Instructions: The narrative below presents a hypothetical situation where you are an analytics engineer at a SaaS startup and poses some scenarios in a chronological order. Use your previous work experience and judgment to augment the details provided. Build your own narrative around the fictional SaaS company and your role there, and then detail how you would build an analytics foundation that solves their data needs. Answers do not need to be isolated to the immediate preceding prompt; you may use previously given information as well as context from your previous answers.

DogDB is a California based company that has developed a novel NoCATS database and offers a managed, hosted solution as a monthly SaaS subscription with free, medium ($50/mo), and enterprise tiers ($500/mo). DogDB is seeing their number of customer accounts skyrocket ("up and to the right") and have hired you as the first dedicated analytics engineer to help them understand and scale their data capabilities, in anticipation of an incipient funding round.

DogDB sells their service through a web-facing rails application. Here, a DogDB customer can sign up for an account, choose a pricing tier, and configure their NoCATS deployment. The accounting settings and configurations are stored in a PostgreSQL database.

Currently, DogDB has tables which are sampled below.

Table **customer_accounts**

account_id	email	current_tier	created_at	updated_at
1	abc@123.com	Free	2019-05-01 21:13:05.156042+00	2019-05-01 21:13:11.804514+00
2	123@abc.com	Medium	2019-07-12 16:05:02.414454+00	2020-01-04 17:23:05.594305+00
3	hello@world.com	Enterprise	2019-07-23 12:26:47.571431+00	2019-09-17 04:32:32.493065+00

Table **customer_interactions**

account_id	channel	category	service_rep	status	created_at	completed_at
1	web	Tech Support	Andy	resolved	2021-01-25 19:11:35.295813+00	2021-01-25 19:13:52.812371+00
1	email	Billing	Jillian	open	2021-04-06 22:23:09.581234+00	*NULL*
3	web	Billing	Monica	resolved	2021-11-13 06:25:54.821374+00	2021-11-15 12:19:33.882136+00
7	phone	Account Change	Derek	canceled	2022-02-14 15:02:47.219352+00	2022-02-20 09:22:48.145523+00

Table **customer_licenses**

account_id	license_data	created_at	updated_at
1	{ “license_id”: “d17cb11cda9ba249c22f67e4aed65d0f65f1a80c”, “role”: “analyst”, “status”: “active” }	2022-03-12 02:56:37.652093+00	2022-03-12 02:56:37.652093+00
6	{ “license_id”: “be49ad8f4a68fbbdd1674b41da20759f54b0e930”, “role”: “developer”, “status”: “active” }	2021-05-28 04:42:58.955093+00	2021-05-28 04:42:58.955093+00
6	{ “license_id”: “8541866bb3a4c4ecf070b2c1b2f7bb9c0934d287”, “role”: “admin”, “status”: “active” }	2022-10-30 21:33:46.353060+00	2022-10-30 21:33:46.353060+00
35	{ “license_id”: “60831f59a531eef325e525ad58bae0e5e8c2d75a”, “role”: “developer”, “status”: “disabled” }	2021-03-26 02:38:02.136033+00	2022-07-21 23:03:29.862040+00

1) Based on the table design above, what are your initial thoughts about DogDB’s data tracking? What are some of the advantages (if any) of their data models, and what are the shortcomings (if any) you foresee in DogDB’s future?

Good start for data tracking. Great that there is some information on customer accounts, how many licenses per account and interactions with customer support. With tables as is so far, DogDB is able to start segmenting customers into free and paid and do a quick analysis on what are common help desk topics. However, some opportunities for improvement upon first glance would be to change some of the input data: for customer_licenses, change license_data column from nested JSON to unnested columns per key/value pair and to truncate timestamps since seconds rounded to whole number are precise enough. Last, but not least would need to create more tables tracking user behavior to pin down what the business is doing well in hypergrowth and what it is not doing so well to make data-informed, evidence-based strategic decisions.

Given that the company is in hypergrowth stage, the company’s virality stage has been met. So leaders at DogDB are wondering how the customer base became so and how long it could last before leveling off. Below

are metrics of interest. After going through what needs to be measured during this stage of the company, will finalize suggestions on what data models need to be supplemented and created. This will resolve data tracking shortcomings and ensure analytics is set for heavier, more complex analytics.

- What needs to be measured in general for SaaS start-up:
  Customer Life Cycle:
  - Attention
  - Enrollment
  - Stickiness
  - Conversion
  - Revenue (Per customer)
  - Customer Acquisition Cost
  - Virality
  - Upselling
  - Uptime and Reliability (running without interruption)
  - Churn
    - definition: who has not logged-in in the past 90 days
    - number of churns per period/number of customers at beginning of period
    - track free and paid separately with same cadence
    - be careful of variable growth rate
      - churn is not normalized for behavior and size (can get different churn rates for the same type of behavior)
      - fix this by taking average of customers at beginning and end of pd being analyzed
        - this can get tricky if have hockey stick growth/churn period is defined
          - solution: measure by COHORT or measure churn PER DAY
  - Lifetime Value (LTV)

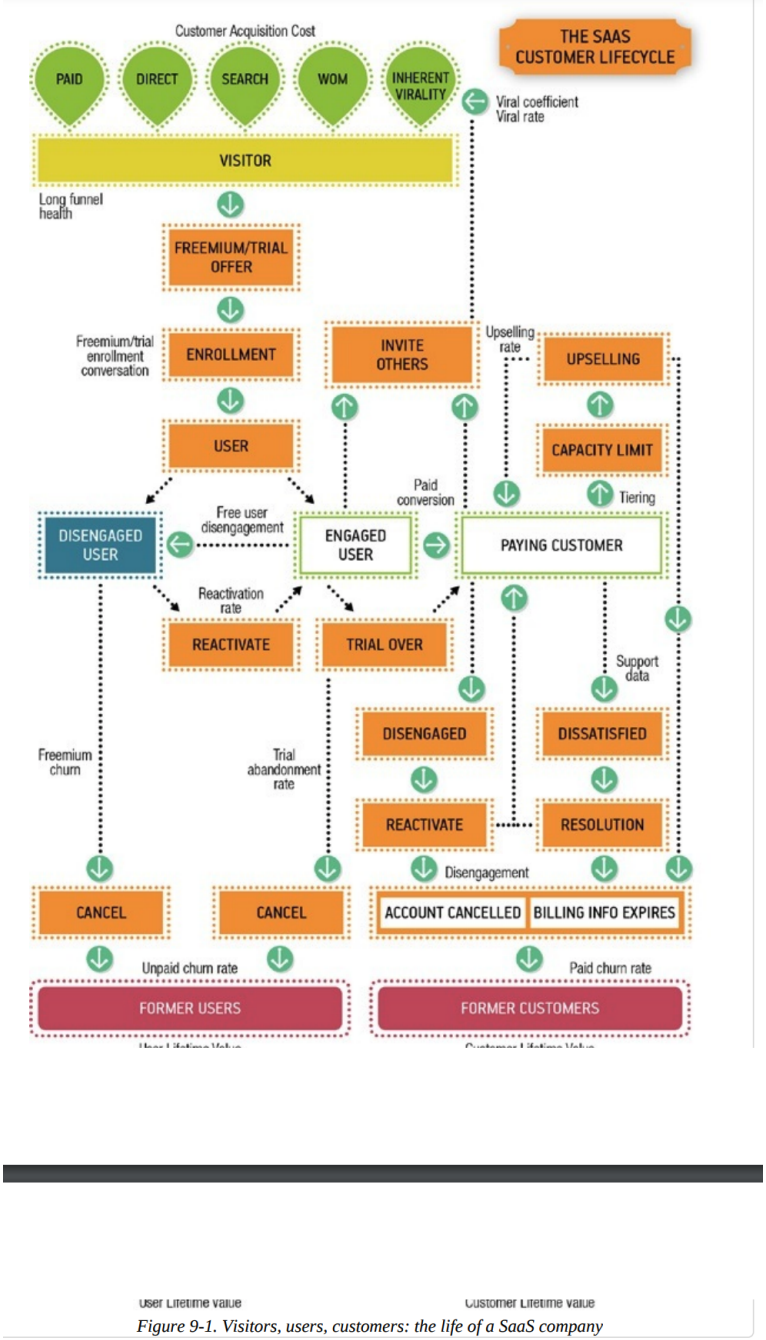
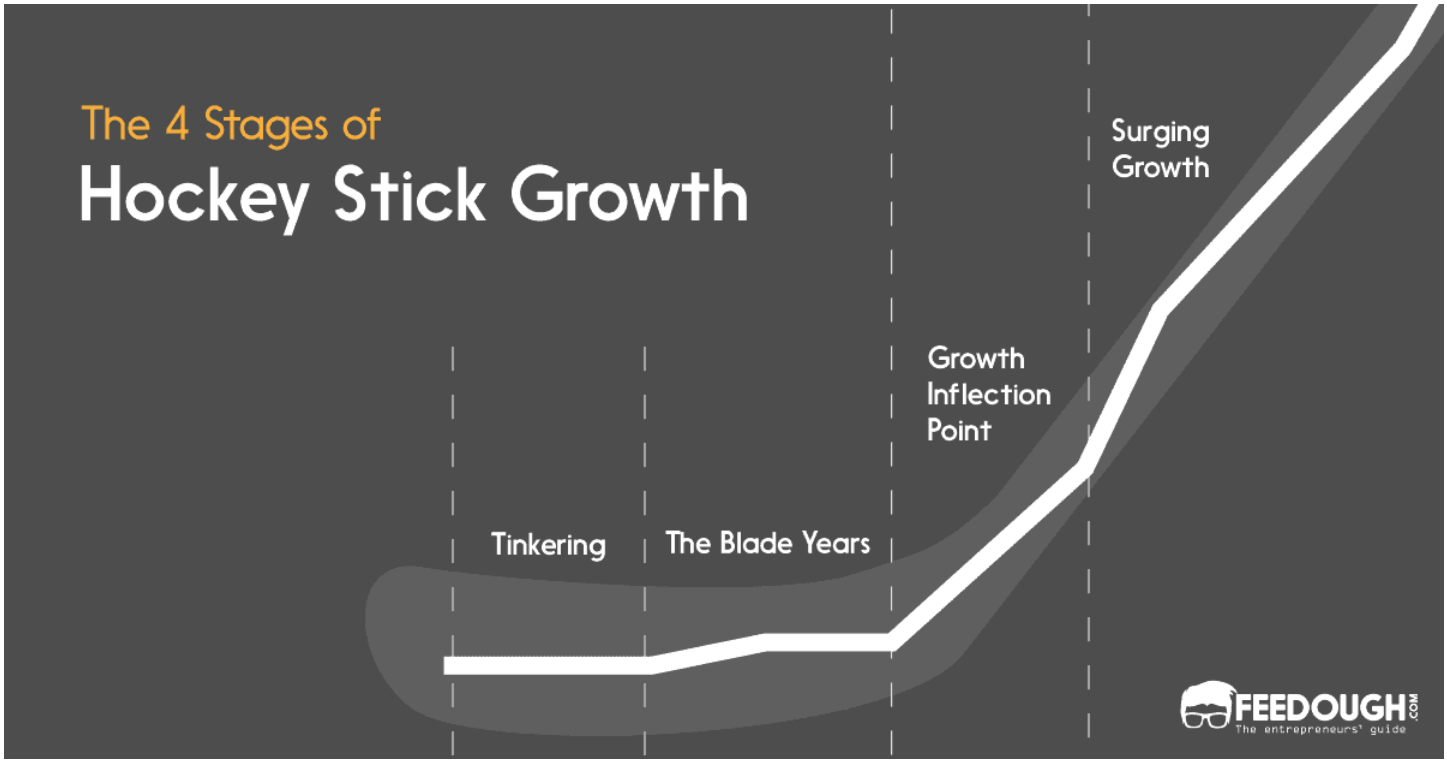


Figure 9-1. Visitors, users, customers: the life of a SaaS company

- Stage of the company: pre-IPO (re-consider financial model- consumption and compute cycles, but harder to predict and explain business)
    - Hockey stick growth - inflection point after subdued growth (blade), shaft part



    - Virality/Revenue Stage
    - Now, need to figure out cause:
        - internal: feature, price change, change in business model
        - external: law/regulation that caused wide scale but previously unseen need that product can satisfy

- What needs to be tracked (for now):
    - number of customers/accounts
    - invoices to track revenue (includes monthly tier)
    - account for costs -> profitability leads to growth (Customer Acquisition Cost (CAC) lower than LTV)
    - channel customer came from
    - # subscriptions
    - stickiness (how much engagement per day)
        - define engagement (3 log-in's/day)
        - what are they doing when inside NoCATS db
        - how long are they spending in NoCATS db - set meaningful threshold for happy customer
        - specific features used and others that are not being used
        - where is the highest engagement
            - What do these customers have in common? Re-focus on their needs and grow from there
    - conversion (paid customer)
    - did customer come back by themselves or based on a marketing nudge/notification
    - segment customers based on free/paid
        - do they all live in the same city
        - did customers who eventually pay come from one specific marketing channel?
        - are customers causing virality all under 30 years old

- from cohort analysis, with any behavior linked to paid customer (increased revenue), perform A/B testing with desired product feature change to try and increase stickiness across customer base
  - Also, any poor behavior from paid customer/free tier cohorts, would want to know which aspects of the business are too risky to not invest in those features further

Given that the customer base has skyrocketed in number, the priorities right now would be stickiness and conversion.

In other words, need to track virality and revenue:

1. VIRALITY - this is what was just accomplished. need to track how it came about and what the future holds based on measurements
   a. what type of users (just have customer accounts with exponential growth), need to segment based on tiers (free, medium, enterprise)- are they mostly free? and if free, are they engaging or thinking the product is something else?
      i. look at actions within product that drive virality and make sure you're measuring them properly and have lines in the sand that you're targeting
      ii. what does big usage and high revenue customers have in common
   b. measure churn to see if there is ROI
   c. How was virality accomplished?
      i. Inherent - built into product and happens as function of use (e.g. Instagram, but not for NoCats db)
         1. -usually brings in engaged customers (most natural) turning into revenue
      ii. Artificial - virality is forced, and often built into a reward system (e-comm app - Mercari, possibly for NoCats db - more friends on platform, more credits to use for next subscription renewal or even for compute/storage)
         1. brings in traffic, but how about engagement
      iii. Word-of-mouth - virality is conversations generated by satisfied users, independent of your product/service (Reddit sub's, which is highly likely for NoCats db)
   d. To measure viral growth: Viral coefficient
      i. calculation involves:
         1. invitation rate: number of invites sent divided by number of users there are
         2. acceptance rate: number of signups divided by number of invites
         3. multiple these together
      ii. consider cycle time (how long it takes for users to invite others)
      iii. goal: >1 which means product is self-sustaining
   e. Net Promoter Score (NPS) - how likely is user going to tell friends about product; compares strong advocates to those who are unwilling to recommend it (referral, marketing collateral)
   f. Lead Indicator (to determine future growth and revenue) - look at groups who stuck and did not stick
      i. e.g. return frequency (days since last visit)
      ii. needs to be tied to part of business model (daily traffic)
      iii. should come early in user's lifecycle/conversion funnel
      iv. early extrapolation to get prediction earlier

2. REVENUE - moving from proving idea is right to proving can make money in a scalable, consistent, self-sustaining way
   a. How?

        i. Revenue/customer -> indicator of actual business health
            1. if revenue is going up and rev/customer is going down, need lot more customers to sustain revenue growth
            2. look at subscriptions, LTV compared to CAC
    b. experimenting with getting price right: bundles, sub tiers, discounts
    c. look at return on investment of marketing dollar
        i. divide how much you changed annual recurring revenue in the past quarter by what it cost DogDB to do so
        ii. want to be >1 to increase sales and marketing spend
    d. how to increase revenue: Given that NoCats db is a subscription model, and if NoCats db is fighting churn, then upselling customers to higher-capacity packages with broader features is its best way of growing existing revenues, so customers will spend a lot of time on more offerings
    e. money customer brings in minus cost of acquiring customer -> drives growth
        i. tweaking business model: how, when and what is charged
    f. keep in mind business ratios
    g. business value based on money in/money out and maximum money can be put in
    h. focus on revenue/customer, more customers, more efficiencies, greater frequency and so on
    i. if revenue is going down, pivot to different market instead of bloating product with features or starting from scratch
    j. goal is to grow and break even since anything beyond paying bills is survival

Metrics to look at for investors to get a signal on whether or not to fund a SaaS company that is mid-stage and going through hypergrowth:

- how many of the customers are Fortune 500
- what is the % contribution to total revenue of Fortune 500 companies
- how big are these customers (Annual Contract Values - ACV) vs total customer count - YoY
- Net Retention Rate (Revenue by cohort of customers in 2nd yr/same cohort in 1st yr)
    - According to Redpoint Ventures, top quartile of net retention rate for SaaS companies is 124%
        - adding new logos (large accounts?)
        - customers constantly spending more on platform
- LTV/CAC
    - CAC - Sales and Marketing spend in quarter by new customer count in same period
    - LTV
        - amortized sales commissions period
        - average contract is revenue/customer at quarter * 4 * net retention rate (to capture future contract expansions)
    - ratio should not be below 3 meaning business is growing at a healthy ratio, capturing market share and is poised to generate future positive cash flows
- typical contracts: 12-36 months
- Profit and Losses (Financials)

In order to measure such, I would need to work with DE to get into the database more data points such as what is below.

1. invoice table

- account_id, invoice created, invoice item type (tier), price, payment method, payment total, how long subscription/contract is

2. customer_accounts table

    - add channel (used to get information on how customer landed on NoCats db), customer: industry, country of operation, head count, is it a Fortune 500 company, total number of licenses, if they opted into marketing drip email list/notifications

3. platform event log table

    - all login times (this will help determine active/churn/resurrected accounts), all platform touch points (creating a view, storing tables)
    - user journeys
        - features users are using lot of and features not used a lot
        - attribution of resurrected/churned clients possible after adding event log

4. marketing/sales table

    - to get marketing/ad spend to calculate CAC, number of invites sent, NPS score

Every month, DogDB's accountant receives a report from the engineering team to assist with accounting. Customers on the medium tier are charged by credit card, but the accountant must know ahead of time what the expected charge will be. Customers on the enterprise tier are sent an invoice from DogDB.

2) Describe what the engineering team is most likely doing currently to support accounting in terms of process. Include the queries they are running if you think you can take a guess at what they are. What are some of the shortcomings of the current process?

It seems like DogDB is asking the Engineering team to try and save the company money by producing two types of transactions based on the customer's tier type. With credit cards, DogDB would have the customer pay a fee to complete credit card payment and with invoice that would not be necessary. There is no service fee with invoices. At the same time, with the credit card process, there is a higher probability of recurring membership with possible automated credit card payment. Also, credit card payment is just more convenient and efficient. However, with invoice there is ability to give higher paying customers flexibility in payment-whether it be credit card/ACH. Shortcoming of the current process is that it is two different accounting processes across different tiers. There can be payment set-up issues once customer upgrades/downgrades from tier. This could negatively affect payment to DogDB's account, which would affect overall revenue/growth metrics for quarterly reports. If there were identical payment options across multiple tiers, it would possibly make accounting less prone to errors.

Possible queries run by Engineering to provide report to Accounting:

Under Results, Query #1 from: https://www.db-fiddle.com/f/cGXjLQJ3Gz1vy7ucVMjngx/6

The customer service team has asked one of the data analysts to develop a dashboard to illustrate the monthly interaction volume of the accounts with the *10 highest number of active developer licenses*. They are looking

for various breakdowns by *account*, *month*, *interaction channel*, *category*, *service representative*, and *interaction status*.

3) You have been tasked with designing a model to provide the data for the analyst. How would you structure the output? Include the query you would use to create the model.

Under Results, Query #2 from: https://www.db-fiddle.com/f/cGXjLQJ3Gz1vy7ucVMjngx/6

The analysts have been complaining that one of the view models created before you arrived is taking too long to run. This view is used in several dashboards across various business departments, and for ad-hoc analyses on a regular basis.

4) What are the troubleshooting steps you would take to identify the problem? Based on experience you've had in the past, develop a hypothetical narrative for identifying what the problem is and then the steps to address fixing it.

The troubleshooting steps I would take to ID the problem would be:

1. Some initial questions:
    a. Why is PostgreSQL still being used as DB since it does not scale well?
    b. What is the BI visualization tool: Looker/Mode? These two scale well with a business
2. Possibly change the reference table:
    a. if there is a similar main table that is on a scheduled build, reference that table instead of building a view from scratch
    b. see if view is built from scratch too frequently; if so switch build when dashboard is loaded only
    c. try and utilize cache function if there is one
    d. (if BI tool is: Looker ☝)utilize persist_for:, sql_trigger_value:, and sql_trigger_if:to modify table based on current table's column conditions (if max(date) in current table is < max(date) in referenced table) so that whole view is not built from scratch
3. Analyze the query that builds the view:
    a. with each CTE, look at how long it takes to run
    b. with CTE's that run relatively the longest see if can use partitioned part of referenced table (WHERE date column is between beginning_date and end_date)
    c. see if window functions can be simplified to aggregate functions if columns do not need to be compared row-wise
    d. select certain columns
    e. if able to aggregate, do so
    f. eliminate any ORDER BY
    g. check overall run time after modifications and hopefully there is improvement
4. Overall BI tool governance
    a. shut down any tables on scheduled builds for dashboards that are no longer in use
    b. reduce number of tiles per dashboard
    c. take off any jobs in queue (dashboard requests) that are taking too long/taking up too many resources and request job be done during less high traffic times (at night)

5) Based on what you understand of DogDB so far, what would your first week at DogDB look like? What is your number one priority to try and change?

Given that company is most likely mid-stage with approximately 50-150 employees, there would need to be a focus on implementing SQL-based data modeling, figuring out marketing attribution, having robust web analytics if it does not exist yet and tackling forecasting challenges.

Considering I am one of the first analytic hires for the data team at a SaaS company going through hypergrowth, the following would be my first week's tasks:

1. Meet with all business stakeholders since they probably have a lot of urgent questions. Understand what the current and near future business requirements are.
2. Ask business stakeholders how they access their data and how they generate their reports as of now. Most likely, there are performance issues because BI tool is using Postgres DB for analytic queries. Postgres was not designed to do that. So a better DB for BI would be BigQuery/Snowflake/Azure/Redshift. Also, hopefully Looker/Mode is used as BI tool. Otherwise, may pay for scaling inefficiencies down the line.
3. Meet with DE team and see where all the data is stored and where it goes to, how it is processed. Understand if there is just a single DB, or is there also a DWH and a Data Lake; Ask how data extraction, loading and transformation are done. Get an idea of what the data tech stack is and see if they are going to move BI DB to columnar DWH versus sticking with OLTP Postgres DB which is not meant for scaled business analytics.
4. Document tables. Have a data dictionary for business definitions.
5. Create communication channels on Slack for AE regarding data governance - set up some rules for how data is to be accessed and processed, etc.
6. Make sure there is a data code repo.
7. Explore the current state of DB– what are the most used tables.
8. Define main tables for SaaS company: accounts, subscriptions, payments, event log, marketing, sales, customer service/help desk (ZenDesk).
9. Based on dashboard performance issues, start purging Looker's views and defining some main reference and derived models.
10. Consolidate dashboards– one-off, ad-hoc table builds/dashboards can be placed in dormant mode unless they become mainstream.

My number one priority to try and change would be BI tooling and data models:

1. BI/Analytics database from Postgres to columnar one that can handle data analytics at scale.
2. Push for Looker/Mode as BI tool for scale as well.
3. Define main tables.
    a. supplement already existing ones (accounts, customer service)
    b. create new tables (subscriptions, payments, event log, marketing, sales)