Hi, I’m Claire 👋

@clairebcarroll

FISHTOWN

ANALYTICS

Data engineers shouldn’t write DDL

Claire Carroll, dbt Community Manager

What is DDL?

- Data definition language
- Queries that define or alter the objects in a database
- Any query other than `select`:
  - `create`, `alter`, `drop`
  - `truncate`, `insert` and `update`*

```sql
create schema sandpit_claire;

create table sandpit_claire.stg_payments as (
   select
     id as payment_id,
     order_id,
     payment_method,

     -- amount is currently stored in cents
     amount / 100 as amount

  from raw_jaffle_shop.payments
);
```

How to write DDL

```sql
select
  id as payment_id,
  order_id,
  payment_method,

  -- amount is currently stored in cents
  amount / 100 as amount

from raw_jaffle_shop.payments;
```

```sql
create table sandpit_claire.stg_payments as (
  select
    id as payment_id,
    order_id,
    payment_method,

    -- amount is currently stored in cents
    amount / 100 as amount

  from raw_jaffle_shop.payments
);
```

```sql
create schema sandpit_claire;

create table sandpit_claire.stg_payments as (
  select
    id as payment_id,
    order_id,
    payment_method,

    -- amount is currently stored in cents
    amount / 100 as amount

  from raw_jaffle_shop.payments
);
```

```sql
create schema if not exists sandpit_claire;

drop table sandpit_claire.stg_payments;

create table sandpit_claire.stg_payments as (
  select
    id as payment_id,
    order_id,
    payment_method,

    -- amount is currently stored in cents
    amount / 100 as amount

  from raw_jaffle_shop.payments
);
```

```sql
create schema if not exists sandpit_claire;

drop table if exists sandpit_claire.stg_payments;

create table sandpit_claire.stg_payments as (
  select
    id as payment_id,
    order_id,
    payment_method,

    -- amount is currently stored in cents
    amount / 100 as amount

  from raw_jaffle_shop.payments
);
```

```sql
create schema if not exists sandpit_claire;

begin;

drop table if exists sandpit_claire.stg_payments;

create table sandpit_claire.stg_payments as (
  select
     id as payment_id,
     order_id,
     payment_method,

     -- amount is currently stored in cents
     amount / 100 as amount

  from raw_jaffle_shop.payments
);

commit;
```

```sql
create schema if not exists sandpit_claire;

begin;

drop table if exists sandpit_claire.stg_payments;

create table sandpit_claire.stg_payments as (
  select
    id as payment_id,
    order_id,
    payment_method,

    -- amount is currently stored in cents
    amount / 100 as amount

  from raw_jaffle_shop.payments
);

commit;
```

```sql
create view sandpit_claire.payments_agg as (
  select
    order_id,

    count(*) as n_payments,
    sum(
      case
        when payment_type = 'credit_card'
        then amount
        else 0
      end
    ) as credit_card_amount

  from sandpit_claire.stg_payments

  group by 1
);
```

```sql
create schema if not exists sandpit_claire;

begin;

drop table if exists sandpit_claire.stg_payments;

create table sandpit_claire.stg_payments as (
  select
    id as payment_id,
    order_id,
    payment_method,

    -- amount is currently stored in cents
    amount / 100 as amount

  from raw_jaffle_shop.payments
);

commit;
```

```sql
create view sandpit_claire.payments_agg as (
  select
    order_id,

    count(*) as n_payments,
    sum(
      case
        when payment_type = 'credit_card'
        then amount
        else 0
      end
     ) as credit_card_amount

  from sandpit_claire.stg_payments

  group by 1
) with no schema binding;
```

DDL optimizations

- Order of execution
- Atomic transactions
- Parameterized schemas
- Performance optimization
- Incremental builds
- Introspecting the warehouse

Good DDL is complex

Technical challenges

- Harder to do performance optimization
- Harder to switch platforms
- Any features on top of this require a lot of work:
  - Testing
  - Automated documentation

Organizational challenges

- Hard for others to understand

```
create schema if not exists sandpit_claire;

begin;

drop table if exists sandpit_claire.stg_payments;

create table sandpit_claire.stg_payments as (
  select
    id as payment_id,
    order_id,
    payment_method,

    -- amount is currently stored in cents
    amount / 100 as amount

  from raw_jaffle_shop.payments
);

commit;
```

```
create view sandpit_claire.payments_agg as (
  select
    order_id,

    count(*) as n_payments,
    sum(
      case
        when payment_type = 'credit_card'
        then amount
        else 0
      end
     ) as credit_card_amount

  from sandpit_claire.stg_payments

  group by 1
) with no schema binding;
```

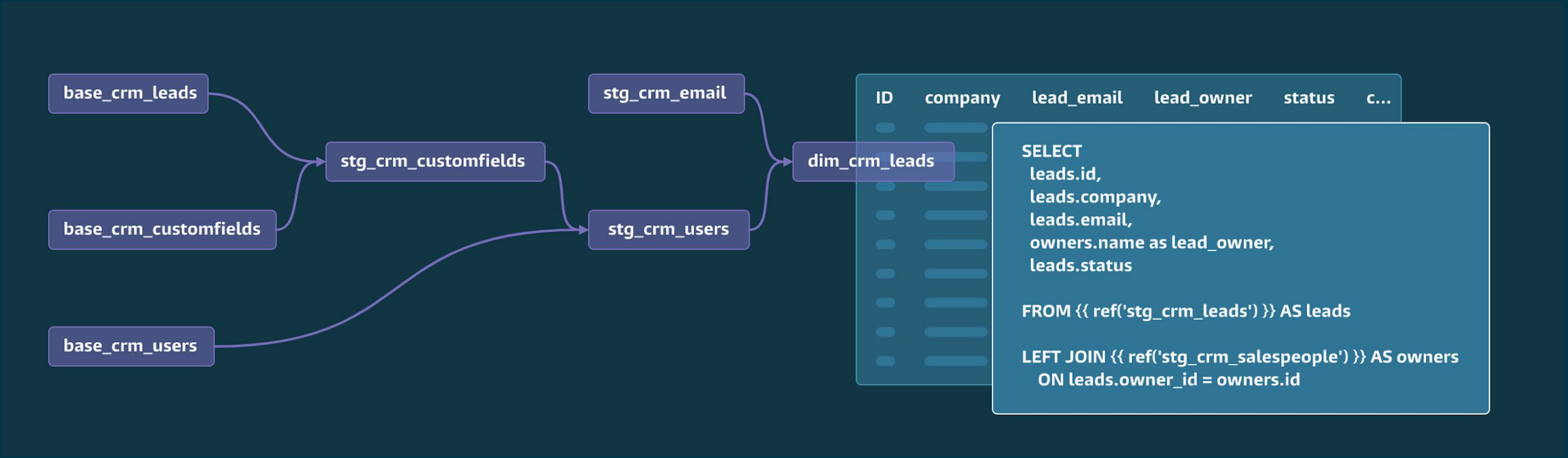
Organizational

- Hard for others to understand
- Become a bottleneck
- Low job satisfaction

How to not write DDL

Use a framework

- The framework generates the boilerplate code
- Options:
  - dbt (🙋‍♀️)
  - Roll your own (🙅‍♀️)
  - General DE platform: Airflow / Dagster / Prefect / Luigi
- Further reading: Maintainable ETLs – StitchFix



```
SELECT
 leads.id,
 leads.company,
 leads.email,
 owners.name as lead_owner,
 leads.status

FROM {{ ref('stg_crm_leads') }} AS leads

LEFT JOIN {{ ref('stg_crm_salespeople') }} AS owners
  ON leads.owner_id = owners.id
```

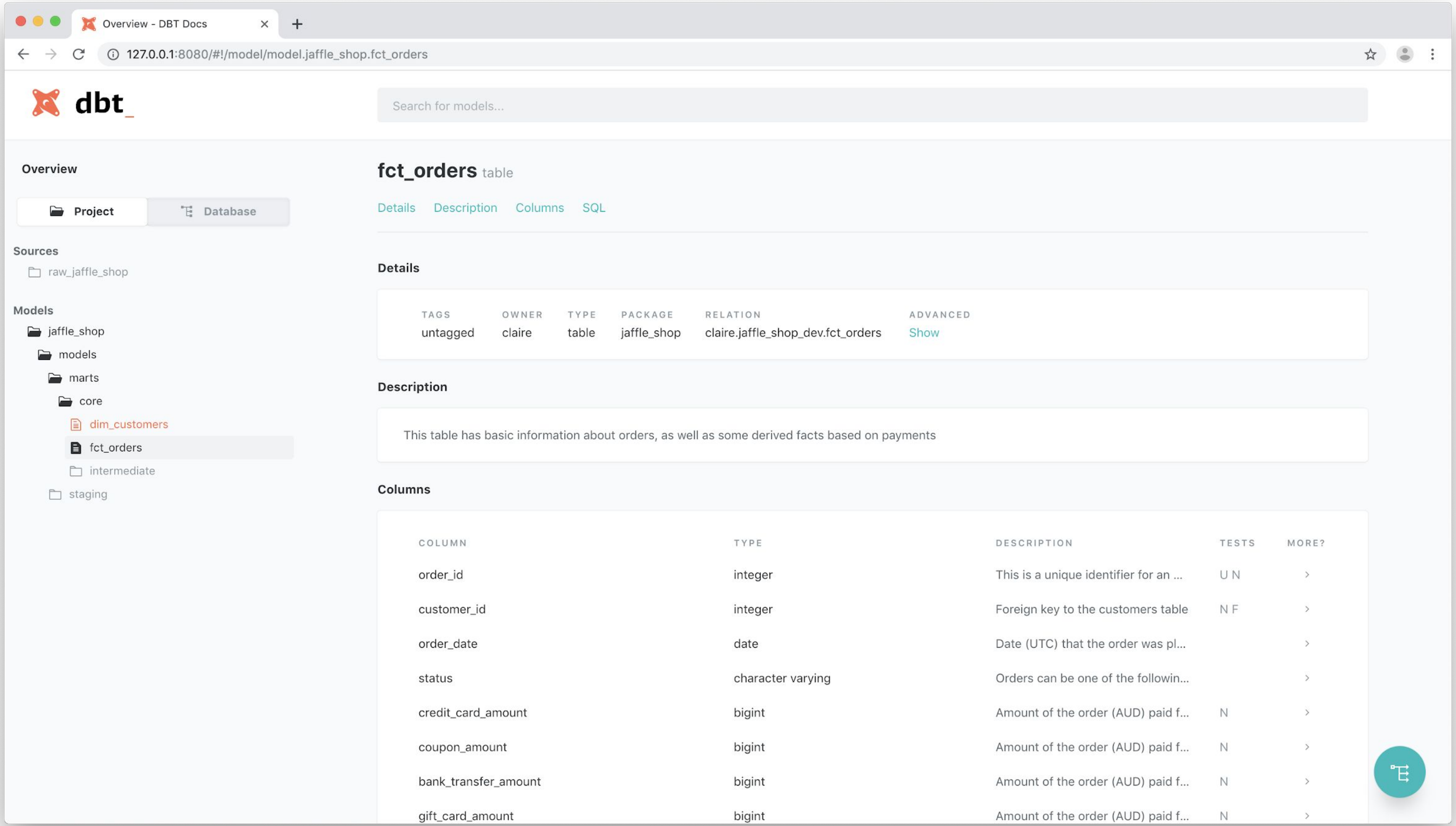
Benefits of a framework

Technological

- Most probems have already been thought about
- Good frameworks are extensible
- Allow you to focus on core business logic

Organizations

- Analysts are empowered to own the transformation layer
- Emergence of the analytics engineer


Overview - DBT Docs

127.0.0.1:8080/#!/model/model.jaffle_shop.fct_orders

dbt_

Search for models...

Overview

Project | Database

Sources
- raw_jaffle_shop

Models
- jaffle_shop
  - models
    - marts
      - core
        - dim_customers
        - fct_orders
      - intermediate
    - staging

fct_orders table

Details Description Columns SQL

Details

TAGS	OWNER	TYPE	PACKAGE	RELATION	ADVANCED
untagged	claire	table	jaffle_shop	claire.jaffle_shop_dev.fct_orders	Show

Description

This table has basic information about orders, as well as some derived facts based on payments

Columns

COLUMN	TYPE	DESCRIPTION	TESTS	MORE?
order_id	integer	This is a unique identifier for an ...	U N	›
customer_id	integer	Foreign key to the customers table	N F	›
order_date	date	Date (UTC) that the order was pl...		›
status	character varying	Orders can be one of the followin...		›
credit_card_amount	bigint	Amount of the order (AUD) paid f...	N	›
coupon_amount	bigint	Amount of the order (AUD) paid f...	N	›
bank_transfer_amount	bigint	Amount of the order (AUD) paid f...	N	›
gift_card_amount	bigint	Amount of the order (AUD) paid f...	N	›

Benefits of a framework

Technological

- Most probems have already been thought about
- Good frameworks are extensible
- Allow you to focus on core business logic

Organizational

- Analysts are empowered to own the transformation layer
- Emergence of the analytics engineer

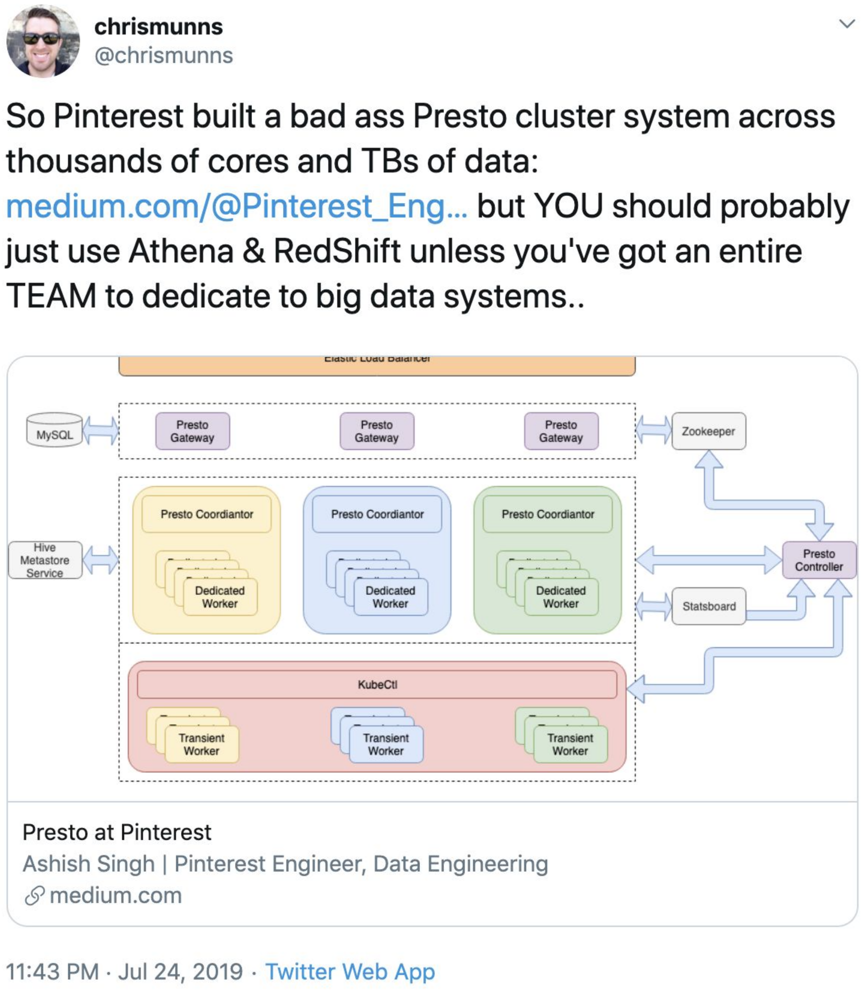
Data engineers shouldn’t write DDL, but...

Data engineers should *deeply understand* DDL

Concurrent trends

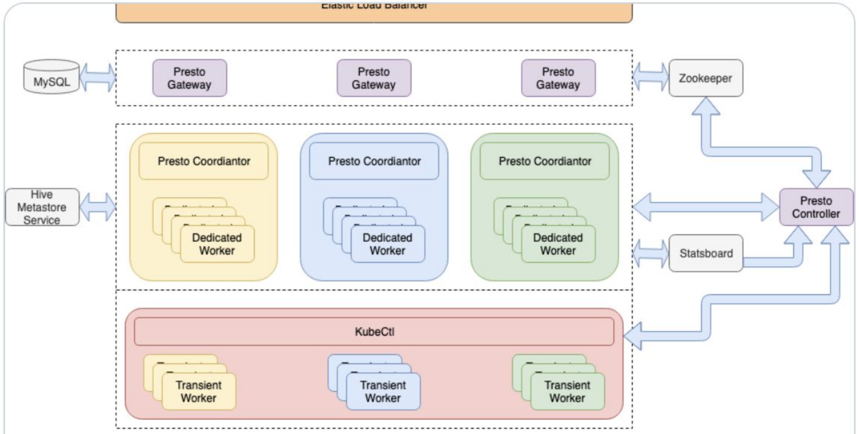
Where possible:

- Data engineers shouldn't write data ingestion
- Data engineers shouldn’t write data collectors
- Data engineers shouldn’t build data warehouse technology



chrismunns
@chrismunns

So Pinterest built a bad ass Presto cluster system across thousands of cores and TBs of data: medium.com/@Pinterest_Eng... but YOU should probably just use Athena & RedShift unless you've got an entire TEAM to dedicate to big data systems..



Presto at Pinterest
Ashish Singh | Pinterest Engineer, Data Engineering
medium.com

11:43 PM · Jul 24, 2019 · Twitter Web App

So what *do* data engineers do?

Extending frameworks

- Custom materializations in dbt
- Building features for dbt

Data warehouse performance

- Performant data model design
- Performance tuning

Deployment

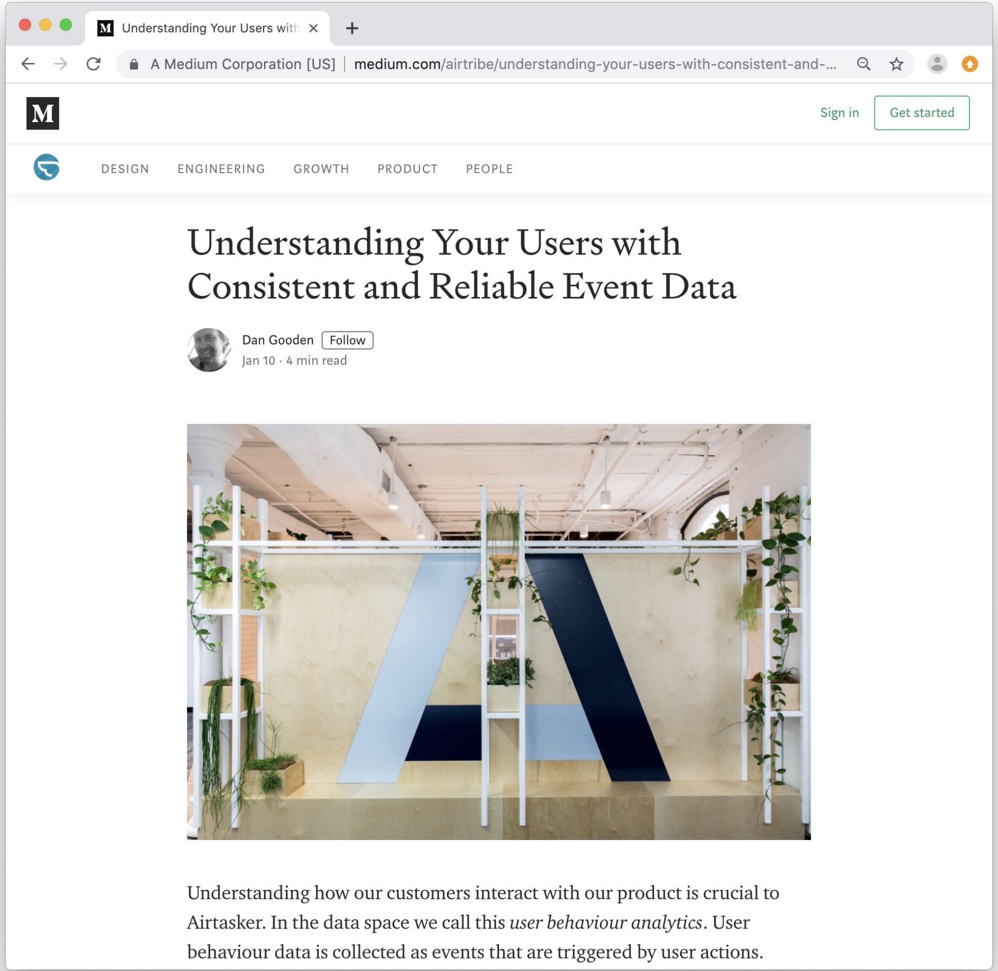
- CI
- CD
- Orchestration
- Credentials storage
- Monitoring
- Alerting

Custom data ingestion

- For APIs that aren’t on Stitch / Singer
- Writing Singer taps

Source data design

- Working with core product team
- Data quality enforcement


Understanding Your Users with
A Medium Corporation [US] | medium.com/airtribe/understanding-your-users-with-consistent-and-...
Sign in
Get started
DESIGN
ENGINEERING
GROWTH
PRODUCT
PEOPLE

Understanding Your Users with Consistent and Reliable Event Data

Dan Gooden Follow
Jan 10 · 4 min read

Understanding how our customers interact with our product is crucial to Airtasker. In the data space we call this *user behaviour analytics*. User behaviour data is collected as events that are triggered by user actions.

Data platform infrastructure

- Luigi to Dagster?
- Redshift to Snowflake?

Working with analysts

- Teach them engineering concepts
- Learn how to make their lives more productive

The role of data engineers

- Extend frameworks
- Data warehouse performance
- Deployment
- Custom data ingestion
- Source data design and data quality enforcement
- Data platform infrastructure
- Working with analysts

Thanks!

Any questions?

- @getdbt
- getdbt.com
- learn.getdbt.com