

CS 6400 A

Database Systems Concepts and Design

Lecture 22

11/12/25

Case Study: AWS US-EAST-1 Service Disruption

The incidence that brought down canvas, gradescope, and many more

What happened



One line summary:

the main errors we are seeing are “Could not resolve host: dynamodb.us-east-1.amazonaws.com”

The problem with this particular failure is that lots of internal AWS services use DynamoDB to store their information, so many additional services and customer applications broke.

Source: <https://aws.amazon.com/message/101925/>

The root cause

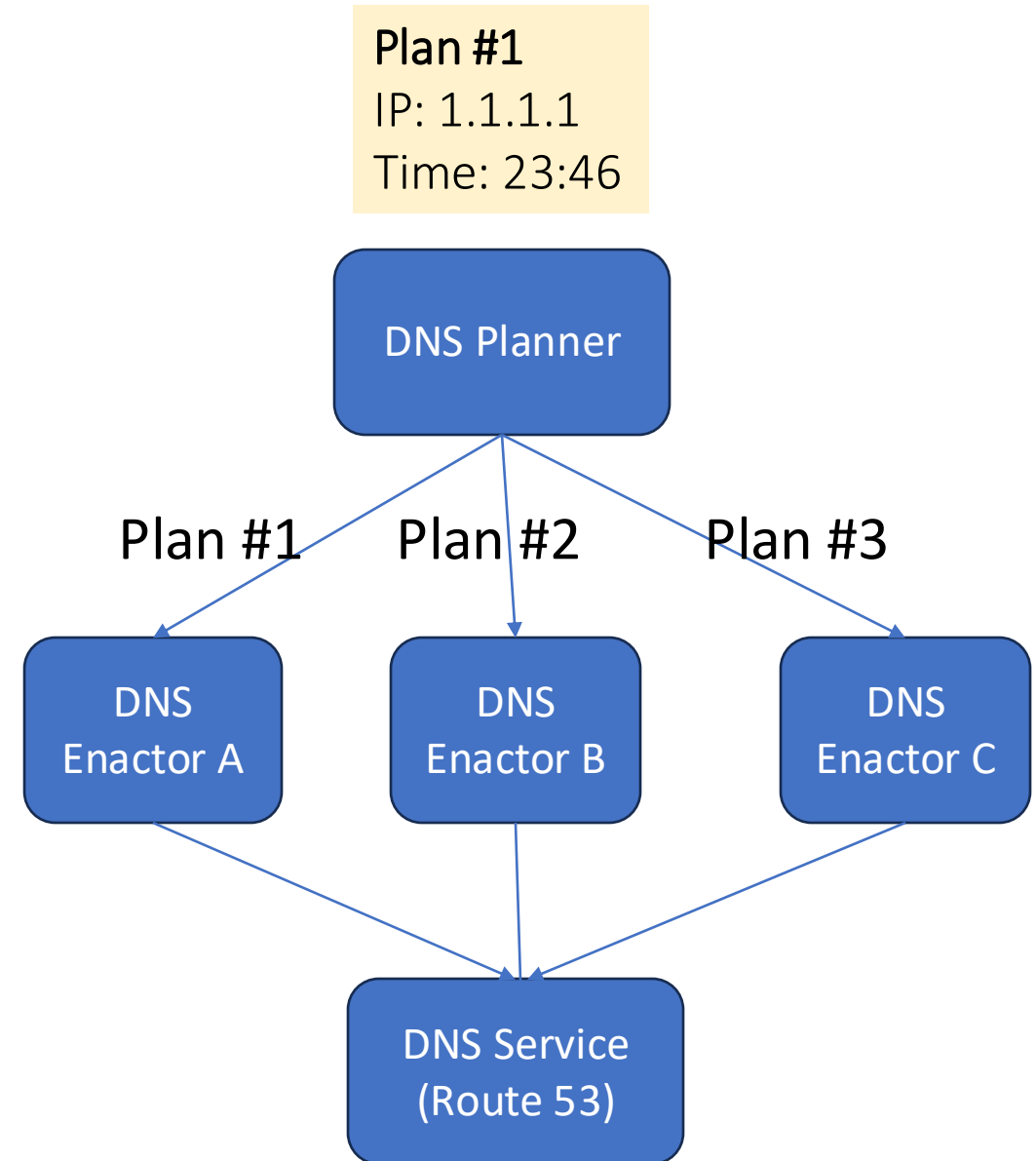
Context: DynamoDB database service uses a sophisticated automated system to manage its DNS records. DynamoDB's automation constantly updates these records to add capacity and handle failures.

The root cause of this issue was **a latent race condition** in the DynamoDB DNS management system that resulted in an incorrect empty DNS record for the service's regional endpoint (**dynamodb.us-east-1.amazonaws.com**) that the automation failed to repair.

Source: <https://aws.amazon.com/message/101925/>

DNS Management

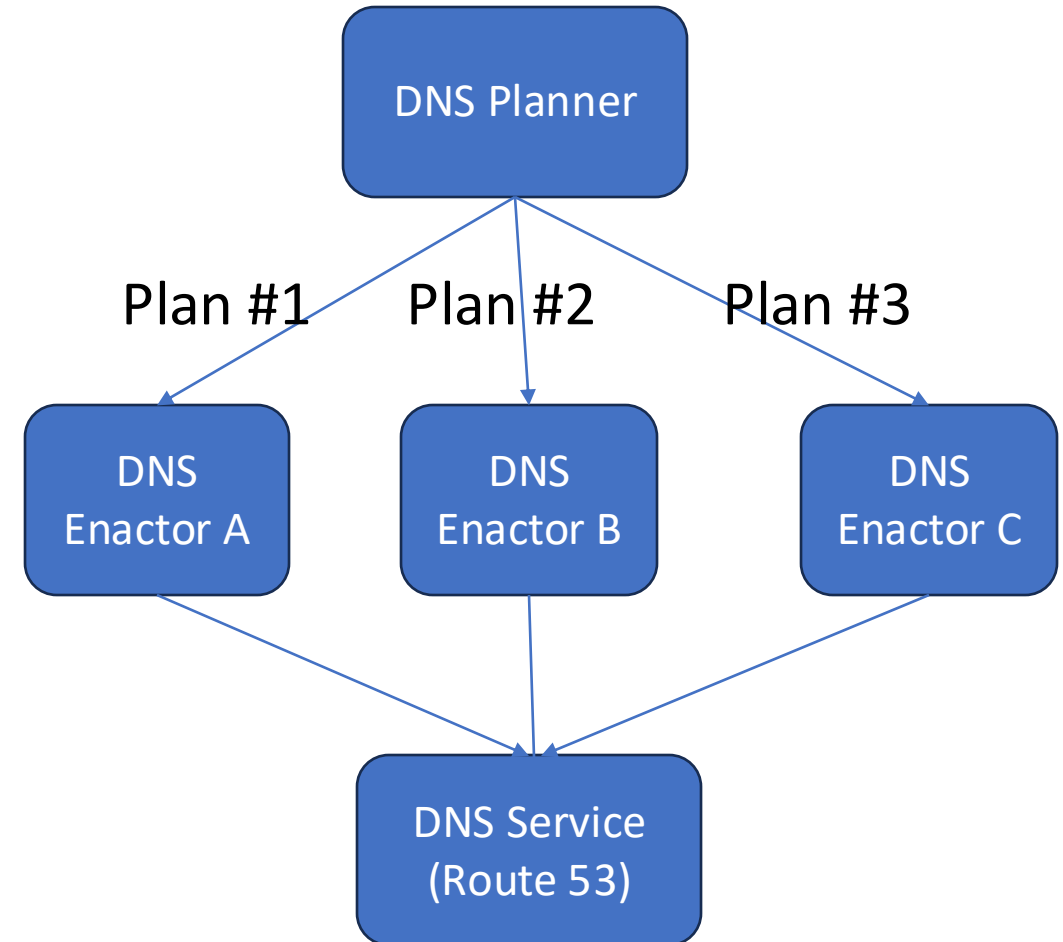
- **The DNS Planner:** This component monitored the health of load balancers and created a new “DNS plan,” a set of instructions for where to send traffic.
- **The DNS Enactor:** This component carried out the plan by updating the central DNS service.



DNS Management

DNS Enactor

0. Check if the current plan is more recent than the one in DNS service
1. Enacting plan (update DNS record)
2. Clean up older plans



DNS Planner

Plan #1

IP: 1.1.1.1

Time: 23:46

DNS
Enactor A

1. Enact Plan

Delayed

DNS
Enactor B

DNS
Enactor C



DNS Planner

Plan #2
IP: 2.2.2.2
Time: 23:47

Plan #1
IP: 1.1.1.1
Time: 23:46

DNS
Enactor A

1. Enact Plan

DNS
Enactor B

1. Enact Plan

2. Clean Up

DNS
Enactor C



Domain Name	IP	Time
dynamodb.us-east-1.amazonaws.com	2.2.2.2	23:47

DNS Planner

DNS
Enactor A

1. Enact Plan

Plan #1
IP: 1.1.1.1
Time: 23:46

DNS
Enactor B

2. Clean Up

Plan #2
IP: 2.2.2.2
Time: 23:47

DNS
Enactor C



Domain Name	IP	Time
dynamodb.us-east-1.amazonaws.com	2.2.2.2	23:47



Domain Name	IP	Time
dynamodb.us-east-1.amazonaws.com	1.1.1.1	23:46



Domain Name	IP	Time
dynamodb.us-east-1.amazonaws.com		

DNS Planner

Plan #3
IP: 3.3.3.3
Time: 23:48

DNS
Enactor A

DNS
Enactor B

DNS
Enactor C

0. Check Current



Domain Name	IP	Time
dynamodb.us-east-1.amazonaws.com	2.2.2.2	23:47



Domain Name	IP	Time
dynamodb.us-east-1.amazonaws.com	1.1.1.1	23:46



Domain Name	IP	Time
dynamodb.us-east-1.amazonaws.com		

DNS Planner

Plan #4
IP: ...

DNS
Enactor A

0. Check Current



Domain Name	IP	Time
dynamodb.us-east-1.amazonaws.com	2.2.2.2	23:47



Plan #5
IP: ...

DNS
Enactor B

0. Check Current



Domain Name	IP	Time
dynamodb.us-east-1.amazonaws.com	1.1.1.1	23:46



Plan #6
IP: ...

DNS
Enactor C

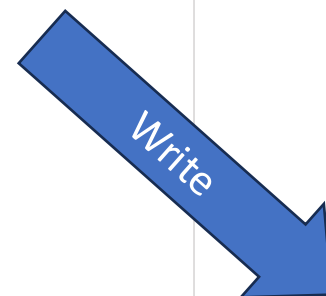
0. Check Current



Domain Name	IP	Time
dynamodb.us-east-1.amazonaws.com		

DNS Planner

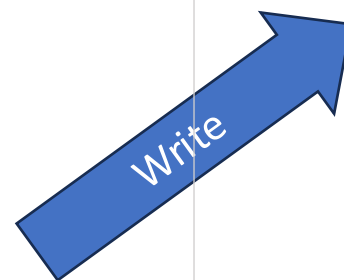
DNS
Enactor A



DNS
Enactor B



DNS
Enactor C



Domain Name	IP	Time
dynamodb.us-east-1.amazonaws.com		

The Cascading Effect

Issue resolved after 15 hrs

- AWS EC2 (DWFM):
 - Relies on Dynamo for configuration management
 - Can not add new instances
 - Selective restarts
- Network manager
 - Can not keep up with sudden increase in demand from newly started EC2 instances
- Health checker failure on NLB
 - Before network configuration arrives to instances, load balancer believes the nodes are dead and remove them from DNS

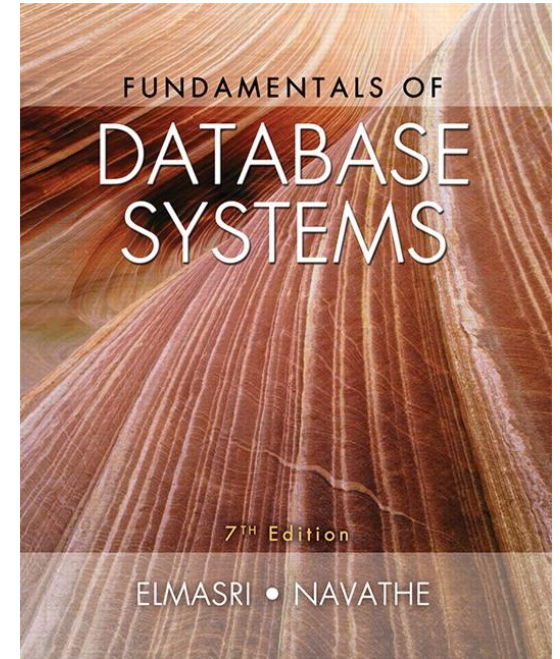
Agenda

1. OLTP vs OLAP
2. Data Cubes
3. HTAP

Reading Materials

Fundamental of Database Systems (7th Edition)

- Chapter 29 - Overview of Data Warehousing and OLAP



1. OLTP vs OLAP

So far we've been dealing with OLTP

OLTP: OnLine Transactional Processing

- Often used to store and manage relevant data to the day-to-day operations of a system or company
 - e.g., ATM transactions, online hotel bookings
- INSERT, UPDATE, DELETE commands
- Handles real-time transactions (response times often in milliseconds)
- ACID properties are often important

This is where relational databases shine!

Ok, so what's OLAP?

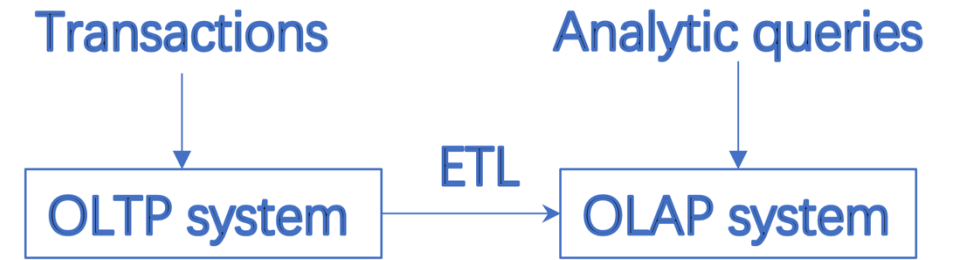
OLAP: OnLine Analytical Processing

- Also known as decision support or business intelligence (BI), but now BI has grown to include more (e.g., AI)
- A specialization of relational databases that prioritizes the reading and summarizing large volumes (TB, PB) of relational data to understand high-level trends and patterns
 - E.g., the total sales figures of each type of Honda car over time for each county
- “Read-only” queries

Contrast this to OLTP

- “Read-write” queries
- Usually touch a small amount of data
 - e.g., append a new car sale into the sales table

How is OLAP Performed?

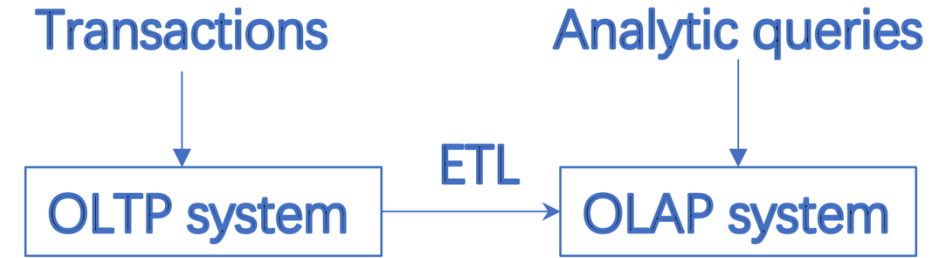


Usually, OLAP is performed on a **separate** data warehouse away from the critical path of OLTP transactions (a live/transactional database).

This data warehouse is periodically updated with data from various sources (e.g., once an hour or once a day)

- This is through a process of ETL (Extract, Transform, Load)
 - Extract useful business that needs to be summarized, transform it (e.g., canonicalize values, clean it up), load it in the data warehouse
- By doing it periodically, this data warehouse can become stale

How is OLAP Performed?



Usually, OLAP is performed on a **separate** data warehouse away from the critical path of OLTP transactions (a live/transactional database).

Why?

- Because OLAP queries end up reading most of the data, and will prevent OLTP queries from taking precedence
 - it is more important to ensure that sales are not prevented than to make sure that a report for a manager is generated promptly
 - the latter will anyway take a long time, so might as well have them wait a bit longer
- It's OK if the warehouse data is a bit stale.

OLAP in Data Warehouses

Here are some data warehouses that you might have heard of

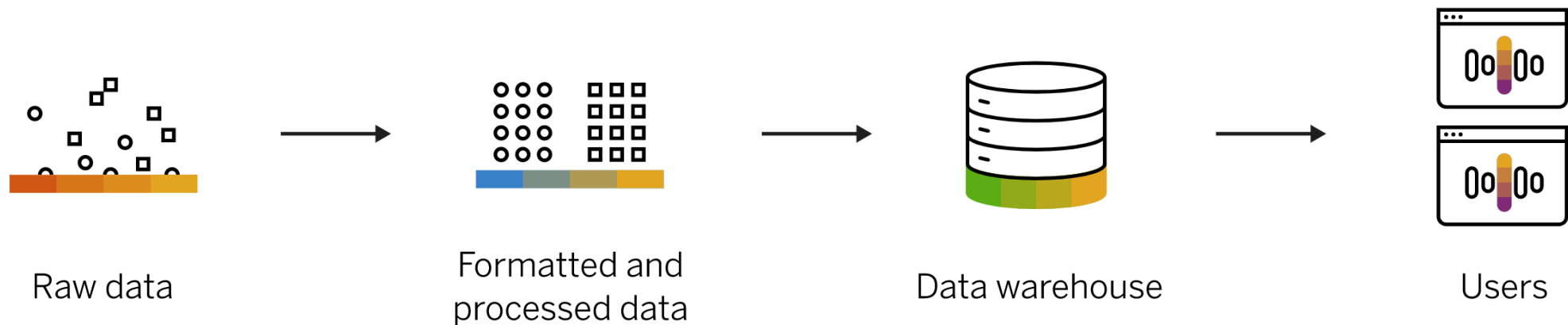


Data Warehouse vs Data Lake

Data warehouse:

structured data (schema-on-write)
expensive for large data volumes
managers and business analysts

Data warehouse



Data Lake vs Data Warehouse

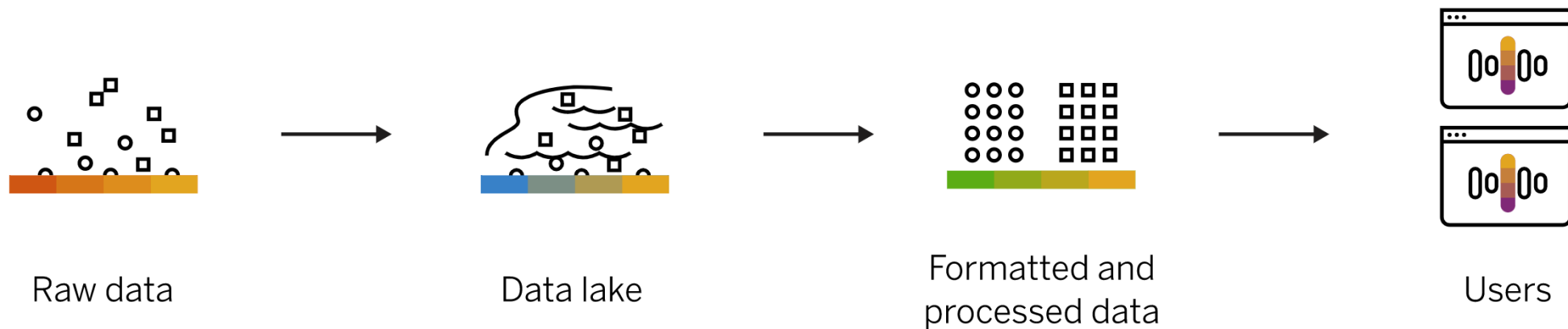
Data lake:

raw data, can be unstructured (schema-on-read)

low-cost storage, but no transactions, data quality checks

data scientists and engineers

Data lake



We will focus on the following two aspects of OLAP systems

#1 Data Model

- Relational vs multi-dimensional schema

#2 Storage Format

- Row vs column store

#1 Data Model: Multi-dimensional Model

The multi-dimensional data model includes two types of tables:

- **Fact table**

- Contain the actual metrics or measurements of business processes. Store quantitative data (numbers that can be aggregated).
 - E.g., Sales amount, Quantity sold
- Typically has many rows but fewer columns
- Usually contains foreign keys that link to dimension tables

- **Dimension table**

- Contain descriptive attributes that provide context to facts
- Store qualitative data
 - E.g., Product dimension (name, category, brand), Time dimension (date, month, year)

Multi-dimensional Schemas

Star schema:

- Consists of a fact table with a single table for each dimension.

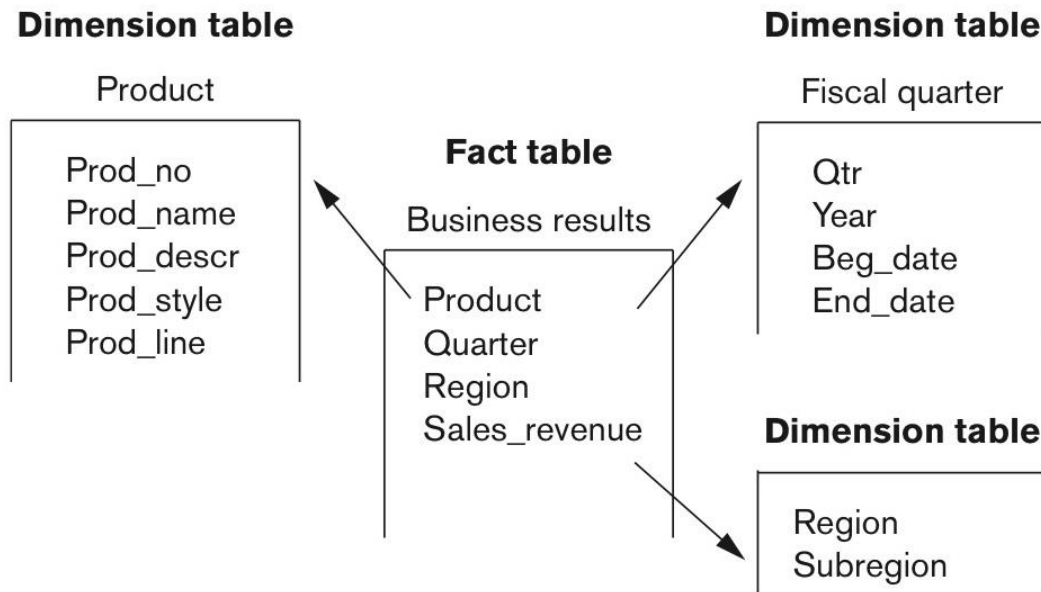


Figure 29.7 A star schema with fact and dimensional tables.

Multi-dimensional Schemas

Snowflake Schema:

- It is a variation of star schema, in which the dimensional tables from a star schema are normalized to eliminate redundancy

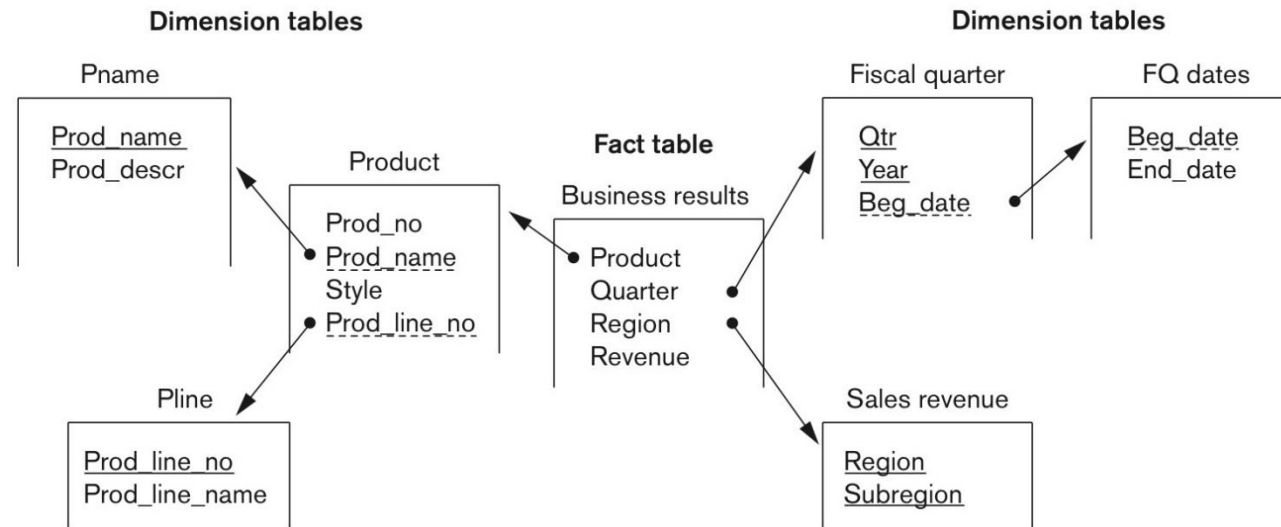


Figure 29.8 A snowflake schema.

Comparison with the Relational Model

The relational model (used by OLTP) keeps tables normalized

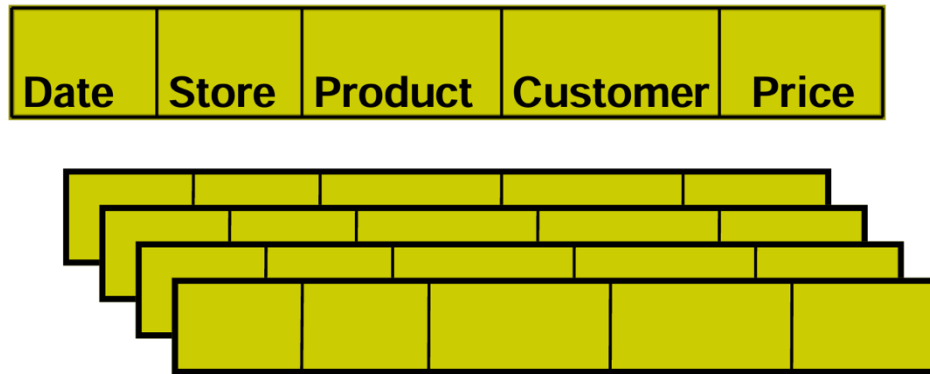
- Minimal updates required for data inserts and deletes => help improve transaction performance

In the multi-dimensional model, the fact table is often in 3NF, but the dimensional tables are denormalized.

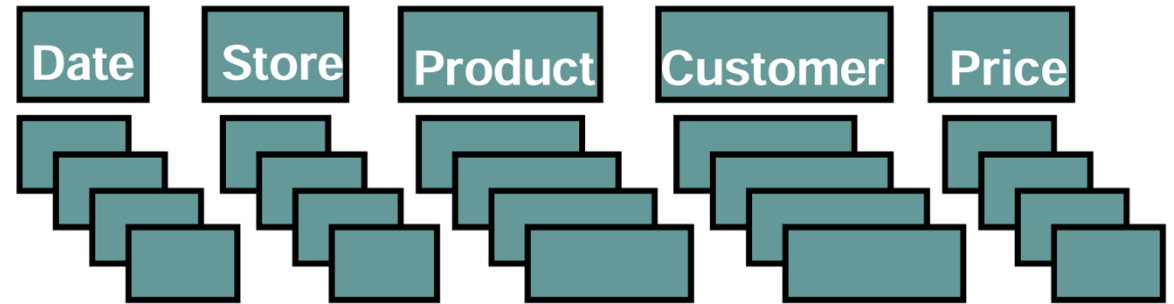
- This means columns in a table contains data which is repeated throughout the table => help improve read performance.
- Data is stored in fewer tables, which removes the overhead of having to perform complex joins => helps improve read performance.

#2 Storage Format: Row vs Column Store

row-store



column-store

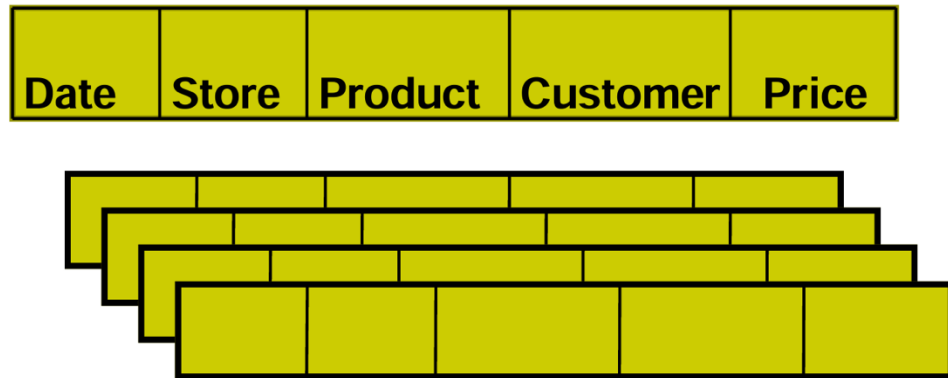


- Store all attributes of a tuple together
- Storage like “row-major order” in a matrix
- Store all rows for an attribute together
- Storage like “column-major order” in a matrix

Q: Which format do you think is better suited for OLTP vs OLAP?

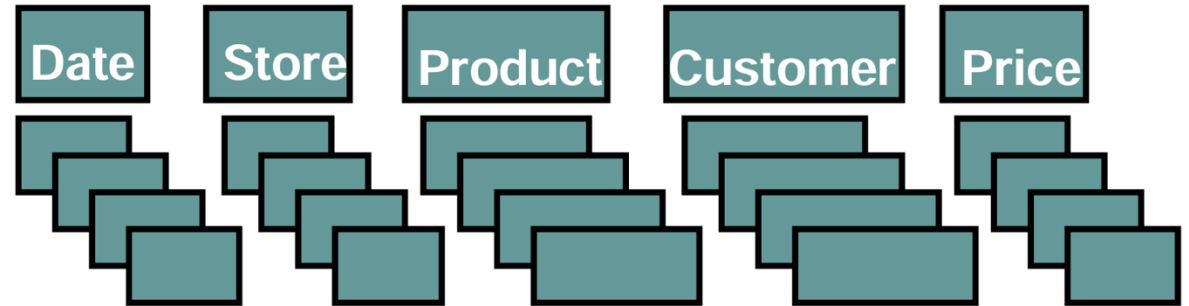
#2 Storage Format: Row vs Column Store

row-store



- + easy to add/modify a record
- need to read unnecessary data

column-store



- + only need to read in relevant data
- tuple write require multiple accesses

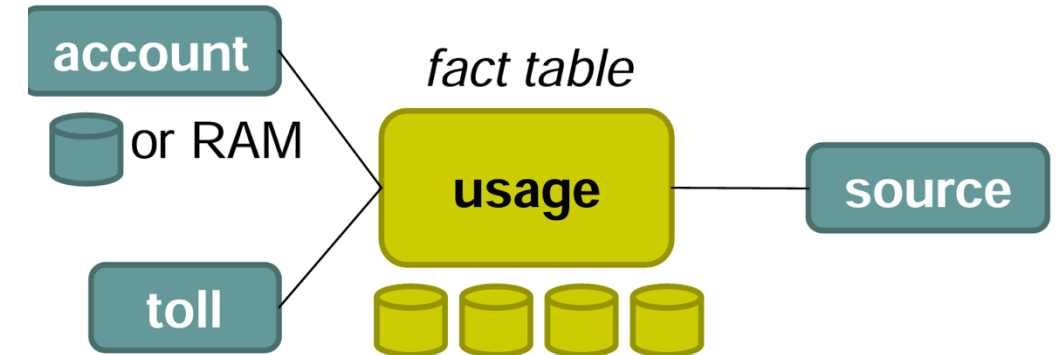
Telco Data Warehousing example

“One Size Fits All? - Part 2: Benchmarking Results”
Stonebraker et al. CIDR 2007

QUERY 2

```
SELECT account.account_number,  
       sum (usage.toll_airtime),  
       sum (usage.toll_price)  
FROM usage, toll, source, account  
WHERE usage.toll_id = toll.toll_id  
AND usage.source_id = source.source_id  
AND usage.account_id = account.account_id  
AND toll.type_ind in ('AE', 'AA')  
AND usage.toll_price > 0  
AND source.type != 'CIBER'  
AND toll.rating_method = 'IS'  
AND usage.invoice_date = 20051013  
GROUP BY account.account_number
```

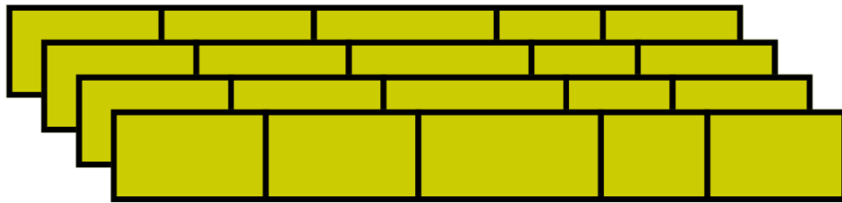
dimension tables



	<i>Column-store</i>	<i>Row-store</i>
<i>Query 1</i>	2.06	300
<i>Query 2</i>	2.20	300
<i>Query 3</i>	0.09	300
<i>Query 4</i>	5.24	300
<i>Query 5</i>	2.88	300

Telco example explained: read efficiency

row store

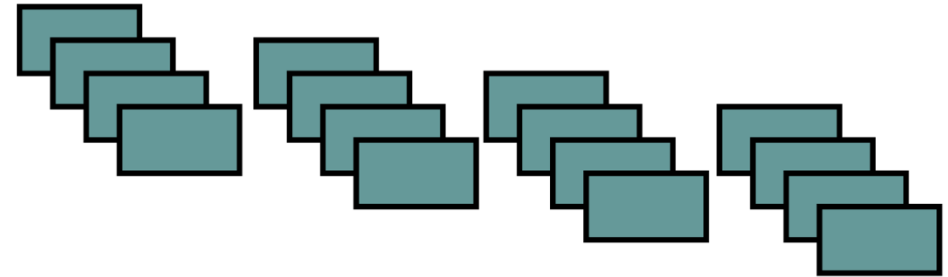


read pages containing entire rows

one row = 212 columns!

is this typical? (it depends)

column store



read only columns needed

in this example: 7 columns

caveats:

- "select * " not any faster
- clever disk prefetching
- clever tuple reconstruction

Telco example explained: compression efficiency

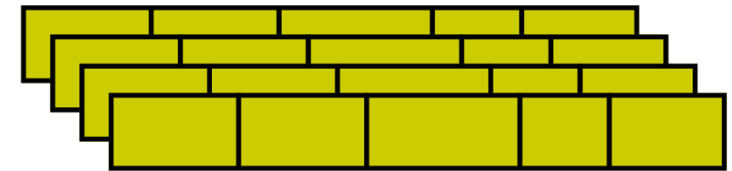
Columns compress better than rows

- Typical row-store compression ratio 1 : 3
- Column-store 1 : 10

Why?

- Rows contain values from different domains
 - More entropy, difficult to dense-pack
- Columns are much more homogeneous
- Caveat: CPU cost (use lightweight compression)

row store



column store



2. Data Cubes

Introducing Data Cubes

Dimensions

Item

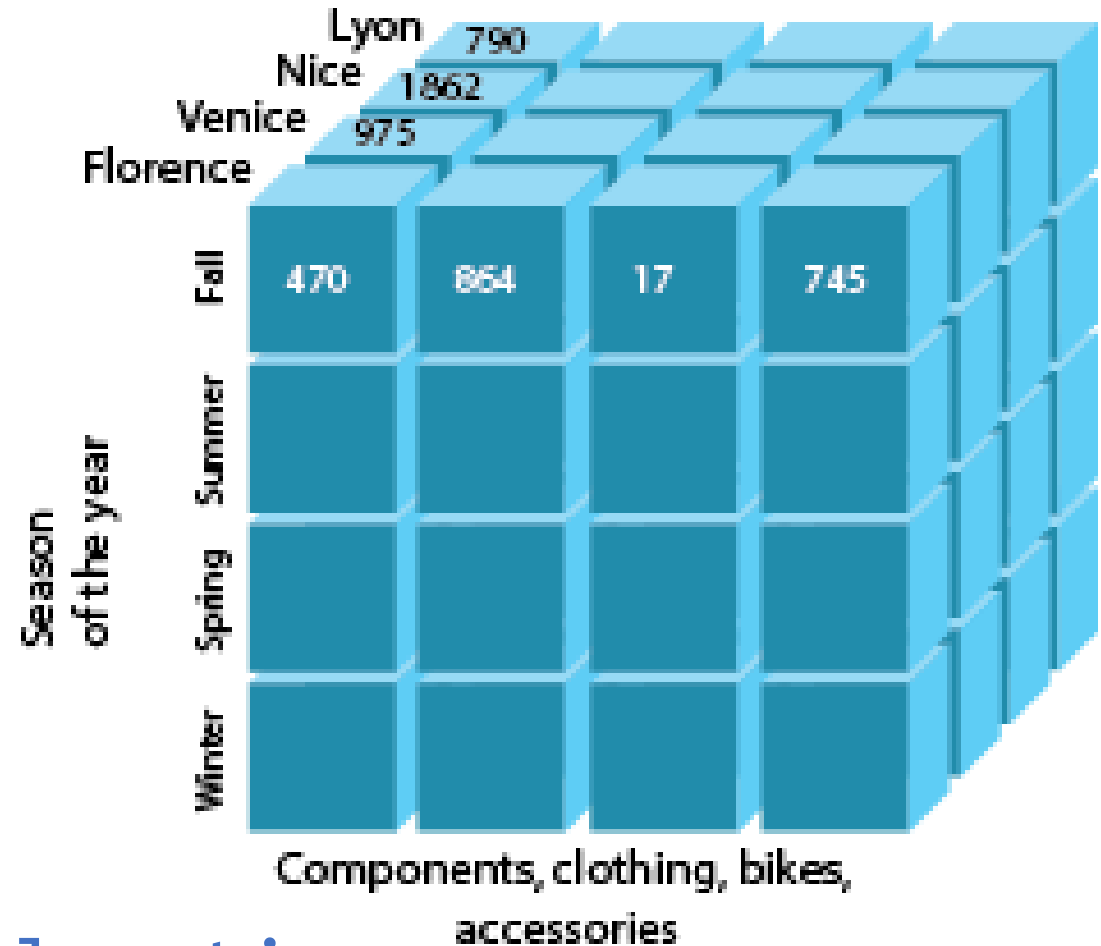
season

location

Measure

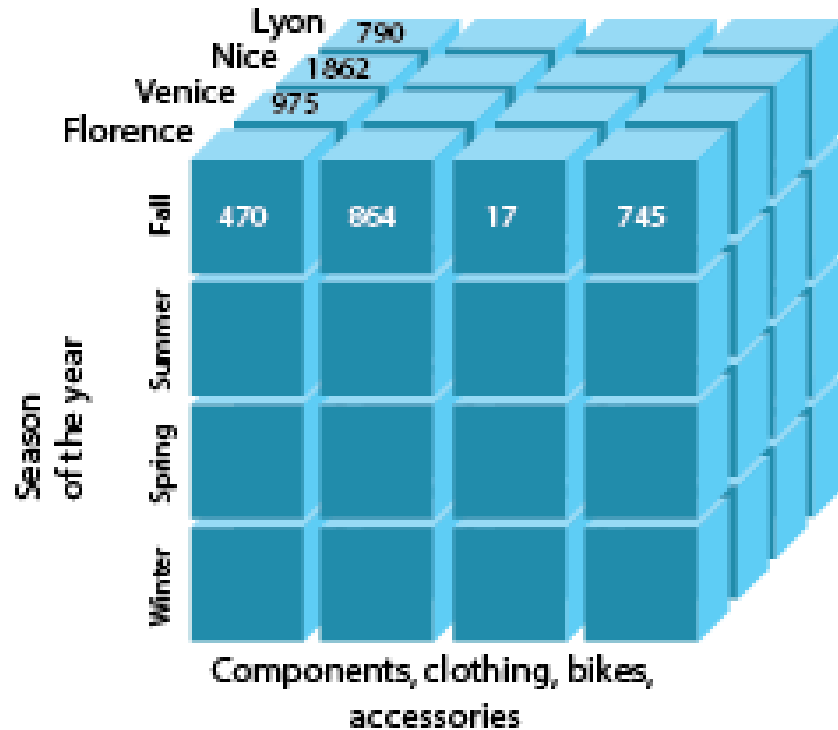
volume

```
SELECT SUM(volume) ...  
GROUP BY item, season, location
```

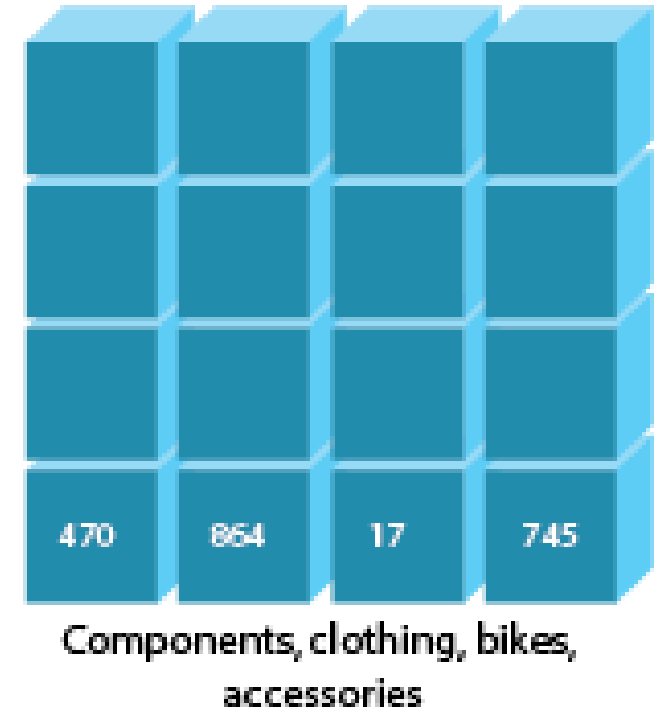
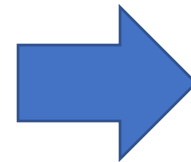


Operations: Slicing and Dicing

Slicing: select a single value in one of the dimensions to create a smaller cube with one less dimension

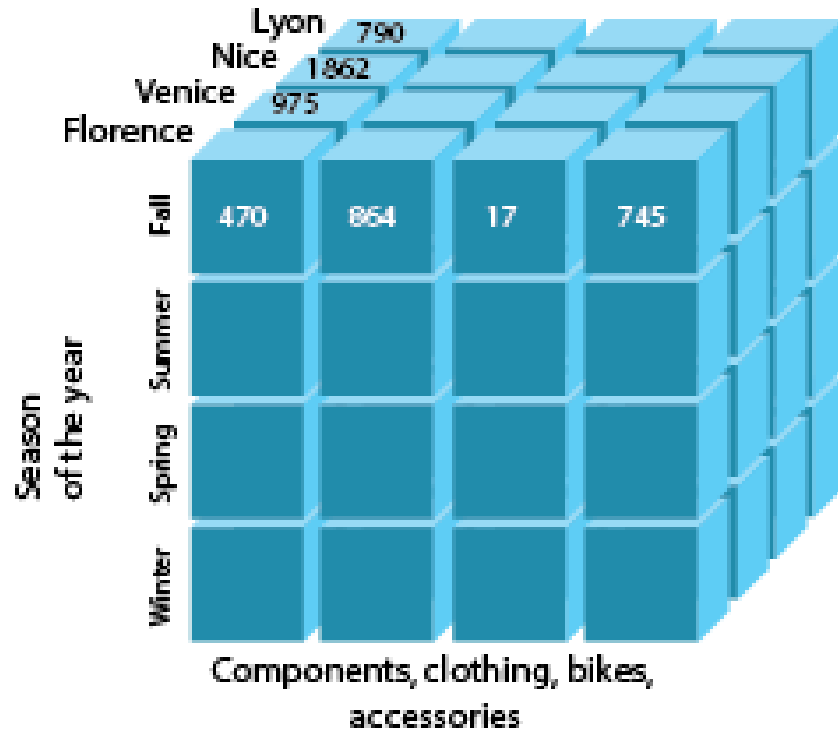


Slice for
(season = winter)

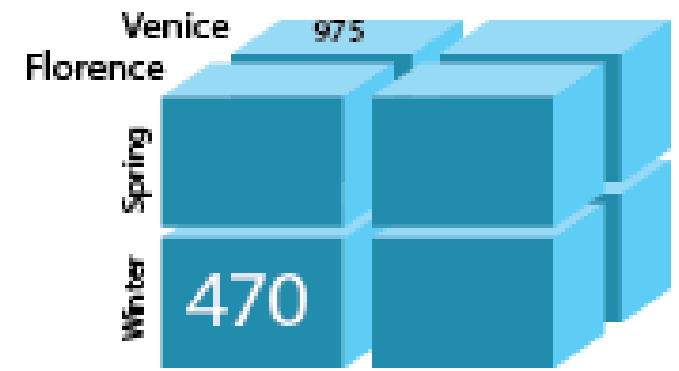


Operations: Slicing and Dicing

Dicing: select specific values of multiple dimensions to create a new subcube

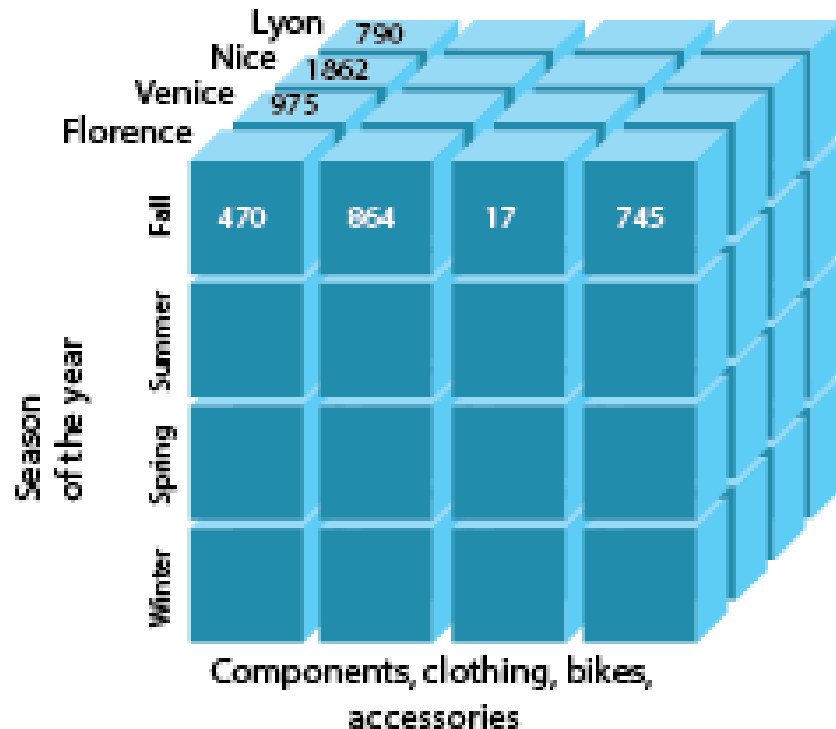


Dice for
(season = winter or spring)
and
(location = Venice or Florence)
and
(item = components or clothing)

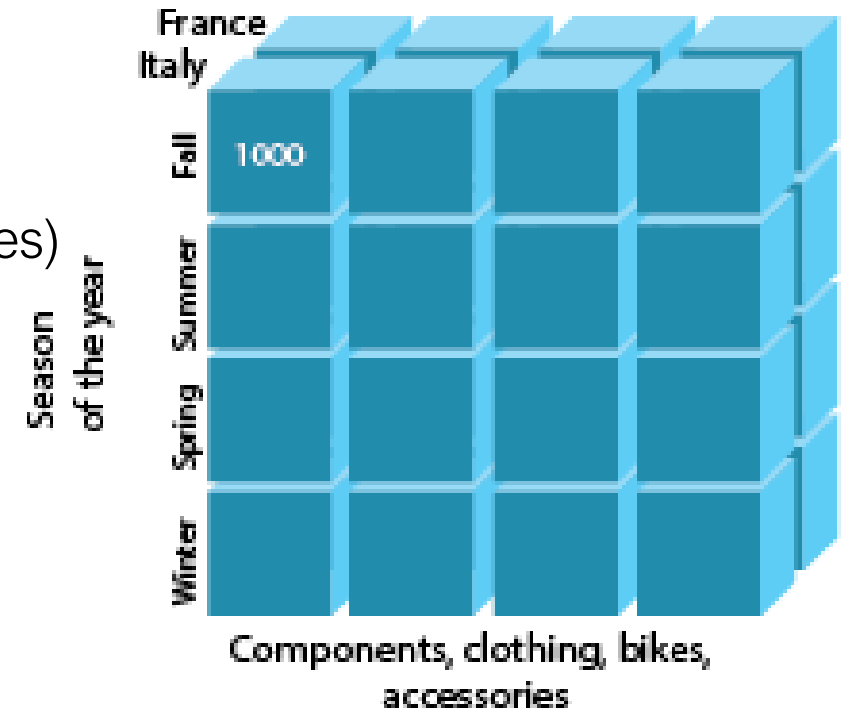
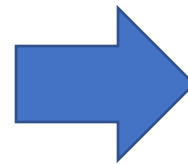


Operations: Roll up and drill down

Rollup: Moving from a finer to a coarser granularity

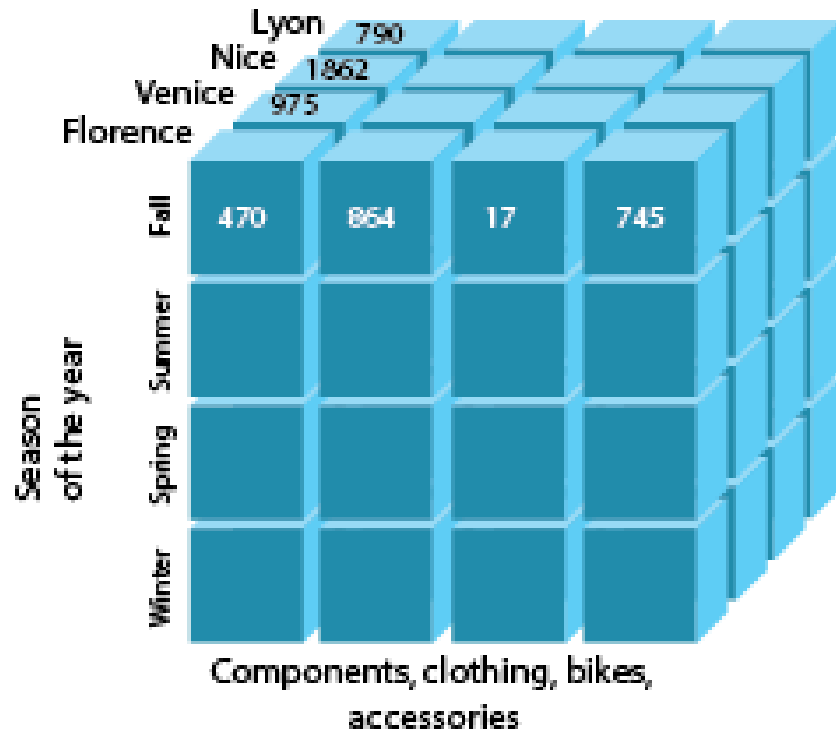


Roll up location
(from cities to countries)

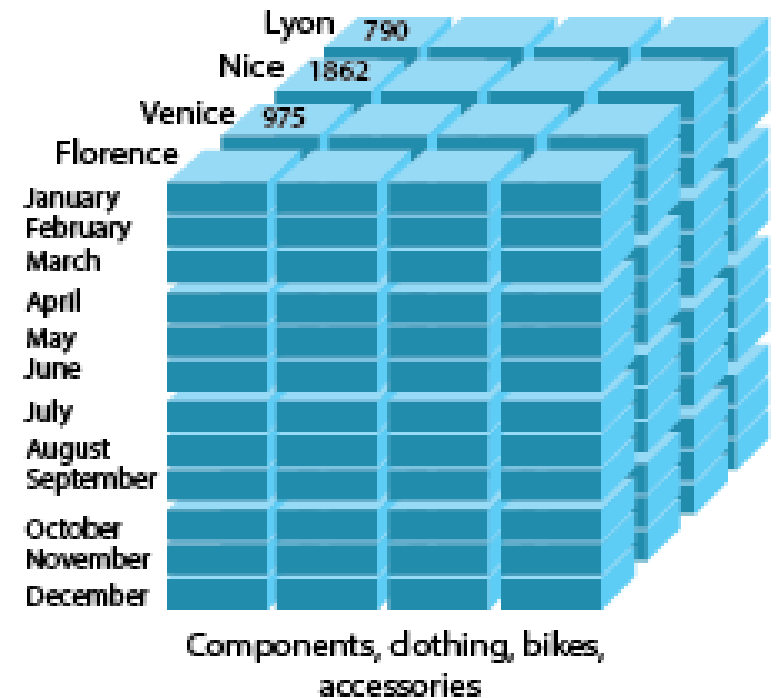
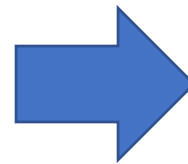


Operations: Roll up and drill down

Drill-down: Moving from a coarser to a finer granularity



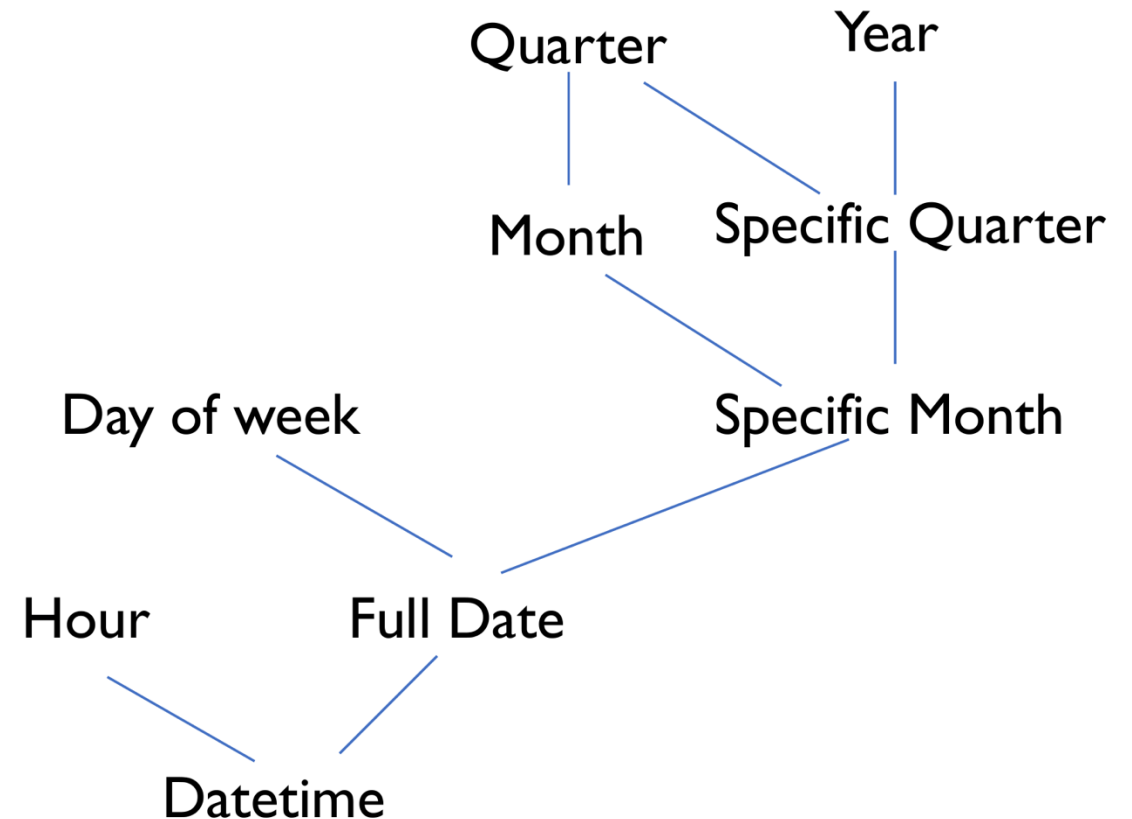
Drill down time
(from quarters to months)



Partitioning granularity in data cubes

Different partitioning may be useful for different applications

- Q: I want the aggregates per month. What if I set my partitioning based on Full Date and computed the data cube? Can I avoid recomputing the cube?
- Q: What about if I had set my partitioning based on Year?



How to pick the right partitioning granularity?

First, why does this matter?

- If this query is being run once on a petabyte sized warehouse, it is important to get it right!

Think about all the ways you want to slice and dice your data

Pick as fine a granularity of partitioning so that you can recreate all the aggregates, but not so fine as to blow up the query result

- The query result size grows exponentially in the attributes
- This is known as the curse of dimensionality

OLAP summary

OLAP is a specialization of relational databases to support analytical processing and report generation

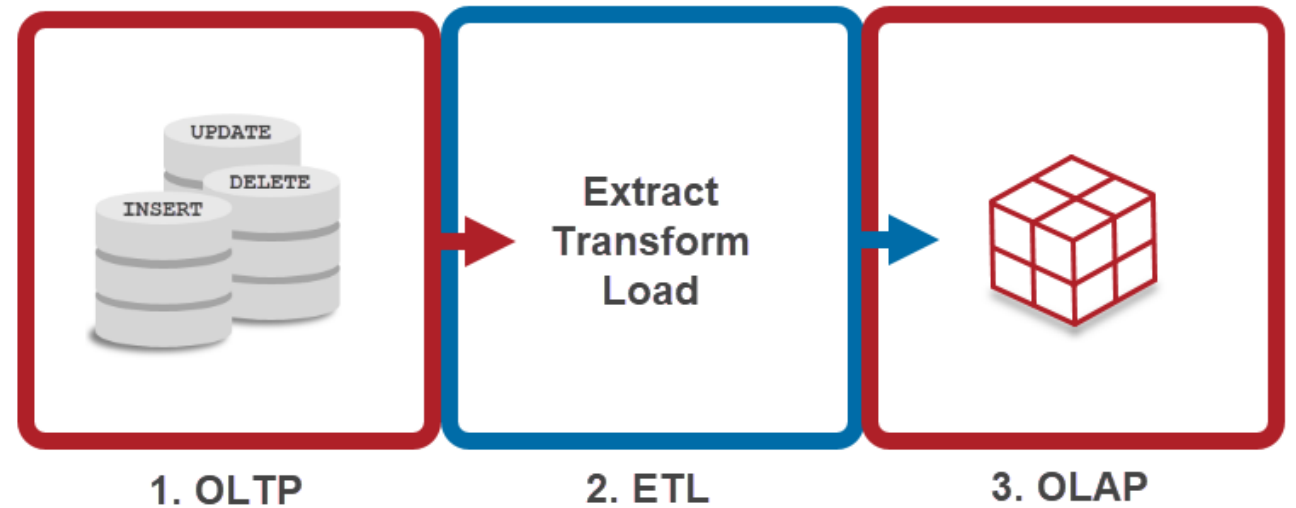
- Typically done in large “batch” operations on the entire database
- Rule of thumb: pick as “coarse-grained” materialized views or query results as will allow you to construct all the cross-tabs that may be necessary

The concepts of OLAP data cubes, hierarchies, slicing/dicing, and rollup/drilldown are valuable to describe what you’re doing when you are exploring your data

OLAP summary

OLAP vs OLTP

- Data Model
- Storage Format



Data warehouses vs Data Lakes

Data cubes

- Slicing and dicing
- Rollup and drill-down
- Selecting partitioning granularity

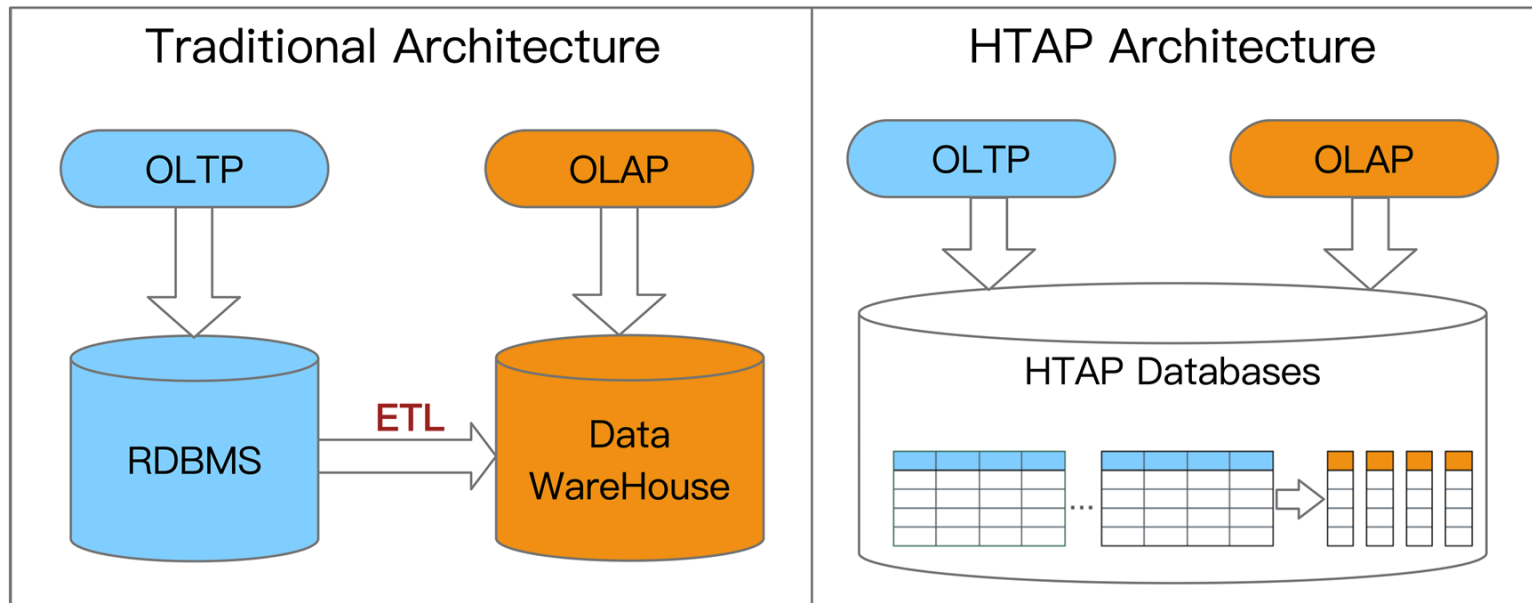
3. A brief intro to HTAP Databases

Slides adapted from *SIGMOD'22 Tutorial HTAP Databases: A Tutorial* by
Guoliang Li and Chao Zhang

Motivation

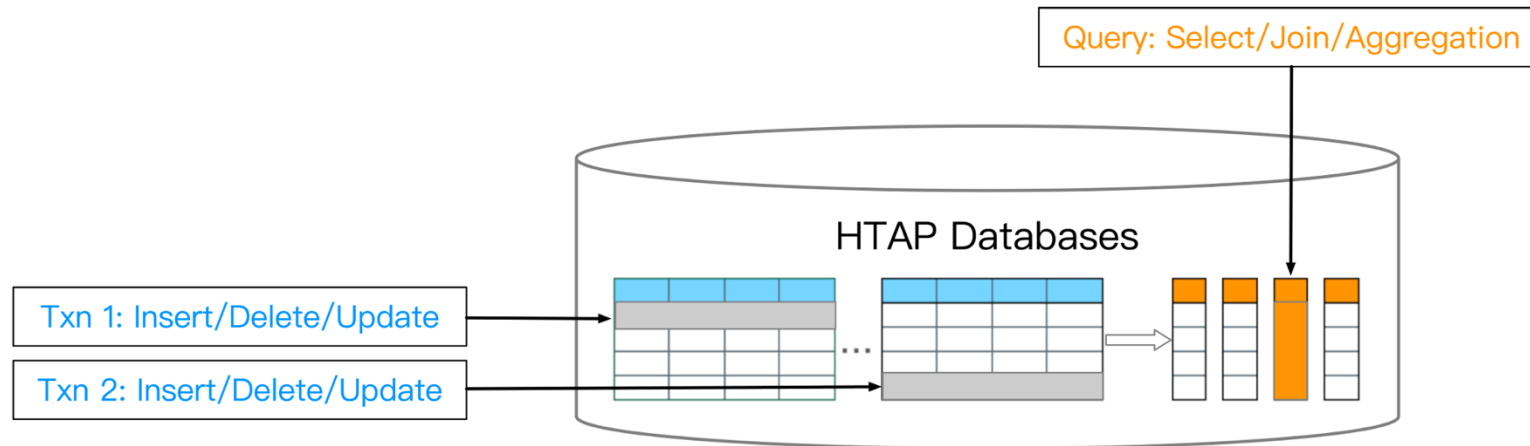
HTAP: Hybrid Transaction Analytical Processing

- Gartner's definition in 2018: supports weaving analytical and transaction processing techniques together as needed to accomplish the business task.



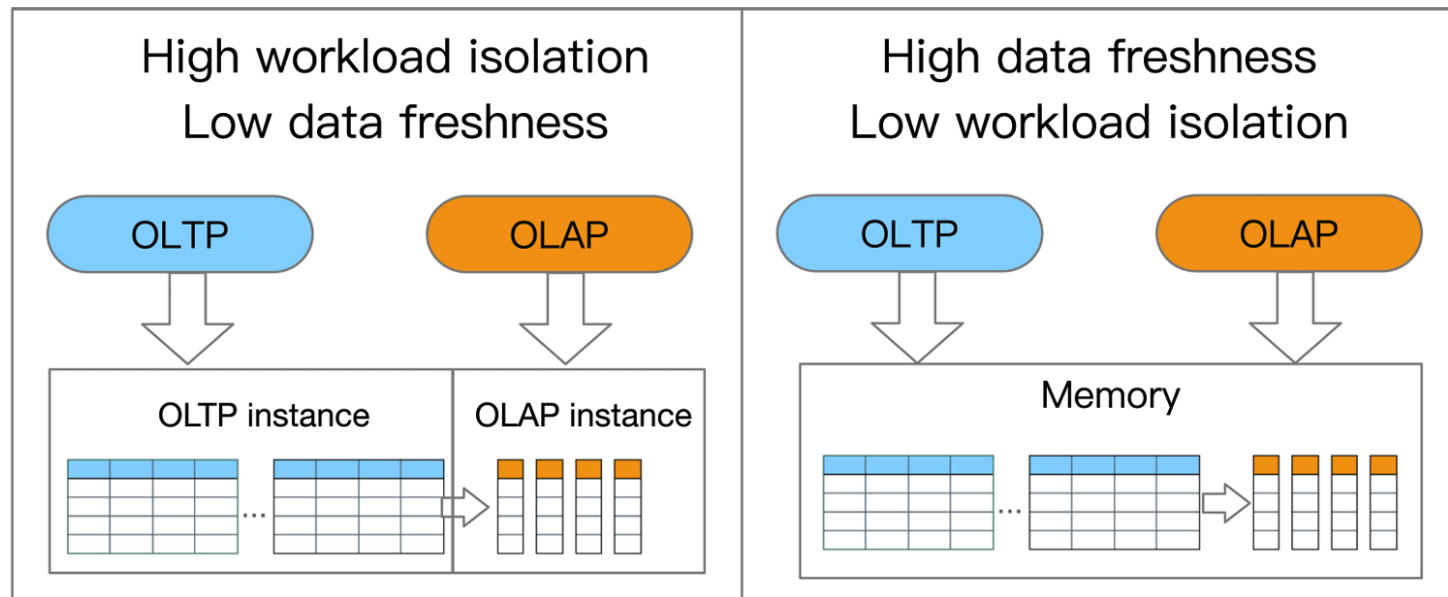
Motivation

- Rule of thumb 1: row store is ideal for OLTP workloads
 - Row-wise, update-heavy, short-lived transactions
- Rule of thumb 2: column store is best suited for OLAP workload
 - Column-wise, read-heavy, bandwidth-intensive queries



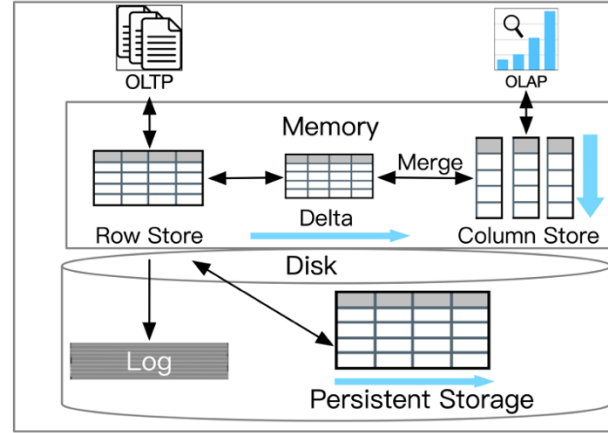
A trade-off for HTAP databases

- Workload isolation: the isolation level of handling the mixed workloads
- Data freshness: the portion of latest transaction data that is read by OLAP
- Trade-off for [workload isolation](#) and [data freshness](#)
 - High workload isolation leads to low data freshness
 - Low workload isolation results in high data freshness



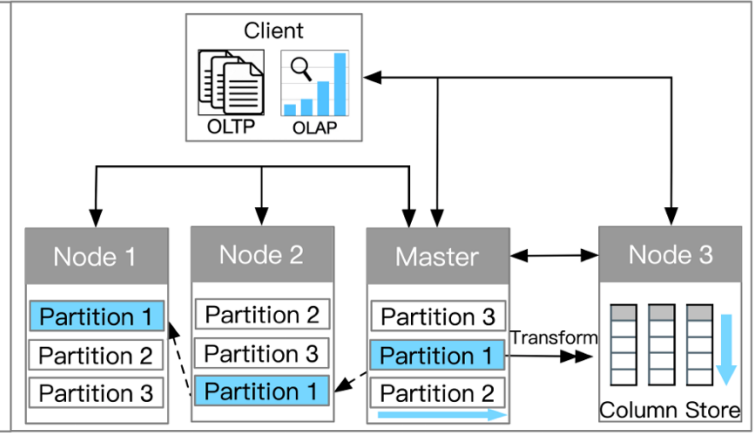
An Overview of HTAP Architectures

(a) **Primary Row Store + In-Memory Column Store**



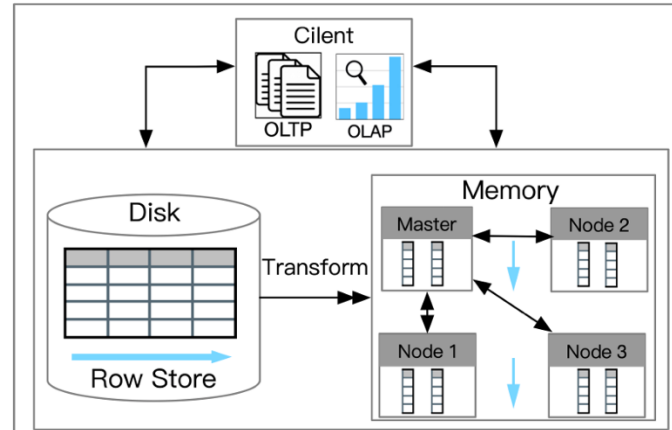
(a) Primary Row Store+In-Memory Column Store

(b) **Distributed Row Store + Column Store Replica**



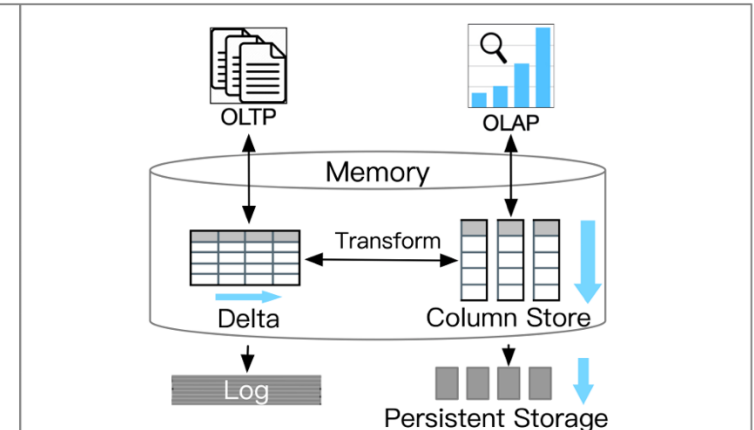
(b) Distributed Row Store+Column Store Replica

(c) **Disk Row Store + Distributed Column Store**



(c) Disk Row Store+Distributed Column Store

(d) **Primary Column Store + Delta Row Store**



(d) Primary Column Store+Delta Row Store

A summary of HTAP databases

Category	HTAP Databases	OLTP Throughput	OLAP Throughput	OLTP Scalability	OLAP Scalability	Workload Isolation	Data Freshness
Primary Row Store+ In Memory Column Store	Oracle Dual-Format SQL Server, DB2 BLU	High	High	Medium	Low	Low	High
Distributed Row Store + Column Store Replica	TiDB, F1 Lightning SingleStore	Medium	Medium	High	High	High	Low
Disk Row Store + Distributed Column Store	MySQL Heatwave, Oracle RAC	Medium	Medium	Medium	High	High	Medium
Primary Column Store + Delta Row Store	SAP HANA (without scale-out), Hyper	Medium	High	Low	Medium	Low	High

* Readings: [HTAP Databases: What is New and What is Next](#)