WHY DATA VIRTUALIZATION IS AN ANALYTICS GAME CHANGER
Introduction

Let’s say you’ve just written a letter using a typewriter. As you’re reading it over, you notice a typo. You grab some correction tape, paint over the error, and carefully draw in the right letters. As you keep reading, you notice that a particular sentence is phrased poorly, and another paragraph might make more sense if it were moved up in the document. Now the Wite-Out won’t cut it; the edits are too complicated to make on the fly. You could re-type the entire page from scratch, or just leave it as-is and hope it’s good enough.

Data teams spend 60% of their time cleaning and organizing data in preparation for analysis

This was the state of writing and editing before the word processor. Today, the edits described above would take a few seconds rather than hours of re-work. And it’s more than just time savings: word processors have fundamentally changed the way we write. Instead of carefully pre-planning every keystroke, we’re free to type as quickly as thoughts come to mind, knowing that we can refine, edit, delete, and rearrange our work later as needed.

The modern analytics stack is still stuck in its typewriter phase. Modern data and engineering teams spend countless hours of work on data collection and schematization only to have to re-instrument over and over again. Instead of spending their time building predictive models or finding the signal in the noise, data teams spend 60% of their time cleaning and organizing data in preparation for analysis.
WHY DATA VIRTUALIZATION IS AN ANALYTICS GAME CHANGER

Only 38% of organizations have high confidence in their data and analytics for customer insights

Even worse, all that time spent cleaning isn’t necessarily leading to the promised land of trustworthy, complete data. Only 38% of organizations have high confidence in their data and analytics for customer insights.

However, a powerful new technique called data virtualization is drastically reducing the time that data teams spend on data preparation. Data virtualization not only saves time, but like the word processor, it shifts the way that people interact with data.

In this white paper, we’ll begin by giving a technical overview of what data virtualization is. Then, we’ll discuss the biggest problems related to data organization and cleaning that arise in a modern analytics infrastructure and how data virtualization can address those issues. Finally, we’ll discuss the state of the data virtualization market today via real-world examples of data virtualization being used successfully.
What is Data Virtualization?

Data virtualization is a data management technique that allows users to manipulate and analyze an abstracted or virtualized view of their data rather than interacting with the raw data itself. It’s a layer of indirection in between raw data and analysis that makes the analysis far easier and makes sure your data stack can keep up with a rapidly changing business environment.

Here’s an example to make this more concrete. Let’s say we’re building an analytics warehouse at a retailer. Our retailer has locations in both the US and Canada. In the US, we’re using Acme Corp as our point of sale system, while in Canada, we’re using Example Systems. In our analytics warehouse, we have a table of transaction data from Acme’s system that has the following columns:

(date_of_transaction, amount_paid_in_US_cents, location)

And we have another table of transaction data from Example that look like this:

(time_of_transaction, amount_paid_in_CA_Dollars, location)
If I want to know how much total revenue we made on May 1, 2018, I would need to write the following SQL query:

```sql
SELECT (total_acme + total_example) total_transaction_volume
FROM
(
    SELECT sum(amount_paid_in_US_cents) total_acme
    FROM transactions_acme
    WHERE date_of_transaction = 'May 1, 2018'
    UNION ALL
    SELECT (sum(amount_paid_in_CA_Dollars) * $CAD_to_USD * 100) total_example
    FROM transactions_example
    WHERE time_of_transaction >= 1525158000000
    AND time_of_transaction < 1525244400000
)
```

While this gets the job done, it’s a complex, brittle query that has to navigate the fact that we have transaction data separated into two different tables, with different date formats and units of currency. Even something as simple as changing the date we want to analyze requires touching three different parts of this query. And what if instead of two point of sale systems, we had twenty?

It would be nice if instead we had a single table with the following schema:

```
(date_of_transaction, amount_paid, currency, location, point_of_sale_system)
```

Then this query and many other queries we’d want to run against the data would be far simpler. One approach would be to write a script that copies over the data in these disparate tables into one common table. Unfortunately, this approach is space inefficient and error-prone. This is where data virtualization comes into play.
One simple form of data virtualization that will help us here is SQL’s View feature. In a SQL database, a View is a virtual table that doesn’t actually exist, but is instead the output of a select statement. Here’s how we’d create our desired table using a view:

```
CREATE VIEW transactions_all AS
  SELECT
    date_of_transaction,
    amount_paid_in_US_cents / 100 as amount_paid,
    "USD" as currency,
    location,
    "Acme Corp" as point_of_sale_system
  FROM transcations_acme

UNION

SELECT
  date(time_of_transaction) as date_of_transaction,
  amount_paid_in_CA_Dollars as amount_paid,
  "CAD" as currency,
  location,
  "Example Systems" as point_of_sale_system
FROM transcations_example
```

Now, we can query the virtual table `transactions_all` directly instead of having to do messy joins between the other tables.

Using SQL Views to create a virtual table is just a basic example of virtualization. Data virtualization doesn’t solely have to exist within a data warehouse. Any time data is flowing from point A to point B, we have an opportunity to create a virtualization layer in the middle. Advanced data virtualization systems can virtualize analytics events themselves to enable data teams to overcome the major limitations they face today.
Four Data Problems That Data Virtualization Can Solve

The previous section introduced the idea of data virtualization through a simple use case. When moving beyond its most basic form, data virtualization can have a deep impact on some of the most common and pernicious problems related to data organization and cleaning today. In this section, we cover some of the primary problems hamstringing data teams that data virtualization can alleviate.

Problem #1: Schema

Schema problems are perhaps the most common problems related to data organization. The example we explored earlier where we had two different transaction tables with different schemas is a version of this problem. More generally, this is a class of data problems where our data is schematized in a way that’s less than ideal for the analytical questions we want to ask of our data. This often arises from analytics tools that require traditional events that you need to define upfront. As your business or analytics questions change, the events you’ve previously defined are no longer useful.

As we showed in the example in the previous section, Views in SQL are one way to resolve simple schema mismatches between subsets of our data. More advanced virtualization systems such as LookML from Looker can go even further in this direction, as we’ll explore later.

These problems are most troublesome when there’s a change in the underlying schema, often driven by a change in the overall business. For example, let’s say we’re at an e-commerce business, and we have a “purchase” table in our data warehouse corresponding to all the purchases people have ever made. The table contains a field indicating whether the purchase was made successfully, what amount, what items, etc. From here, we can easily calculate our total gross merchandise volume (GMV).
But later on we decide to introduce a subscription option. Now, we’re collecting recurring revenue, and all payments after the first one for a customer are made without a new “purchase” event ever happening. Our GMV report, along with other related reports, ends up breaking. We’ll need to reconfigure how our data is structured to account for this change. And after making these schema changes, our old GMV reports won’t reconcile with the new reports unless we migrate over all of our old data.

The Data Virtualization Solution

This sort of situation is where the benefits of data virtualization can be directly felt. If you are automatically capturing all your user behavior data, an analytics tool that enables you to define Virtual Events that work retroactively can handle underlying schema changes with no issue. Since the events are virtualized and have definitions that live in your analytics tool rather than in your product, updating schemas is a matter of clicking a button. Having a layer of virtualization between your raw data and your semantic data that you manipulate means that the raw data won’t break should the semantic data no longer fit reality.

Problem #2: Accuracy

Sometimes data is just plain wrong. Accuracy problems are ones where the data in your system of record doesn’t match the true value. These problems are often the result of issues at the source: data entry errors, broken logging code, etc. If this is the case, it’s likely impossible to fix inaccurate data in the cleaning phase.

While many accuracy problems can’t be fixed during cleaning, there are a few that can. Data inconsistency problems are ones that can be addressed via virtualization. Inconsistency refers specifically to two or more pieces of data in our dataset that contradict each other.

Let’s walk through another example. We’re at a B2B company that asks prospective customers what industry their company is in when they sign up for our product. This is an optional field, so some customers fill it out while others don’t. However, we also use a third-party data enrichment service to automatically infer a customer’s company’s industry. This third-party data is less accurate than the customer-inputted data, but it’s a good stopgap when the customer hasn’t filled in that information themselves.
In our data warehouse, we have a customers table that contains a variety of customer metadata including the customer-inputted industry. A separate table contains data from the third-party service. In order to query the common set of data, we have to write a complex SQL statement that joins these two tables and also overrides the third-party data if and only if the customer-inputted data is present.

**The Data Virtualization Solution**

Data virtualization provides an elegant and more flexible solution to the above example. With a virtualized system, we can create a virtual table that has a column with customer-inputted data if it’s there and third-party data otherwise. We can also add another column to the virtual table that has the source of the data.

Additionally, data virtualization prevents some data accuracy problems in the first place. Data entry errors and other human errors can cause permanent data losses and accuracy errors in a non-virtualized system. But with a virtualized system, data entry and labeling errors can be easily undone, since the underlying raw data was never impacted -- only the virtualized dataset.

**Problem #3: Completeness**

Our dataset can’t be considered complete unless we have all the data we need to accomplish our analytics goals. Even if our data is accurate, consistent, and well-schematized, if some data is missing it isn’t sufficient.

Here’s an example to illustrate how data virtualization can help with catching data completeness issues. Let’s say we’re running analytics for an e-commerce store, and there’s a line of tracking code that fires every time a user clicks “Add To Cart” on any product page. Every time someone clicks “Add To Cart,” a new row is added to a table in our analytics warehouse. One day, a website change causes this tracking code to stop working and that table stops receiving new data from this particular event listener.

We would likely notice this problem when running a query or report that relies on the “Add To Cart” data. If this isn’t one of our frequently used reports, it may be some time before we notice the problem. Once we notice the problem in a report, we have to go and fix the tracking code at the source.
After a while, our merchandising team makes a change so that some items can be automatically added to cart when purchased alongside a complementary item if a user checks a box when clicking Add To Cart. Now, some items are getting added to the cart without an explicit user action. Again, the tracking code is broken. But instead of dropping 100% of the data, it’s dropping a small portion of it. This is much harder to notice in our downstream reports and the issue might go weeks or months before being caught.

The Data Virtualization Solution

Many data virtualization systems have mechanisms to solve the problems in this example. Some virtualization systems, especially those specific to behavioral data (such as Heap), can automatically detect if a certain event has stopped firing. Systems that use Virtual Events can prevent this issue in the first place by defining the event as the “Add to Cart” user action in the analytics tool itself, rather than via a tracking code. Should an overhaul to your website mean that the Virtual Event is no longer complete, you can redefine it in minutes and it will retroactively include past data.

Problem #4: Privacy, Security, and Access Control

Access to data often needs to be limited for reasons of privacy or security. Frameworks such as SOC 2 mandate that organizations only give access to sensitive data to people who need that access to perform their job. Also, regulatory frameworks like GDPR require that we disclose to customers exactly how their data will be used and what processors and sub-processors will have access to their data. Finally, organizations may just want to limit access to certain data to certain people for internal reasons.

Having appropriate access control is tough without a virtualization system. It’s possible to maintain table-level permissions in most SQL databases, but this only goes so far. In a fast-moving business, tables are being created very quickly and table schemas (and what data lives in those tables) can also change quickly. It’s burdensome to have to review table-level permissions with each database migration, putting a severe strain on agility and efficiency.
Table-level permissions also don’t solve the problem of how that data gets used downstream. Let’s say we’re feeding data in our warehouse into a real-time personalization engine that alters a user’s website or app experience. GDPR requires that we get explicit consent to use a user’s personal data in this manner. Table-level permissions also don’t solve the problem of how that data gets used downstream. Let’s say we’re feeding data in our warehouse into a real-time personalization engine that alters a user’s website or app experience. GDPR requires that we get explicit consent to use a user’s personal data in this manner. A virtualization layer can help make sure only approved pieces of personal data are sent downstream into the personalize engine.

The Data Virtualization Solution

With data virtualization, you gain fine-grained control over the flow of data between systems and what data is accessible by whom. Since the data is virtualized in software, it’s a matter of dictating security and control in a software layer, rather than with the fundamental data itself. In the above example, a virtualization layer can make sure only approved pieces of personal data are sent downstream into the personalize engine.

| username | 
|---|---|
| email_address | 
| ip_address | Never synced to warehouse

Heap’s selective sync feature allows users to configure which events and user properties are sent to which destinations
Data Virtualization Today

The data virtualization landscape has changed considerably in recent years. We’ve already discussed how SQL Views are an example of simple virtualization technology that anyone can take advantage of. In this section, we’ll offer a brief overview of some more advanced approaches.

There are several dedicated, standalone virtualization products available on the market today. We’ll look at two of these below. If you’re interested in more examples, you can check out Forrester’s most recent Wave report on data virtualization.

- DataVirtuality provides direct connectors into over 150 data sources. Typically, querying each of these tools would require learning a separate set of query APIs, each with its own quirks and technical limitations. In addition, the maintenance burden for these would be non-trivial as tools publish API changes all the time. DataVirtuality instead abstracts away these complexities and allows its customers to write SQL queries directly against any of these data sources.

- Denodo connects to several data sources, including even unstructured sources of data like PDF documents. It then provides a powerful system for combining, integrating, and transforming those sources of data into clean business views of that data. This data can be published as SQL views in a warehouse or a variety of other data service formats.

There are also many systems that incorporate elements of data virtualization into their products despite not being dedicated virtualization vendors. We’ll look at two of these vendors below:

- Looker is primarily a business intelligence (BI) vendor that focuses on data visualization. However, Looker also provides a modeling language called LookML that sits on top of the raw SQL data. This allows a data team using Looker to model the SQL data in ways that are conducive to the types of business questions that data consumers within their company are likely to ask.
The Heap property editor allows users to edit how certain pieces of event and user metadata are displayed in its analysis modules as well as downstream in a data warehouse.
• Heap is a customer behavior analytics platform. It automatically collects user behavioral data on websites (e.g. clicks, pageviews), mobile apps (e.g. taps, swipes, gestures), and third-party data sources (e.g. purchases from Shopify). It also offers APIs for directly sending event data and user metadata to Heap. Unlike other behavioral analytics tools, Heap doesn’t merely provide query or warehouse access to this raw data. Instead, Heap has a virtualization layer that sits on top of this raw data. This allows users to define Virtual Events in Heap, which act like traditional events but are also able to be updated retroactively and flexibly, making it easy to check what user action they refer to. The virtualization layer is a powerful method of ensuring that data from disparate sources is appropriately modeled specifically for behavioral questions.

Using one virtualization system isn’t mutually exclusive with using others. Several of these virtualization systems live in different parts of the data stack and can complement each other very well. It’s quite common to do the following, for example:

- Use Heap for collecting and virtualizing behavioral data on your website
- Push Heap data downstream into your data warehouse where it sits alongside other sources of data
- Use SQL Views to further model that data in your warehouse for specific analytical applications
- Use Looker as a visualization layer, and use LookML to model it appropriately for consumption within Looker
Conclusion

Like the leap from typewriter to word processor, data virtualization can deliver a step function change in agility and efficiency for your analytics. It’s crucial to think through your data virtualization strategy when building your enterprise data warehouse and the broader data “stack.” There are major problems with schematization, data accuracy, completeness, and access control that every data team runs into, and these problems are constant battles in a changing business environment. By separating your raw data from the semantic data you run analytics on, data virtualization can heavily mitigate these problems and allow your data team to concentrate on moving the business forward rather than on fighting fire after fire.

To learn more about how Heap is using data virtualization to solve these problems and why we believe user behavior analytics in the near future will require data virtualization, check out Virtual Events: Making Data-Driven Decisions a Reality, a blog post from Heap’s CTO Dan Robinson.

Heap automatically captures every customer touchpoint and automates away the pain of data. Other analytics tools require you to tag events upfront and manually instrument tracking code. Instead, Heap automatically captures everything: clicks, taps, swipes, form changes, and more. Get answers in seconds and make decisions faster.

To learn more about Heap’s data virtualization solutions, contact sales@heapanalytics.com or call +1.415.938.9398.