Building the Business Case

for dbt Cloud

- ▶ **Step 1: Align with company objectives**
- ▶ **Step 2: Benchmark, and propose ideal outcomes**
- ▶ **Step 3: Identify solutions that support key outcomes**
- ▶ **Step 4: Define a time-bound proof of concept**
- ▶ **Step 5: Tabulate results**
- ▶ **Step 6: Present findings**
- ▶ **Step 7: Create the Implementation plan**
- ▶ **Step 8: Prepare to ramp**

dbt Labs

Table of Contents

About this guide

This guide is based on 18 hours of interviews with over 20 dbt Cloud customers. We used their lived experiences to provide a roadmap that can be used by anyone trying to introduce dbt Cloud into their organization. Of course, your situation is going to be unique. We recommend pairing the resources and templates in this guide with a visit to the dbt Community Slack. Here you'll find 48k+ data professionals facing similar challenges. The **#toward-analytics-engineering** channel was the starting point for the survey data included in this report and it's where discussions are happening every day about how to champion the analytics engineering workflow.

Why dbt Cloud?

dbt Cloud enables data teams to adopt analytics engineering best practices like testing, version control, and documentation without the overhead of manual coordination and infrastructure management. Because transformations can be built with SQL through an intuitive Integrated Development Environment (IDE), dbt Cloud encourages a wider variety of people to participate, eliminating bottlenecks and enabling each person on the data team to focus on higher leverage work.

Mapping business goals to data opportunities

Thanks to things like workflow automation and analyst enablement, data teams have cited **20x multipliers** in data velocity and dramatic increases in **data trust**. Despite major benefits, more than ¼ of dbt users surveyed reported facing some initial resistance to this way of work due to: legacy tech debt, competing priorities, and inability to secure budget and buy-in.

There is a common thread at the heart of each of these issues: misalignment between business priorities and the work needed to improve data enablement. The data team knows how broken pipelines affect productivity, but does the rest of the business? How does time spent troubleshooting impact business goals? **What does your organization lose when you lose time?**

This guide was designed to help you map pipeline problems to business goals, quantify projected loss and lift, and secure the support you need to adopt dbt Cloud at your organization. The tips, templates, and tales in these pages have been gathered via dozens of interviews with dbt Cloud customers – two of whom you'll meet below.

Step 1: Align with company objectives

When seeking budget or headcount approval, the best place to start is where the business wants to end – your company objectives. Mapping problems to the goals they jeopardize will not only help you speak the same language as non-data stakeholders, but also provide leverage in asking for resources to help.

Template: Map analytic workflow problems to business goals

Contributing KRs	Owner	Current	Goal	Contributing factors
Increase upsell by 30%	Mktg	10%	30%	Requested changes to account views take 3 weeks to fulfill
				Dashboards take on avg 8 mins to load, frustrating analysts & reps
				Disparities across on-prem & cloud data reduces trust in account info

The template used in this guide was built as a result of conversations with several dbt Cloud customers.
The example inputs in subsequent tables are an abstraction of those conversations.

In the above example, company upsell rates have not progressed as expected. Polling key stakeholders on why they believe they're behind on achieving this goal will help connect business context to data roots. Use trends identified in these interviews to populate the "Contributing Factors" column.

Step 2: Benchmark, and propose ideal outcomes

Consider how improvements to data team workflows can improve issues outlined above. Add quantified benchmarks, and define desired outcomes.

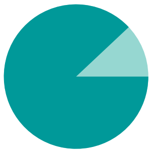
Template: Benchmark current state and set measurable desired outcomes

Problem		Desired Outcome
Requested changes to account views take 3 weeks to fulfill	→	Enable analysts to own transformation workflows and deliver data in 1 week
Dashboards take an average of 8 mins to load, frustrating analysts and reps	→	Eliminate redundant systems to hit sub 10 sec load time
Disparities across on-prem and cloud data reduces trust in account information	→	Migrate everything to a Cloud solution with routine testing for 100% parity

Step 3: Identify solutions that support key outcomes

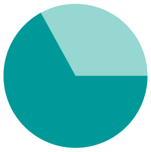
If you're evaluating dbt Cloud against other solutions, use your list of desired outcomes to inform a feature comparison matrix. This will help you narrow the field of potential contenders you will explore in a subsequent POV (proof of value).

In customer interviews, we learned a perceived "switching cost" from legacy tech was the #1 reason why data leaders struggled to secure buy-in on migrating to an analytics engineering workflow. For this reason, we've chosen to populate our comparison matrix with features of a typical "homegrown" transformation solution. We've also shaped subsequent cost/value calculators to consider company uplift along with total cost of ownership.



88%

Percentage of data practitioners that cited "legacy tech debt" as top challenge in implementing analytics engineering workflows.



67%

of dbt Cloud users said that enabling non-engineers to absorb more development work has been a top benefit of implementing analytics engineering workflows

Legacy In-House Tooling vs. dbt Cloud

	Legacy In-House	dbt Cloud
Modularity and portability	• Stored procedures with unknown ownership and dependencies	• SQL and Jinja for macros • Pre-built packages • git-enabled CI/CD to collaborate across teams • API for service integration
Accessible authoring	• Combination of Python and other scripting languages	• User friendly IDE that uses SQL SELECT statements for development
Job scheduling, monitoring, and alerting	• Multiple tasks to invoke jobs • Would require additional orchestration tools and manual processes	• Out of the box scheduler and job monitoring • Alerting for job status • Data freshness checks
Version Control	• No standardized version control	• Out of the box integration with most configuration management tools including GitHib and GitLab • Enables a code promotion path from development to QA to production • Changed code can automatically trigger CI test runs
Documentation	• Ad hoc	• Available, fully-hosted web experience • Automated + always up-to-date to eliminate human error
Dependency management	• Indiscernible lineage	• DAG generates as you code for up-to-date lineage • DAG in the IDE for seamless authoring
Data quality	• Ad hoc testing	• Automated testing • Snapshots to record changes to mutable tables • Lineage with an auto-updating data dictionary
Implementation & Support	• Engineering team	• “dbt Learn, Rapid Onboarding” • Professional services for support, audit, and refactoring
Cost	• Requires additional engineering support for maintenance	• No additional infrastructure cost • Annual license
Security	• On-premise, limited external networking requirements • Custom-developed integrations	• ELT architecture reduces risk / surface area • Enterprise-grade security features like SSO, RBAC, and SOC-2 compliance

dbt Labs

Step 4: Define a time-bound proof of concept

Once you’ve narrowed your prospective solutions, it’s time to put each to the test. Your POV should be structured to show demonstrable impact on the problem you prioritized above. The below list contains some recommended use case examples.

For each, consider things like total time to perform the task, cost of context-switching, resources required, stakeholders involved, cycles to completion, process governance, and quality of outcomes.

Use case examples:

1. A data engineer updates a customer account model to reflect changes in product type definitions
2. A job / model refresh fails midway through the pipeline
3. A financial analyst wants to validate the accuracy of an account’s ARR.

Step 5: Tabulate results

Record key results against your baseline for 2 weeks. We've populated the table with example inputs based on conversations with dbt Cloud customers, but your own may differ.

Template: Key results after one month proof of value test

dbt Cloud Impact			
	KPI	**Legacy In-House**	**dbt Cloud**
Baseline	# of practitioners developing	4 Engineers (E)	1 Engineer (E), 4 Analysts (A)
	Total # hours spent in development	60 (E) hours	20 (E) hours; 40 (A) hours Analysts can develop in the IDE, enabling engineers to serve as mentors
	Average compute hours for transformation workflows	40 hours	20 hours dbt Cloud parallelizes transformations to reduce run time
Accessibility Analysts self-serve validation and change requests to enable engineers to work on higher leverage tasks	# hours spent mapping lineage, answering questions, understanding dependencies	30 (E) hours	5 (E) hours, 10 (A) hours Analysts self-serve answers to lineage, freshness, and validity questions
Accessibility Key Result	**Stakeholder data delivery time**	**3 weeks**	**1 week**
Velocity User-friendly IDE enables more people to develop; DDL/DML abstraction speeds development pipelines	# data marts developed	1	4 4x productivity due to more developers enabled via IDE
	# hours servicing ad-hoc requests	15 (E) hours	2 (A) hr Analysts enabled to self-serve requests
	# hours spent managing infrastructure	5 (E) hours	0 dbt Cloud is fully hosted
Velocity Key Result	**Dashboard load time**	**8 minutes**	**5 seconds**
Quality Automated testing and alerting; snapshots to record changes and automated lineage documentation	# models tested in 1 month	0/5 Break-fix work only done reactively in production	50/50 Construct and automate custom tests with the ability to adjust alert sensitivity
	# hours spent fixing broken pipelines	20 (E) hours	2 (A) hours Alerting for test failure and unified experience for troubleshooting reduces time to fix prior to production
	Data downtime	60% availability	99.9% availability
Quality Key Result	**Support tickets for data validation**	**10/week**	**1/week**
Governance Data lineage and dependencies are clear,well documented, and web-hosted for easy reference	# hours spent documenting	5 (E) hours	0 Documentation generated automatically, and web-hosted for easy consumption
	# hours spent investigating data dependencies	20 (E) hours	0 DAG generated as you code
	# hours spend validating metric accuracy	5 (E) hours	1 (A) hours Analysts enabled to self-serve requests
Governance Key Result	**Support tickets for documentation**	**2/week**	**0/week**
	Total hours expended	**160 (E) hours**	**25 (E) hours; 55 (A) hours**

Company uplift potential

Expanding the set of individuals capable of contributing to, and self-serving information about transformation workflows, while automating data quality controls frees data teams to focus on higher leverage work like platform expansion, pipeline optimization, or real-time processing – activities that could significantly accelerate critical business decisions. The following table includes additional examples of company uplift translated from the calculations above, organized by risk reduction, growth/revenue impact, and operational efficiency.

Template: dbt Cloud impact mapped to organizational goals

Organizational Uplift	dbt Cloud Impact
Risk Reduction	**100%** of data models now tested prior to production (0/5 → 50/50) **40** percentage point reduction in data downtime (dbt Cloud has a 99.9% uptime guarantee)
Growth/Revenue	**15%** increase in speed to market (Engineering hours reallocated to initiatives that accelerate market analysis) **20%** increase in upsell (account dashboards are now more reliable)
Operational Efficiency	**87%** productivity increase for engineers (160→25: Reduction in hours Eng spends on data development, pipeline resolution, and responding to ad-hoc requests) **300%** increase in data marts developed (1→4: More developers contributing to development workflow)

Project cost comparison

Your POV results can be used to project annual costs. Use the average salary for analysts and engineers (or anyone contributing to data development) to calculate cost per hours expended. The below is based on an organization with a data team of ~5. Also check out the interactive Total Cost of Ownership Calculator.

POV Inputs

	Legacy In-House	dbt Cloud
# data marts developed	1	4
Labor expended	160 (E) hours	25 (E) hours; 55 (A) hours
Labor cost*	E: 160 * $78/hr = $12,480	E: 25 * $78/hr = $1,925 A: 55 * $47/hr = $2,585 = $4,510
Average compute hours per week	100 hours	50 hours
Average compute cost**	100 * $16/hr * 4 weeks = $6,400	50 * $16/hr * 4 weeks = $3,200

* Assuming analyst salary = $90k USD ($47/hr); Average data engineer salary = $150k USD ($78/hr)
** Assuming average compute cost of $16/hr

Total Annual Cost

	Legacy In-House	dbt Cloud
# data marts developed annually	12	48
Annual labor cost	$12,480 * 12 = $150k	$4,510 * 12 = $54,120
Annual transformation infra costs	$15k/year	-
Annual license costs	-	~$24k/year
Annual warehouse infra costs	($6,400 * 12) = $76,800	($3,200 * 12) = $38,400
Total annual cost (infrastructure/license + transformation labor hours + warehouse infra)	$241,560	$116,820

Total Annual Cost per Outcome

# data marts developed annually	12	48
Total annual cost (infrastructure/license + transformation labor hours + warehouse infra)	$241,560	$116,820
Cost (including salaried hours) per data mart	$241,560 / 12 = $20,130	$116,820 / 12 = $2,434
Cost (excluding salaried hours) per data mart	(($15k + $76,800) / 12) = $1,890	(($24k + $38,400) / 48) = $1,300

► Step 6: Present findings

Once you've finished your analysis, you should be prepared to present summary findings. The tables above can be used to derive the following key metrics Cloud customers have found to be the most impactful:

Key metrics dbt Cloud customers used to build the business case

% increase in data uptime

*__JetBlue__ reduced maintenance windows from **8-0 hrs***

% increase in data production velocity

*__Firefly Health__ increased production speed by **20x***

% increase in company data trust scores

Envoy** increased their data team NPS by **21%

% increase in engineering productivity

Nasdaq** increased productivity by **30%

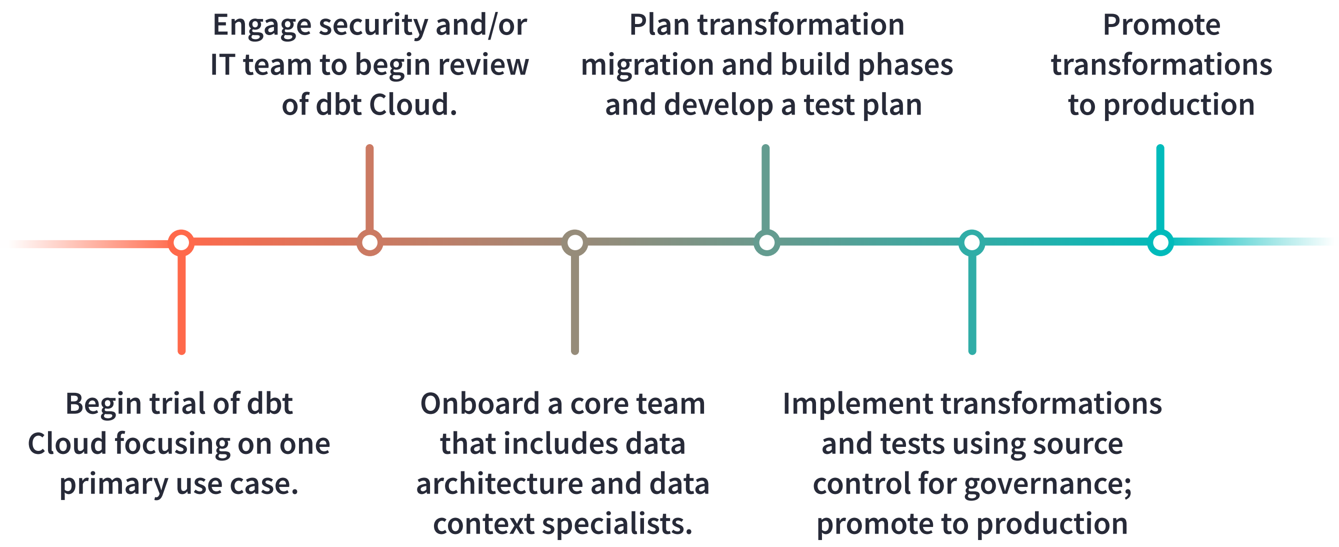
% reduction in change request tickets

Prosper** reduced data change requests by **70%

Step 7: Create the Implementation plan

If your project plan includes migrating off an existing solution, adding an implementation timeline can eliminate last minute questions about resource availability. Consider a phased approach to transformation migration that includes a gradually broadening scope of dbt functionality.

Suggested implementation timeline:



Begin trial of dbt Cloud focusing on one primary use case.

Engage security and/or IT team to begin review of dbt Cloud.

Onboard a core team that includes data architecture and data context specialists.

Plan transformation migration and build phases and develop a test plan

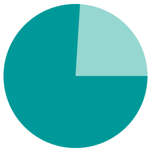
Implement transformations and tests using source control for governance; promote to production

Promote transformations to production

Step 8: Prepare to ramp

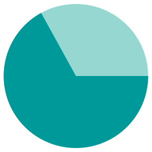
You'll notice we allotted a relatively small window of time for user training in the above example. While training on dbt might sound daunting (57% of survey respondents faced doubts about the ability of non-engineers to own development work), this may be an unfounded fear.

The dbt Cloud IDE, dbt Community, and dbt docs have proven to significantly reduce ramp time for anyone less familiar with the command line or git best practices. 100% of dbt users surveyed estimated that non-engineers needed less than 6 weeks to ramp, with 50% claiming 1-3 weeks was enough. Additionally, the presence of a "power user" or mentor was cited as the #1 most effective aid in further accelerating ramp.



76%

Percentage of data practitioners that estimate non-engineers can ramp on dbt in less than 3 weeks.



67%

say a “power user” mentor is the #1 indicator of success for new users

Summary

As illustrated above, dbt Cloud can dramatically improve data team efficiency through workflow automation and the democratization of development. By infusing software engineering best practices like testing, version control, and documentation into the analytic workflow, and enabling more members of the data team to participate in all areas of development, each member of the data team can refocus on pursuing higher leverage work that elevates the business as a whole.

Top outcomes reported after implementing dbt Cloud:
1. Increased pipeline transparency
2. Increased participation in data development
3. Reduced time spent troubleshooting

Why dbt Cloud?

The solution we had built in-house was great, but as more layers of code were added, we started seeing discrepancies between data sets. When that happens, you lose all trust. Unwinding unnecessary overhead from maintaining a system that just couldn't scale, and rebuilding our pipeline with dbt Cloud at the center, changed the game for us.

Cory Dambert | Director, BI & Analytics at Prosper Marketplace

Top Feature:

Analysts from different groups were siloed from one another–relying on stored procedures that lacked any indication of ownership or dependencies. dbt Cloud provided a centralized development environment where even analysts less familiar with SQL could quickly reference transformation logic and lineage and skill up at their own pace.

Biggest benefit:

The biggest benefit I receive from dbt is analyst self-service. It's normal for data engineers to handle ad hoc requests for data parity, but on the whole that work isn't moving the organization forward. If analysts can be empowered to write code the right way, using tools to maintain a proper process flow, my team gets time back to work on optimization and investigate new solutions. That's huge.

Why dbt Cloud?

We wanted the data engineers to be doing more in-depth analysis and less one-off requests. To do that, we had to enable analysts to own their own models. While we had at least one person in each business unit that knew SQL relatively well, they would still need help with best practices. That's why we chose dbt instead of a traditional drag and drop BI solution.

Top Feature:

The IDE. If I can centralize where people develop so that they don't have to think about any of the setup–they just log in and have immediate access to all of their work–that's a much easier sell than sending someone an installation guide and wishing them luck.

Biggest benefit:

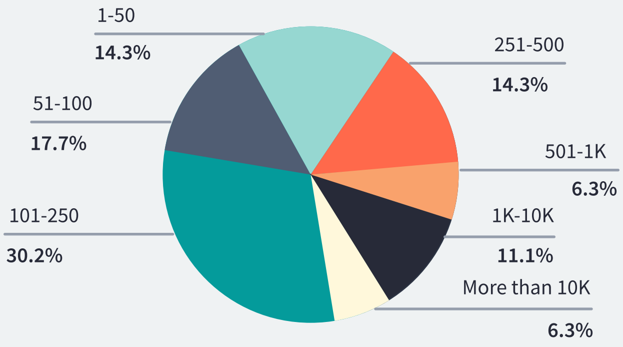
Documentation will be our biggest accelerator. If the economic research team doesn't require a lot of back and forth to understand what each table means, they'll be 30 or 40% more productive right off the bat.

Michael Weiss | Senior Product Manager at Nasdaq

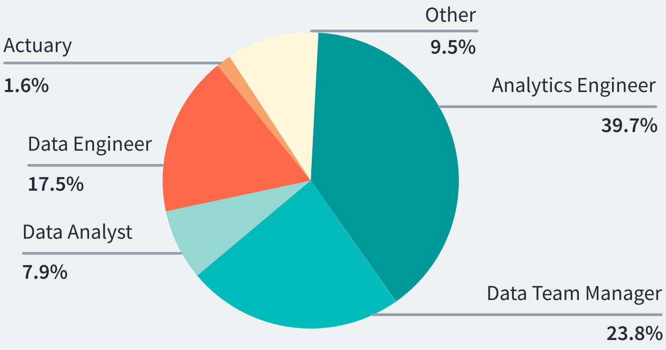
► Appendix:

The data contained in the following visualizations was collected from an informal survey of 80 dbt Community Slack members. While this sample is not large enough for statistical inference, it is consistent with our experiences working with thousands of data teams. We invite you to visit the #toward-analytics-engineering channel in the dbt Slack Community to exchange experiences and advice with the data professionals involved with the survey and this guide.

What is the size of your organization?

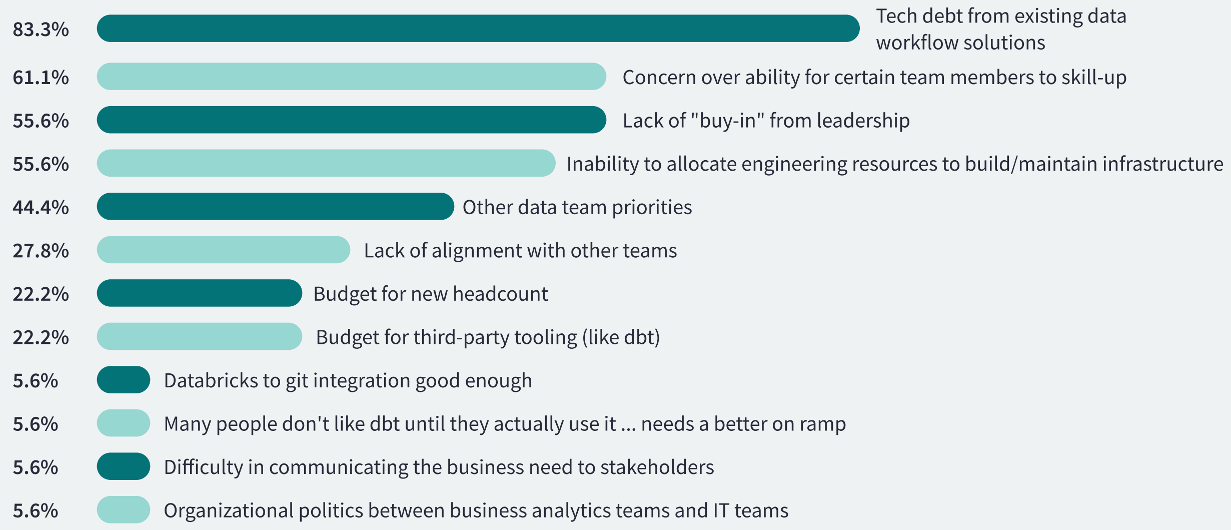


Which best describes your role?

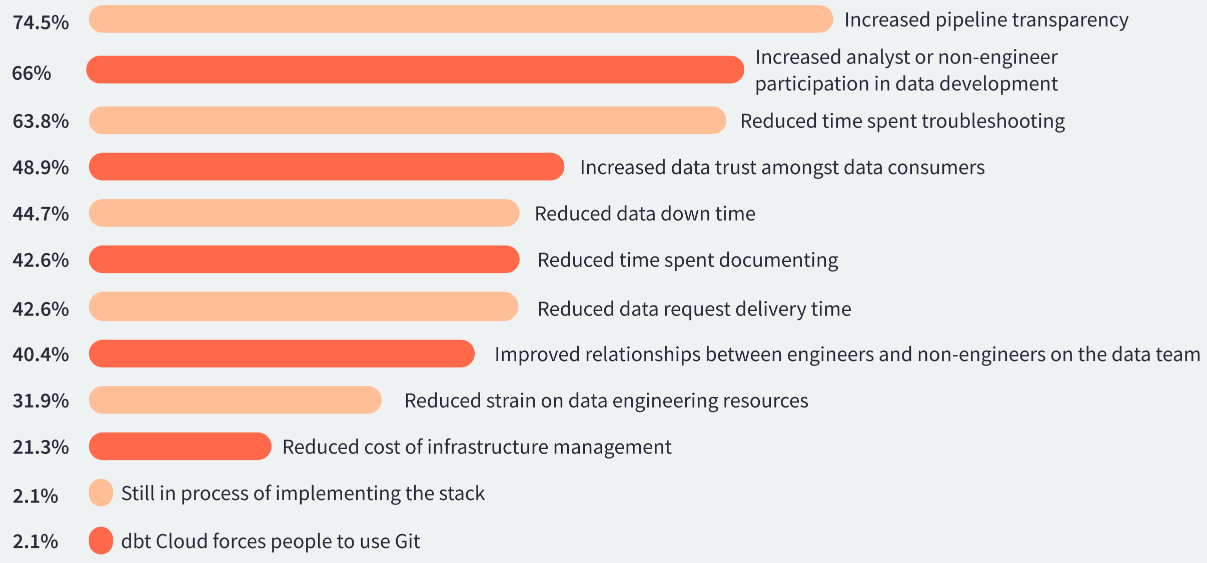


Appendix:

What do you believe contributed to this resistance?

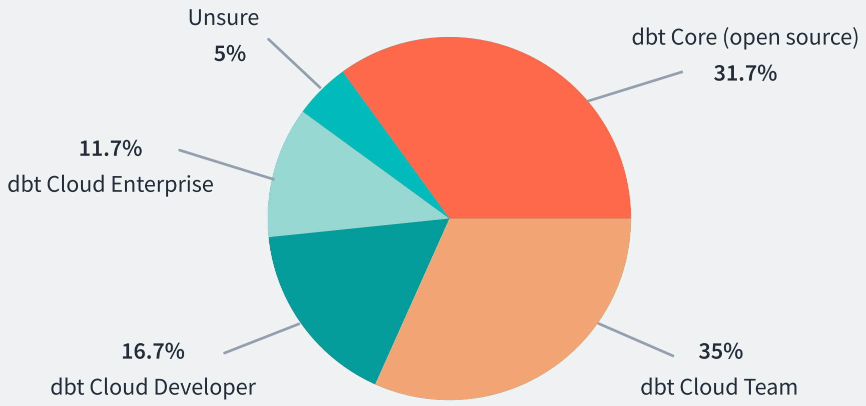


How has the analytics engineering workflow positively impacted your organization?

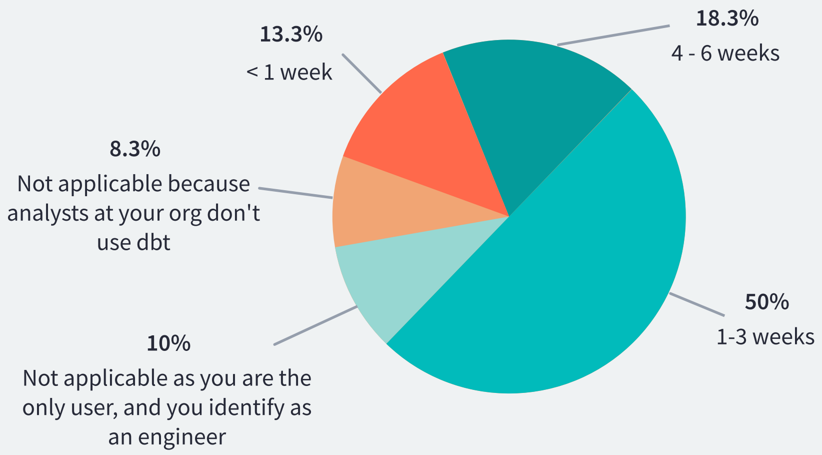


▶ Appendix:

What version of dbt does your organization use?

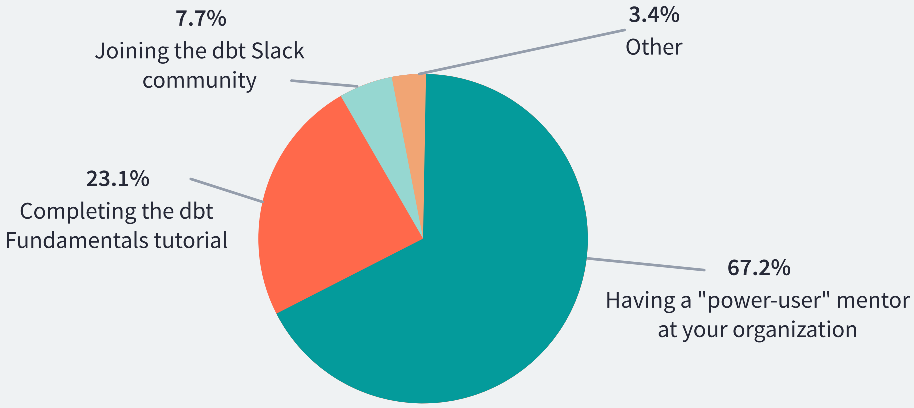


How long would you estimate it takes the average data analyst or non-engineer on your team to start developing in dbt?



► Appendix:

What do you believe has the BIGGEST influence on getting an analyst or non-engineer from “0-1” in dbt?


Joining the dbt Slack community
Other
Completing the dbt Fundamentals tutorial
Having a "power-user" mentor at your organization