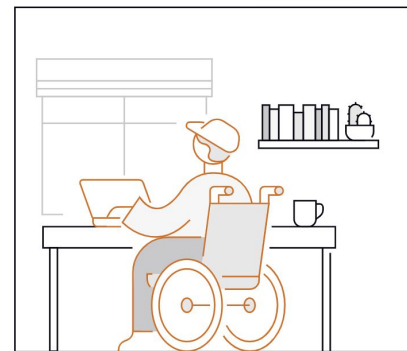
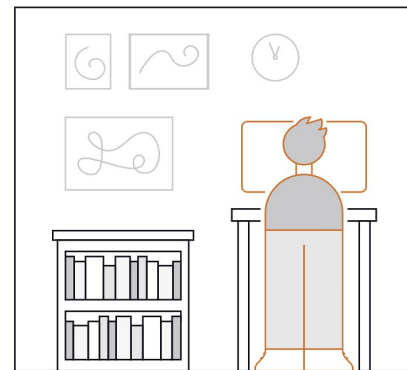
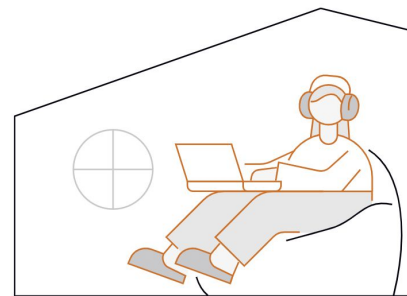


dbt at GitLab

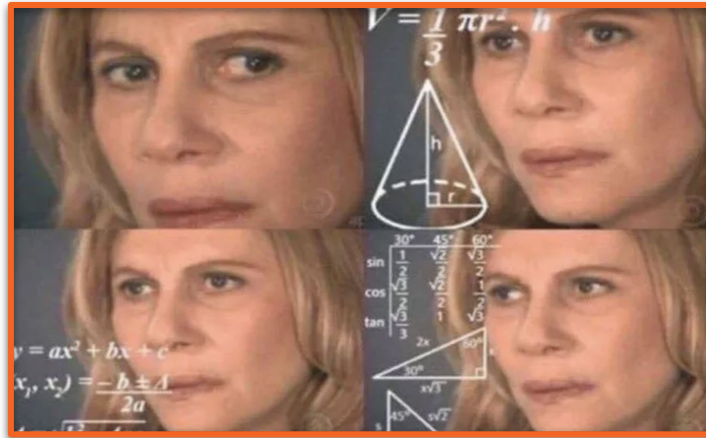
How we use one of the most popular Data tool

Radovan Bacovic
Staff Data Engineer @ GitLab





We are using our product which we are using to build our product to build our data product to analyze the data from our product.



Confusing?

??????????

We are using our product which we are using to build our product to build our data product to analyze the data from our product.

This looks like recursion

Still confusing



We are using our product which we are using to build our product to build our data product to analyze the data from our product.



We are using product **GitLab to build data product (**Enterprise Data Platform**) to analyze the data from **GitLab****



About **GitLab** (company)



What makes GitLab unique?

GitLab's Growth: Deep Transparency Makes a Difference

For this latest episode of The New Stack Makers podcast, GitLab CEO and co-founder Sid Sijbrandij, candidly discussed what the next steps are for GitLab.

Sep 24th, 2019 3:00pm by [Kiran Oliver](#)

Radical Transparency: A Look at GitLab's Company Culture



Business Strategy, Process Management

What makes GitLab unique?

(Probably) the most transparent company in the world

News

GITLAB: THE WORLD'S MOST TRANSPARENT COMPANY

By [Everett Cook](#) Last updated: Feb 15, 2023

[Data Platforms](#), [Case Studies](#) Updated December 8, 2022

How the GitLab Data Team Builds a Culture of Radical Transparency



What makes GitLab unique?

Total transparent operational model - so transparent that it is intimidating



What makes GitLab unique?

All remote from the beginning



What makes GitLab unique?

The biggest all-remote company in the world



What makes GitLab unique?

All-remote and async work advocate



What makes GitLab unique?

A global leader in distributed work

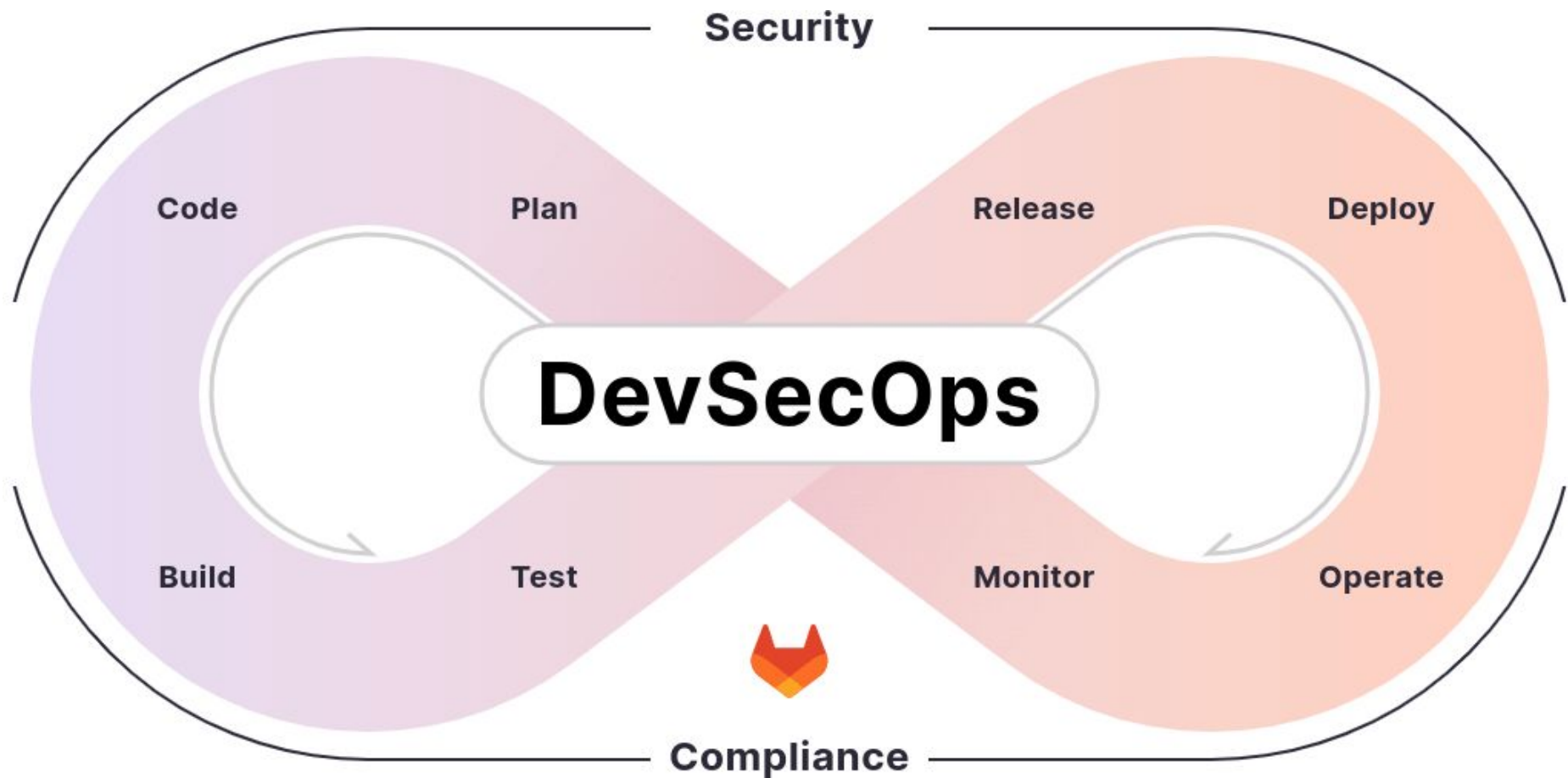


What GitLab is doing?



What GitLab is doing?

The **DevSecOps** Platform delivered as a single application to help you iterate faster and innovate together





Plan



Create



Verify



Package



Secure



Release



Configure Monitor



Govern



Manage

Team Planning	Source Code Management	Continuous Integration (CI)	Package Registry	SAST	Continuous Delivery	Auto DevOps	Metrics	Security Policies	Subgroups
Portfolio Management	Code Review	Code Testing and Coverage	Container Registry	Secret Detection	Pages	Kubernetes Management	Incident Management	Vulnerability Management	DevOps Reports
Service Desk	Wiki	Performance Testing	Helm Chart Registry	Code Quality	Advanced Deployments	Deployment Management	On-call Schedule Management	Dependency Management	Value Stream Management
Requirements Management	Web IDE	Merge Trains	Dependency Proxy	DAST	Feature Flags	ChatOps	Tracing	Audit Events	
Quality Management	Snippets	Review Apps	Git LFS	API Security	Release Evidence	Infrastructure as Code	Error Tracking	Compliance Management	
Design Management		Secrets Management		Fuzz Testing	Release Orchestration		Product Analytics		
				Dependency Scanning	Environment Management				
				Container Scanning					
				License Compliance					



Plan



Create



Verify



Package



Secure



Release



Configure



Monitor

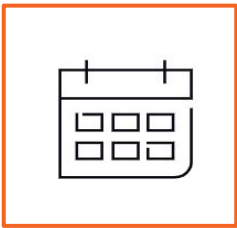


Govern



Manage

GitLab by num83r5



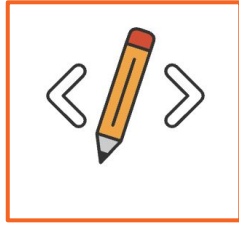
0

We have 0 offices



1600

1600+ team members in more than 65 countries and regions



30+ millions

30+ million registered users



22

Releases a new version of the product on the 22nd of every month.



2000

GitLab handbook has over 2000 web pages of text.



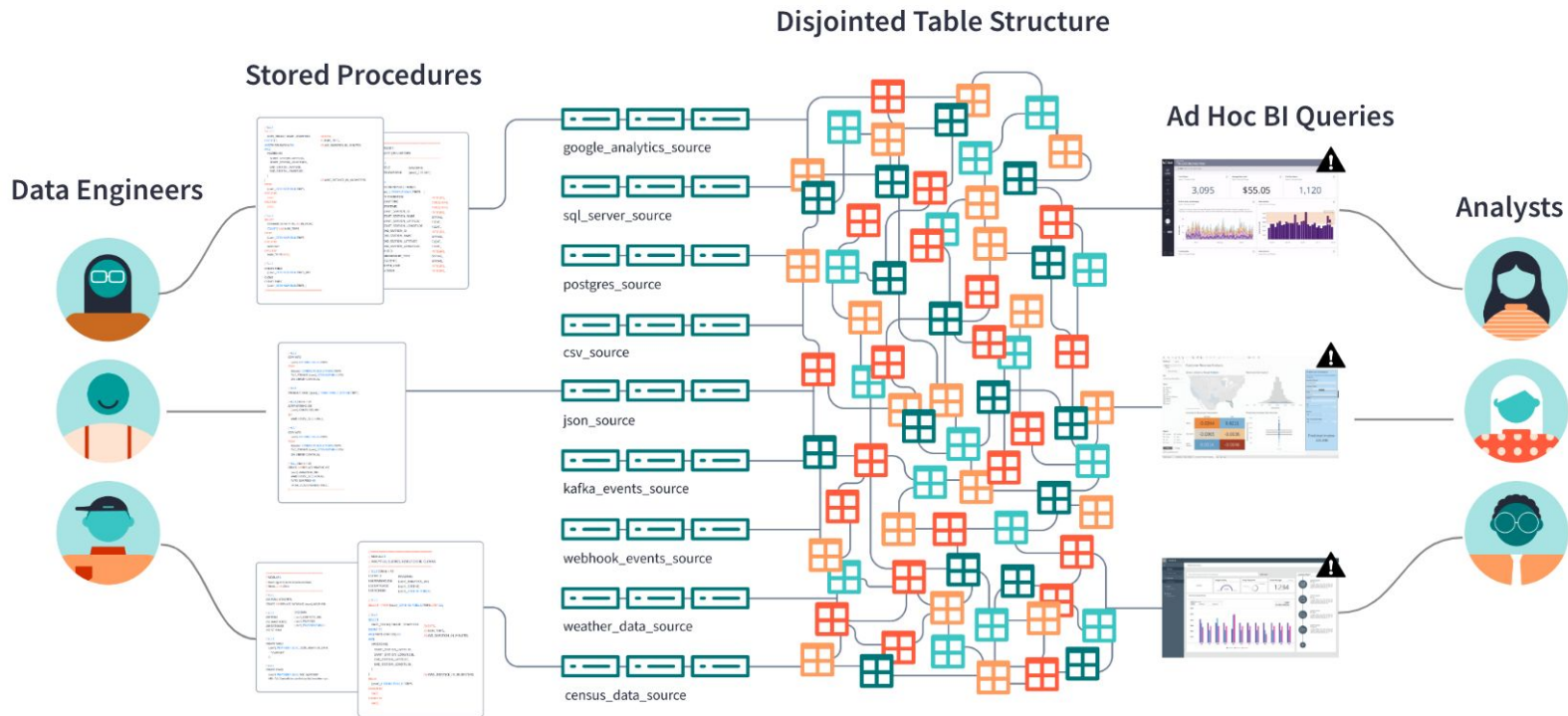
Let's jump in the **dbt machinery**



What is **dbt**?



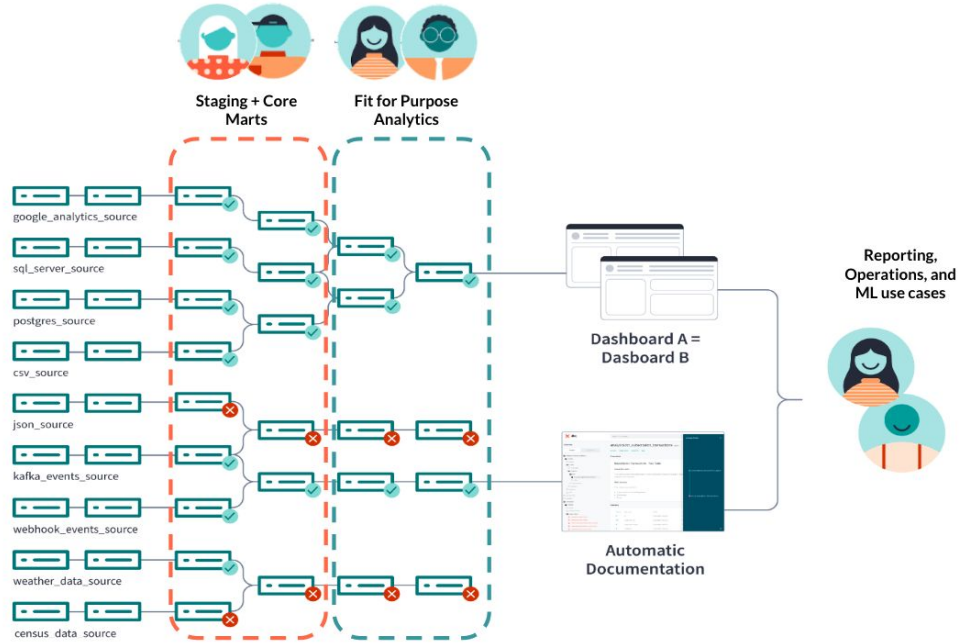
Engineers and analysts have different tools to answer the same question—deepening analytic debt and lengthening time to insight





dbt safely connects data development work so teams move faster, together.

Helping to build an “**expectation of knowing**” with timely data everyone can trust.



01

Reusable SQL, version control, and web-based IDE **speeds development.**

02

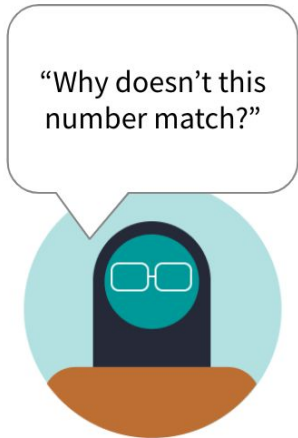
CI/CD, testing and automated dependency management **reduces data downtime.**

03

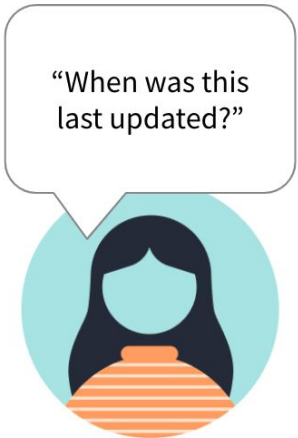
Business-accessible documentation builds trust and **ensures consistency.**



But while data increased, organizational knowledge hasn't.



Inconsistent metrics



Process opaqueness



Process bottlenecks



GitLab **dbt** use case

(mainly) open source tech stack



Google Cloud



Apache
Airflow



SNOWFLOW



Fivetran



Stitch



dbt



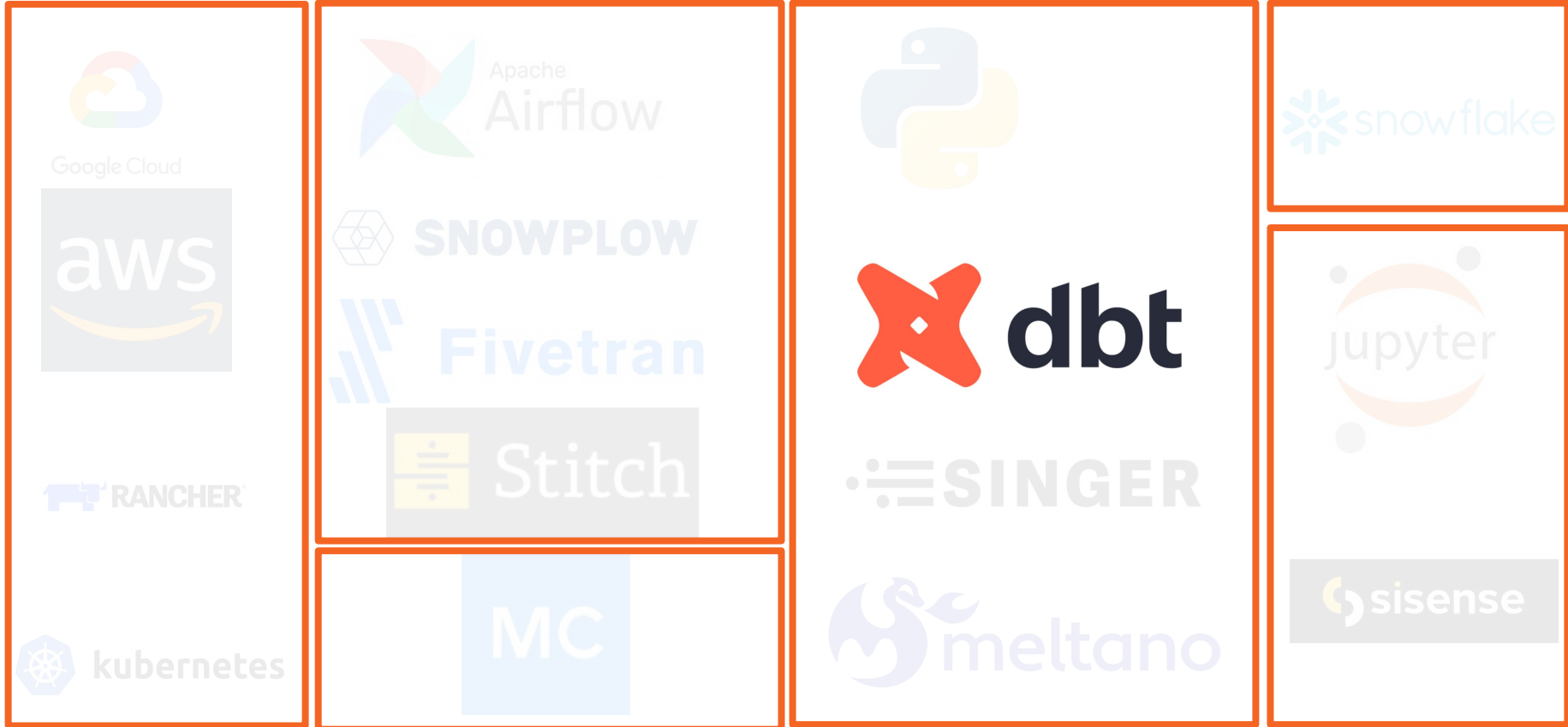
SINGER

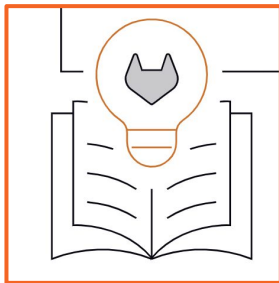


meltano



(mainly) open source tech stack





dbt



snowflake



Why dbt?



1. Open source

Why **dbt**?



1. Open source
2. Easy to learn

Why **dbt**?



1. Open source
2. Easy to learn
3. Strong community



Why **dbt**?



1. Open source
2. Easy to learn
3. Strong community
4. Single Source of truth for your transformation

Why **dbt**?



1. Open source
2. Easy to learn
3. Strong community
4. Single Source of truth for your transformation
5. Portable

Why **dbt**?



Why **dbt**?

1. Open source
2. Easy to learn
3. Strong community
4. Single Source of truth for your transformation
5. Portable
6. Build in Python + Jinja + SQL



Why **dbt**?

1. Open source
2. Easy to learn
3. Strong community
4. Single Source of truth for your transformation
5. Portable
6. Build in Python + Jinja + SQL
7. Usage: Jinja + SQL



Why **dbt**?

1. Open source
2. Easy to learn
3. Strong community
4. Single Source of truth for your transformation
5. Portable
6. Build in Python + Jinja + SQL
7. Usage: Jinja + SQL
8. A plenty of external libraries



Why **dbt**?

1. Open source
2. Easy to learn
3. Strong community
4. Single Source of truth for your transformation
5. Portable
6. Build in Python + Jinja + SQL
7. Usage: Jinja + SQL
8. A plenty of external libraries
9. Support for Python models and ML



Why **dbt**?

1. Open source
2. Easy to learn
3. Strong community
4. Single Source of truth for your transformation
5. Portable
6. Build in Python + Jinja + SQL
7. Usage: Jinja + SQL
8. A plenty of external libraries
9. Support for Python models and ML
10. Extensive



GitLab architecture for dbt

GitLab architecture for dbt



RAW

1. Unfiltered data
2. Raw format
3. Various formats (structured, semi-structured data)
4. Result of ingestion by various tools - depends on the use case

PREP

1. Clean data
2. Formatted data
3. Validated data
4. Result of transformation by **dbt**

PROD

1. Sensitive data
2. Hashed data
3. Separated per function
4. Result of transformation by **dbt**

GitLab architecture for dbt



RAW

Used as a source for the transformation

PREP

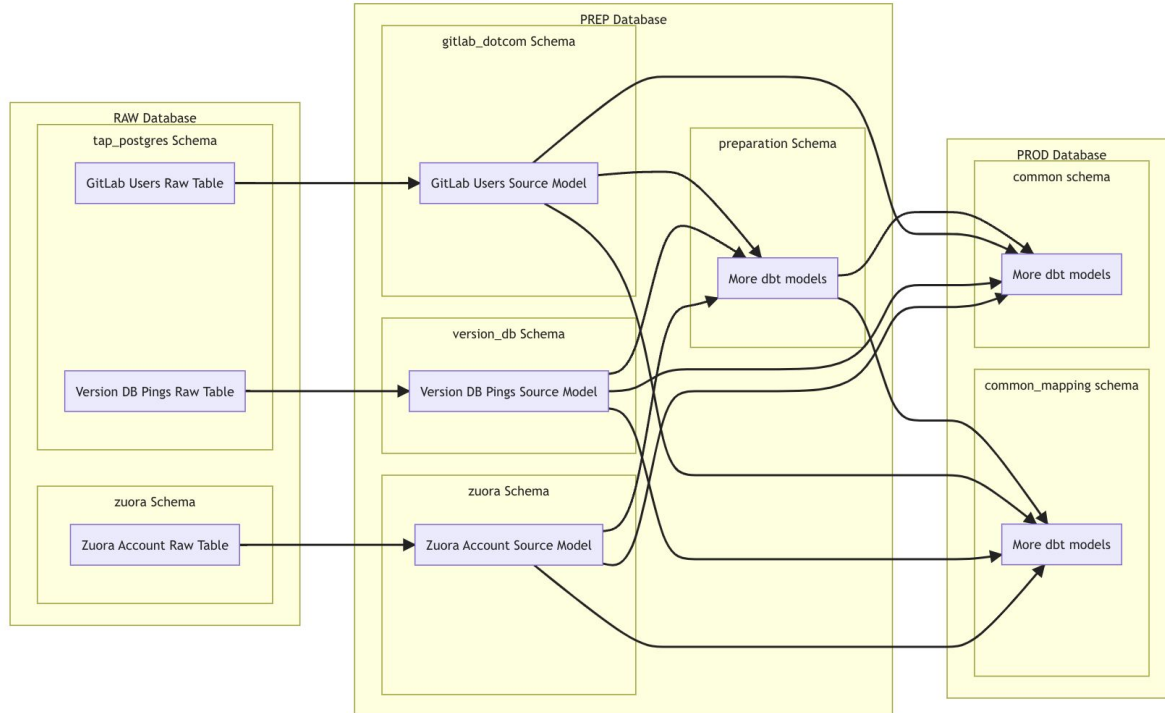
1. Cleaned up data
2. Hashed data
3. Filtered
4. Aggregated data
5. Legacy data

PROD

Data ready for:

1. Visualisation
2. Data Science
3. Further use
4. Modeled data using dimensional modeling

GitLab architecture for dbt





Best practices

Development setup



Best practices

Macros

```
WITH my_cte AS (...)  
  
{% dbt_audit(  
    cte_ref="my_cte",  
    created_by="@gitlab_user1",  
    updated_by="@gitlab_user2",  
    created_date="2019-02-12",  
    updated_date="2020-08-20"  
)}  
  
ORDER BY updated_at
```



Best practices

Tags

```
{{ config(  
    tags=["mmpi", "six_hourly"]  
) }}
```



Best practices

Selectors

```
- name: six_hourly_salesforce_opportunity
description: "Non-incremental Salesforce opportunity models"
definition:
  union:
    - intersection:
      - '@source:salesforce'
      - 'tag:six_hourly'
```



Best practices

Sources

```
SELECT *  
FROM "RAW".██████████.gitlab_db_events
```

```
SELECT *  
FROM {{ source('██████████', 'events') }}
```



Best practices

Sensitive data



Best practices

Sensitive data
- hashing

```
{% macro hash_sensitive_columns(source_table) %}  
  
  {% set meta_columns = get_meta_columns(source_table, "sensitive") %}  
  
  {%- for column in meta_columns %}  
  
    {%- if config.get("materialized") == "view" and config.get("secure") -%}  
  
      {{ hash_of_column_in_view(column) }}  
  
    {%- else -%}  
  
      {{ hash_of_column(column) }}  
  
    {% endif %}  
  
  {% endfor %}  
  
  {{ dbt_utils.star(from=ref(source_table), except=meta_columns) }}  
  
{% endmacro %}
```



Best practices

Sensitive data
- Dynamic masking

```
{%- macro create_masking_policy(database, schema, data_type, policy) -%}  
  {%- set mask = get_mask(data_type) -%}  
  {%- set body %}  
    CASE  
      WHEN CURRENT_ROLE() IN ( ' ' , ' ' ) THEN val -- Set for specific roles that should always have access  
      WHEN IS_ROLE_IN_SESSION('{{ policy }}') THEN val -- Set for the user to inherit access bases on there roles  
      ELSE {{ mask }}  
    END  
  {%- endset %}  
  
  {%- set policy_name %}  
    '{{ database }}'.{{ schema }}.{{ policy }}-{{ data_type }}  
  {%- endset %}  
  
  CREATE MASKING POLICY IF NOT EXISTS {{ policy_name }} AS (val {{ data_type }})  
  RETURNS {{ data_type }} ->  
  {{ body }};  
  
  ALTER MASKING POLICY IF EXISTS {{ policy_name }} SET BODY ->  
  {{ body }};  
  
{%- endmacro -%}
```



Best practices

Warehouse size

Warehouse Size	Credits / Hour	Credits / Second	Notes
X-Small	1	0.0003	Default size for warehouses created using <code>CREATE WAREHOUSE.</code>
Small	2	0.0006	
Medium	4	0.0011	
Large	8	0.0022	
X-Large	16	0.0044	Default for warehouses created in the web interface.
2X-Large	32	0.0089	
3X-Large	64	0.0178	
4X-Large	128	0.0356	
5X-Large	256	0.0711	Preview feature.
6X-Large	512	0.1422	Preview feature.



Best practices

Testing

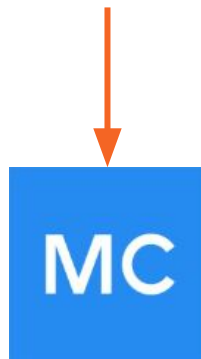
```
- name: instance_sql_errors
  tags: ["product", "service_ping"]
  description: '{{ doc("instance_sql_errors") }}'
  columns:
    - name: run_id
      tests:
        - not_null
    - name: sql_errors
      tests:
        - not_null
    - name: ping_date
      tests:
        - not_null
    - name: uploaded_at
      tests:
        - not_null
```



Best practices

Testing

```
dbt --no-use-colors test --profiles-dir profile --target prod --models workspaces.* ; ret=$?;  
montecarlo import dbt-run --manifest target/manifest.json --run-results target/run_results.json --project-name gitlab-analysis;  
python ../../orchestration/upload_dbt_file_to_snowflake.py test; exit $ret
```



Best practices

Testing





Automation

Automation

Pipelines - testing



Automation

Pipelines - testing

- Clone environments



Automation

Pipelines - testing

- Clone environments
- Run models



Automation

Pipelines - testing

- Clone environments
- Run models
- **SAFE**





Stay on the bright side - **SAFE** framework

- **S**ensitive
- **A**ccurate
- **F**inance
- **E**ffect

Automation

Pipelines - testing

- Clone environments
- Run models
- **SAFE**
- Linters



Clone

Snowflake

 clone_prod

 ! clone_prod_real

  clone_prep_spe...

  clone_prod_spe...

 clone_raw_full

  clone_raw_she...

  clone_raw_post...

  clone_raw_spe...

 force_clone_both

 grant_clones


Run

dbt Run


 specify_model

 specify_L_model




specify_dbt_param...

 specify_xL_model

  specify_raw_m...

 specify_csv_seed

  specify_snapshot



   specify_Lsna...


  run_changed_...

   run_changed_...

   run_changed_...

   specify_selec...

  run_changed_cl...

   clone_modelD...

Misc / testing


dbt Misc

 all_tests


 data_tests


 freshness

 schema_tests

 snapshots

 specify_tests

 manual_seed

 safe_model_script

 macro_name_che...

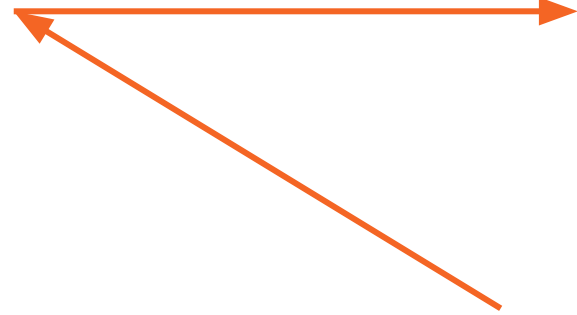
 periscope_query

 dbt_sqlfluff

Documentation

dbt Docs

pages



Automation

Pipelines - production

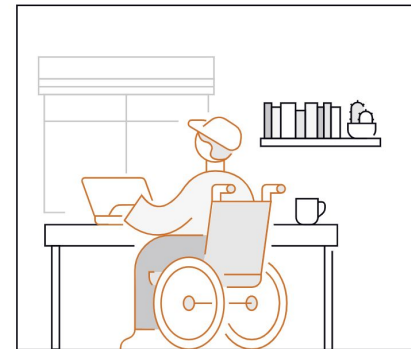
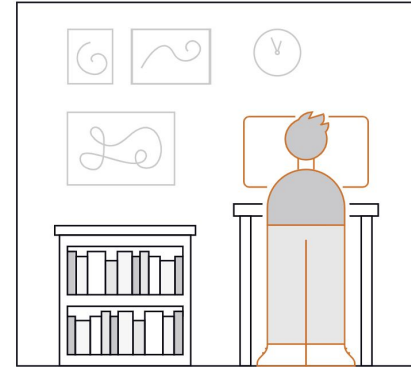
Automation

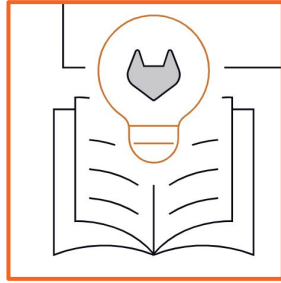
Documentation

<https://dbt.gitlabdata.com/>



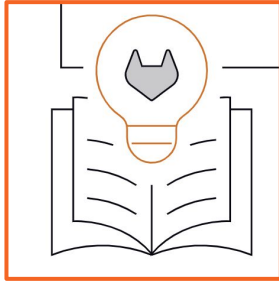
**Let's wrap up
and
gather **takeaways****



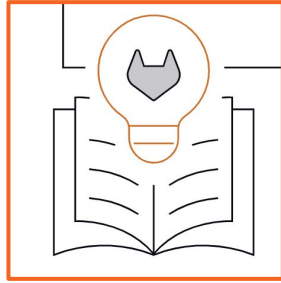


Speaking about transparency through our cultural lens, we are:

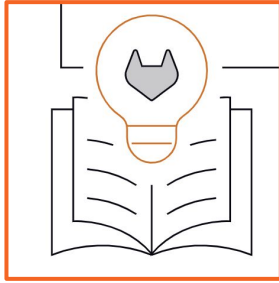
“Short Term **Critical And Long Term **Optimistic**”**



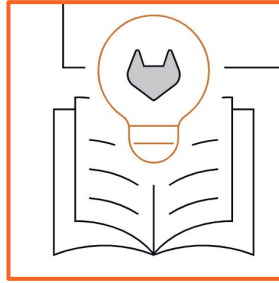
dbt is the proper fit for our
transformation use case



SSOT (Single source of truth) for our needs regarding the data project



Stick with **GitLab philosophy about
DevOps and **Open Source****



Allows us better and **integrative
collaboration among teams**



About me

Find me, ping me, ask me



Thank you!

