ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

1

ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

OLAP, Data Warehouse

2

Why we still study OLAP/Data Warehouse in BI?

- 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)

Let's get back to the root in 70's: Relational Database

Basic Structure

- Formally, given sets D_1, D_2, …. D_n a **relation** r is a subset of
  $D_1 \times D_2 \times \ldots \times D_n$
  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)

7

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

SOICT VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

8

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

In 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.

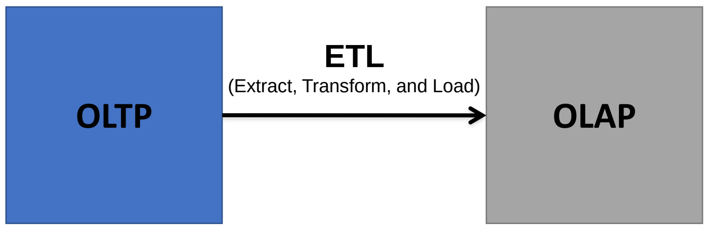
OLAP Examples

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



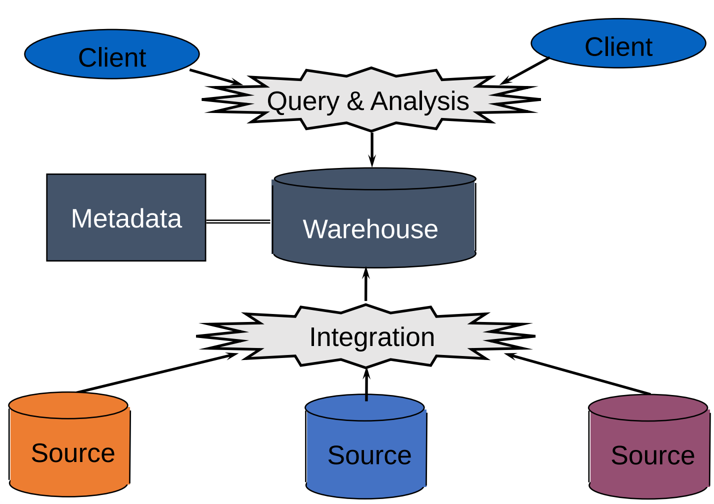
OLTP/OLAP Integration

- 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

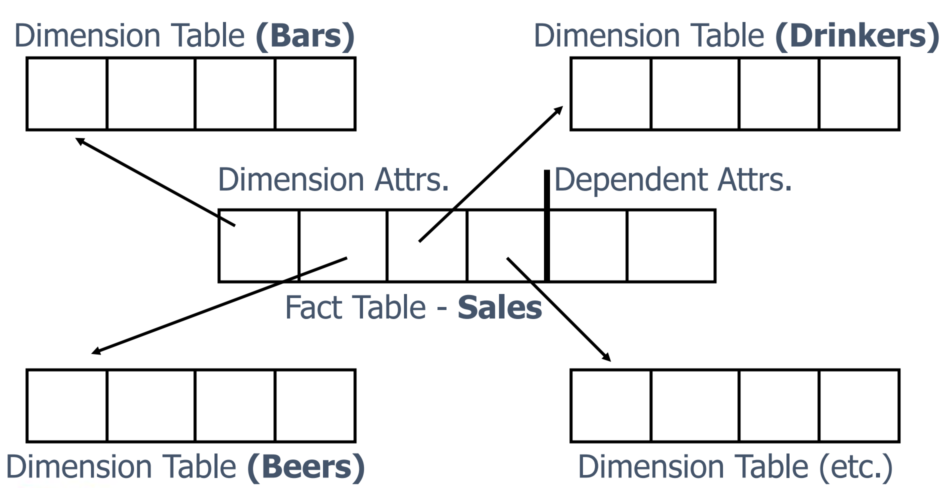
- The dimension tables include information about the bar, beer, and drinker "dimensions":

  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.

Warehouse Models & Operators

- Data Models
  - relations
  - stars & snowflakes
  - cubes
- Operators
  - slice & dice
  - roll-up, drill down
  - pivoting
  - other

Star

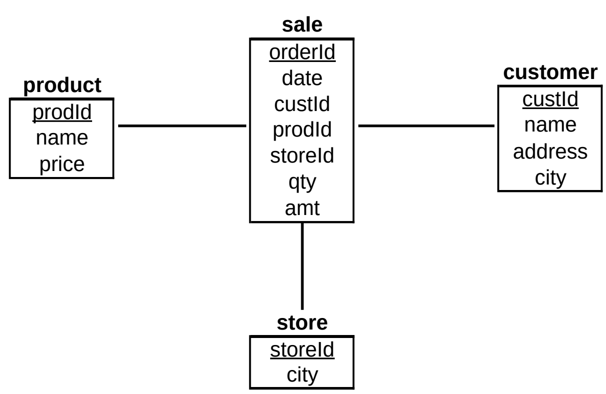
product	prodId	name	price
	p1	bolt	10
	p2	nut	5

store	storeId	city
	c1	nyc
	c2	sfo
	c3	la

sale	oderId	date	custId	prodId	storeId	qty	amt
	o100	1/7/97	53	p1	c1	1	12
	o102	2/7/97	53	p2	c1	2	11
	105	3/8/97	111	p1	c3	5	50

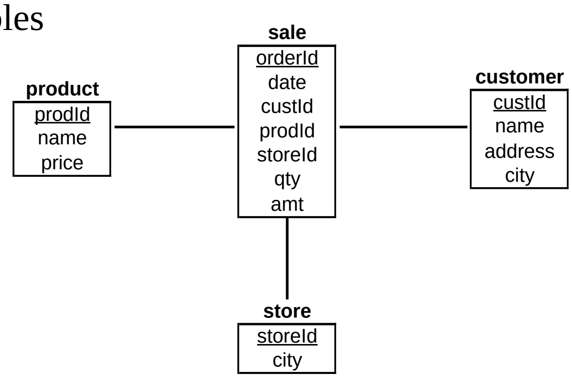
customer	custId	name	address	city
	53	joe	10 main	sfo
	81	fred	12 main	sfo
	111	sally	80 willow	la

Star Schema

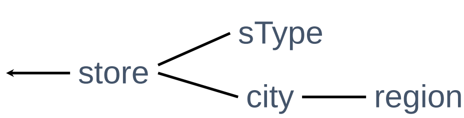


Terms

- Fact table
- Dimension tables
- Measures



Dimension Hierarchies



sType	tId	size	location
	t1	small	downtown
	t2	large	suburbs

store	storeId	cityId	tId	mgr
	s5	sfo	t1	joe
	s7	sfo	t2	fred
	s9	la	t1	nancy

city	cityId	pop	regId
	sfo	1M	north
	la	5M	south

region	regId	name
	north	cold region
	south	warm region

➔ snowflake schema
➔ constellations

Aggregates

- Add up amounts for day 1
- In SQL: SELECT sum(amt) FROM SALE
  WHERE date = 1

sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4

81

Aggregates

- Add up amounts by day
- In SQL: SELECT date, sum(amt) FROM SALE
  GROUP BY date

sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4

ans	date	sum
	1	81
	2	48

Another Example

- Add up amounts by day, product
- In SQL: SELECT date, sum(amt) FROM SALE
  GROUP BY date, prodId

sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4

sale	prodId	date	amt
	p1	1	62
	p2	1	19
	p1	2	48

rollup →

← drill-down

ROLAP vs. MOLAP

- ROLAP:
  Relational On-Line Analytical Processing
- MOLAP:
  Multi-Dimensional On-Line Analytical Processing

Cube

Fact table view:

sale	prodId	storeId	amt
	p1	c1	12
	p2	c1	11
	p1	c3	50
	p2	c2	8

↔

Multi-dimensional cube:

	c1	c2	c3
p1	12		50
p2	11	8	

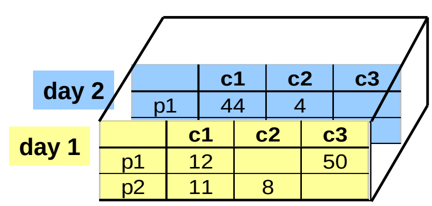
dimensions = 2

3-D Cube

Fact table view:

sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4

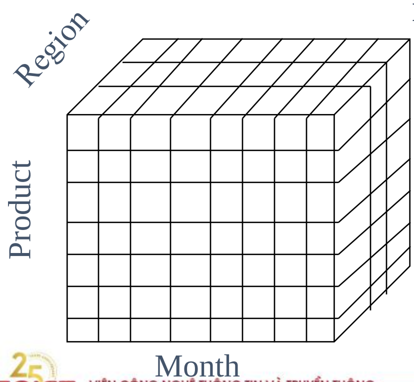
Multi-dimensional cube:



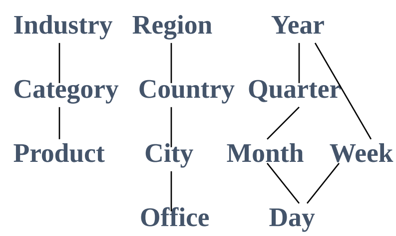
dimensions = 3

Multidimensional Data

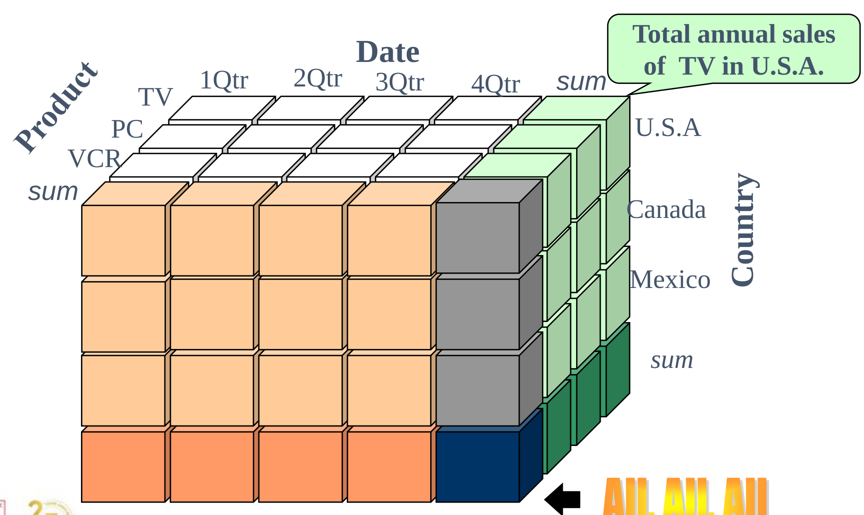
- Sales volume as a function of product, month, and region



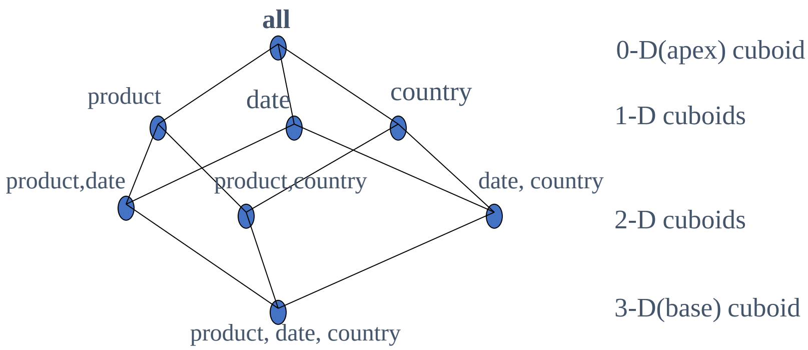
Dimensions: Product, Location, Time
Hierarchical summarization paths



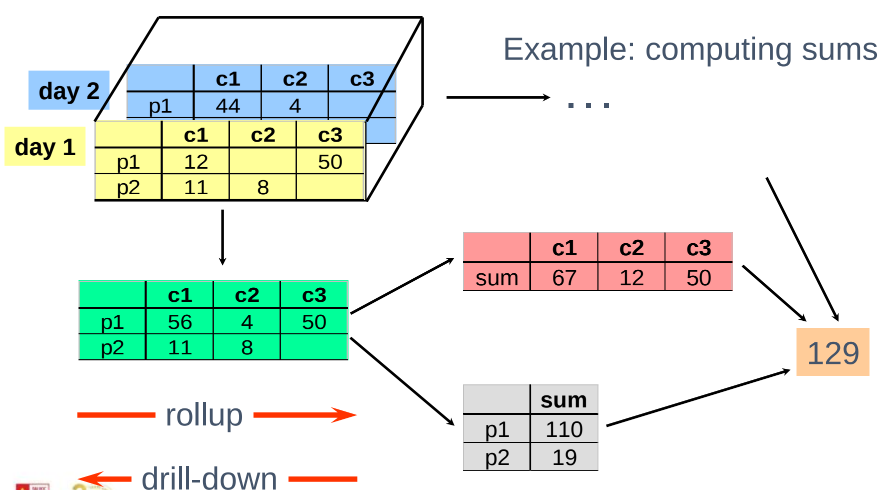
A Sample Data Cube



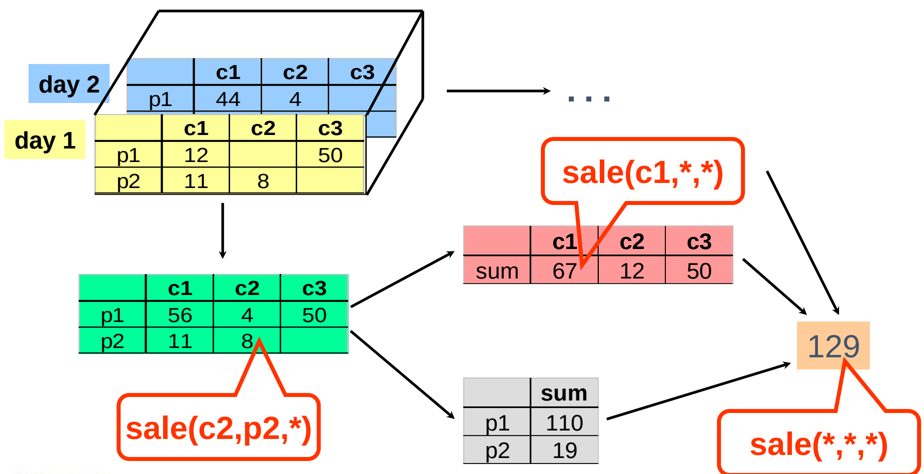
Cuboids Corresponding to the Cube



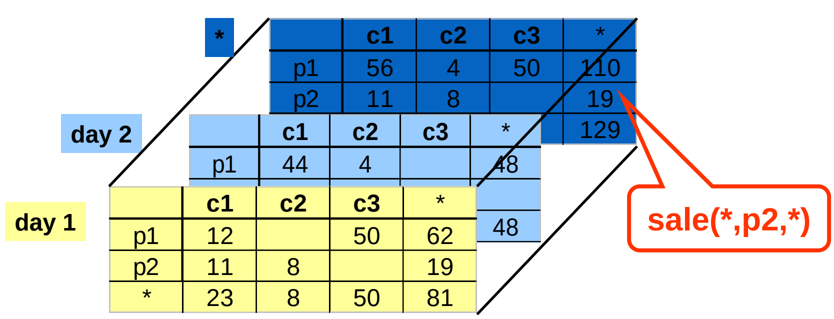
Cube Aggregation



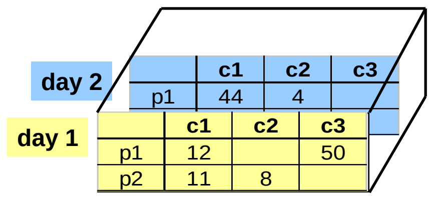
Cube Operators



Extended Cube



Aggregation Using Hierarchies



customer
region
country

	region A	region B
p1	56	54
p2	11	8

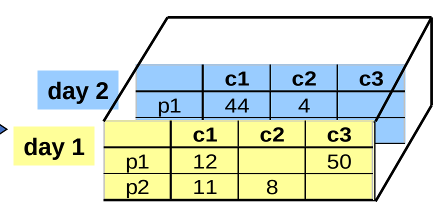
(customer c1 in Region A;
customers c2, c3 in Region B)

Pivoting

Fact table view:

sale	prodId	storeId	date	amt
	p1	c1	1	12
	p2	c1	1	11
	p1	c3	1	50
	p2	c2	1	8
	p1	c1	2	44
	p1	c2	2	4

Multi-dimensional cube:



	c1	c2	c3
p1	56	4	50
p2	11	8	

CUBE Operator (SQL-99)

Chevy Sales Cross Tab				
Chevy	***1990***	***1991***	***1992***	***Total (ALL)***
black	50	85	154	289
white	40	115	199	354
Total (ALL)	90	200	353	1286

SELECT model, year, color, sum(sales) as sales
FROM sales
WHERE model in ('Chevy')
AND year BETWEEN 1990 AND 1992
GROUP BY CUBE (model, year, color);

CUBE Contd.

SELECT model, year, color, sum(sales) as sales
FROM sales
WHERE model in ('Chevy')
AND year BETWEEN 1990 AND 1992
GROUP BY CUBE (model, year, color);

- Computes union of 8 different groupings:
  - {(model, year, color), (model, year), (model, color), (year, color), (model), (year), (color), ()}

Aggregates

- Operators: sum, count, max, min, median, average
- "Having" clause
- Cube (& Rollup) operator
- Using dimension hierarchy
  - average by region (within store)
  - maximum by month (within date)

Query & Analysis Tools

- Query Building
- Report Writers (comparisons, growth, graphs,…)
- Spreadsheet Systems
- Web Interfaces
- Data Mining

Other Operations

- Time functions
  - e.g., time average
- Computed Attributes
  - e.g., commission = sales * rate
- Text Queries
  - e.g., find documents with words X AND B
  - e.g., rank documents by frequency of words X, Y, Z

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

52

52

Data Warehouse Implementation

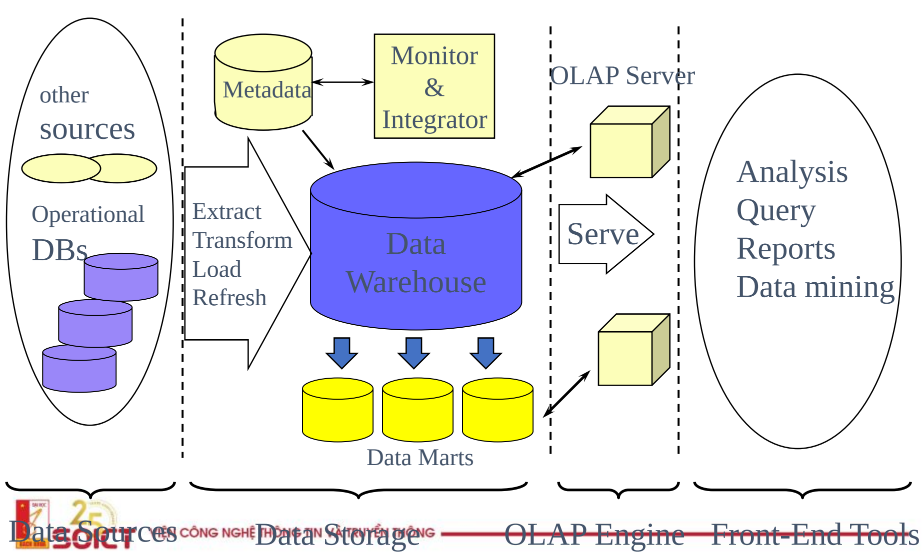
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

53

Implementing a Warehouse

- *Monitoring*: Sending data from sources
- *Integrating*: Loading, cleansing,...
- *Processing*: Query processing, indexing, ...
- *Managing*: Metadata, Design, ...

Multi-Tiered Architecture



Monitoring

- Source Types: relational, flat file, IMS, VSAM, IDMS, WWW, news-wire, …
- Incremental vs. Refresh

customer	id	name	address	city
	53	joe	10 main	sfo
	81	fred	12 main	sfo
	111	**sally**	**80 willow**	**la**

new

Data Cleaning

- Migration (e.g., yen ⇨ dollars)
- Scrubbing: use domain-specific knowledge (e.g., social security numbers)
- Fusion (e.g., mail list, customer merging)
- Auditing: discover rules & relationships (like data mining)

billing DB → customer1(Joe) → merged_customer(Joe)

service DB → customer2(Joe) → merged_customer(Joe)

Loading Data

- Incremental vs. refresh
- Off-line vs. on-line
- Frequency of loading
  - At night, 1x a week/month, continuously
- Parallel/Partitioned load

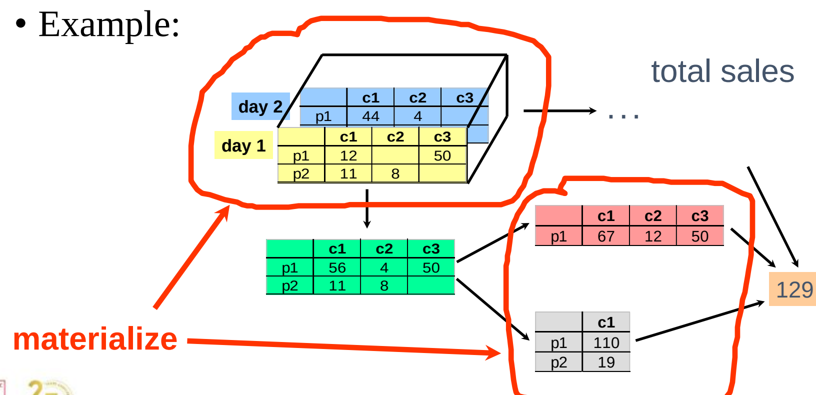
OLAP Implementation

Derived Data

- Derived Warehouse Data
  - indexes
  - aggregates
  - materialized views (next slide)
- When to update derived data?
- Incremental vs. refresh

What to Materialize?

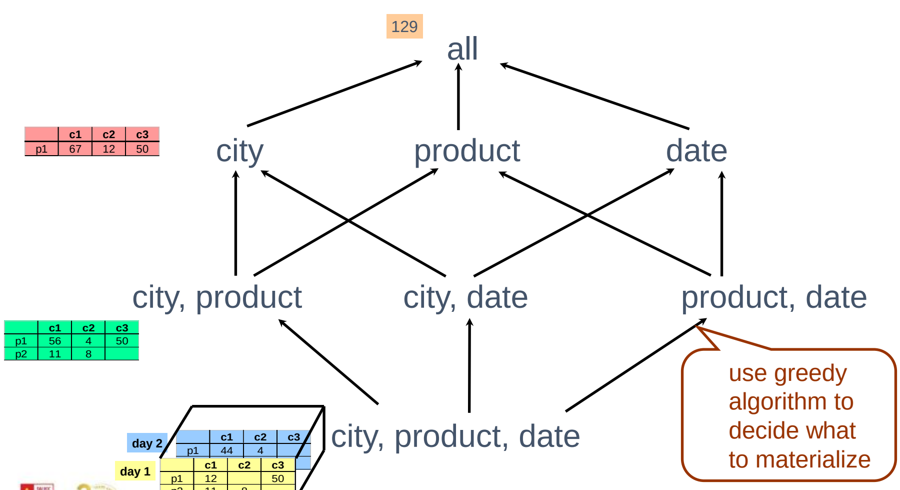
- Store in warehouse results useful for common queries
- Example:



Materialization Factors

- Type/frequency of queries
- Query response time
- Storage cost
- Update cost

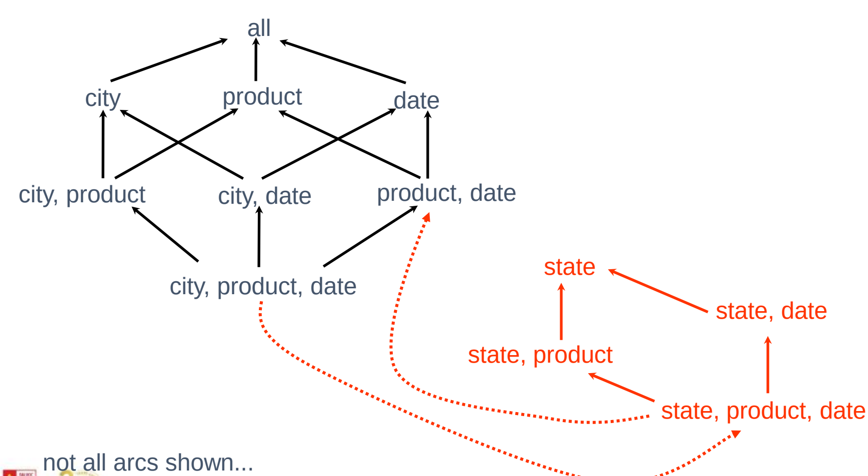
Cube Aggregates Lattice



Dimension Hierarchies

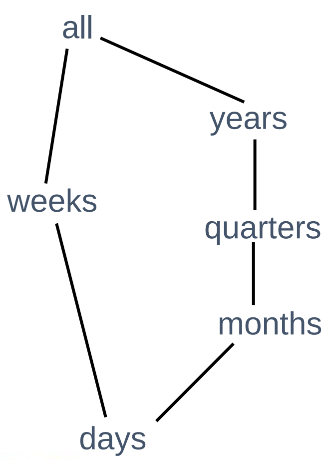
cities	city	state
	c1	CA
	c2	NY

Dimension Hierarchies



not all arcs shown...

Interesting Hierarchy



time	day	week	month	quarter	year
	1	1	1	1	2000
	2	1	1	1	2000
	3	1	1	1	2000
	4	1	1	1	2000
	5	1	1	1	2000
	6	1	1	1	2000
	7	1	1	1	2000
	8	2	1	1	2000

conceptual dimension table

Indexing OLAP Data: Bitmap Index

- Index on a particular column
- Each value in the column has a bit vector: bit-op is fast
- The length of the bit vector: # of records in the base table
- The *i*-th bit is set if the *i*-th row of the base table has the value for the indexed column
- not suitable for high cardinality domains

Base table

Cust	Region	Type
C1	Asia	Retail
C2	Europe	Dealer
C3	Asia	Dealer
C4	America	Retail
C5	Europe	Dealer

Index on Region

RecID	Asia	Europe	America
1	1	0	0
2	0	1	0
3	1	0	0
4	0	0	1
5	0	1	0

Index on Type

RecID	Retail	Dealer
1	1	0
2	0	1
3	0	1
4	1	0
5	0	1

Need of Bitmap Indexing

- A company holds an employee table with entries like EmpNo, EmpName, Job, New_Emp and salary.
- Assuming that the employees are hired once in the year, therefore the table will be updated very less and will remain static most of the time.
- But the columns will be frequently used in queries to retrieve data like : No. of female employees in the company etc.
- In this case we need a file organization method which should be fast enough to give quick results.

How Bitmap Indexing is done

- The column New_Emp has only two values **Yes** and **No** based upon the fact that the employee is new to the company or not.
- Similarly let us assume that the Job of the Employees is divided into 4 categories only i.e Manager, Analyst, Clerk and Salesman. Such columns are called columns with low cardinality. Even though these columns have less unique values, they can be queried very often.
- **Bit:** Bit is a basic unit of information used in computing that can have only one of two values either 0 or 1 . The two values of a binary digit can also be interpreted as logical values true/false or yes/no.

Take-home messages

- OLAP
  - Multi-relational Data model
  - Operators
  - SQL
- Data warehouse (architecture, issues, optimizations)

ĐẠI HỌC BÁCH KHOA

25 YEARS ANNIVERSARY SOICT

HA NOI UNIVERSITY OF SCIENCE AND TECHNOLOGY
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Thank you for your attentions!

soict.hust.edu.vn/ fb.com/groups/soict

72