HypDB: A Demonstration of Detecting, Explaining and Resolving Bias in OLAP queries

Babak Salimi [1] Corey Cole [1] Peter Li [1] Johannes Gehrke [2] Dan Suciu [1]

[1] Department of Computer Science & Engineering
University of Washington
{bsalimi, suciu, pzli97}@cs.washington.edu, coreylc@uw.edu
[2] Microsoft
johannes@microsoft.com

ABSTRACT

On line analytical processing (OLAP) is an essential element of decision-support systems. However, OLAP queries can be biased and lead to perplexing and incorrect insights. In this demo, we present HypDB, the first system to detect, explain and resolve bias in OLAP queries. Our demonstration, shows several examples of OLAP queries from real world datasets that are biased and could lead to statistical anomalies such as Simpson's paradox. Then, we demonstrate step-by-step how HypDB: (1) detects whether an OLAP query is biased, (2) explains the root causes of the bias and reveals illuminating insights about the domain and the data collection process and (3) eliminates the bias via query rewriting and generates decision-support insights.

PVLDB Reference Format:
Babak Salimi, Corey Cole, Peter Li, Johannes Gehrke, Dan Suciu. HypDB: A Demonstration of Detecting, Explaining and Resolving Bias in OLAP queries. *PVLDB*, 11 (12): 2062 - 2065, 2018.
DOI: https://doi.org/10.14778/3229863.3236260

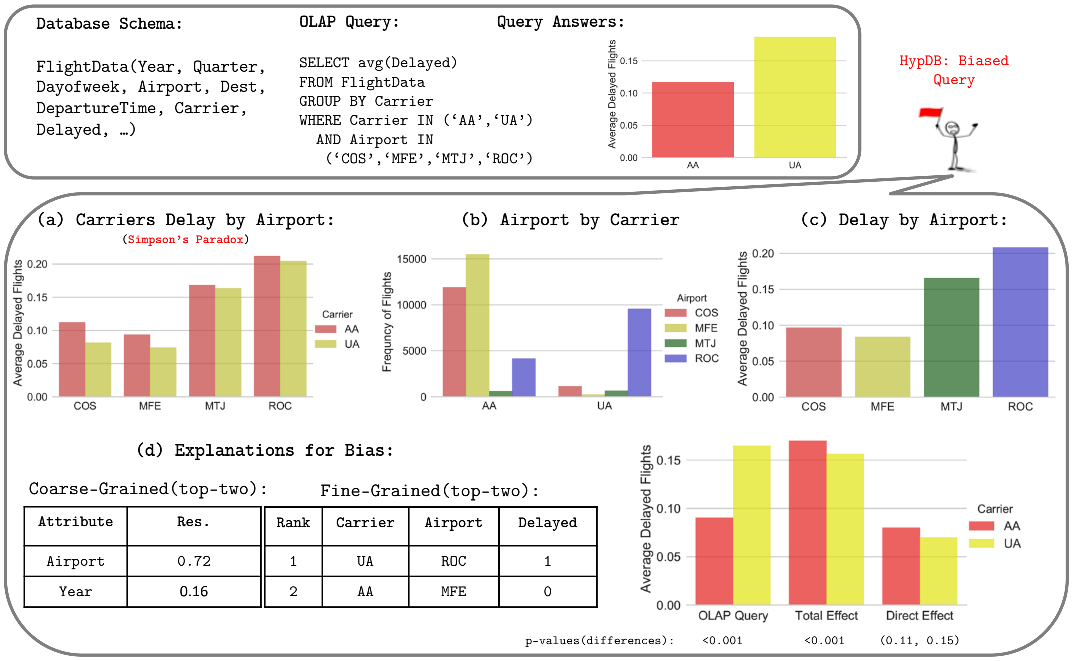
1. INTRODUCTION

On line analytical processing (OLAP) is an essential element of decision-support systems. OLAP tools enable the capability for complex calculations, analyses, and sophisticated data modeling; this aims to provide the insights and understanding needed for improved decision making. Despite the huge progress OLAP research has made in recent years, the question of whether these tools are truly suitable for decision making remains unanswered [3, 2]. The following example shows how insights obtained from OLAP queries can lead to incorrect business decisions.

EXAMPLE 1.1. *Suppose a company wants to choose between the business travel programs offered by two carriers, American Airlines (AA) and United Airlines (UA). The company operates at four airports: Rochester (ROC), Montrose (MTJ), McAllen Miller (MFE) and Colorado Springs (COS) and wants to choose the carrier with the lowest rate of delay at these airports. To make this decision, the company's data analyst uses FlightData [7], she runs the group-by query shown in Fig. 1 to compare the performance of the carriers. Based on the analysis at the top of Fig. 1, she recommends choosing AA because it has a lower average flight delay. Surprisingly this is a poor decision. AA has, in fact, a higher average delay than UA at each of the four airports Fig. 1(a). This trend reversal, is known as Simpson's paradox, occurs as a result of* confounding influences*. The Airport has a confounding influence on the distribution of the carriers and departure delays because its distribution differs for AA and for UA (Fig. 1 (b) and (c)): AA has many more flights from airports that have relatively few delays, like COS and MFE, while UA has more flights from ROC, which has relatively many delays. Thus AA seems to have an overall lower delay only because it has many flights from airports that in general have few delays. At the heart of the issue is an incorrect interpretation of the query; While the analyst's goal is to compare the* causal effect *of the carriers on delay, the OLAP query measures only their* association.

In this demonstration, we propose HypDB, the first system to detect, explain, and resolve bias in OLAP queries. HypDB systematically performs the type of analysis exemplified in Fig. 1. It interprets OLAP queries as queries about testing causal hypotheses, those most often required for making business decision. The gold standard for testing causal hypothesis is a *randomized experiment* or an *A/B test*, called as such because the treatments are assigned to subjects randomly. In contrast, business data is *observational*, defined as data recorded passively and subject to selection bias. Built upon *observational studies* in statistics [8, 6], HypDB detects bias in an OLAP query by automatically inferring *confounding* or *covariate variables.* It, then resolves the bias by rewriting the query into an unbiased query that correctly performs the hypothesis test that the analyst intended. Finally, it generates ranked explanations for the bias, to explain its finding. An important application of HypDB, which we demonstrate on adult census data [5] and Berkeley data [1], is to detect *unfairness* and *disparate impact* [11].

Proceedings of the VLDB Endowment, Vol. 11, No. 12
Copyright 2018 VLDB Endowment 2150-8097/18/08.
DOI: https://doi.org/10.14778/3229863.3236260



```
FlightData(Year, Quarter,
Dayofweek, Airport, Dest,
DepartureTime, Carrier,
Delayed, …)
```

```
SELECT avg(Delayed)
FROM FlightData
GROUP BY Carrier
WHERE Carrier IN ('AA','UA')
  AND Airport IN
   ('COS','MFE','MTJ','ROC')
```

Coarse-Grained(top-two):

Attribute	Res.
Airport	0.72
Year	0.16

Fine-Grained(top-two):

Rank	Carrier	Airport	Delayed
1	UA	ROC	1
2	AA	MFE	0

Figure 1: **Scenario explained in Ex. 1.1.**

2. SYSTEM ARCHITECTURE

Fig 2 shows HypDB architecture. Essentially, HypDB is an add-hoc analysis tool in an OLAP system; it accepts a query Q that computes the average of Y GROUP BY T with an arbitrary WHERE condition. The HypDB's Hypothesis Tester (HT) module assumes that an OLAP database is a random sample of some population and supports efficient statistical tests to determine whether the answers to Q indicate statistically significant dependence between T and Y. If the answers are significant, Q is passed through the HypDB's Bias Detector (BD) module, which checks whether Q is biased and the indicated correlation between T and Y is spurious. For biased queries, BD automatically detects a set of confounding attributes that is responsible for the bias and sufficient for bias elimination. Then, the confounding attributes are passed through HypDB's Query Rewriter (QR) and Explanation Generator (EG) modules. The former eliminates bias by rewriting Q into an unbiased query that controls for the confounding influences. The latter generates fine and coarse-grained explanations for the bias and ranked them by their responsibility.

3. OVERVIEW OF THE SYSTEM

This section, provides a general description of the internal of HypDB. A detailed description of the system can be found on [10]. HypDB assumes that a database D is a uniform sample from a large population and interprets the answers to a query of the form Q : `SELECT avg(Y) FROM D WHERE C GROUP BY T` as $\mathbb{E}[Y|T=t_0, C]$ and $\mathbb{E}[Y|T=t_1, C]$ for $T=\{t_0, t_1\}$. As shown in Fig. 1, Q is not useful for making judgments about the effect of choosing between two alternatives, $T=t_0$ or $T=t_1$, on some outcome of interest, Y. A principled business decision, instead, should rely on comparing Y in two *counterfactual* worlds, where T is set to t_0 and t_1. HypDB models these counterfactuals following Rubin's causality framework [8] by assuming two attributes $Y(t_0)$ and $Y(t_1)$, the *potential outcomes* of Y if T were hypothetically set to t_0 and t_1, respectively. Then, the causal effect of T on Y can be measured by $\mathbb{E}[Y(t_0)] - \mathbb{E}[Y(t_1)]$. In Ex. 1.1, HypDB assumes that each flight has *two* delay attributes, $Y(\text{AA})$ and $Y(\text{UA})$, representing the delay if the flight were serviced by AA, or by UA respectively. Of course, each flight was operated by either AA or by UA, hence Y in the database is either $Y(\text{AA})$ or $Y(\text{UA})$; the other value is missing, and we can only imagine it in an alternative, counterfactual world. It can be shown that if a sufficient set of confounding attributes $\mathbf{Z}$ is known, $\mathbb{E}[Y(t_i)]$ for $i=\{0,1\}$ can be computed from data by conditioning on $\mathbf{Z}$, i.e., $\mathbb{E}[Y(t_i)] = \mathbb{E}_{\mathbf{z}}[\mathbb{E}[Y|T=t_i], \mathbf{Z}=\mathbf{z}]$. Thus, a major challenge in causal inference is to identify a sufficient $\mathbf{Z}$.

A principled approach for selecting sufficient confounding attributes is based on *causal diagram*[6], in which the complete causal structure of attributes is represented with Directed Acyclic Graphs(DAGs). In the causal DAG shown in Fig. 3: the direct edge from Lung Cancer to Fatigue means lung cancer is a *direct cause* of fatigue; The directed path from Lung Cancer to Car Accident means lung cancer causes car accident, however, the effect is *indirect* and *mediated* by Fatigue; Nodes that are not connected with a directed path are not casually related. Given a causal DAG G, it can be shown that to compute the *total effect* of T on Y, i.e., *the effect through all directed paths* from T to Y, it is sufficient to condition on the parents of T in G. Moreover, to compute the *direct effect* of T on Y, i.e., *the effect* only *though the direct arrow* from T to Y, it is sufficient to condition on the parents of T and Y in G.

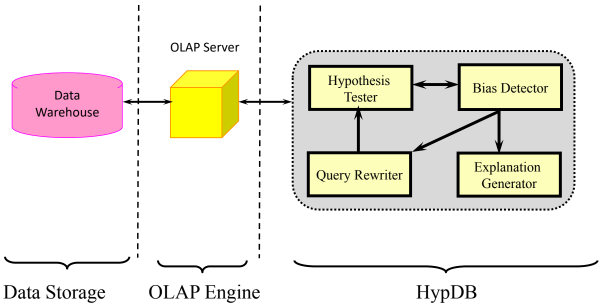


Figure 2: **HypDBArchitecture.**

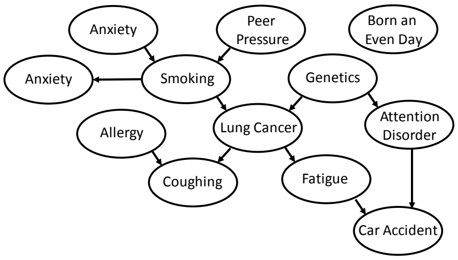


Figure 3: **Causal DAG underling CancerData.**

HypDB develops an efficient method for detecting parents of a node that does not compute the entire causal DAG. Instead, it first computes the Markov boundary of T, which consists of the set of all parents, children and parents of its children in a causal DAG. For instance, Markos boundary of Lung cancer in Fig. 3 consists of all colored nodes. Second, HypDB learns the parents of T from its Markov Boundary by performing number of independence tests. To generate explanations, HypDB ranks the confounding attributes by a metric called *responsibility*, which quantifies the confounding effect of an attribute. To eliminate bias HypDB rewrites the query to an unbiased query which conditions on the confounding attributes.

Finally, HypDB uses a suite of optimization techniques, ranging from using pre-computed OLAP data cubes, on line view materialization, cashing intermediate results and efficient non-parametric independence test based on permutation to detect, resolve and explain bias in an OLAP query interactively at query time.

4. DEMONSTRATION DETAILS

We demonstrate HypDB by providing a walkthrough that investigate several OLAP queries on datasets in Table 1. Note that in our demo the time to explain and resolve the bias is always under 1 second, while the time to detect the bias may be larger, depending on the dataset.

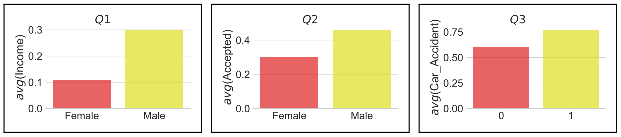
Queries. We start the demo with the following queries (below in SQL): Q_1 computes average income (Income=1 iff income$> 50k$) by Gender; Q_2 computes the acceptance rate in 1973 at UC Berkeley by Gender; Q_3 computes the rate of car accidents among groups with and without lung cancer.

```
Q1:                     Q2:                       Q3:
 SELECT avg(Income)      SELECT avg(Accepted)      SELECT avg(Car_Accident)
 FROM AdultData          FROM BerkeleyData         FROM CancerData
 GROUP BY Gender         GROUP BY Gender           GROUP BY Lung_Cancer
```

Dataset	Columns [#]	Rows[#]
AdultData [5]	15	48842
BerkeleyData [1]	3	4428
CancerData [4]	12	2000

Table 1: **List of datasets used in the Demo**

Answer to the queries:. The answers to Q_1, Q_2 and Q_3 are visualized below:



It is quite tempting to interpret these results as follows: Q_1 suggests a strong disparity with respect to females' income. Indeed, using this AdultData, several prior works in algorithmic fairness have reported gender discrimination, e.g., [11]. Q_2 also suggests a huge disparate impact on female applicants. Indeed, in 1973, UC Berkeley was sued for discrimination against females based on this interpretation. Q_3 suggests that lung cancer affects the rate of car accidents.

Detecting bias. We next use HypDB to check whether the queries are biased. HypDB shows that: Q_1 is biased and identifies attributes such as MaritalStatus, Education, Occupation, etc. as mediating and confounding attributes; Q_2 is biased w.r.t. Department; Q_3 is biased and identifies the attributes Genetics and Fatigue as confounding and mediating attributes, which complies with the causal DAG in Fig. 3, upon which CancerData was generated.

Explaining bias. Fig. 5 shows the explanations generated by HypDB for the bias of Q_1, Q_2 and Q_3. For Q_1, explanations show that Maritalstatus accounts for most of the bias, followed by Education. The top fine-grained explanations for MaritalStatus reveal surprising facts: there are more married males in the data than married females, and marriage has a strong positive association with higher incomes. It turns out that the income attribute in US census data reports the adjusted gross income as indicated in an individual's tax forms, which depends on filing status (jointly and separately), could be household income. Thus, *AdultData is inconsistent and should not be used to investigate gender discrimination.* HypDB explanations also show that males tend to have higher educations than females and higher educations are associated with higher incomes. The explanation generated for Q_2 reveal that females tended to apply to departments such as F that have lower acceptance rates, whereas males tended to apply to departments such as A and B that have higher acceptance rates. Q_3 explanations show that Fatigue is the most responsible attribute for bias; and people with lung cancer tend to be fatigued, which is highly associated with car accidents.

Resolving bias.. At this phase, HypDB uses the confounding and mediating variables to remove bias by rewriting the query into queries that compute the total and direct effects. For instance, the rewritten query associated with Q_2 is shown in Listing 1. It partitions BerkeleyData into blocks that are homogeneous on Department. It then computes

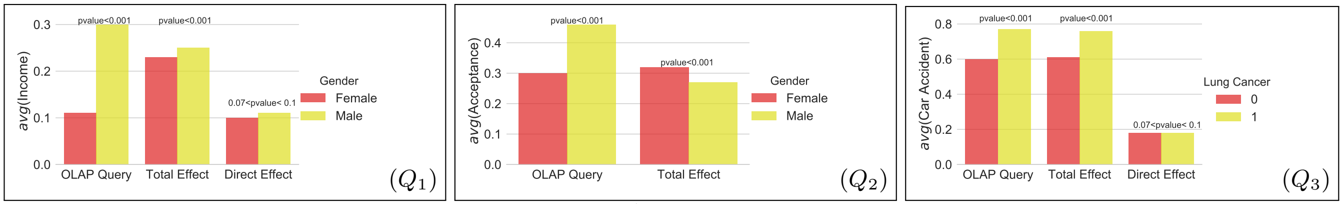


Figure 4: **Rewritten query answers.**

<table>
<tr><th></th><th colspan="2"><i><b>Explanation</b></i></th></tr>
<tr><td></td><td><i><b>Coarse-grained</b></i></td><td><i><b>Fine-grained</b></i></td></tr>
<tr><td>Q1</td><td>
<table>
<tr><th><i>Attribute</i></th><th><i>Res.</i></th></tr>
<tr><td><i>MaritalStatus</i></td><td><i>0.58</i></td></tr>
<tr><td><i>Education</i></td><td><i>0.13</i></td></tr>
<tr><td><i>CapitalGain</i></td><td><i>0.07</i></td></tr>
<tr><td><i>HoursPerwWeek</i></td><td><i>0.04</i></td></tr>
<tr><td><i>Age</i></td><td><i>0.04</i></td></tr>
</table>
</td><td>
<table>
<tr><th>Rank</th><th>MarialStatus</th><th>Gender</th><th>Income</th></tr>
<tr><td>1</td><td>Married</td><td>Male</td><td>1</td></tr>
<tr><td>2</td><td>Single</td><td>Female</td><td>0</td></tr>
</table>
<table>
<tr><th>Rank</th><th>Education</th><th>Gender</th><th>Income</th></tr>
<tr><td>1</td><td>Bachelors</td><td>Male</td><td>1</td></tr>
<tr><td>2</td><td>SomeCollage</td><td>Female</td><td>0</td></tr>
</table>
</td></tr>
<tr><td>Q2</td><td>
<table>
<tr><th>Attribute</th><th>Res.</th></tr>
<tr><td>Department</td><td>1</td></tr>
</table>
</td><td>
<table>
<tr><th>Rank</th><th>Gender</th><th>Accepted</th><th>Department</th></tr>
<tr><td>1</td><td>Male</td><td>1</td><td>A</td></tr>
<tr><td>2</td><td>Male</td><td>1</td><td>B</td></tr>
<tr><td>3</td><td>Female</td><td>0</td><td>F</td></tr>
</table>
</td></tr>
<tr><td>Q3</td><td>
<table>
<tr><th>Attribute</th><th>Res.</th></tr>
<tr><td>Distance</td><td>1</td></tr>
</table>
</td><td>
<table>
<tr><th>Rank</th><th>Income</th><th>Price</th><th>Distance</th></tr>
<tr><td>1</td><td>0</td><td>1</td><td>Far</td></tr>
<tr><td>2</td><td>1</td><td>0</td><td>Near</td></tr>
</table>
</td></tr>
</table>

Figure 5: **Bias explanations.**

the average Acceptance Group BY Gender in each block. Finally, it aggregates the block's averages by taking their weighted average, where the weights are probabilities of the blocks. See [9] for a detailed investigation of representing biased elimination techniques in SQL.

Listing 1: Rewritten query associated to Q_2.

```
WITH Blocks
AS(
SELECT Gender,Department) AS Avge
FROM BerkeleyData
GROUP BY   Gender,Department),
Weights
AS(
SELECT Department, count(*)/ (SELECT
     count(*) FROM D)  AS W
FROM BerkeleyData
GROUP BY Department
SELECT Gender,sum(Avge * W)
FROM Blocks,Weights
WHERE Blocks.Department = Weights.
     Department
GROUP BY Carrier
```

Answer to the rewritten queries:. Fig 4 shows the answers to the rewritten queries associated to Q_1, Q_2 and Q_3. As depicted, the huge disparity against females suggested by Q_1 and Q_2 was due to bias and explained by confounding attributes. However, note that due to the reported inconsistency in AdultData and lack of demographic information in BerkeleyData, drawing causal conclusions from these datasets is challenging. The answers to Q_3 show that lung cancer has no significant direct effect on the rate car accidents, however, it has a significant total effect (results comply with the ground truth in Fig. 3).

5. CONCLUSION

In this demonstration we introduced HypDB, the first system to detect, explain, and resolve bias in decision-support OLAP queries. We demonstrated that biased queries can be perplexing and lead to statistical anomalies, such as Simpson's paradox. HypDB develops novel techniques to find explanations for the bias, thereby assisting the analyst in interpreting the results. It supports an automated method for rewriting the query into an unbiased query that correctly performs the hypothesis test that the analyst intended.

6. REFERENCES

[1] P. J. Bickel, E. A. Hammel, J. W. OConnell, et al. Sex bias in graduate admissions: Data from berkeley. *Science*, 187(4175):398–404, 1975.

[2] C. Binnig, L. D. Stefani, T. Kraska, E. Upfal, E. Zgraggen, and Z. Zhao. Toward sustainable insights, or why polygamy is bad for you. In *CIDR*, 2017.

[3] A. A. Freitas. Are we really discovering interesting knowledge from data. *Expert Update (the BCS-SGAI magazine)*, 9(1):41–47, 2006.

[4] I. Guyon. Lung cancer simple model, 10 2009.

[5] M. Lichman. Uci machine learning repository, 2013.

[6] J. Pearl. *Causality: models, reasoning, and inference.* Cambridge University Press, 2000.

[7] D. A. O.-T. Performance. `http://www.transtats.bts.gov/`.

[8] D. B. Rubin. Causal inference using potential outcomes: Design, modeling, decisions. *Journal of the American Statistical Association*, 100(469):322–331, 2005.

[9] B. Salimi, C. Cole, D. R. Ports, and D. Suciu. Zaliql: causal inference from observational data at scale. *PVLDB*, 10(12):1957–1960, 2017.

[10] B. Salimi, J. Gehrke, and D. Suciu. Bias in olap queries: Detection, explanation, and removal. In *Proceedings of the 2018 International Conference on Management of Data*, pages 1021–1035. ACM, 2018.

[11] F. Tramer and et al. Fairtest: Discovering unwarranted associations in data-driven applications. In *Security and Privacy (EuroS&P), 2017 IEEE European Symposium on*, pages 401–416. IEEE, 2017.