kubit

EBOOK

A CTO’S GUIDE TO DATA MODELING FOR PRODUCT ANALYTICS

Use cases and best practices

What’s the perfect data model for product analytics?

If you’re asking this question, you’re not alone. Many of our customers have asked us the same over the years. The short answer: It doesn’t exist. While there certainly are some principles and blueprints to follow, a universally perfect data model simply does not exist. Just as your product is continually evolving, the best data model must also evolve, tailored to your analytics use cases through an iterative process.

In this ebook, we’ll recommend best practices for implementing a data model that works for your business and your product in their current state today.

Table of Contents

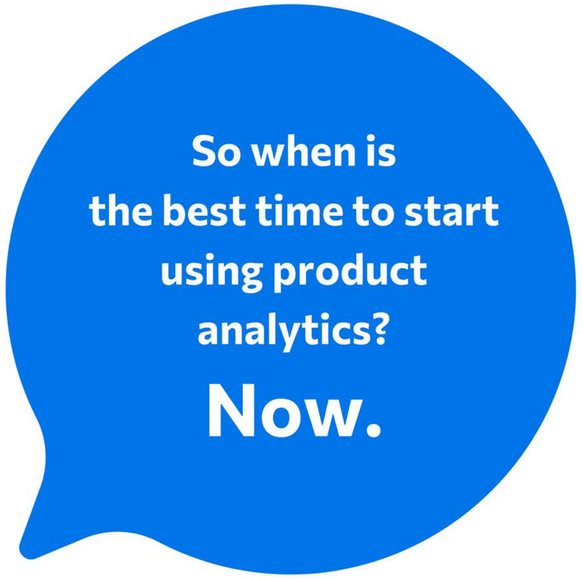
The best time to start using product analytics

Our customers are often under the impression that they need to first improve their current data model before deploying product analytics. But this is not the case. Product analytics is not a one-time project; it's an iterative process that changes as you fine-tune your data. Without testing and verifying against your actual use cases, the "perfect data model", even in the best circumstances, will require ongoing revisions.

An initial product analytics data model can be as simple as having all the raw data in the proper schema to support context-rich and flexible analysis. As long as you follow the basic principles outlined here, your data model should be able to adapt to changes easily.

There is no requirement to have "clean" data for product analytics to be effective. Raw data is perfect for running some quick and dirty analyses to get instant insights. It's also great for figuring out and verifying your analytics needs and, more importantly, for troubleshooting at the lowest possible granular level.

A product analytics tool can work on top of the current "raw" data model while building the "clean" one. It should be able to support them both concurrently and give you the iterative path to migrate and evolve your product analytics.



Welcome to the NEW stage of data modeling

Some of the misconceptions around data modeling stem from misconceptions created by the first generation of product analytics tools. During the boom of mobile apps and the mobile app ecosystem, most developers lacked the technologies and expertise to quickly build large-scale data models themselves. Instead, they settled for SDKs, letting the tool collect their product data and store it in a proprietary data silo. But this infrastructure has a lot of limits. When the quantity of data you need to analyze increases, this solution requires more API integrations or complex ETL jobs to ingest all the data sources into the silo.

Over time, this approach revealed its limits, including duplicated data copies and endless batch jobs.

Meanwhile, major developments were happening related to both data regulations and data infrastructure. A growing focus on data privacy and security eventually resulted in data privacy policies like GDPR and CCPA and the phasing out of third-party cookies.

Kubit—as the first product analytics tool that fully leverages cloud data warehouses and data-sharing capabilities of the modern data stack—offered a new way for companies to do product analytics.

The best way of getting started with Kubit is not by copying a proprietary data model developed by one of the traditional platforms. Those models were designed a decade ago for accommodating hundreds of customer use cases.

The new stage of data modeling is specifically designed for your product and your business and can adapt to your specific needs.

Product analytics: 10 use cases

Before we dive into the principles of designing a product analytics data model, let’s consider typical use cases of product analytics—a specialty that’s all about understanding the full life cycle of users of apps or other digital products.

Product analytics: 10 use cases

1 User Behavior

Analyze all the action events your users take, explicitly (in the UI) or implicitly (backend or third party). With this detailed tracking information, you can get a variety of insights.

For example:

- App launch
- Login
- Page/Screen view
- Clicks
- Purchase
- Subscription renewal
- Likes

2 User Engagement

Count the occurrences of certain events and compare their ratios through a time series to understand how your users engage with your products over time.

For example:

- **DAU, MAU:** Number of unique users per day or month
- **Streams (Events) / DAU:** Number of streams per daily active users
- **Viewing minutes:** Sum of the total time across multiple viewed events
- **Revenue/Order:** Average purchase amount across multiple order events

3 Conversion

Measure the conversion/drop-off rate between the steps of key user flows like registration, checkout, or social loop funnels.

For example:

- **Registration success rate:** Percentage of guest users who complete the registration flow
- **Attach rate:** Percentage of users who complete the subscription or purchase flow after seeing the paywall

4 Retention

Measure how often your users come back and why.

For example:

- **Dn, Wn Retention**: Percentage of new users who open the app again on D1/D2, or W1/W2 (used to compare the effectiveness of user acquisition or reactivation campaigns)
- **Viewing retention**: Percentage of users who continuously watch certain shows at different time intervals (daily, weekly, monthly)
- **Subscription retention**: Percentage of your subscribers who renew over time

5 Filters and Breakdowns

Filter or group measures by certain properties (attributes) to analyze and compare.

For example:

- **App properties:** Platform, app version, app ID
- **Device properties:** Device model, OS version, locale, country
- **User properties:** Age, gender, language, install date, persona
- **Context properties:** Page, flow, campaign

6 Cohort

Segment users by certain criteria to enable dynamic identification and tracking of the behavior of specific groups. Use this data to inform product decisions.

For example:

- **Active viewers:** Users who watched more than one hour of a certain show in the last seven days
- **Churned subscribers:** Users who didn't renew their subscriptions
- **Dormant users:** Users who didn't open the app for more than a week

7 Attribution

Identify and attribute the sources of new users of your digital products. Track and pinpoint which channels your new users come from.

For example:

- Paid advertisement
- YouTube/Instagram campaign
- Social posts
- Organic installs from app store or your website
- Content promotion within your apps

Once the attribution source is identified, these campaigns can be applied as filters or breakdowns in users' engagement, conversion, and retention to measure their impact and effectiveness.

8 Reactivation

Measure re-engagement of customers lost through churn. Adjust your product based on the impact of your push notifications and email campaigns.

For example:

- **Open rate:** Percentage of push notifications or emails opened
- **Click and conversion rate:** Percentage of users who reacted to the message
- **Retention rate:** Impact of a campaign, if any, on user retention

9 Experiments

Use instant insights from your experiments (e.g., A/B tests) for continuous product design. The ability to compare the outcome of different treatments with statistical analysis is a key part of data-driven decision-making.

For example:

- Run constant experiments with the large volume of user data available
- Compare KPIs between test groups and compute statistical significance

10 Content

Test different content (e.g., different shows) through various presentation methods (e.g., catalogs, carousels, push notifications) and measure the impact on critical success factors.

For example:

- Engagement
- Retention
- Virality
- Eventual subscriptions
- Lifetime value (LTV)

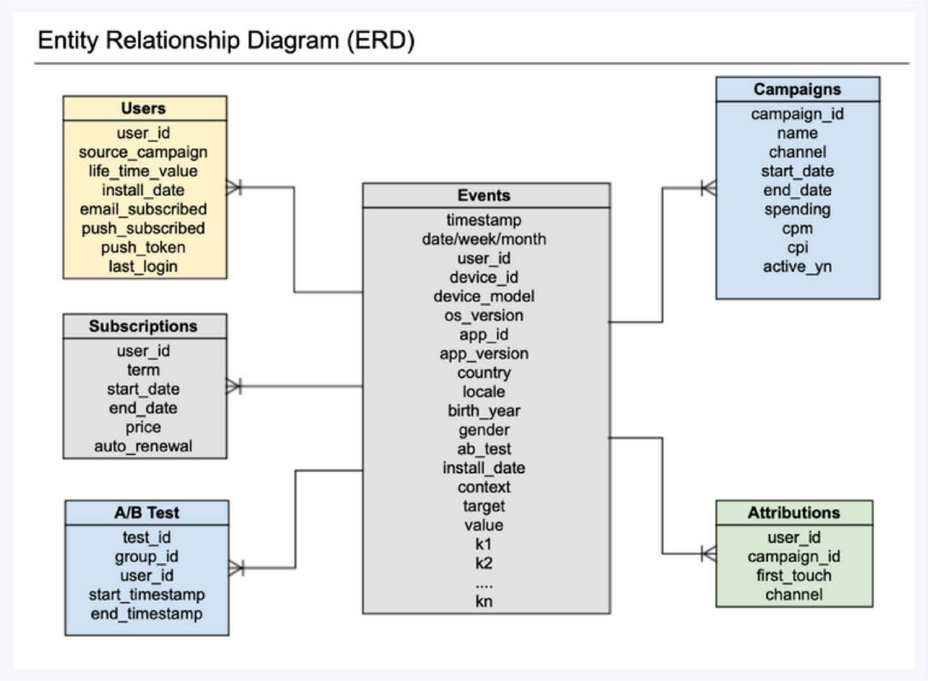
Principles of product analytics data modeling

As you see already, product analytics is quite complex, and the data model can be even more so—because no two digital products have the same use cases, data sources, or vendors. Instead of pushing a “perfect data model,” follow these general principles and best practices, based on years of our experience.

Star Schema

The cornerstone of the analytical data model is the star schema, which simply refers to the shape of the Entity Relationship (ER) diagram, where a centralized fact table (what happened) is surrounded by many dimension tables (the properties to analyze by), connected by distinct IDs.

Here is an example:



Typically, the central fact table captures all the action events your users perform. Then, you have many dimension tables containing all of the properties for each use case. All of these tables can be joined (associated) through a common user ID. You can have multiple fact tables, treating each as the center of its own star schema or unioned together to serve other purposes. For example, Segment typically leverages one table per event type. Your product analytics vendor should be able to directly use these tables as they are, instead of forcing you to develop ETL jobs to create a single gigantic fact table.

Unlike tables in transactional databases (MySQL, PostgreSQL, Oracle, SQL Server, etc.), tables in data warehouses have several special traits specifically for analytical purposes.

Append-Only

Analytical insights must be repeatable and verifiable. Not only are data warehouse tables mostly read-only, they are also append-only. This means data flows into these tables like a log: every entry is immutable, and you only append new entries instead of updating/changing what happened in the past.

Denormalized

Data warehouses are specially designed to store denormalized data efficiently, and to utilize it for very fast analytical queries over billions of rows (terabytes) of data. The fact table should be denormalized to avoid unnecessary table joins. This means you should capture as many properties as possible when the event happens and store them as separate fields/columns in the fact table.

For example, each event should contain all the properties that are available at the time—including user, app, device, and context. Dimension tables come into account for after-the-fact properties—those that are coming from vendors, that are only available in the backend, or that are computed through batch jobs like machine learning algorithms (e.g., each user's Persona).

Data Governance

Proper data governance is required to manage the complex data models for product analytics.

Data governance best practices (which are often ignored) include maintaining a data dictionary and monitoring data quality. Without these steps, your data risks being mostly “garbage in, garbage out.”

Data Dictionary

All of your events and their properties must be documented thoroughly and ready to use. The documentation provides the context, detailed explanations, and history of any modifications. In product development, change is the only constant—and keeping track of every version of your data dictionary entry is critical to separate noise from meaningful insights.

Data Quality

Everything that can possibly fail, will. Your data pipeline may stop; your code changes may introduce bugs; and your data may be lost or duplicated. All of these possible incidents should be monitored and reported to guarantee the accuracy and completeness of your data—ideally before others have wasted days trying to figure out anomalies.

Time

Most product analytics use cases deal with time-series data, which should be captured and stored at the lowest granular level possible, and ideally context-free. Regardless of the time zone that the analysis uses, of the calendar month (four weeks or 30 days), or of the fiscal quarter...your raw data should contain the basic information needed to satisfy these use cases.

Typically, your events should have their timestamps captured and stored (accurate down to the millisecond), and they also should be in sync with your other data sources (e.g., dimension tables) for joining purposes. This is especially important in the attribution and experiment use cases, where the campaign or A/B test treatment should only impact the events that happened after the touch point.

Unfortunately, this kind of straightforward causal dependency is often neglected or even not supported by product analytics tools (because it requires a special yet expensive join using “Correlated Subquery.”

Slow-Changing Dimension

One frequently used design pattern for the product analytics data model is Slow Changing Dimension (SCD), specifically “Type 2: Add rows with effective date."

For example, your users may have been touched multiple times by your acquisition or social campaigns, in which case you can append a new row for every touchpoint—when it happens, along with an effective date to mark the effective time range of the entry.

This gives the ultimate flexibility at analysis time, allowing you to apply different kinds of attribution strategies like first-touch, last-touch, or multi-touch.

Pre-Compute Vs. On-the-Fly

Some measures like session length and total viewing time must be computed across multiple events, such as “Session start/end” and “View start/end.” However, in real life, the ending events have to be derived based on certain thresholds like idleness or no action.

It can be tempting to pre-compute such measures into their own tables, often because your product analytics tool can’t handle them on-the-fly. Unless you have a very fixed definition, usually it’s a bad idea to invest in heavy-duty ETL jobs, not to mention another table with “duplicated” information. How about when you want to see the “what-if” scenario? Or what if you want to troubleshoot some bugs and thus need to change these definitions dynamically?

The new generation of warehouse-native product analytics tools should be able to let you define such measures as reusable components and easily compute them on-the-fly to give you the maximum flexibility, while hiding the complexity.

Another similar use case is combining legacy events into one logical event. For example, over time, you may have dozens of variants of login events with different names, but for analytical purposes, they should always be treated as a single event. Your product analytics tool should allow you to group these together during query time dynamically.

Data Pipeline

You don’t want to put all your eggs in one basket and be constrained by your product analytics vendor’s SDKs. Choosing the best tool for the job is always a better idea, instead of getting locked into something that claims to be universal, but which really can’t get anything done well.

For event instrumentation, you don’t need to reinvent the wheel. There are already very mature customer data platforms (CDPs), including commercial and open-source solutions to provide you with the ultimate freedom to collect and send the data anywhere. With a simple flip of a switch, you can decide which data to collect and send that data to a cloud data warehouse or other vendor as the destination. Even better, you can choose to replay your history data if there are any mishaps or needs.

The days are long gone when we had to invoke API calls or build designated batch jobs to consume data from your vendors (those handling your acquisition, experiment, and reactivation campaigns). That way of working was expensive, fragile, and a headache to maintain.

All cloud data warehouses are headed toward data-sharing, and your vendor should be able to share your data securely and let you query it in real-time without ever needing to make a copy. All it takes is several lines of SQL code, and you can join with these vendors’ dimension tables live in all of your product analytics.

Use case:

How a video-streaming provider successfully modeled its data for product analytics

Background

A top video-streaming platform (“AcmeTV”) with millions of daily active users started its product analytics journey using Segment as its CDP to instrument and collect user behavior events data. Its initial plan was to get insights into its products using Tableau (a business intelligence reporting tool), hiring many analysts to build reports.

Quickly, AcmeTV realized that this solution couldn’t scale. Without the ability to do self-service analytics, the product and marketing teams couldn’t get their questions answered in a timely manner. They were experiencing delays that significantly impaired the speed of decision-making and of adapting to changes in product and marketing.

Challenge

Almost all self-service product analytics tools would have required AcmeTV to send billions of events daily to proprietary data silos. From the beginning, this was a no-go for two main reasons:

- Sending billions of events daily would have carried an astronomical price tag. Also, they were not open to the option of working just with sampled data, because some of their key business metrics rely on heartbeat signals (for example, loyalty calculated based on viewing seconds).

- AcmeTV’s growth relies heavily on its content strategy and user engagement campaigns (such as push notifications). An enormous amount of data already existed in its Snowflake instance that could be used dynamically, joining it with user behavior events. Building and maintaining ETL jobs to sync this other business data to a third party would be too daunting, error-prone, and expensive.

Solution

Instead of putting the project on hold for 6-12 months while preparing a "proper data model," we recommended that "right now" is the best time for AcmeTV to start deploying product analytics.

We provided them an impactful POC with minimal effort on their part, which went live within a week. All they had to do was flip a switch in Segment to send their event data into Snowflake. The data was raw, and the volume was huge—spread over hundreds of tables—but Kubit managed all the complexity and immediately delivered a perfectly functional self-service product analytics solution.

Results

As it launched a new platform, AcmeTV got critical insights, specifically into understanding the subscription funnel and how different content drives revenue and retention.

Over the course of the next several months, the solution evolved rapidly—guided by new insights into their data, guided by Kubit. These included:

- Discoveries that many of their events had been instrumented incorrectly or even misspelled. Luckily, Kubit provided many ways to abstract and remove such complexity from the product team.
- At instrumentation time, some critical properties weren't available to the client. Kubit allowed them to join with other dimension tables to filter and slice-and-dice. There were no ETL or backfill jobs required.
- Everything happened dynamically. Interestingly, many metrics or KPIs the team had defined before the platform launch were proven ineffective or even wrong. Kubit helped them refine their requirements and adjust their data strategy—from instrumentation to measurement, and then decision-making.

At the end of the third month of using Kubit, AcmeTV had a very clear grasp of its users' engagement and retention, and it was able to build a very effective and mature content/push strategy based on those insights. They quickly realized that more than half of their original assumptions were either completely wrong, or lacked enough consideration. Kubit helped them to iteratively improve their understanding of data and utilize it to make data-driven decisions in a timely manner.

Kubit brought about all of these impacts on AcmeTV without any development work from its data engineering team. The Kubit platform can adapt to changes and make them live within days, all thanks to its warehouse-native architecture.

Now, with most of AcmeTV's use cases clearly defined and battle-tested in real life, both teams are working together to build an improved, cleaner data model for the future. Meanwhile, their raw schema still functions as the workhorse for ad-hoc analysis, instant troubleshooting, and hypothesis verification.

If AcmeTV had waited for a "better data model" to get started, it would have completely missed its platform launch and some of its most critical product insights.

About the author:

Alex Li is the Founder and CEO of Kubit, the first warehouse-native product analytics platform.

Alex created Kubit out of his own experiences and frustrations as a CTO at Smule, during which he realized that traditional product analytics tools simply can’t cope with the massive volume and scale of product data in modern enterprises.

It was a market gap serious enough to inspire Alex to build his own solution—one that empowers enterprises to take full advantage of all their product data by leveraging the latest advancements in data warehouse and data-sharing technologies.

You can learn more about getting started with product analytics, or improving your current product analytics approach, at kubit.ai. Or, feel free to drop a note to Alex and his team at info@kubut.ai.

kubit

www.kubit.ai