Business Intelligence: OLAP, Data Warehouse, and Column Store

Thomas Heinis

Imperial College London

Why we still study OLAP/Data Warehouse in Big Data?

- Understand the Big Data history
  - How does the requirement of (big) data analytics/business intelligence evolve over the time?
  - What are the architecture and implementation techniques being developed? Will they still be useful in Big Data?
  - Understand their limitation and what factors have changed from 90's to now?
- NoSQL is not only SQL ☺
- Hive/Impala aims to provide OLAP/BI for Big Data using Hadoop

Highlights

- OLAP
  - Multi-relational Data model
  - Operators
  - SQL
- Data warehouse (architecture, issues, optimizations)
- Join Processing
- Column Stores (Optimized for OLAP workload)

Back to the 70's: Relational Databases

Basic Structure

- Formally, given sets D_1, D_2, ... D_n a **relation** r is a subset of
  D_1 x D_2 x … x D_n
  Thus, a relation is a set of n-tuples (a_1, a_2, …, a_n) where each $a_i \in D_i$
- Example:

  customer_name = {Jones, Smith, Curry, Lindsay}
  customer_street = {Main, North, Park}
  customer_city = {Harrison, Rye, Pittsfield}

  Then r = { (Jones, Main, Harrison),
  (Smith, North, Rye),
  (Curry, North, Rye),
  (Lindsay, Park, Pittsfield) }

  is a relation over
  customer_name , *customer_street*, *customer_city*

Relation Schema

- $A_1, A_2, \ldots, A_n$ are *attributes*

- $R = (A_1, A_2, \ldots, A_n)$ is a *relation schema*

  Example:

  Customer_schema = (*customer_name, customer_street, customer_city*)

- *r*(*R*) is a *relation* on the *relation schema R*

  Example:

  customer (Customer_schema)

Relation Instance

- The current values (*relation instance*) of a relation are specified by a table
- An element *t* of *r* is a *tuple*, represented by a *row* in a table

attributes (or columns)

customer_name	*customer_street*	customer_city
Jones	Main	Harrison
Smith	North	Rye
Curry	North	Rye
Lindsay	Park	Pittsfield

tuples (or rows)

customer

Database

- A database consists of multiple relations
- Information about an enterprise is broken up into parts, with each relation storing one part of the information

  account : stores information about accounts
  depositor : stores information about which customer owns which account
  customer : stores information about customers

- Storing all information as a single relation such as *bank*(*account_number, balance, customer_name*, ..) results in repetition of information (e.g., two customers own an account) and the need for null values (e.g., represent a customer without an account)

Banking Example

branch (branch-name, branch-city, assets)

customer (customer-name, customer-street, customer-city)

account (account-number, branch-name, balance)

loan (loan-number, branch-name, amount)

depositor (customer-name, account-number)

borrower (customer-name, loan-number)

Relational Algebra

- Primitives
  - Projection (π)
  - Selection (σ)
  - Cartesian product ($\times$)
  - Set union ($\cup$)
  - Set difference (-)
  - Rename (ρ)
- Other operations
  - Join ($\bowtie$)
  - Group by… aggregation
  - …

What happens next?

- SQL
- System R (DB2), INGRES, ORACLE, SQL-Server, Teradata
  - B+-Tree (select)
  - Transaction Management
  - Join algorithm

Early 90's:
OLAP & Data Warehouse

Database Workloads

- OLTP (online transaction processing)
  - Typical applications: e-commerce, banking, airline reservations
  - User facing: real-time, low latency, highly-concurrent
  - Tasks: relatively small set of “standard” transactional queries
  - Data access pattern: random reads, updates, writes (involving relatively small amounts of data)
- OLAP (online analytical processing)
  - Typical applications: business intelligence, data mining
  - Back-end processing: batch workloads, less concurrency
  - Tasks: complex analytical queries, often ad hoc
  - Data access pattern: table scans, large amounts of data involved per query

OLTP

- Most database operations involve *On-Line Transaction Processing* (OTLP).
  - Short, simple, frequent queries and/or modifications, each involving a small number of tuples.
  - Examples: Answering queries from a Web interface, sales at cash registers, selling airline tickets.

OLAP

- Of increasing importance are *On-Line Application Processing* (OLAP) queries.
  - Few, but complex queries --- may run for hours.
  - Queries do not depend on having an absolutely up-to-date database.

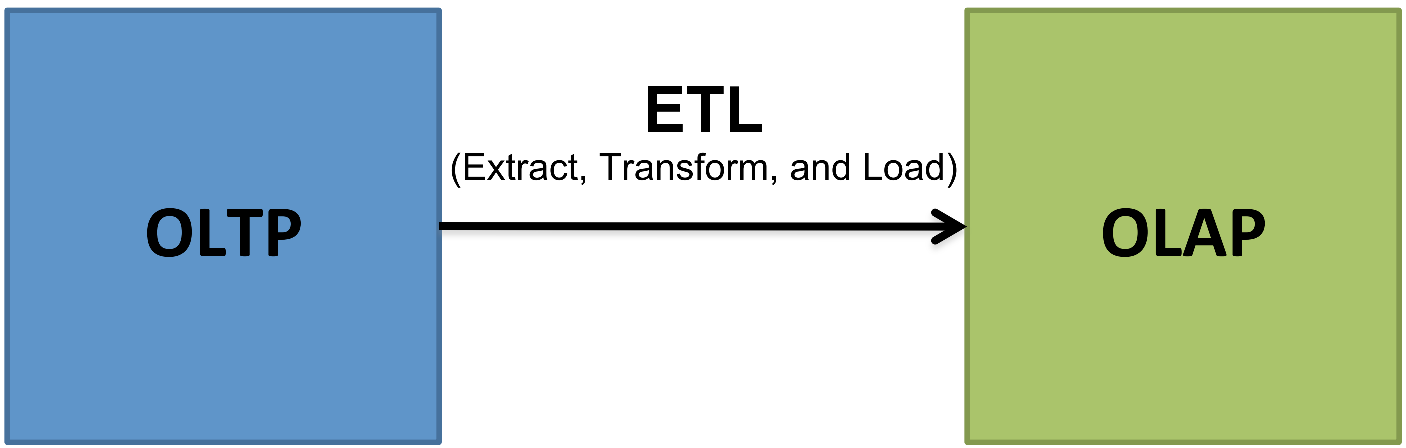
OLAP Examples

1. Amazon analyzes purchases by its customers to come up with an individual screen with products of likely interest to the customer.
2. Analysts at Wal-Mart look for items with increasing sales in some region.

One Database or Two?

- Downsides of co-existing OLTP and OLAP workloads
  - Poor memory management
  - Conflicting data access patterns
  - Variable latency
- Solution: separate databases
  - User-facing OLTP database for high-volume transactions
  - Data warehouse for OLAP workloads
  - How do we connect the two?

OLTP/OLAP Architecture



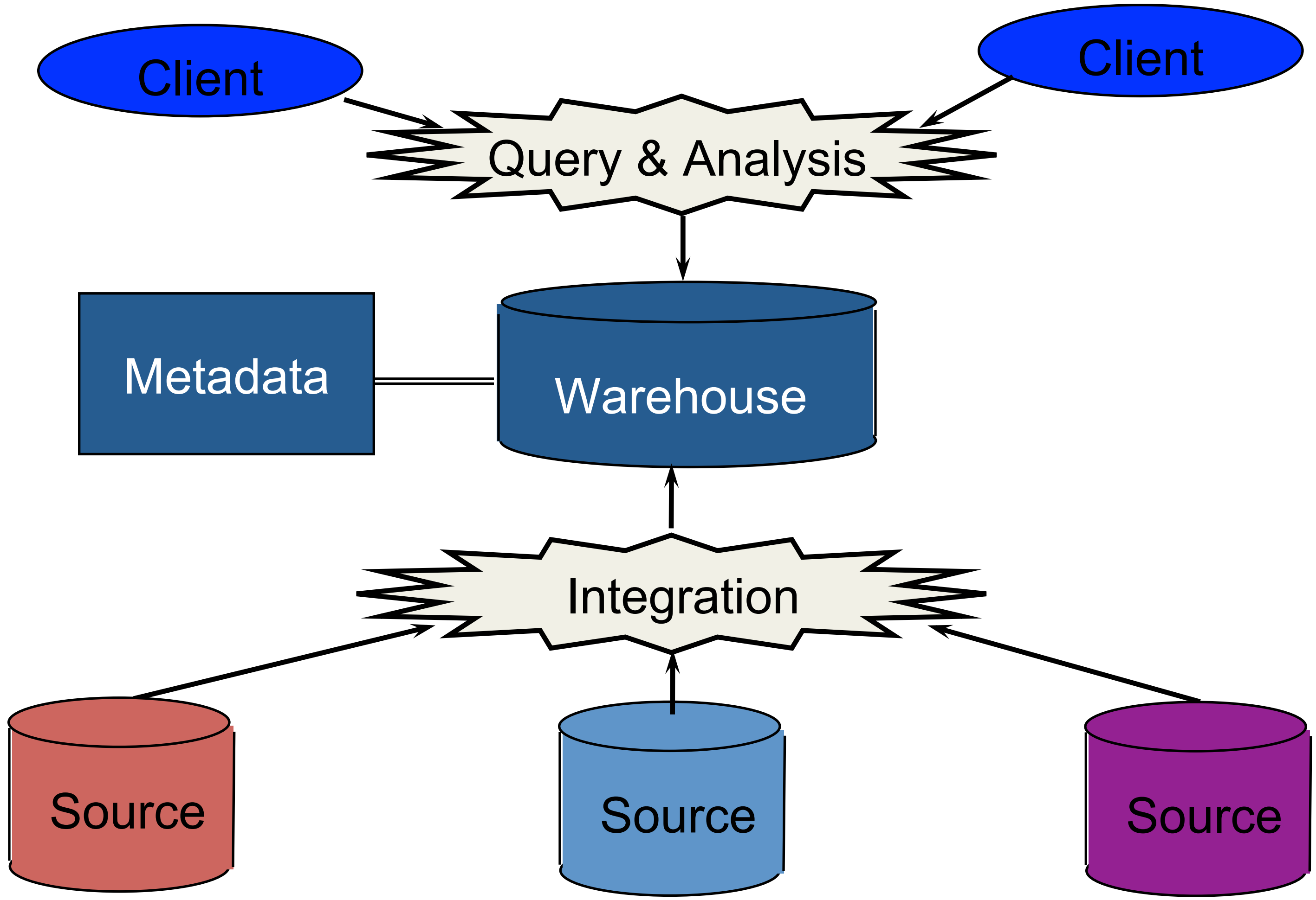
OLTP/OLAP Integration

- OLTP database for user-facing transactions
  - Retain records of all activity
  - Periodic ETL (e.g., nightly)
- Extract-Transform-Load (ETL)
  - Extract records from source
  - Transform: clean data, check integrity, aggregate, etc.
  - Load into OLAP database
- OLAP database for data warehousing
  - Business intelligence: reporting, ad hoc queries, data mining, etc.
  - Feedback to improve OLTP services

The Data Warehouse

- The most common form of data integration.
  - Copy sources into a single DB (*warehouse*) and try to keep it up-to-date.
  - Usual method: periodic reconstruction of the warehouse, perhaps overnight.
  - Frequently essential for analytic queries.

Warehouse Architecture



Star Schemas

- A *star schema* is a common organization for data at a warehouse. It consists of:
  1. *Fact table* : a very large accumulation of facts such as sales.
     - Often "insert-only."
  2. *Dimension tables* : smaller, generally static information about the entities involved in the facts.

Example: Star Schema

- Suppose we want to record in a warehouse information about every beer sale: the bar, the brand of beer, the drinker who bought the beer, the day, the time, and the price charged.
- The fact table is a relation:

  Sales(bar, beer, drinker, day, time, price)

Example, Continued

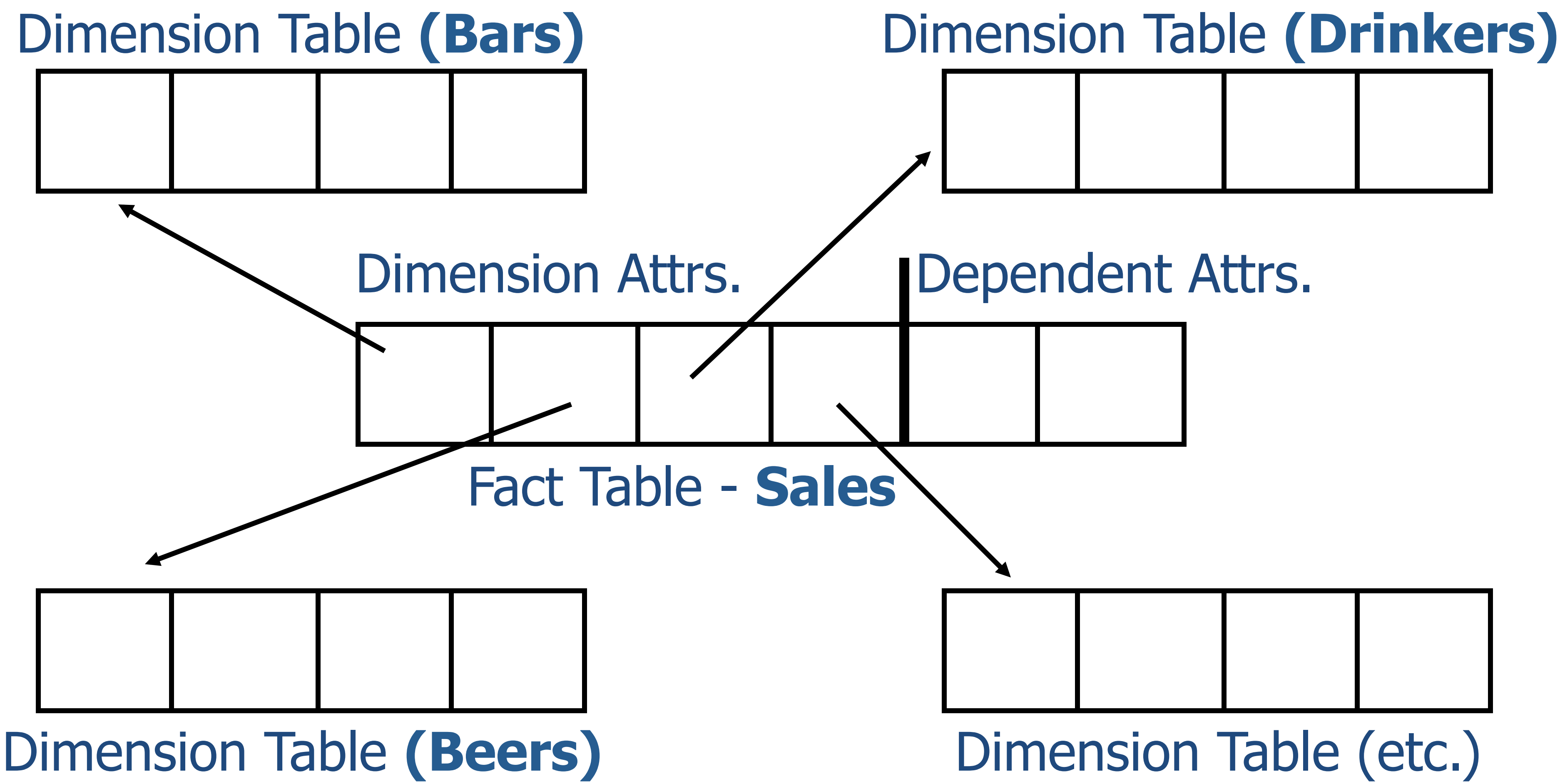
- The dimension tables include information about the bar, beer, and drinker “dimensions”:

  Bars(bar, addr, license)

  Beers(beer, manf)

  Drinkers(drinker, addr, phone)

Visualization – Star Schema



Dimensions and Dependent Attributes

- Two classes of fact-table attributes:
    1. *Dimension attributes* : the key of a dimension table.
    2. *Dependent attributes* : a value determined by the dimension attributes of the tuple