segment

Go back to Growth & Marketing

Choosing a Database for Analytics

Stephen Levin on November 23rd 2015

When your analytics questions run into the edges of out-of-the-box tools, it's probably time for you to choose a database for analytics. It's not a good idea to write scripts to query your production database, because you could reorder the data and likely slow down your app. You might also accidentally delete important info if you have analysts or engineers poking around in there.

You need a separate kind of database for analysis. But which one is right?

In this post, we'll go over suggestions and best practices for the average company that's just getting started. Whichever set up you choose, you can make tradeoffs along the way to improve the performance from what we discuss here.

Working with lots of customers to get their DB up and running, we've found that the most important criteria to consider are:

the type of data you're analyzing
how much of that data you have
your engineering team focus
how quickly you need it

What types of data are you analy

into a Word Doc?

If you answered Excel, a relational database like Postgres, MySQL, Amazon Redshift or BigQuery will fit your needs. These structured, relational databases are great when you know exactly what kind of data you're going to receive and how it links together — basically how rows and columns relate. For most types of user analysis, relational databases work well. User traits like names, emails, and billing plans fit nicely into a table as do user events and their properties.

On the other hand, if your data fits better on a sheet of paper, you should look into a non-relational (NoSQL) database like Hadoop or Mongo.

Non-relational databases excel with extremely large amounts of data points (think millions) of semi-structured data. Classic examples of semi-structured data are texts like email, books, and social media, audio/visual data, and geographical data. If you're doing a large amount of text mining, language processing, or image processing, you will likely need to use non-relational data stores.

Choosing a Database		
Criteria	Relational	Non-relational
Type of Data	Structured	Unstructured
Would fit in massive	Excel Sheet	Word Doc
The schema	Stays the same	Changes often
Works well with data like	User data, Inventory	Email content, Photos, Video
For analysis like	User paths, Funnel analysis	Text mining, Language processing
Can query with	SQL	MapReduce, Python

How much data are you dealing with?

The next question to ask yourself is how mu more you have, the more helpful a non-relational data

Here's a handy chart to help you figure out which option is right for you.

Database Options by Scale				
Data Size	< 1TB	2TB-64TB	64TB-2PB	#alloftheData
Database that's a good fit	Postgres MySQL	Amazon Aurora	Amazon Redshift Google BigQuery	Hadoop

These aren't strict limitations and each can handle more or less data depending on various factors — but we've found each to excel within these bounds.

If you're under 1 TB of data, Postgres will give you a good price to performance ratio. But, it slows down around 6 TB. If you like MySQL but need a little more scale, Aurora (Amazon's proprietary version) can go up to 64 TB. For petabyte scale, Amazon Redshift is usually a good bet since it's optimized for running analytics up to 2PB. For parallel processing or even MOAR data, it's likely time to look into Hadoop.

That said, AWS has told us they run Amazon.com on Redshift, so if you've got a top-notch team of DBAs you may be able to scale beyond the 2PB "limit."

What is your engineering team focused on?

This is another important question to ask yourself in the database discussion. The smaller your overall team, the more likely it is that you'll need your engineers focusing mostly on building product rather than database pipelines and management. The number of folks you can devote to these projects will greatly affect your options.

With some engineering resources you have more choices — you can go either to a relational or non-relational database. Relatic e than NoSQL.

If you have some engineers to work on the setup, but can't put anyone on maintenance, choosing something like Postgres, Google SQL (a hosted MySQL option) or Segment Warehouses (a hosted Redshift) is likely a better option than Redshift, Aurora or BigQuery, since those require occasional data pipeline fixes. With more time for maintenance, choosing Redshift or BigQuery will give you faster queries at scale.

Side bar: You can use Segment to collect customer data from anywhere and send it to your data warehouse of choice. **See how it works here** 👉

Relational databases come with another advantage: you can use SQL to query them. SQL is well-known among analysts and engineers alike, and it's easier to learn than most programming languages.

On the other hand, running analytics on semi-structured data generally requires, at a minimum, an object-oriented programming background, or better, a code-heavy data science background. Even with the very recent emergence of analytics tools like Hunk for Hadoop, or Slamdata for MongoDB, analyzing these types of databases will require an advanced analyst or data scientist.

How quickly do you need that data?

While "real-time analytics" is all the rage for use cases like fraud detection and system monitoring, most analyses don't require real-time data or immediate insights.

When you're answering questions like what is causing users to churn or how people are moving from your app to your website, accessing your data with a slight lag (hourly or daily intervals) is fine. Your data doesn't change THAT much minute-by-minute.

Therefore, if you're mostly working on after- for a database that is optimized for analytics like

to quickly read and join data, making queries fast. They can also load data reasonably fast (hourly) as long as you have someone vacuuming, resizing, and monitoring the cluster.

If you absolutely need real-time data, you should look at an unstructured database like Hadoop. You can design your Hadoop database to load very quickly, though queries may take longer at scale depending on RAM usage, available disk space, and how you structure the data.

Postgres vs. Amazon Redshift vs. Google BigQuery

You've probably figured out by now that for most types of user behavior analysis, a relational database is going to be your best bet. Information about how your users interact with your site and apps can easily fit into a structured format.

`analytics.track('Completed Order')` — `select * from ios.completed_order`

IOS.COMPLETED_ORDER				
USER_ID	PRODUCT_NAME	PRODUCT_SIZE	PRODUCT_	REVENUE
SLKJ79SF	POLKA DOT SKIRT	6	1837	59.99
LKJ2847F	ELEPHANT BOXERS	S	8301	29.99
LDN834FS	ARGYLE SOCKS	M	7361	9.99
78SKJU1S	LEATHER BELT	M	8472	24.99

So now the question is, which SQL database to use? There are four criteria to consider.

Scale vs. Speed

When you need **speed**, consider Postgres: U... o...

That's why when you need **scale**, we usually recommend you check out Redshift. In our experience we've found Redshift to have the best cost to value ratio.

Flavor of SQL

Redshift is built on a variation of Postgres, and both support good ol' SQL. Redshift doesn't support every single data type and function that postgres does, but it's much closer to industry standard than BigQuery, which has its own flavor of SQL.

Unlike many other SQL-based systems, BigQuery uses the comma syntax to indicate table unions, not joins according to their docs. This means that without being careful regular SQL queries might error out or produce unexpected results. Therefore, many teams we've met have trouble convincing their analysts to learn BigQuery's SQL.

Third-party Ecosystem

Rarely does your data warehouse live on its own. You need to get the data into the database, and you need to use some sort of software on top to analyze it. (Unless you're a-run-SQL-from-the-command-line kind of gal.)

That's why folks often like that Redshift has a very large ecosystem of third-party tools. AWS has options like Segment Data Warehouses to load data into Redshift from an analytics API, and they also work with nearly every data visualization tool on the market. Fewer third-party services connect with Google, so pushing the same data into BigQuery may require more engineering time, and you won't have as many options for BI software.

You can see Amazon's partners here, and Google's here.

That said, if you already use Google Cloud S ou may benefit from staying in the Google ecosyste a

won't be a deal breaker either way, it's definitely easier if you already use one to stay with that provider.

Getting Set Up

Now that you have a better idea of what database to use, the next step is figuring out how you're going to get your data into the database in the first place.

Many people that are new to database design underestimate just how hard it is to build a scalable data pipeline. You have to write your own extraction layer, data collection API, queuing and transformation layers. Each has to scale. Plus, you need to figure out the right schema down to the size and type of each column. The MVP is replicating your production database in a new instance, but that usually means going with a database that's not optimized for analytics.

Luckily, there are a few options on the market that can help bypass some of these hurdles and automatically do the ETL for you.

But whether you build or buy, getting data into SQL is worth it.

Only with your raw user data in a flexible, SQL format can you answer granular questions about what your customers are doing, accurately measure attribution, understand cross-platform behavior, build company-specific dashboards, and more.

Share

Go back to Growth & Marketing

Download The Customer Data Platform Report 2020

Packed full of market trends, analysis, and insights that we've summarized from talking to our thousands of customers.

Get the report

Become a data expert.

Get the latest articles on all things data, product, and growth delivered straight to your inbox.

Your work email

Subscribe

segment

Privacy

Terms of

Website Data Collec

Thanks for stopping by! 👋 Interested in checking out the IDC report that names Segment the #1 CDP?

1