Postgres BI

Андрей Фефелов

mastery.pro

Содержание

- Постановка задачи
- Open source решения
- ROLAP
- Обзор нашей архитектуры
- Особенности Postgres для BI
  - ETL vs ELT (stage-nds-ddm)
  - Column data storage
  - Configuration
  - Фишечки
- Плюсы/минусы решения

Постановка задачи

- Заказчик – крупнейшая в Ирландии фармацевтическая группа
- 4 типа аптечного ПО
- 250 аптек
- Анализировать:
  - Заказы (Orders)
  - Рецепты (Scripts)
  - Возмещения (Claims)
- Цели:
  - Оптимизации товарной политики
  - Маркетинг

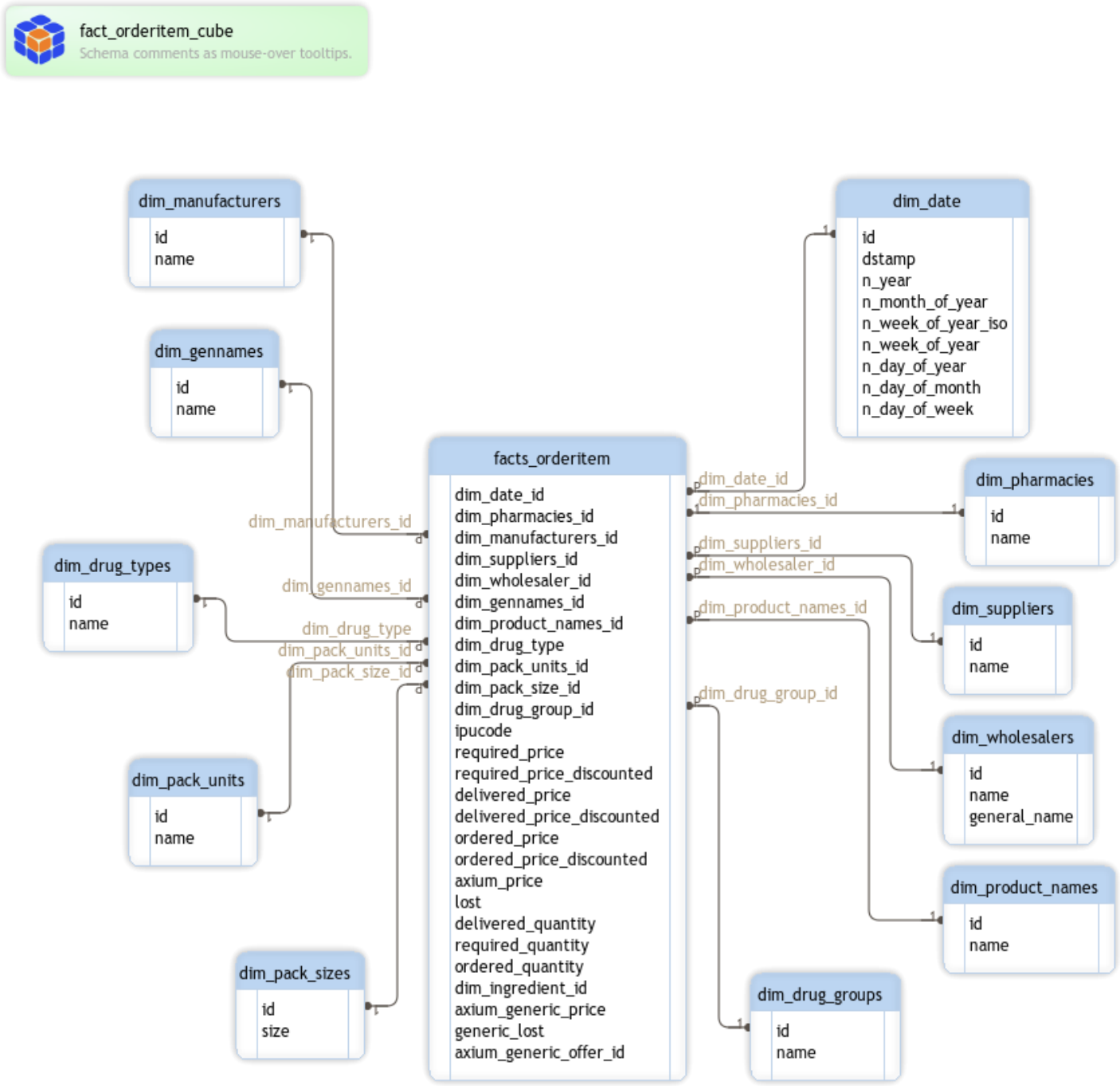
Open source

- SpagoBI
- Pentaho
- Mondrian
- Saiku
- Cubes (databrewery)

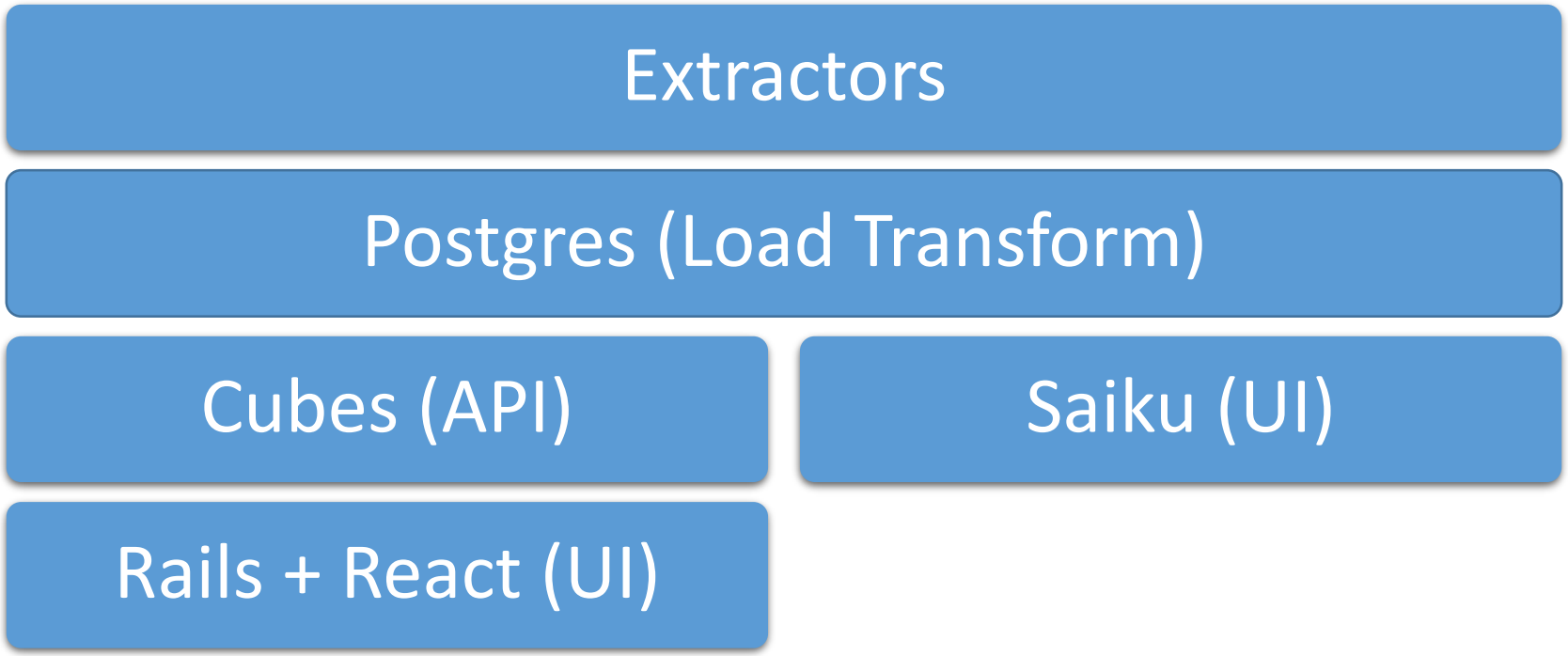
ROLAP (R-ROLAP)

- Звезда
  - Факты
  - Измерения
  - Меры
- No pre-calculated aggregates
- SSD
- Колоночное хранение
- ???
- Profit!

ROLAP

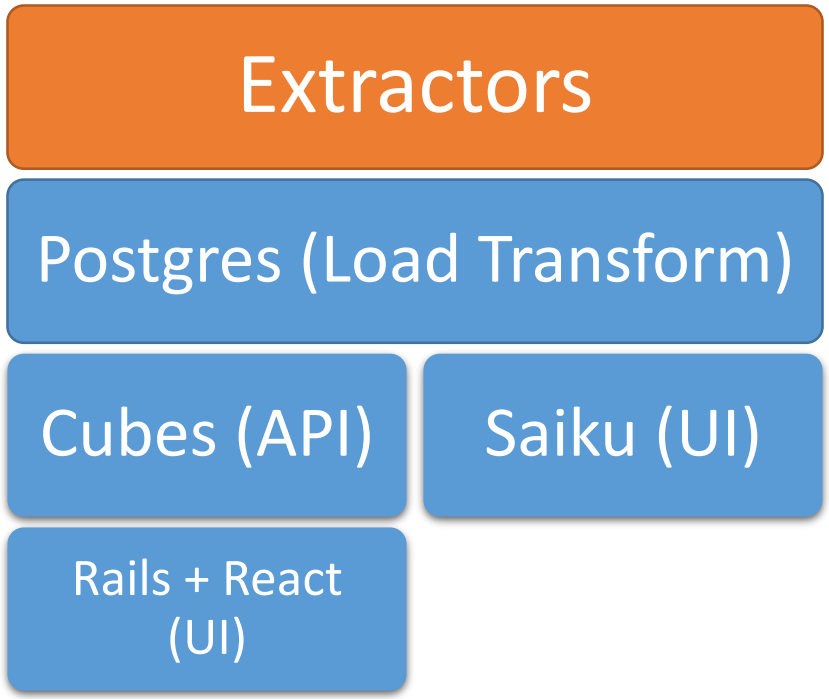

fact_orderitem_cube
Schema comments as mouse-over tooltips.
dim_manufacturers
id
name
dim_gennames
id
name
dim_drug_types
id
name
dim_pack_units
id
name
dim_pack_sizes
id
size
dim_manufacturers_id
dim_gennames_id
dim_drug_type
dim_pack_units_id
dim_pack_size_id
facts_orderitem
dim_date_id
dim_pharmacies_id
dim_manufacturers_id
dim_suppliers_id
dim_wholesaler_id
dim_gennames_id
dim_product_names_id
dim_drug_type
dim_pack_units_id
dim_pack_size_id
dim_drug_group_id
ipucode
required_price
required_price_discounted
delivered_price
delivered_price_discounted
ordered_price
ordered_price_discounted
axium_price
lost
delivered_quantity
required_quantity
ordered_quantity
dim_ingredient_id
axium_generic_price
generic_lost
axium_generic_offer_id
dim_date_id
dim_pharmacies_id
dim_suppliers_id
dim_wholesaler_id
dim_product_names_id
dim_drug_group_id
dim_date
id
dstamp
n_year
n_month_of_year
n_week_of_year_iso
n_week_of_year
n_day_of_year
n_day_of_month
n_day_of_week
dim_pharmacies
id
name
dim_suppliers
id
name
dim_wholesalers
id
name
general_name
dim_product_names
id
name
dim_drug_groups
id
name

Архитектура



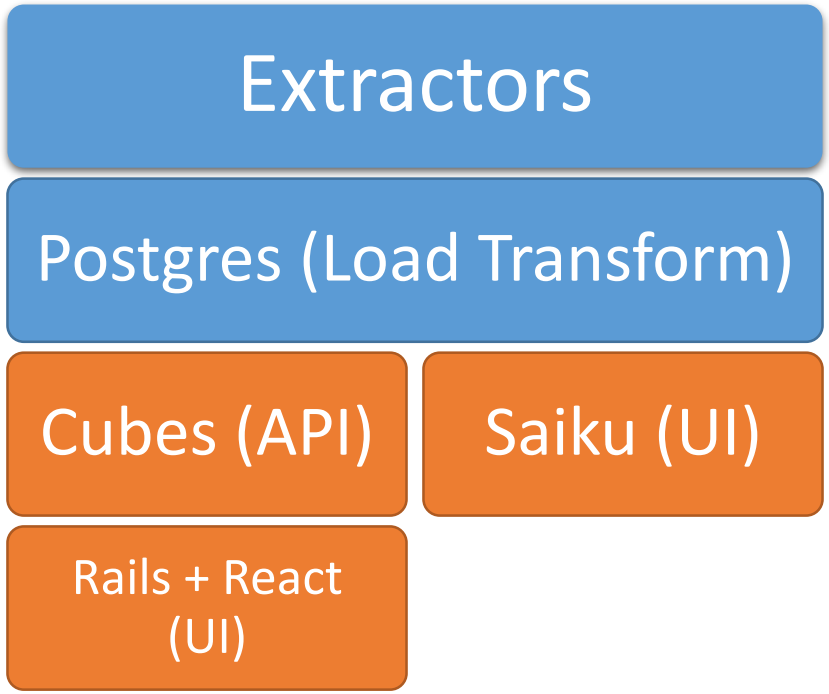
Архитектура - extractors

- Cyclone_client
  - Mssql (2008-2012)
  - Golang
  - CSV + rsync over ssh
- Kachok
  - Web scrapper
- Skytools replication
  - From existing products

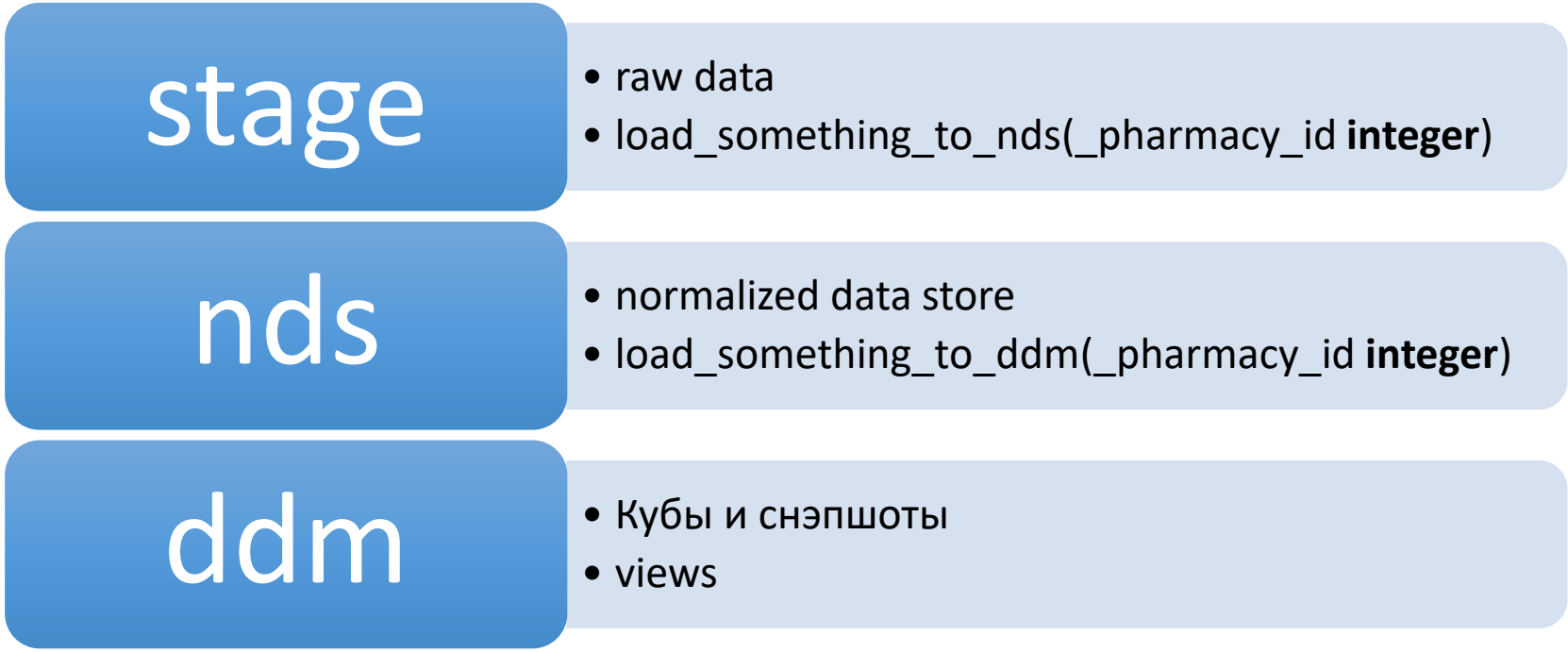


Архитектура – API + UI

- Cubes - cubes.databrewery.org
  - Easy drilling-down
  - Slicing and dicing
  - Serves aggregates, dimension details, facts
  - Provides all necessary metadata for a reporting application
- Rails, React
  - Авторизация
  - d3, dc, crossfilter
- Saiku
  - Только для бэк-офиса



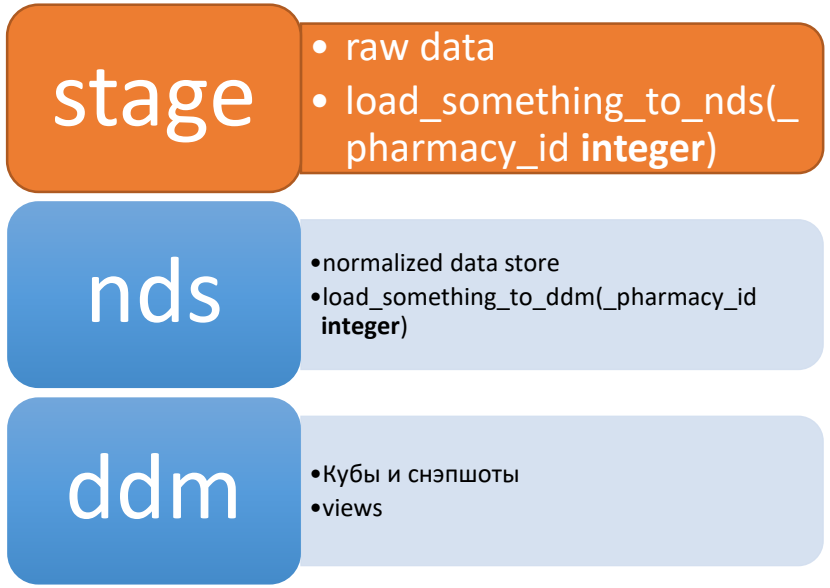
Архитектура – Postgres (load, transform)



Архитектура – Postgres (load, transform)

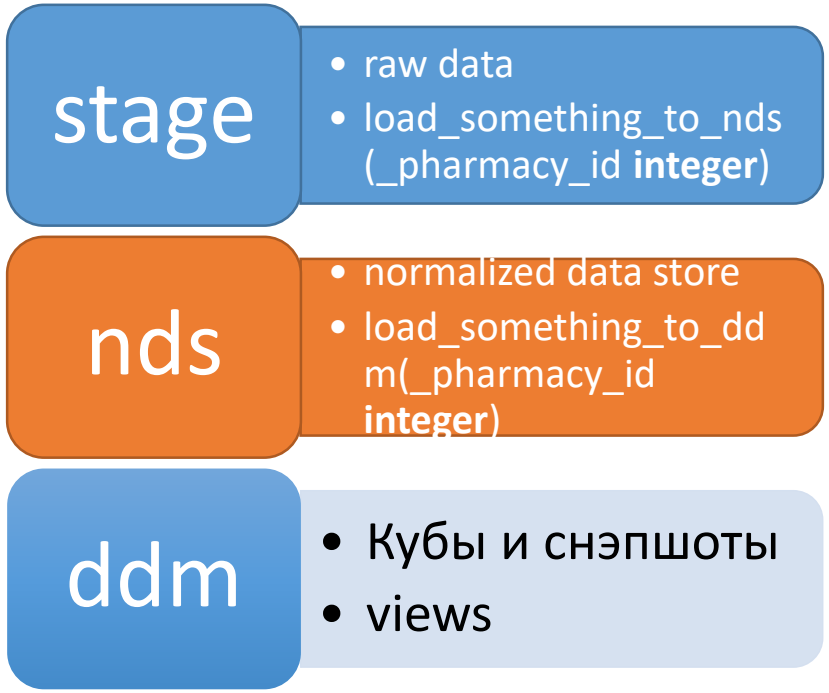
Stage

- «Сырые» данные
- Полностью очищается в каждом цикле ELT
- Служит источником для nds



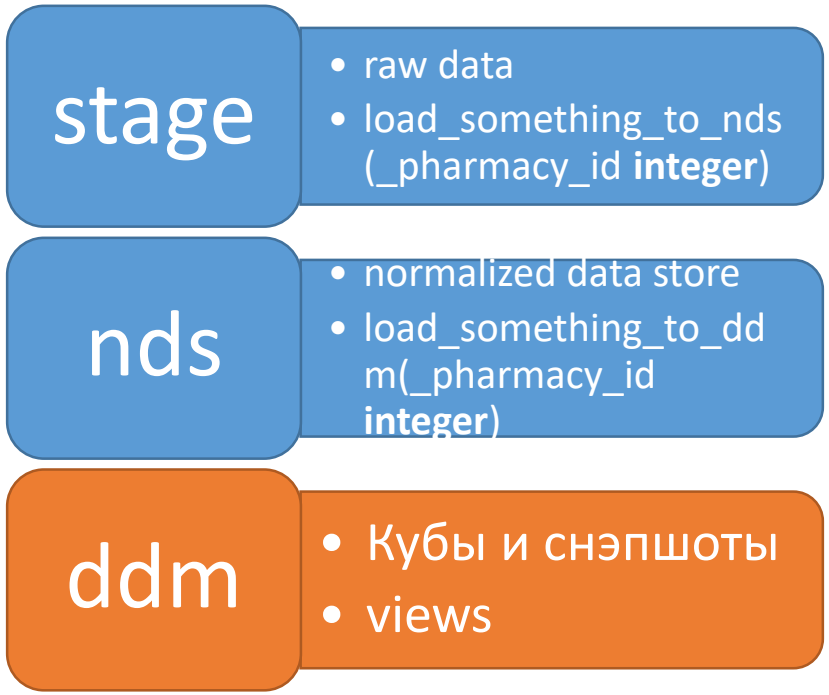
Архитектура – Postgres (load, transform)

- Normalized Data Store
  - Данные нормализуются и валидируются
  - Служит источником для ddm
    - Вычисляются меры перед загрузкой в ddm
  - Вычисляется delta для загрузки в ddm на основе last_updated

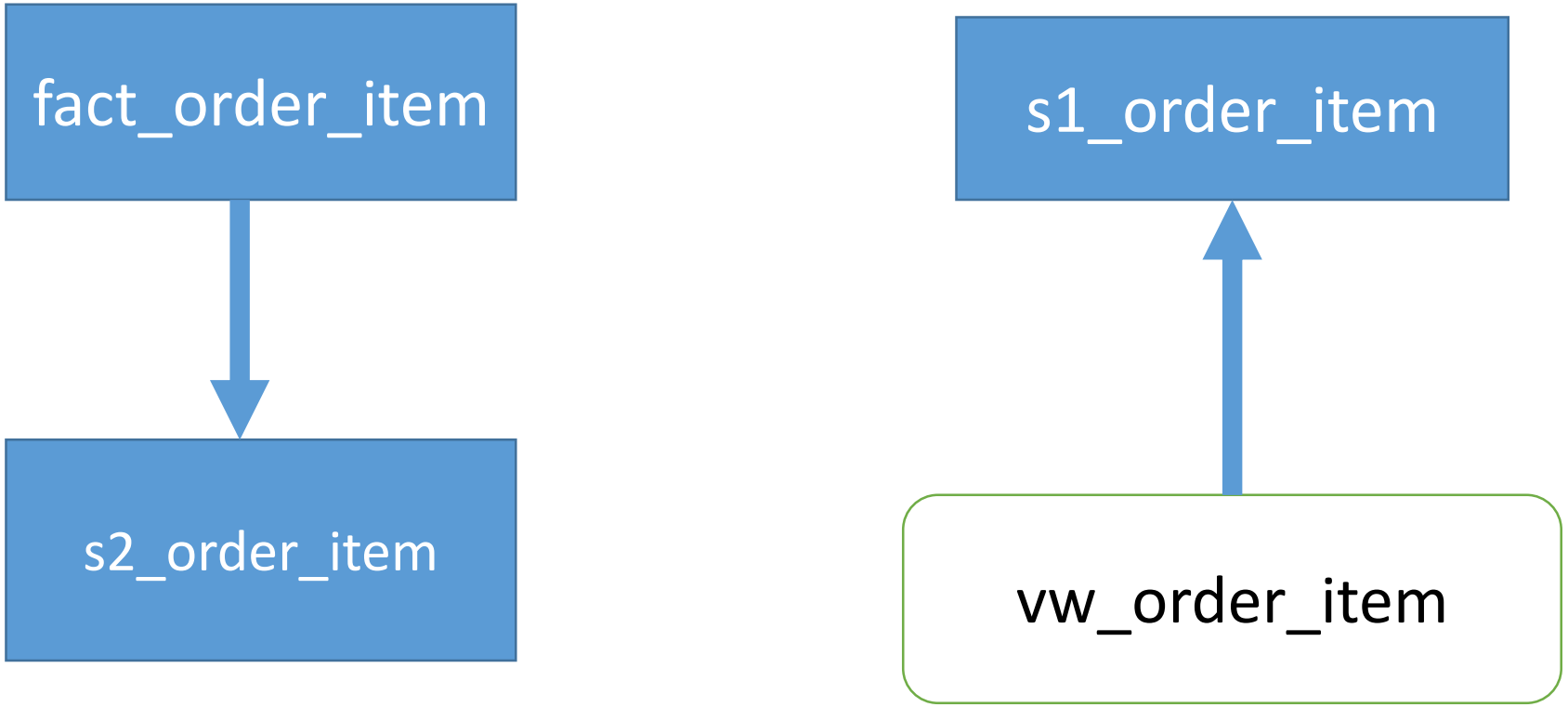


Архитектура – Postgres (load, transform)

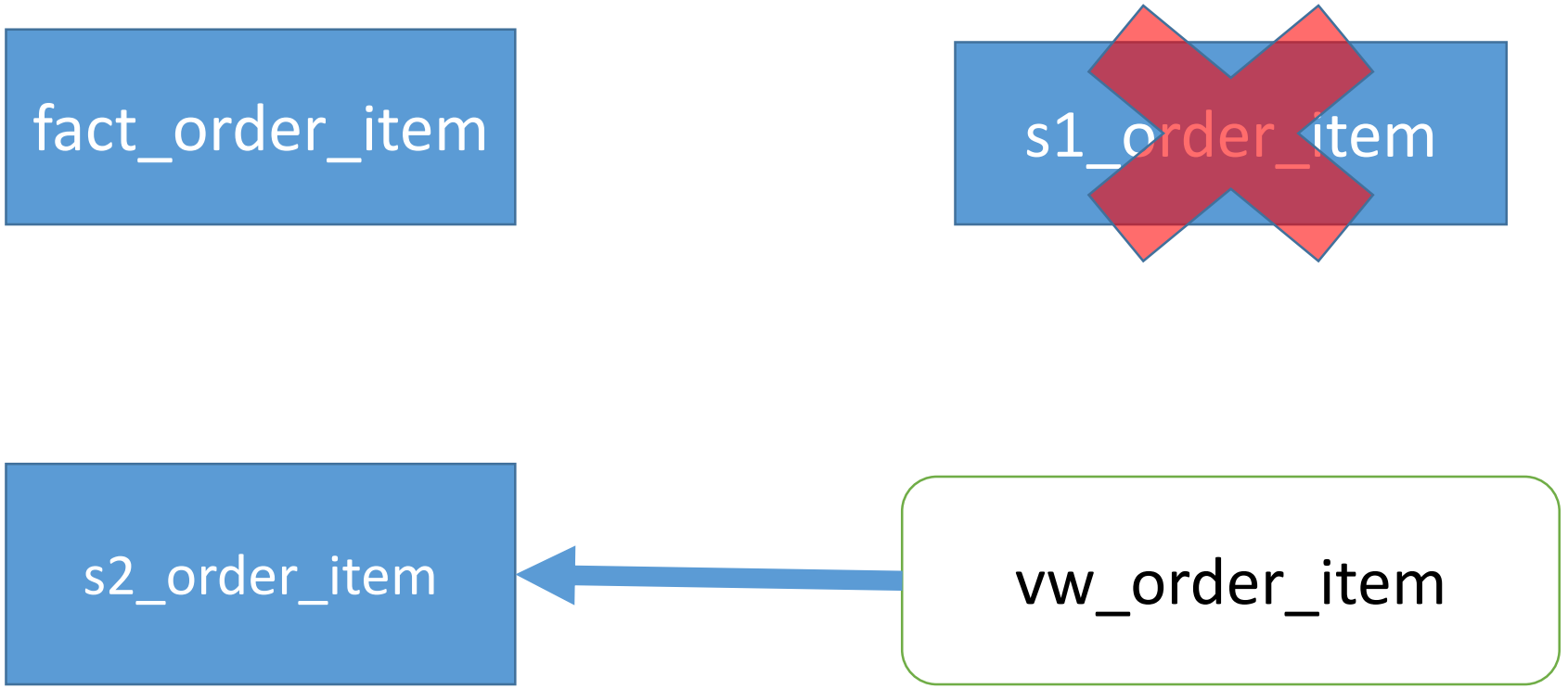
- Dimensional data model
  - Кубы
  - Снэпшоты тоже здесь
    - Спокойно делать релизы
    - Анализировать состояния «до-после» релиза
    - View как точка входа для приложения



Архитектура – Postgres (snapshots)



Архитектура – Postgres (snapshots)



Column storage

- Подходит для задачи:
  - агрегаций
  - вывод фиксированного набора полей из кубов
- cstore_fdw -> https://github.com/citusdata/cstore_fdw
  - Compression: Reduces in-memory and on-disk data size by 2-4x. Can be extended to support different codecs.
  - Column projections: Only reads column data relevant to the query. Improves performance for I/O bound queries.
  - Skip indexes: Stores min/max statistics for row groups, and uses them to skip over unrelated rows.

Column storage

- Наш опыт:
  - Не быстрее на нашем профиле, чем “ванильный” postgres (привет, cubes)
  - Размер кубов уменьшился в 12 раз. Wow.
  - Не бэкапится стандартным образом (no need?)
  - Не поддерживает delete/update (snapshots)

Конфигурация

- Профиль нагрузки:
  - Большой объем RW I/O
  - Основной объем I/O в stage, nds
  - DDM нагружена слабо

- shared_buffers = ½ RAM
- work_mem = 1GB
- maintenance_work_mem = 2GB
- temp_buffers = 2GB
- effective_cache_size = ½ RAM
- checkpoint_segments = 128

Фишечки

- DDM может быть вынесена на отдельный сервер (londiste)
- Используйте COPY/BULK INSERTS не используйте UPDATE (ха ха ха)
- Думайте о горизонтальном и вертикальном партиционировании поищите хорошие ключи для этого
- Думайте о параллелизме с самого начала
- Используйте TABLESPACES/PARTIAL INDEXES (и больше дисков)
- Вам нужна политика хранения данных
- Статистику пишите в tempfs

Фишечки ч2

- Используйте миграции – sqitch by theory
- Тестируйте ELT - sqitch by theory
- Анализируйте pg_stat_statements (добавьте в мониторинг)
- Профилируйте процедуры – PLPROFILER3
- Иногда, вам (не) нужен cstore_fdw
- Иногда, вам (не) нужны unlogged tables

Плюсы и минусы решения

- Минусы
  - Плохо масштабируется горизонтально
  - Сложный деплой

- Плюсы
  - Local data (no big network transfers)
  - Effectively parallelized (thanks to pharmacy_id)
  - **PL**/pgSQL

Спасибо за внимание

andy@mastery.pro

Speed limit

- cubes не быстрый (сериализация)
  - json (12 sec)
  - ujson (4 sec)
  - postgres json output (1.5 sec) db self time 0.3-0.7 sec