Intro to Data Modeling for Analytics

Nick Johnson

May 15, 2024

Why do we make software applications?

We want to capture data that can be used for something!

Cases for data modeling

- Transactional (Application) uses
- Analytics
- ML
- others

What should I put in my analytics platform?

- PIES test
- Performance
- Integration
- Ease of Use
- SLA consideration

Importance of Data Modeling for Analytics

Too often, analytics is an after-thought

Think about the problem you are trying to solve

What questions do you want the data to answer?

Who will be consuming the data model?

Spend time getting it right

Dimensional Modeling Intro

- Define decoupled keys
  - If needed, define keys for your analytics environment to decouple from source - "hubs" store the mapping
- Dimensions
  - Filters and Grouping/Slicing
  - Type I and Type II Slowly Changing Dimensions
- Facts
  - Things we summarize
  - Numbers, Counts, Amounts

Ground rules

No nulls in dimensions

Decide on the grain first

What questions are you trying to answer?

What is the domain of the questions you're trying to answer?

Grain

The finer the grain, the more detail we store, and we can always summarize up.

All facts and dimensions need to be defined at the same grain

May need to aggregate to the highest grain you have.

```
Date, Product
Week, Product
Month, Category
Date, Category
```

Some Recommendations

Build a date dimension that can be re-used

User-friendly field names

Always include update timestamps for auditing and debugging

Design dimensions first, then bring dimension keys over to fact

Pay attention to cardinality

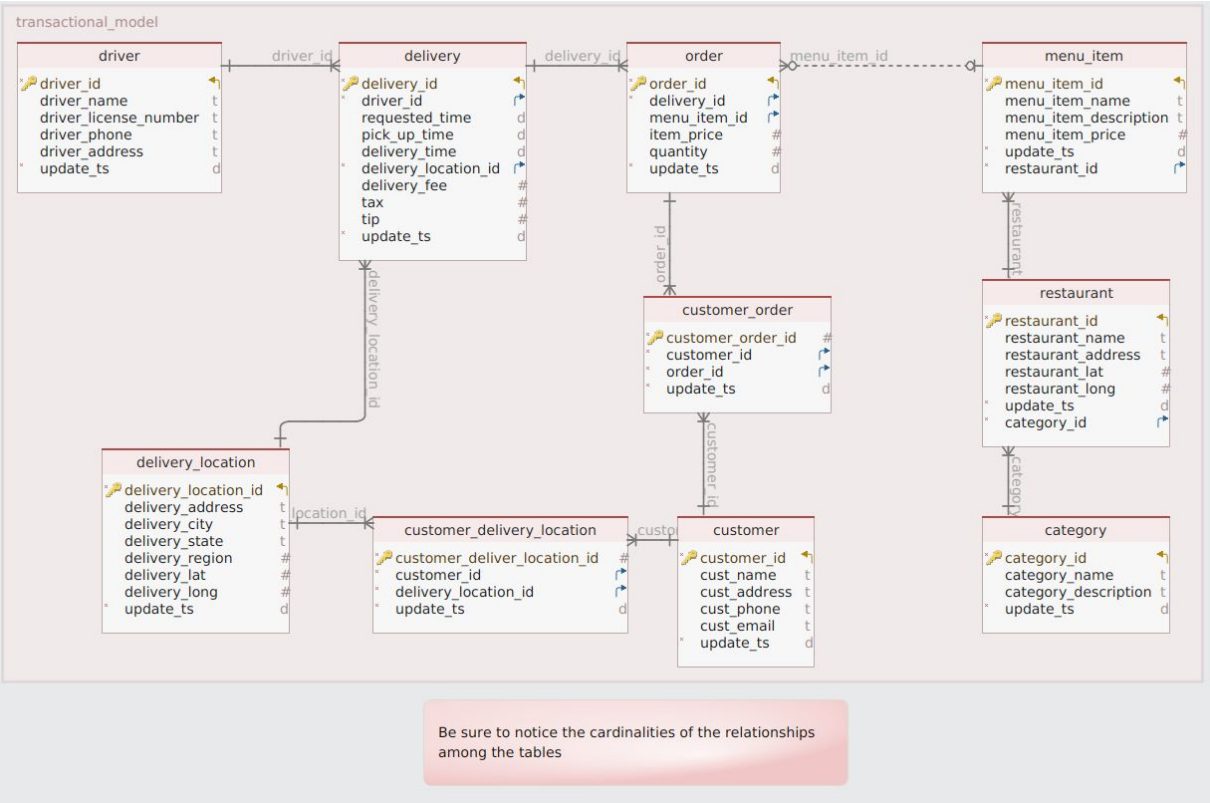
Do not enforce key constraints in analytics platform

Work Through Example Business Case

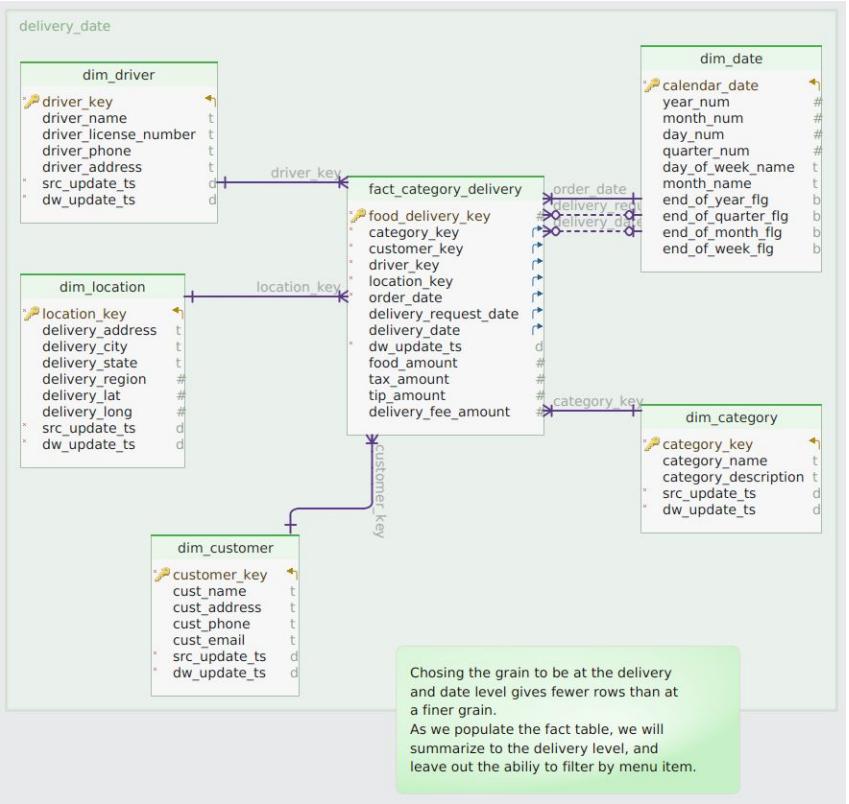
Food Delivery Platform

Their business concepts include customers, restaurants, restaurant categories, menu items, orders, deliveries, delivery locations, and drivers.

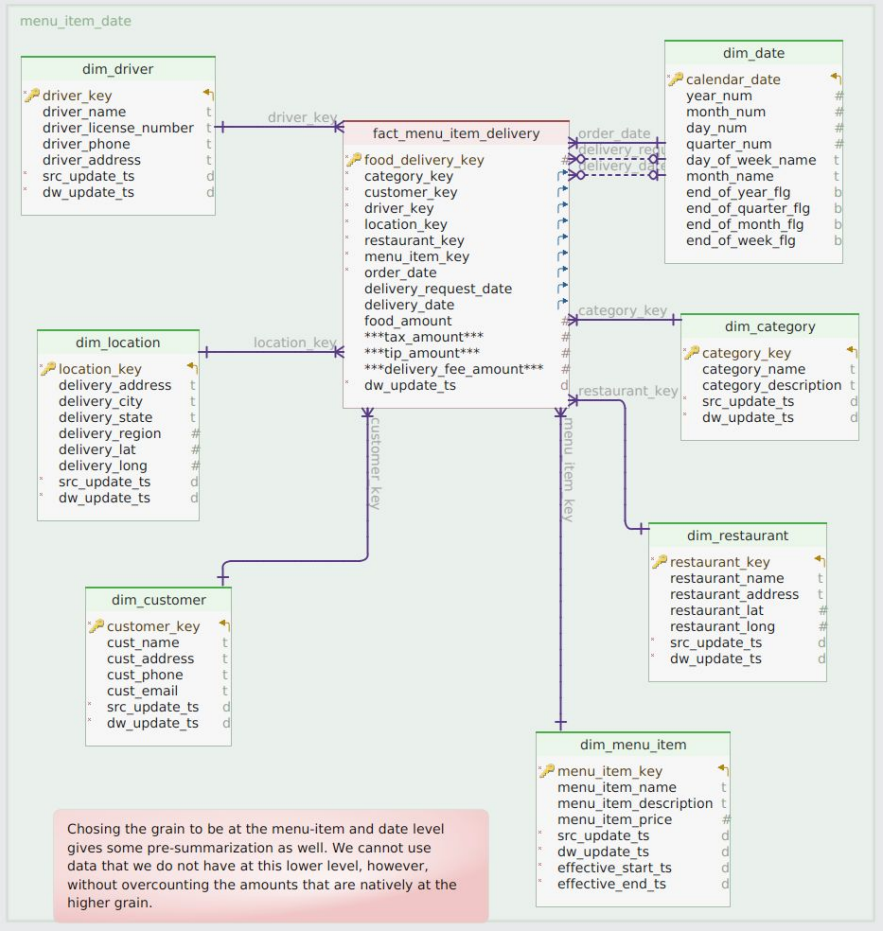
Transactional Model



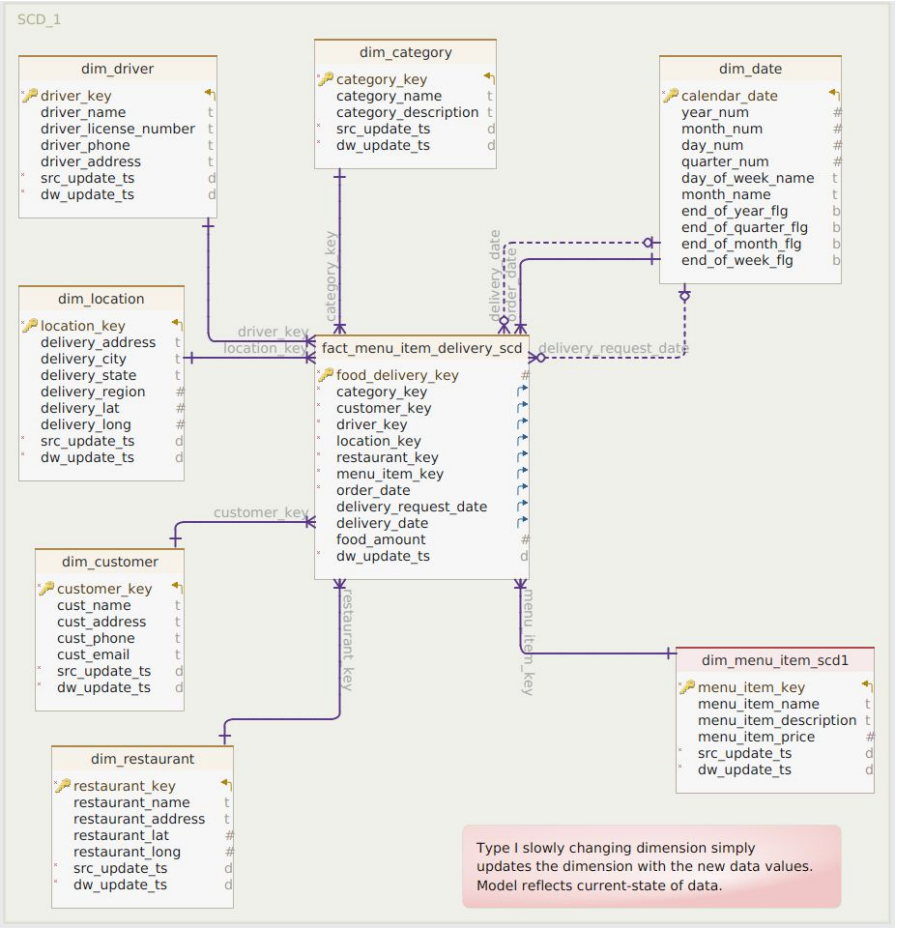
Grain - delivery, date



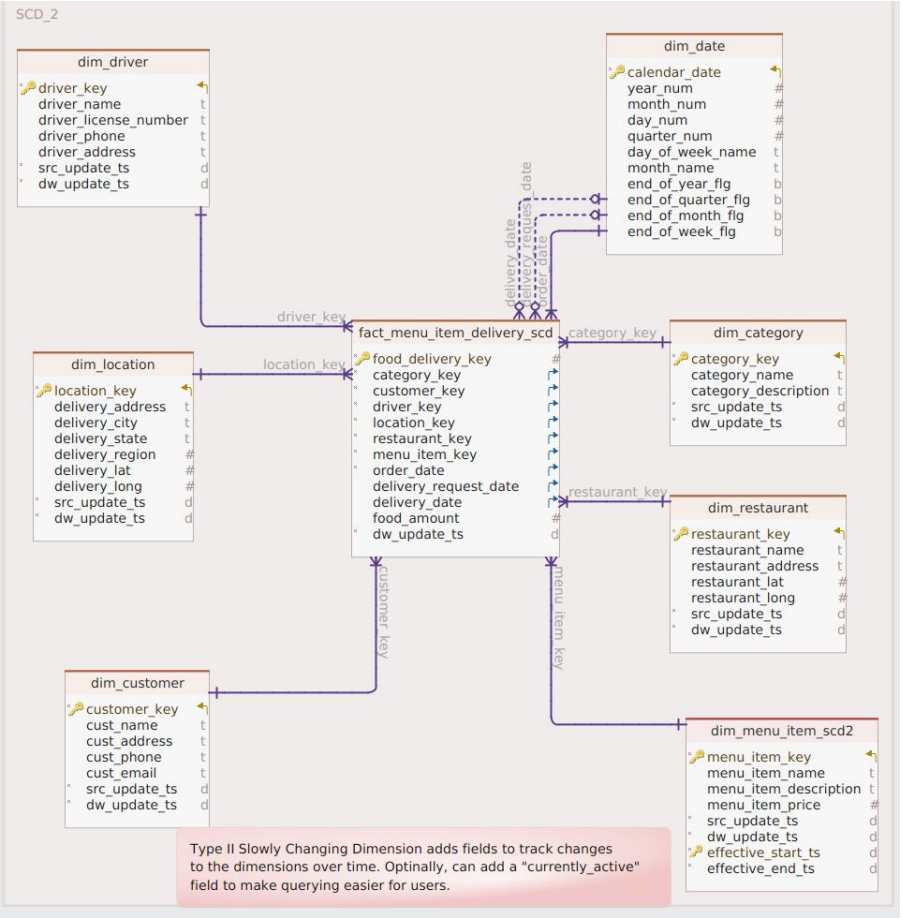
Grain - menu_item, date



Type I Slowly Changing Dimension



Type II Slowly Changing Dimension

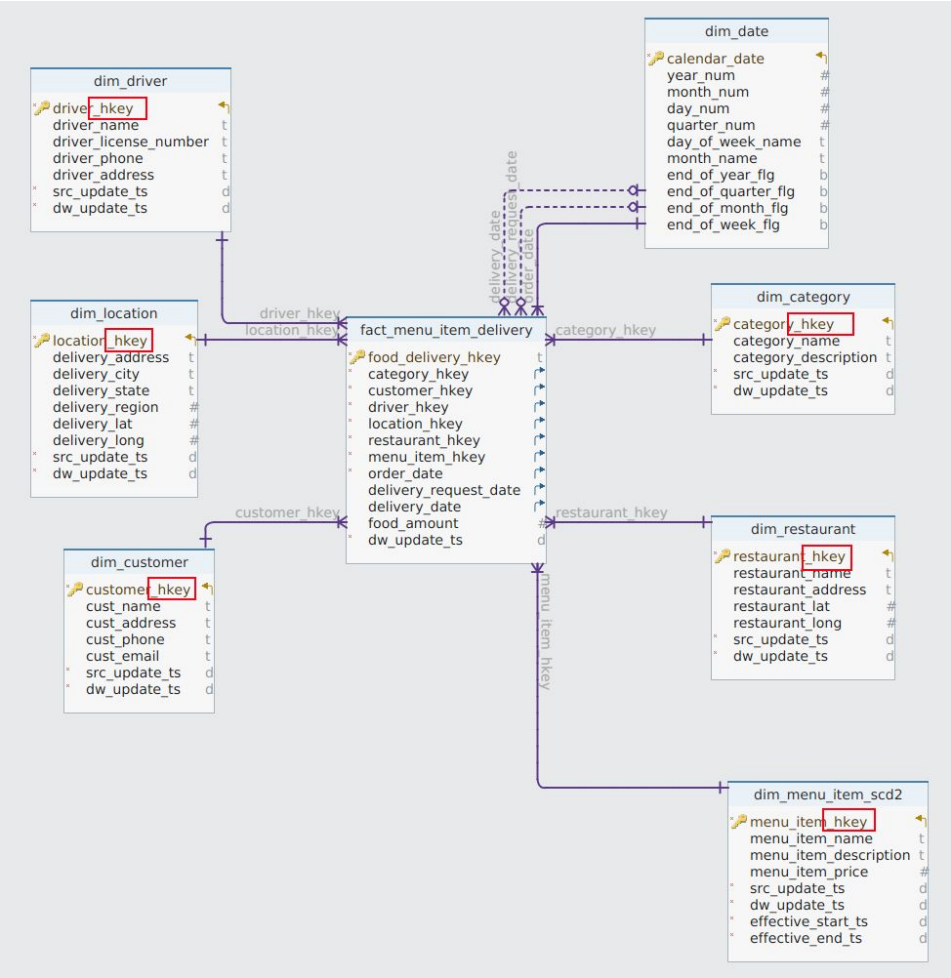


Borrow Concepts from Data Vault 2.0

Insert Only

Hashes that allow parallel loading

Prevention of duplicates



Borrow concepts from Data Vault 2.0

```
-- Instead of using a decoupled integer or a simple copy of the source system's
--   primary key for the dimension key, which requires you to complete the
--   loading of the dimension tables prior to loading the fact table,
--   you can use a hash of the source key and a source system identifier.
--  This enables you to load in parallel.  The hashdiff example field
--     is something that can be used to help identify changed records
--     when you're doing an insert-only approach.

select
       md5(nvl(upper(trim(source_key::string)),'-1')
           ||';'|| source_system_id::string) as DIM_SAMPLE_HKEY
     , source_key as SOURCE_KEY
     , md5(                        source_key::string
           ||';'|| nvl(to_varchar(interesting_date)::date, 'yyyy-mm-dd'),'2100-01-01')
           ||';'|| nvl(upper(trim(interesting_number::string)),'-1')
           ||';'|| nvl(upper(trim(interesting_text::string)),'-1')
           ||';'|| nvl(to_varchar(source_update_date)::date, 'yyyy-mm-dd'),'2100-01-01')
         ) as hashdiff
     , interesting_date
     , interesting_number
     , interesting_text
     , source_update_date
     , now() as update_ts
  from source_system_table

```