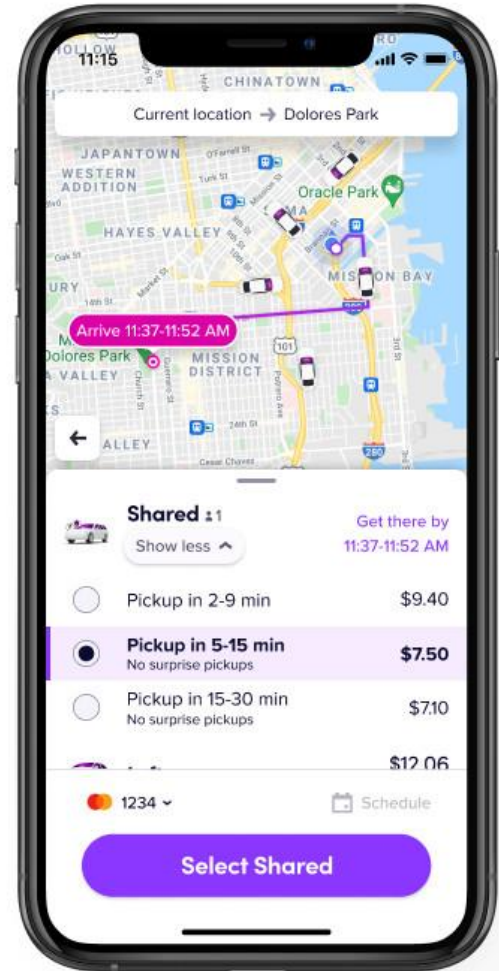


Feed the alligators with the lights on: How Data Engineers can see who really uses data

Mark Grover
CEO, Co-founder, stemma.ai

stemma

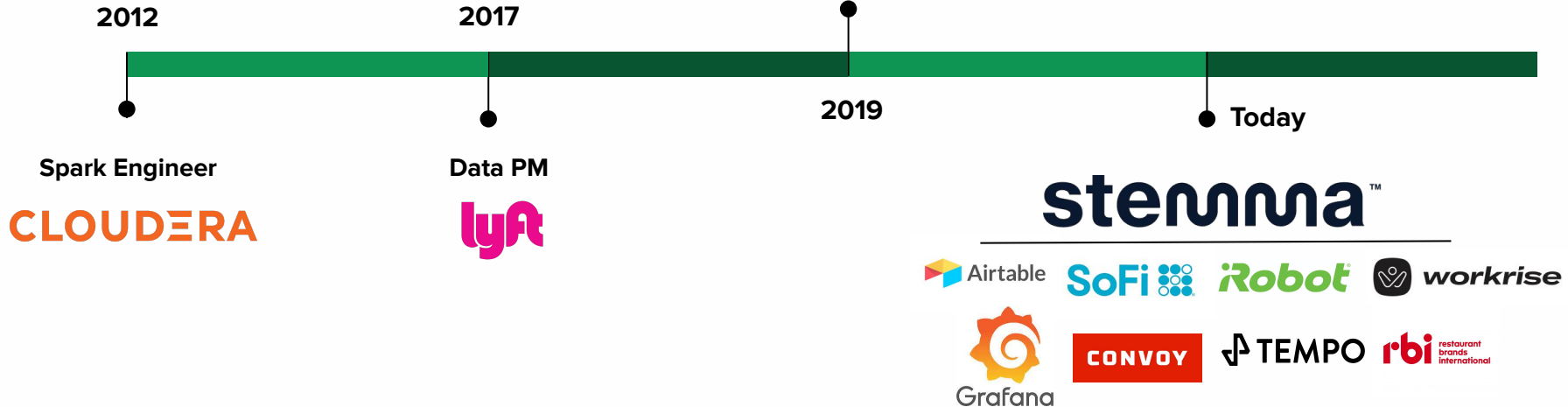


About Me

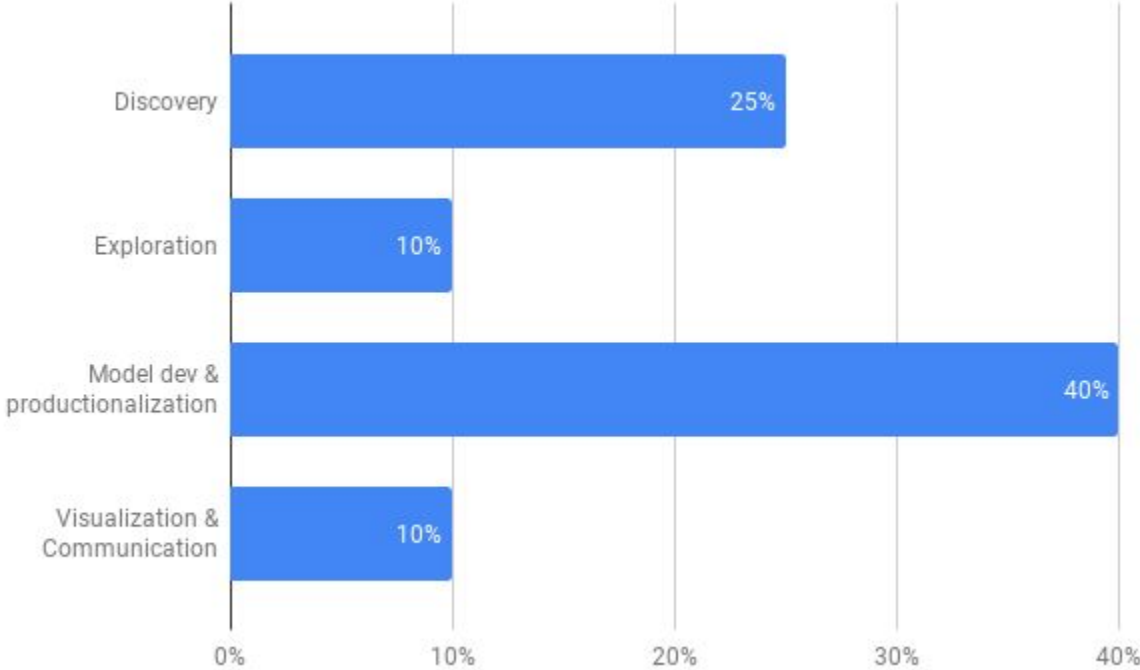


Mark Grover
Co-Founder, CEO,
Stemma

Previously, Co-creator of
Amundsen @Lyft



The discovery problem: Lots of wasted tech & biz users time

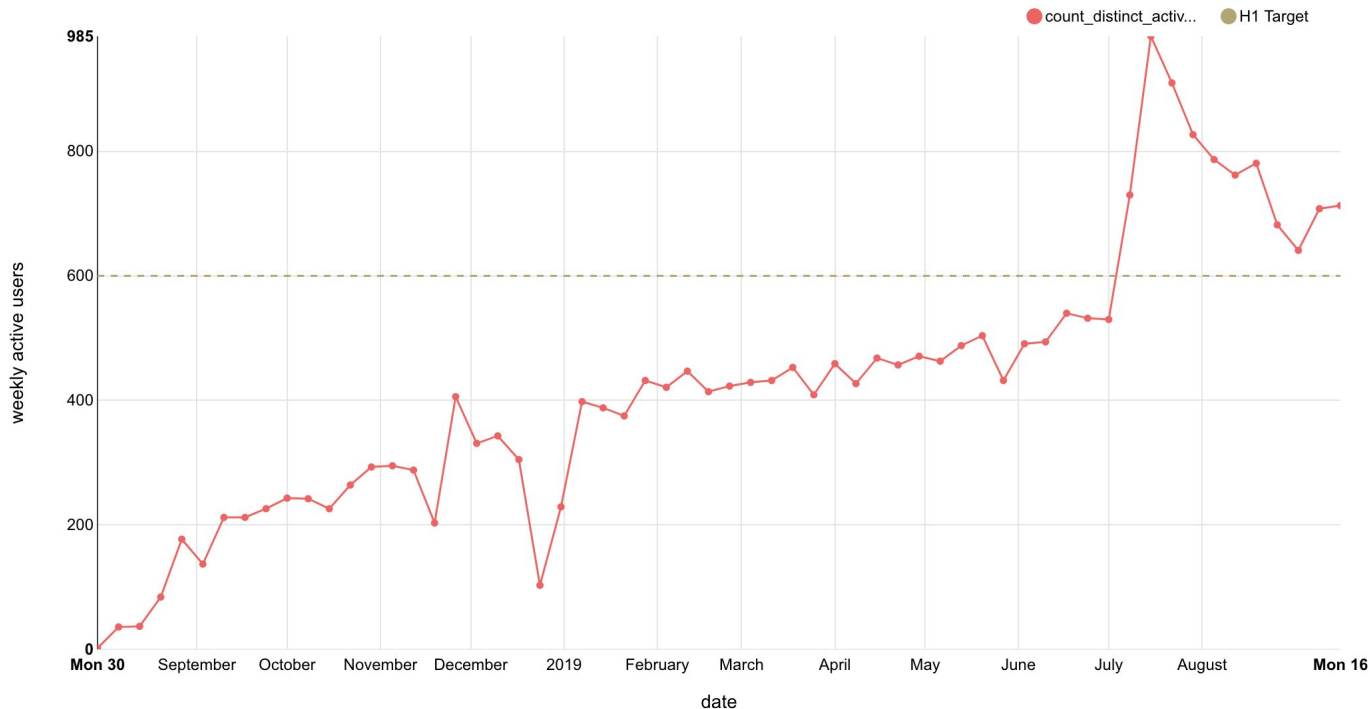


Analyst/DS workflow and time spent on each step

Amundsen @ Lyft: 750+ WAUs, 150k+ tables, 4k+ employee pages

Weekly active users ☆ Altered s

60 rows 00:00:07.67 🔗 </> .json .csv ☰



“This is God’s work” - George X, ex-head of Analytics, Lyft

“I was on call and I’m confident 50% of the questions could have been answered by a simple search in Amundsen” - Bomee P, DS, Lyft

Amundsen Open Source Community



Active community



BAM: Discovery is solved! No more problems!

- Automating the catalog shows the mess but does not clean up the mess (we still need DEs for that)
- We keep adding hay to the haystack
- Smooth changes require knowledge of how data is used

But this gets worse ...

- The more you self-serve data, the less you know about how it is being used
- Breaking changes = angry users and roll-backs

Two perspectives on data changes



Example 1: SaaS biz software company

- Maintained a sandbox schema so analytics could field ad hoc questions
- A table, intended for one-off use, became popular with other analysts

But...

- Data team was unaware of the table and its popularity, so they did not groom the data
- Department heads came to meeting with different figures for the same measurement due to using the sandbox data

So now...

- Data is blamed for untrustworthy data

Example 2: Product-led dev tool company

- Set out to reduce ETL for terabytes of data that was not being used
- Analytics team rigorously vetted and approved the reduction

But...

- A service account created by the Growth team was running a reverse ETL job to support the PLG funnel
- The deletion caused the marketing campaign to mistarget users

So now...

- Data gets unscheduled priority work from the Growth team

Researching how teams make changes to data

Interviewed dozens of Data Engineers across software, hardware, service companies

Questions:

- **When:** How often do you make changes to data? What triggers them?
- **How:** What are the commonly followed processes?
- **Pain:** How well does it work?

What we learned

Best use of time

data modeling, solving bugs, scoping long-term projects



Worst use of time

rolling back and cleaning up breaking changes



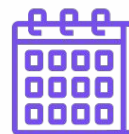
Difficulty of data changes

3.0 *on average*
out of 5

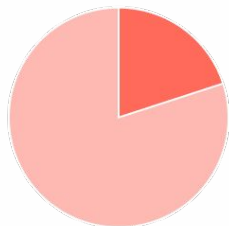
Difficulty of user migration

4.5 *on average*
out of 5

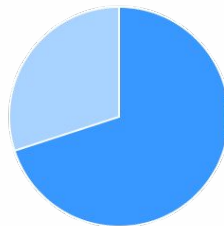
Frequency of changes



approx. once
per month



20% had **no form of impact analysis** ahead of changes



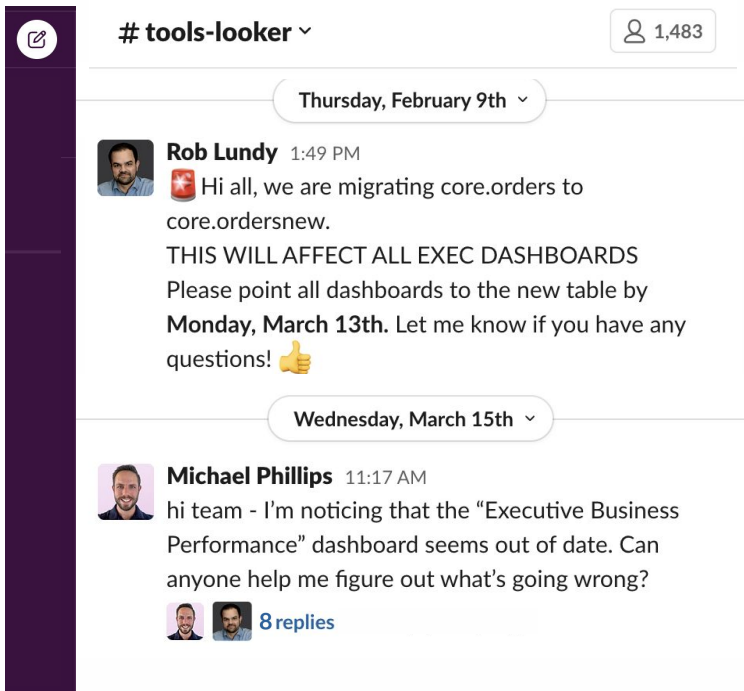
70% **rely on users** to see a broad message and make changes

Takeaway: Slack blasts don't work

70% rely on broad blasts but...

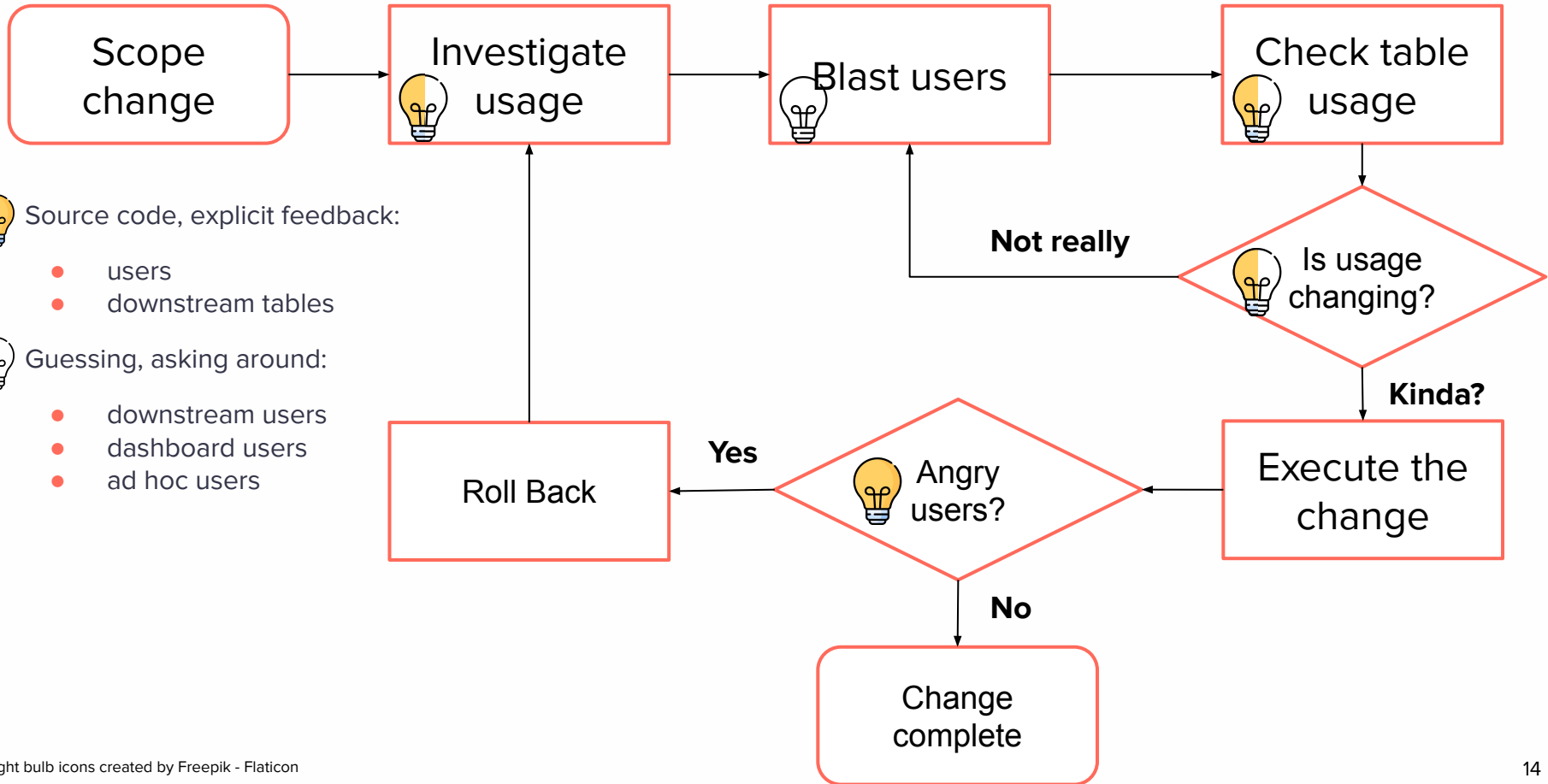
- Poor knowledge of users
= vague description of impact
- Difficult to know specific date of change
= vague timing in early messages
- Broadcasts reach unaffected users
= become noise over time

“We had a major change that we broadcast every week for six months. We still had analysts caught off guard when we cut over.”



The screenshot shows a Slack channel interface for #tools-looker with 1,483 members. A date separator indicates Thursday, February 9th. A message from Rob Lundy at 1:49 PM reads: "Hi all, we are migrating core.orders to core.ordersnew. THIS WILL AFFECT ALL EXEC DASHBOARDS Please point all dashboards to the new table by Monday, March 13th. Let me know if you have any questions! 👍". A second date separator indicates Wednesday, March 15th. A message from Michael Phillips at 11:17 AM reads: "hi team - I'm noticing that the 'Executive Business Performance' dashboard seems out of date. Can anyone help me figure out what's going wrong?". Below this message, there are two profile pictures and the text "8 replies".

Many “dark” phases in the current process



But we can do better! Three Types of Usage

ETL usage

Disruptions break downstream tables

- dbt
- Orchestration systems (Airflow, Dagster, Prefect, etc.)

Analytics usage

Disruptions break a decision

- User accounts from BI tools
- Ad hoc queries - can come from data warehouse, BI tool, or platform tool

Machine usage

Disruptions break a process, or application

- ML models, Reverse ETL

It's possible to see collect and analyze this usage

Data Warehouses

Query DWH Query Logs to understand data use

BI tools

Use APIs to understand how data is being used

ETL tools / dbt

Catalog.json / Manifest.json contain table-level lineage information

Snowflake: Who ran what queries over the last 7 days

```
SELECT qh.end_time,
       qh.query_type,
       qh.user_name,
       qh.query_text
FROM   snowflake.account_usage.access_history ah
JOIN   snowflake.account_usage.query_history qh
ON     ah.query_id = qh.query_id ,
       lateral flatten(base_objects_accessed) f1
WHERE  ah.query_start_time >= dateadd('day', -7, CURRENT_TIMESTAMP())
AND    qh.end_time >= dateadd('day', -7, CURRENT_TIMESTAMP())
AND    f1.VALUE:"objectDomain"::string='Table'
AND    objects_modified IS NULL
OR     objects_modified = []
AND    replace(f1.VALUE:"objectName"::string, '') = concat('<CATALOG NAME>', '.', '<SCHEMA NAME>', '.', '<TABLE
NAME>');
```

Finding Table users through Query Logs

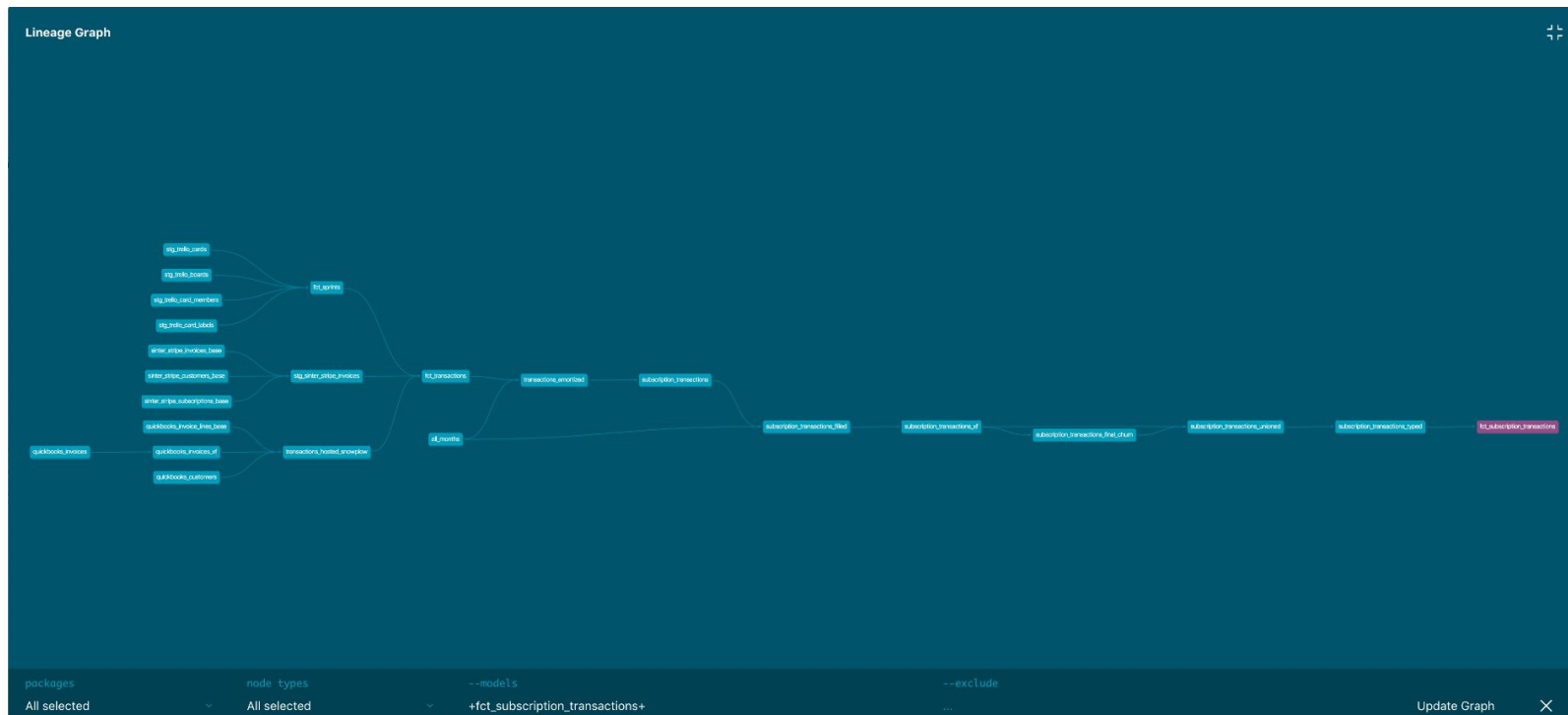
The tricky part

- Only works for select warehouses (ex. Snowflake, BigQuery)
- Need to know how Snowflake querytables are populated to get useful information

Pro Tip: Use a range of 90 days because some data is only used in quarterly reporting

Getting table lineage from dbt

dbt docs generate; dbt docs serve



Finding downstream tables

Tricky part

- dbt docs aren't very performant or user friendly
- No column level lineage

Pro Tip: Focus on finding downstream table owners for biggest impact

Share responsibility!

Getting usage from BI tool (Tableau)

```
# The tables upstream to this Workbook
#
# Arguments
# filter: Filter by GraphQL field and given value
# orderBy: Sort by given fields. The sort orders defined first in
# the list will take priority. If there are no given sort orders or a tie on the
# final sorted field then the resulting set will be sorted by ID in ascending
# order.
# permissionMode: Results filtering mode.
upstreamTables(
  filter: DatabaseTable_Filter,
  orderBy: DatabaseTableSortOrder,
  permissionMode: PermissionMode
): [DatabaseTable]!
```

Finding Dashboard users

Tricky part

- API cumbersome to use
- Pre-parsed list of tables only available for Tableau.
- If you don't use Tableau, you need to do your own query parsing

Process if Software Engineers made Data changes

(Lots of manual work)

- 1) Understand exact users impacted by a data change
 - Find usage in data warehouse, BI tools, dbt
- 2) Where possible, make downstream changes and submit for review by table & dashboard owners
- 3) For other downstream changes:
 - DM owners and important users
 - Create jira for each table and dashboard owner
 - Explicit instructions for change
 - Include “change by” date
- 4) Follow up & validate downstream owners have approved changes
- 5) Execute the change

Data Ecosystem is evolving to address impact analysis

Data Catalogs like Stemma automate impact analysis and messaging

- track usage across dbt, BI tools, ad hoc queries
- Usage models to stage migration in order of impact
- Detect owners and users via lineage, message via Slack and email

BI tools are adding dashboard modification via API, opening way to fully automate migration for the end user

- Mode, Amazon Quicksight, Hex

Use a modern catalog to

- Table details
- Column descriptions, including Autodescribed
- Other catalog context like glossary terms

The screenshot displays a modern data catalog interface. At the top, there is a navigation bar with 'Discover', 'Glossary', and 'Admin' tabs, a search bar, and an 'Advanced Search' button. The breadcrumb path is 'Snowflake / food_delivery / core / orders'. The interface includes buttons for 'Bookmarked', 'Message', and 'Lineage Graph'. Below the breadcrumb, there are tabs for 'About', 'Lineage', 'Usage', 'History', and 'Additional Fields'. The 'About' tab is active, showing '8 Upstream | 8 Downstream', '63,988 Queries | 7 Users', and '2 Fields'. The main content area is titled 'Contents' and features an 'Airflow' button. A table provides summary information: 'Last updated: Mar 24, 2023 5am CDT', 'Date range: N/A', 'Row count: 33,688,906', and 'Table size: 12.53 GB'. Below this is a search bar and a table of columns with headers 'Column', 'Related Terms', 'Type', and 'Actions'. The columns listed are 'id' (primary_key, text), 'user_id' (number), and 'driver_id +:' (foreign_key, number). There are also sections for 'Column Description', 'Related Terms', and 'Autodescribed via upstream lineage'. The bottom of the screen shows 'address +:' (text) and 'star_rating' (number).

| Column | Related Terms | Type | Actions |
|--|-----------------|--------|---------|
| id <small>(primary_key)</small> ID for the order, applies to all orders. | Arr | text | |
| user_id | 2 Related terms | number | |
| Column Description Edit this column to add a description | | | |
| Related Terms Arpu / Arpa (Revenue Metrics), Average Deal Size (Revenue Metrics) | | | |
| driver_id +: <small>(foreign_key)</small> | | number | |
| Autodescribed via upstream lineage (core.driver_docs.driver_id) ID of the delivery driver | | | |
| Related Terms Link this column to a glossary term to have it appear here | | | |
| address +: Full address where the order is delivered. Format includes postal code at the end. | 2 Related terms | text | |
| star_rating Rating of 1-5 stars. Used to promote ordering from popular restaurants. | | number | |

Measure and rank activity by user

- User details
- Includes service accounts
- Number of queries, date of last query
- Usage context from Slack

The screenshot displays a Snowflake user activity dashboard. The top navigation bar includes 'Discover', 'Glossary', and 'Admin'. A search bar is present with the text 'Search for users, tables and dashboards...'. The breadcrumb trail shows the path: Snowflake / food_delivery / core / orders. The 'Usage' tab is selected, showing '63,988 Queries | 7 Users'. Below this, a 'Frequent Users' section features a search bar and a table with the following data:

| User | Num. of queries | Query Source | Most recent query |
|----------------------------------|-----------------|--------------|-----------------------------------|
| Mark Grover | 15,600 | Snowflake | a day ago Mar 23, 2023 5am CDT |
| Michael Phillips | 14,040 | Snowflake | a day ago Mar 23, 2023 4am CDT |
| Dorian Johnson | 12,480 | Snowflake | a day ago Mar 23, 2023 4am CDT |
| janice@stemma.ai | 10,906 | Snowflake | a day ago Mar 23, 2023 4am CDT |
| Rob Lundy | 9,240 | Snowflake | a day ago Mar 23, 2023 4am CDT |
| hightouch_srvc_snowflake | 1,500 | Snowflake | a day ago Mar 23, 2023 4am CDT |

Below the table, it indicates 'Showing usage for last 90 days' and 'Rows per page 10' with '1-6 of 6' items. The 'Slack Conversations' section shows two messages:

- #general** • Feb 07, 2023 5pm CST
Mark Grover Hey, I am a new analyst here. Something looks off with exec dashboard - it looks out of date, can someone help me? [View In Slack](#)
- #data-catalogue** • Feb 07, 2023 2pm CST
Michael Phillips

List all dependencies down to the column level

- Filter lineage by column
- See distant relatives by column
- Table and column popularity

The screenshot displays the Snowflake interface for a lineage graph. At the top, there's a navigation bar with 'Discover', 'Glossary', and 'Admin' tabs, and a search bar. Below that, the breadcrumb path is '/ Snowflake / food_delivery / core / orders'. The main content area is titled 'dbt Compiled Query' and shows the following SQL code:

```
1 SELECT *
2 FROM core.restaurants r
3 JOIN core.order_items i ON (r.id = i.restaurant_id)
4 WHERE type != 'SIMULATED';
```

Below the query, there are two sections: 'Upstream' and 'Downstream'. Both sections have a search bar and a 'Filter By Column' dropdown set to 'driver_id'. The 'Upstream' section shows a table with the following data:

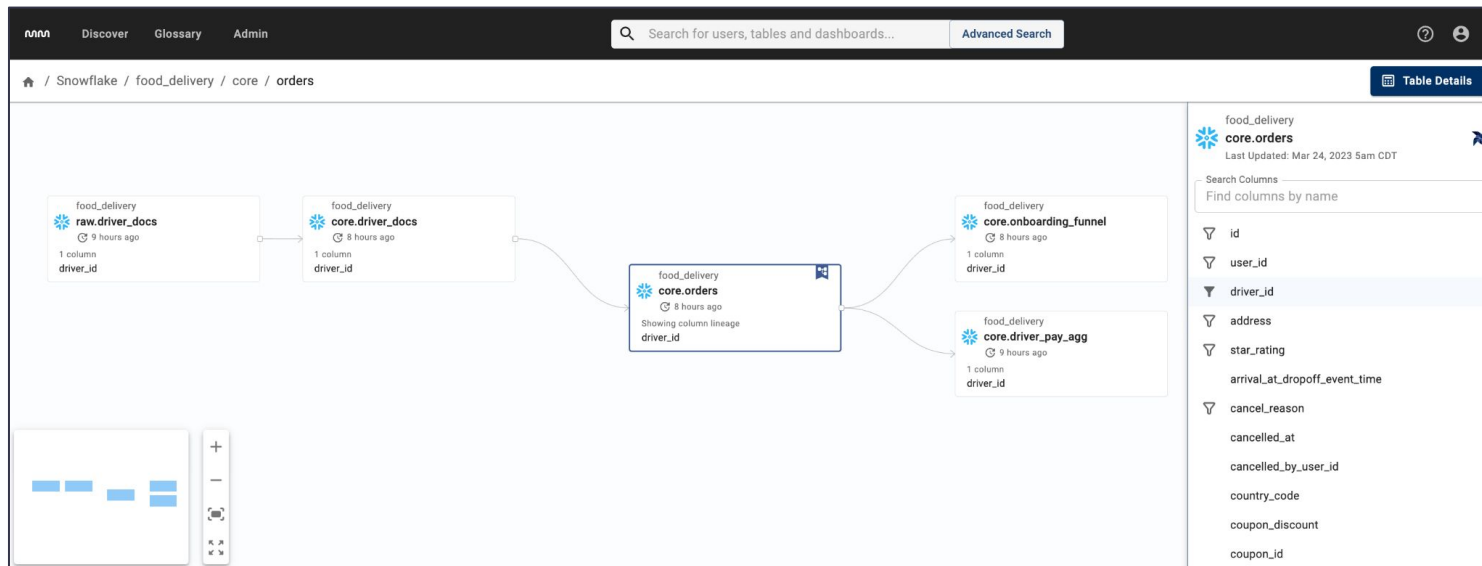
| Upstream Asset | Distance | Upstream Columns | Popularity* | Status | Last Updated | Owner |
|-----------------------------------|----------|------------------|-------------------------|--------|----------------------|-------|
| FOOD_DELIVERY core.driver_docs | 1st | driver_id | 0 queries 0 users | | Mar 24, 2023 5am CDT | |
| FOOD_DELIVERY raw.driver_docs | 2nd | driver_id | 1,078 queries 1 user | | Mar 24, 2023 5am CDT | |

The 'Downstream' section shows a table with the following data:

| Downstream Asset | Distance | Downstream Columns | Popularity* | Status | Last Updated | Owner |
|--------------------------------------|----------|--------------------|-------------------------|--------|----------------------|-------|
| FOOD_DELIVERY core.driver_pay_agg | 1st | driver_id | 3,542 queries 1 user | | Mar 24, 2023 5am CDT | |
| FOOD_DELIVERY | 1st | driver_id | 3,388 queries | | Mar 24, 2023 5am CDT | |

*Popularity is calculated for tables over 90 days and dashboards for all time. Rows per page: 10. 1-2 of 2.

See dependencies with visual lineage



- Highlight relationships in complex topologies
- Filter by column
- See table and dashboard relationships

Quickly message users and owners

- Send Email and Slack from the table
- Target owners, users, downstream users, Slack groups

The screenshot shows a data catalog interface with a 'Send message to...' dialog box open. The dialog lists the following recipients:

- Owners (1 user and 1 team): Mark Grover, #data-catalogue
- Frequent users (4 users): Mark Grover, Michael Phillips, Dorian Johnson, Rob Lundy
- Owners of dashboards using this table (1 user): Mark Grover
- Owners of downstream tables (3 users and 1 team): Rob Lundy, Michael Phillips, Mark Grover, #data-catalogue

A text input field labeled 'Message Text *' contains the placeholder text 'Enter a message to send to the selected recipients'. Below the input field, a note states: 'Users will be notified via email. Teams will be notified via Slack.' At the bottom right of the dialog are 'Cancel' and 'Send Message' buttons.

The background interface shows a 'dbt Compiled Query' for the table 'orders'. The query is:

```
1 SELECT *
2 FROM core.orders
3 JOIN core.orders ON core.orders.id = core.orders.id
4 WHERE type = 'order'
```

The interface also displays 'Upstream' and 'Downstream' sections. The 'Downstream' section shows a table with columns: 'Downstream Asset', 'Distance', 'Downstream Columns', 'Popularity*', 'Status', and 'Last Updated'. The table contains one row for 'FOOD_DELIVERY core.driver_pay_agg' with 1st distance, 'driver_id' column, 3,542 queries popularity, and 1 user.

Thanks



Mark Grover
Co-Founder, CEO,
Stemma

mark@stemma.ai

www.stemma.ai

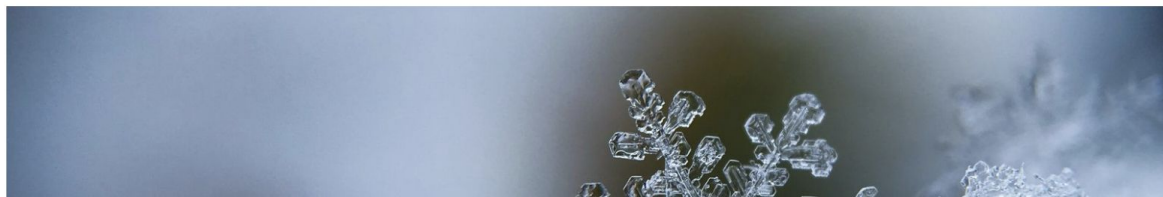
For more snowflake queries visit our blog:

How to fix your ETL to lower Snowflake Costs



by Mark Grover

CO-FOUNDER, CEO OF STEMMA



Thank You

stemma™