the query Q^* and dir, we try to find explanations that have either very high or very low value. It is important to consider both directions high and low because query answers might have both direct and inverse relationships. By considering the extremes of both high and low, we try to find the best explanations. Dividing by the mean of the query across time is important for normalization, since different query values might have different scales and comparing them without normalization is not very effective.

3.2 Examples

Now let's look at some examples. Consider a database D on card transactions with following relations:

Consumers(consumerID, cardNumber, name, expDate, address)
Transactions(transactionID, consumerID, cardNumber, amount, typeOfSpending, month)

Also consider a query Q^*, which returns total amount spent by Anne on furniture during February:

SELECT sum(amount)
FROM Consumers C, Transactions T
WHERE C.consumerID=T.consumerID AND name='Anne' AND month='Feb' AND typeOfSpending='furniture'

Now assume a user asks *"Why Anne's spending on furniture is so high in Feb?"*. Formally, a user question is $(Q^*, low, name)$.

Then one of the possible **direct** explanations could be $\phi = [\textbf{month} =' Feb' \wedge \textbf{typeOfSpending} =' housing']$. Corresponding Q is:

SELECT sum(amount)
FROM Consumers C, Transactions T
WHERE C.consumerID=T.consumerID AND name='Anne' AND month='Feb' AND typeOfSpending='housing'

In this case there is a direct relationship between Q^* and Q, high value of Q^* is explained by a high value of $Q/\bar{Q}_t$. This is indeed intuitive in this example: *"Since Anne bought a house, she needs to buy furniture"*

Now let's consider another example where explanation is found by removing one of the attributes from the where clause of the original query Q^* given by:

SELECT sum(amount)

FROM Consumers C, Transactions T

WHERE C.consumerID=T.consumerID AND name='Anne' AND month='Jan' AND typeOfSpending='clothing'

So user asks why *"Why Anne's spending on clothing is low in January?"*, i.e. $(Q^*, low, name)$.

Then one of the possible explanations which also has a direct relationship could be $\phi = [\textbf{month} =' Jan']$. Suppose if $Q/\bar{Q}_t$ is high for ϕ, this means that not only Anne's spending on clothing is high, but also her total spending over its average across time is also unusually high. This gives us room for additional guesses, maybe because her salary increased.

In the two examples above we considered explanations by either changing the value of one of the attributes or completely removing the attribute. In both cases we came up with a new query Q and divided its answer over its average across time $\bar{Q}_t$. One important assumption we should make now is only consider past values of time t in both cases: while computing the averages across time and also when substituting the value of a_1 to a different time period. Since we are looking at causal explanations, (of course we do not claim that this is an exact causal relationship which can be only verified by making actual experiments), explaining past values based on future values does not seem logical. Also it might be crusial to put some limit on how far we can go when we are considering earlier periods, because events that have happened too far in the past usually have very small effect on the current values, and they might also introduce some bias to the average. And finally, we do not remove the time attribute when trying to find explanations.

4 Algorithm/Implementation

Our method can be described by the following algorithm.

Algorithm

Input: (Q^*, dir, a_0)

Step 1: *Check if it is indeed an outlier:*

- if $Q^* < \bar{Q}_t^*$ ($Q^* > \bar{Q}_t$) and $dir = high(low)$
  - ***return*** not an outlier
- if $Q^* > \bar{Q}_t^*$ ($Q^* < \bar{Q}_t$) and $dir = high(low)$

Step 2: *Look for candidate explanations:*

- $\forall Q$ that differ from Q^* in one attribute value compute $Q/\bar{Q}_t$
- ***return*** top $k/2$ ϕ_{dir} and top $k/2$ ϕ_{inv}

So the first step is to check if the question is actually meaningful, for instance if the input is $(Q^*, high, a_0)$ we need to confirm that the value of Q^* is higher than its average across time. If this is indeed true the algorithm will return $k/2$ explanations that are have the highest value compared to their average and return another $k/2$ that are the lowest compared to their average.

4.1 Implementation

Values of all possible explanatory queries and their averages across time can be calculated at once just by running the GROUP BY CUBE query. To calculate the average quantities across time we simply need to divide total sums over all periods by the number of periods. Let's revisit the first example with clothing. GROUP BY CUBE query that of interest is given by:

SELECT month, typeOfSpending, sum(amount)
FROM Consumers C, Transactions T
WHERE C.consumerID=T.consumerID AND name='Anne'
GROUP BY CUBE(month, typeOfSpending)

5 Experiments and Results

5.1 DBLP Dataset

The DBLP dataset, which contains a large amount of data about publications in different areas of computer science and their authors, was used for experiments. In particular two aggregate queries that will be analyzed in the next subsection were based on Inproceedings and Authorship tables.

Inproceedings (pubkey, title, booktitle, year, area)

Authorship (authorname, pubkey)

Inproceedings table contains 1712660 records, and Authorship table contains 8806789 records.

5.1.1 Qualitative/Performance Evaluation

Let's consider the follwoing query which returns the number of papers published in Theory by Jeffrey Scott Vitter during the period 1995-1999.

```
SELECT count(*)
FROM Authorship A, Inproceedings I
WHERE I.pubkey=A.pubkey AND
author='Jeffrey Scott Vitter' AND
area = 'Theory' AND floor(year/5)*5=1995;
```

The number returned by the query is 12, and suppose the user thinks that this amount is too high and provides $(Q^*, high, author)$ as an input to the algorithm. Then the algorithm runs the following GROUP BY CUBE query.

```
SELECT floor(year/5)*5 AS decade_st, area, count(*)
FROM Authorship A, Inproceedings I
WHERE I.pubkey=A.pubkey AND
author='Jeffrey Scott Vitter'
AND area<>'UNKNOWN' AND floor(year/5)*5<=1995
GROUP BY CUBE(decade_st, area)
ORDER BY decade_st, area;
```

Records with area UNKNOWN were not considered. From the output of the query illustrated below we can see that $\bar{Q}_t^* = 7$, which is less than 12 and thus this might indeed be an outlier.

```
 decade_st |   area   | count
```

```
-----------+----------+-------
      1980 | Theory   |     5
      1980 |          |     5
      1985 | ML-AI    |     1
      1985 | Theory   |     3
      1985 |          |     4
      1990 | Database |     3
      1990 | Theory   |     8
      1990 |          |    11
      1995 | Database |     8
      1995 | ML-AI    |     1
      1995 | Theory   |    12
      1995 |          |    21
           | Database |    11
           | ML-AI    |     2
           | Theory   |    28
           |          |    41
```

Since there are not that many explanatory queries for this example, let k=4. Then according to the first two explanations of the algorithm provided below Jeffrey's number of publication in Theory during 1995-1999 is high because he has also published more in Database and generally in all fields during that period. The last two explanations also suggest the reason that he was publishing less in the past. Interestingly, all this explanations are indeed true. Because during this period Jeffrey Scott Vitter was helding a distinguished Professorship position at Duke University, where among other contributions he has made to the prospect of the university he has achieved rise in sponsored research expenditures up to 250%, which most probably implies that he was also involved in many different types of research during that period of time.

$\phi_{direct} =$

$[area =' Database' \wedge decade_st = 1995]$: 2.909
$[decade_st = 1995]$: 2.048

$\phi_{inverse} =$

$[area =' Theory' \wedge decade_st = 1980]$: 0.714
$[area =' Theory' \wedge decade_st = 1985]$: 0.429

Now consider the following query with input $(Q^*, high, author)$

```
SELECT floor(year/5)*5 AS decade_st, area, count(*)
FROM Authorship A, Inproceedings I
WHERE I.pubkey=A.pubkey AND
author='Jeffrey D. Ullman' AND
 area='Database' AND floor(year/5)*5=1995;
```

Again $Q^* > \bar{Q}_t^*$ since $Q^* = 16$ and $\bar{Q}_t^* = 5.143$. The algorithm suggests two meaningful explanations in this case, because he published less in Theory and also less in the previous year in Database. Again this person turns out to be a famous professor in a database field. Of course we can't expect the algorithm to give us whole biography of the person, but it still provides some interesting patterns in the data.

$\phi_{direct} =$

$[decade_st = 1995]$: 1.867
$[area =' Database' \wedge decade_st = 1985]$: 1.750

$\phi_{inverse} =$

$[area =' Theory' \wedge decade_st = 1995]$: 0
$[area =' Database' \wedge decade_st = 1990]$: 0.389

6 Future Work and Conclusions

This paper considers another interesting approach for explaining aggregate query answers. As results from the previous section indicate the explanation can provide very meaningful insights. Key for this might be the importance of normalizing values, which allows comparison of different query answers that might have different distributions and scales. There are many different yet interestig possibilities for future work, which includes changing values of several attributes at the same time or considering several queries and comparing interesting patterns in their values.

References

[1] R. M. C. Silverstein, S. Brin and J. Ullman. Scalable techniques for mining causal structures. *Data Min. Knowl. Discov.*, 2000.

[2] G. D. M. Das, S. Amer-Yahia and C. Yu. Meaningful interpretations of collaborative rankings. *PVLDB*, 2011.

[3] M. B. N. Khoussainova and D. Suciu. Perfxplain: Debugging map-reduce job performance. *VLDB*, 2012.

[4] S. Roy and D. Suciu. A formal approach to finding explanations for database queries. *SIGMOD*, pages 1579–1590, 2014.

[5] L. O. Sudeepa Roy and D. Suciu. Explaining query answers with explanation-ready databases. *VLDB*, 2016.

[6] E. Wu and S. Madden. Scorpion: Explaining away outliers in aggregate queries. *PVLDB*, 2013.