

## Chỉ mục và thiết kế kho dữ liệu



### Mục tiêu:

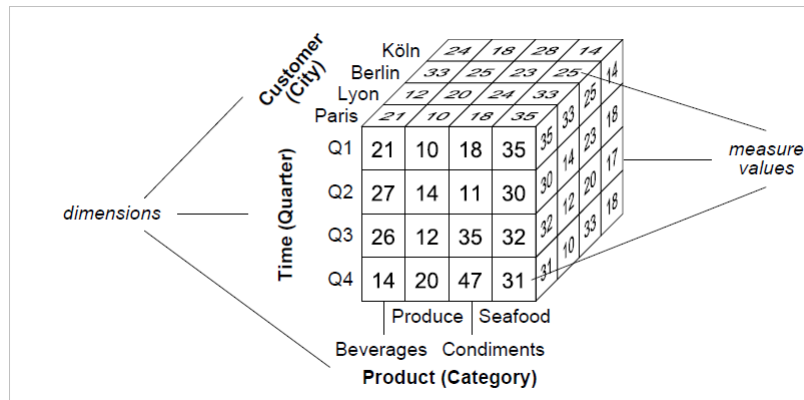
- Index
- Hiểu ý nghĩa của chỉ mục cho OLAP
- Thiết kế, định nghĩa kho dữ liệu
- Các ví dụ

## Đặt vấn đề



- Nguyên tắc của chỉ mục: Ánh xạ các giá trị khóa tới các bản ghi để phối hợp truy cập trực tiếp.
- Mục đích: Tăng tốc độ truy cập dữ liệu
- Yêu cầu: Sử dụng thêm bộ nhớ
- Trong các hệ CSDL quan hệ đều ưa chuộng kỹ thuật index B+-Tree

## Mô hình đa chiều

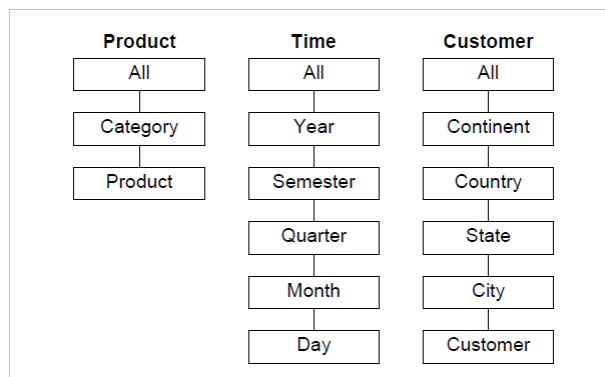


Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

3

3

## Phân cấp

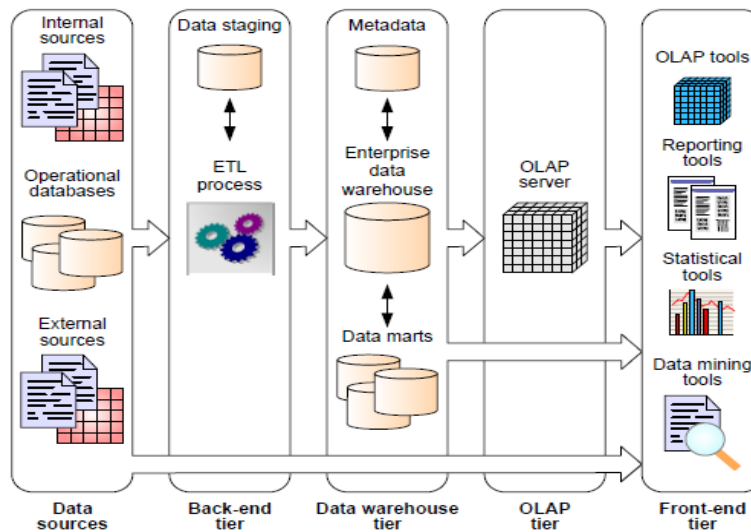


Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

4

4

## Kiến trúc kho dữ liệu



Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

5

5

## Index trong CSDL



- **Clusted index:** Bản ghi được chèn đúng thứ tự của cột được index.
- **Non clustered index:** Sử dụng cấu trúc dữ liệu Heap. Dữ liệu được thêm vào cuối bảng
  - **Đồng (heap)** là một cấu trúc dữ liệu được sử dụng để lưu trữ một tập hợp các phần tử sao cho các phần tử này luôn tuân theo một quy tắc sắp xếp được xác định trước. Cụ thể, trong một đồng, mỗi phần tử có giá trị lớn hơn hoặc bằng (hoặc nhỏ hơn hoặc bằng) tất cả các phần tử con của nó.
  - Heap được sử dụng khi cần tìm kiếm phần tử lớn nhất (hoặc nhỏ nhất) trong một tập hợp phần tử và các thao tác thêm, xóa phần tử thường xuyên.

Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

6

6

## Clusterd index



- B+Tree
- Sắp xếp vật lý các bản ghi
- Nhiều nhất chỉ có 1 clustered index trên một bảng
- Ràng buộc PK là Clustered Index
- Minh họa
- `CREATE CLUSTERED INDEX <index_name> ON <table_name>(<column_name> ASC/DESC)`

## Non Clusterd index



- B+Tree
- Sắp xếp tăng/giảm theo bảng chỉ mục, cần thêm bộ nhớ (tạo ra 1 file index). **Không sắp xếp vật lý các bản ghi**
- Tương tự index của 1 cuốn sách
- Ràng buộc Unique là Non clustered index
- Minh họa
- `CREATE NONCLUSTERED INDEX <index_name> ON <table_name>(<column_name> ASC/DESC)`

## Index trong OLAP



- Môi trường OLAP chủ yếu là đọc dữ liệu với lượng lớn để trả lời các truy vấn online (time đáp ứng chấp nhận được)
- Cần lựa chọn kỹ thuật đánh chỉ mục (Index) phù hợp
- Có hai kỹ thuật Index hay dùng trong OLAP: Bitmap index và Join Index

Bảng cơ sở

Bitmap index trên Region

Bitmap Index trên Type

Cust	Region	Type	RecID	Asia	Europe	America	RecID	Retail	Dealer
C1	Asia	Retail	1	1	0	0	1	1	0
C2	Europe	Dealer	2	0	1	0	2	0	1
C3	Asia	Dealer	3	1	0	0	3	0	1
C4	America	Retail	4	0	0	1	4	1	0
C5	Europe	Dealer	5	0	1	0	5	0	1

Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

9

9

## Chỉ mục Bitmap



- Đánh chỉ mục trên các cột cụ thể
- Mỗi giá trị trong cột có một vector bit
- Độ dài của mỗi bảng vector bit bằng số bản ghi có trong bảng cơ sở
- Bit thứ  $i$  được thiết lập (=1) nếu dòng thứ  $i$  của bảng cơ sở có giá trị cho cột index
- Bitmap index phù hợp cho các cột có miền giá trị thưa

Bảng cơ sở

Bitmap index trên Region

Bitmap Index trên Type

Cust	Region	Type	RecID	Asia	Europe	America	RecID	Retail	Deale
C1	Asia	Retail	1	1	0	0	1	1	0
C2	Europe	Dealer	2	0	1	0	2	0	1
C3	Asia	Dealer	3	1	0	0	3	0	1
C4	America	Retail	4	0	0	1	4	1	0
C5	Europe	Dealer	5	0	1	0	5	0	1

Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

10

10

## Chỉ mục Bitmap



Base table			Index on Region				Index on Type		
Cust	Region	Type	RecID	Asia	Europe	America	RecID	Retail	Dealer
C1	Asia	Retail	1	1	0	0	1	1	0
C2	Europe	Dealer	2	0	1	0	2	0	1
C3	Asia	Dealer	3	1	0	0	3	0	1
C4	America	Retail	4	0	0	1	4	1	0
C5	Europe	Dealer	5	0	1	0	5	0	1

Region	Bitmap vector
Asia	1 0 1 0 0
Europe	0 1 0 0 1
America	0 0 0 1 0

Type	Bitmap vector
Retail	1 0 0 1 0
Dealer	0 1 1 0 1

Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

11

11

## Chỉ mục Bitmap



- Cho quan hệ Employees(EmpID, Name, Gender, Rating)

EmpID	Name	Gender	Rating
201	Xuân	M	3
202	Hạ	M	5
203	Thu	F	5
205	Đông	M	4

Index trên Gender

Gender	Bitmap vector
M	1 1 0 1
F	0 0 1 0

Index trên Rating

Rating	Bitmap vector
1	0 0 0 0
2	0 0 0 0
3	1 0 0 0
4	0 0 0 1
5	0 1 1 0

Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

12

12

## Chỉ mục Bitmap



- Cho quan hệ Employee(EmpID, Name, Gender, Rating)

Rating	Bitmap vector
1	0 0 0 0
2	0 0 0 0
3	1 0 0 0
4	0 0 0 0
5	0 1 1 0

Cho biết số nhân viên có thứ hạng nhỏ hơn 3?

Thao tác trên Bimap vector:

0 0 0 0
0 0 0 0 OR
0 0 0 0

Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

13

13

## Chỉ mục Bitmap



- Cho quan hệ Employee(EmpID, Name, Gender, Rating)

Gender	Bitmap vector
M	1 1 0 1
F	0 0 1 0

Rating	Bitmap vector
1	0 0 0 0
2	0 0 0 0
3	1 0 0 0
4	0 0 0 0
5	0 1 1 0

Cho biết số nhân viên có thứ hạng nhỏ hơn 3 và giới tính là Nam?

Thao tác trên bit-vector:

0 0 0 0
1 1 0 1 AND
0 0 0 0

Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

14

14

## Chỉ mục Bitmap



- Cho quan hệ Employee(EmpID, Name, Gender, Rating)

Gender	Bitmap vector
M	1 1 0 1
F	0 0 1 0

Cho biết số phần trăm của các nam nhân viên?

$$\text{Count(M)/Length(M)} = \frac{3}{4} = 75\%$$

## Chỉ mục Bitmap



- Thuận lợi của bitmap index:
  - Cho phép sử dụng hiệu quả các phép toán trên bit khi trả lời các truy vấn
  - Giảm không gian lưu trữ hơn kỹ thuật B+tree index
  - Có thể sử dụng mã loạt dài để lưu trữ
  - Sản phẩm thương mại hỗ trợ Bitmap index: Oracle,
  - Làm việc tốt trên các thuộc tính có miền giá trị thưa (Cột có lực lượng nhỏ, lực lượng  $\leq 0.1\%$  là tốt nhất, từ  $0.2\% - 1\%$  thì cần xem xét)
- Điểm yếu của bitmap index
  - Không hiệu quả trên các thuộc tính có miền giá trị dày đặc
  - Khó bảo trì, tức là khi kích thước quan hệ thay đổi thì phải tạo bitmap index lại



## Chỉ mục Bitmap



- Xét quan hệ Tài\_Khoản(Số\_tài\_khoản, Tên\_chi\_nhánh, Số\_dư):

Số_tài_khoản	Tên_chi_nhánh	Số_dư
A_01	Ha Noi	750
A_05	Nghe An	500
A_04	Nghe An	600
A_06	Dong Thap	700
A_09	Ho Chi Minh	400
A_10	Ho Chi Minh	900
A_19	Ho Chi Minh	700
A_30	Da Nang	700
A_68	Long An	350

- Xây dựng Bitmap index trên thuộc tính Tên\_chi\_nhánh
- Xây dựng Bitmap index trên thuộc tính số\_dư với các mức: dưới 250, từ 250 đến dưới 500, từ 500 đến dưới 750 và từ 750 trở lên
- Cho biết số lượng tài khoản ở chi nhánh Nghe An có số dư từ 500 trở lên. Cụ thể hóa các bước trong câu trả lời truy vấn trên và chỉ ra kết quả cuối dựa vào các vector bit

Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

17

17

## Thiết kế kho dữ liệu



- Cách tiếp cận
  - Top-Down: Cho toàn doanh nghiệp, sau đó đi vào từng Data Mart riêng biệt
  - Bottom-up: Bắt đầu từ các yêu cầu cụ thể, sau đó hợp nhất chúng
- Quy trình thiết kế



Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

18

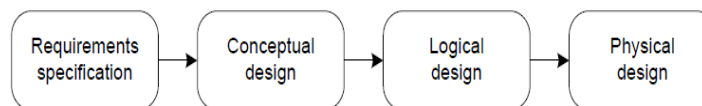
18

## Thiết kế kho dữ liệu



### ■ Pha đặc tả yêu cầu

- Trả lời câu hỏi: Dữ liệu nào đã có? Tổ chức theo những khía cạnh nào?
- Các truy vấn mà người dùng quan tâm theo chủ thể của hệ thống
- Xác định được các phần tử quan trọng: các chiều, độ đo

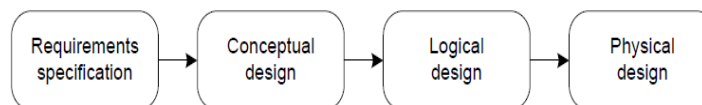


## Thiết kế kho dữ liệu



### ■ Pha thiết kế khái niệm

- Trả lời câu hỏi: Dữ liệu nào đã có? Tổ chức theo những khía cạnh nào?
- Các truy vấn mà người dùng quan tâm theo chủ thể của hệ thống
- Xác định được các phần tử quan trọng: các chiều, độ đo



## Thiết kế kho dữ liệu



### ■ Ví dụ

Dimensions and measures	Hierarchies and levels	Analysis scenarios											
		1a	1b	1c	2a	2b	2c	3a	3b	3c	4a	4b	4c
Employee	<b>Supervision</b>												
	Subordinate → Supervisor	—	—	—	—	—	—	—	—	—	✓	✓	✓
	<b>Territories</b>												
Date	Employee ⇄ City →												
	State → Country → Continent												
	<b>Calendar</b>												
Product	Day → Month →	—	—	—	✓	✓	✓	✓	✓	—	✓	✓	✓
	Quarter → Semester → Year												
	<b>Categories</b>												
Customer	Product → Category	—	—	✓	—	✓	—	✓	✓	✓	✓	—	—
	<b>Geography</b>												
	Customer → City →	✓	✓	✓	✓	✓	✓	—	—	—	—	—	—
Supplier	State → Country → Continent												
	<b>Geography</b>												
	Supplier → City →	✓	—	—	—	—	—	—	—	—	—	—	—
Quantity	State → Country → Continent												
	—	—	—	—	—	—	—	—	—	—	✓	—	—
	Discount	—	—	—	—	—	—	—	✓	—	—	—	—
SalesAmount	—	—	—	—	—	—	—	—	—	—	—	—	—
	—	✓	✓	✓	✓	—	—	—	—	—	✓	✓	✓
	UnitPrice	—	—	—	—	—	✓	—	—	—	—	—	—

Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

21

21

## Thiết kế kho dữ liệu



### ■ Pha thiết kế khái niệm

- Mục đích là thể hiện một tập hợp các yêu cầu dữ liệu một cách rõ ràng và ngắn gọn

Source table	Source attribute	DW level	DW attribute	Transformation
Products	ProductName	Product	ProductName	—
Products	QuantityPerUnit	Product	QuantityPerUnit	—
Products	UnitPrice	Product	UnitPrice	—
...	...	...	...	...
Customers	CustomerID	Customer	CustomerID	✓
Customers	CompanyName	Customer	CompanyName	—
...	...	...	...	...
Orders	OrderID	Order	OrderNo	✓
Orders		Order	OrderLineNo	✓
Orders	OrderDate	Date	—	✓
...	...	...	...	...

Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

22

22

## Thiết kế kho dữ liệu



- Pha thiết kế logic
  - Xác định lược đồ logic:
  - Cách thức ELT: chuyển các dữ liệu đã có vào lược đồ logic (xem chi tiết phần sau)
- Pha thiết kế vật lý
  - Tiếp cận theo nghiệp vụ
  - Tiếp cận theo dữ liệu

## Thiết kế kho dữ liệu



- Xây dựng mô hình logic
  - Khối dữ liệu (Cube)
  - Chiều (Dimension)
  - Phân cấp chiều (Hierarchies)
  - Độ đo (Measures)
- Xây dựng mô hình vật lý
  - Sơ đồ hình sao
  - Sơ đồ bông tuyết
  - Sơ đồ chòm sao

## Thiết kế kho dữ liệu



- Mục đích của mô hình logic
  - Xác định các khối, các chiều, các độ đo
  - Tính chỉnh bảng dữ kiện, bảng chiều theo các chủ đề định trước trong bước thiết kế khái niệm
  - Xây dựng các phân cấp chi tiết cho mỗi chiều
- Ví dụ
  - Các khối: Sales, Price, Inventory
  - Các chiều: Product, Time, Geography, Customer
  - Các độ đo: Sum(), AVG(), Count(), Max(),...

## Thiết kế kho dữ liệu

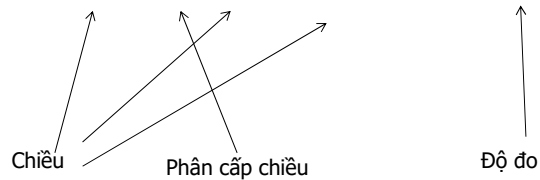


- Các chiều thường là các thực thể
  - Mỗi chiều có thể được dùng cho nhiều hơn một khối
  - Các chiều có thể có phân cấp chi tiết
- Ví dụ
  - Chiều: Time
  - Phân cấp chi tiết:  
Year → Quarter → Week → Day  
Hoặc  
Year → Quarter → Month → Day

## Thiết kế kho dữ liệu



- **Khối**
  - Khối là sự kết hợp của các chiều và/hoặc các độ đo
- **Ví dụ**
  - Sales((Product, Day, Store, Customer), (Quality))



## Thiết kế kho dữ liệu



- **Kết nhập theo các chiều – phép CUBE**
- **Ví dụ với Cube\_Sales((Item, City, Year), (Quality))**  
biểu diễn 3 chiều (Item, City, Year)
  - 1 sự kết nhập 3 chiều (Item, City, Year)
  - 3 sự kết hợp 2 chiều (Item, City), (Item, Year) và (Year, City)
  - 3 sự kết hợp 1 chiều (Item), (City), (Year)
  - 1 sự kết hợp 0 chiều ( )
- Có  $2^n$  sự kết hợp giữa n chiều (số tập con của tập n phần tử)

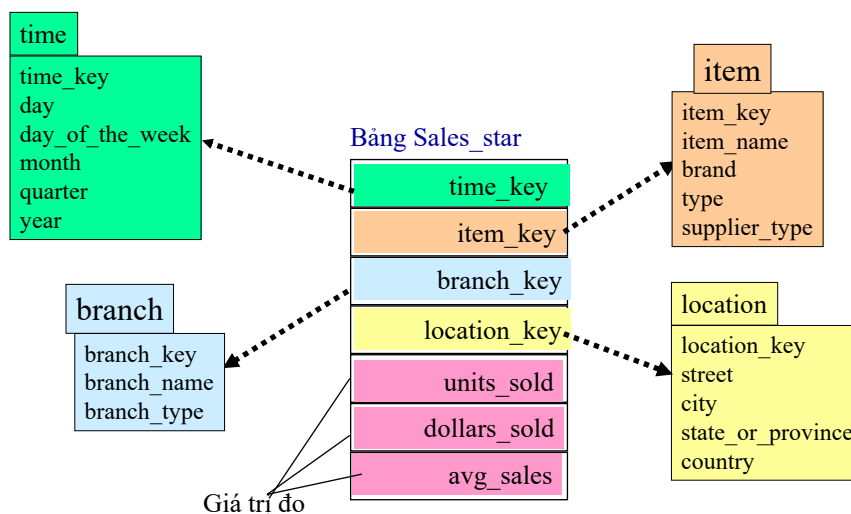
## Sơ đồ kho dữ liệu



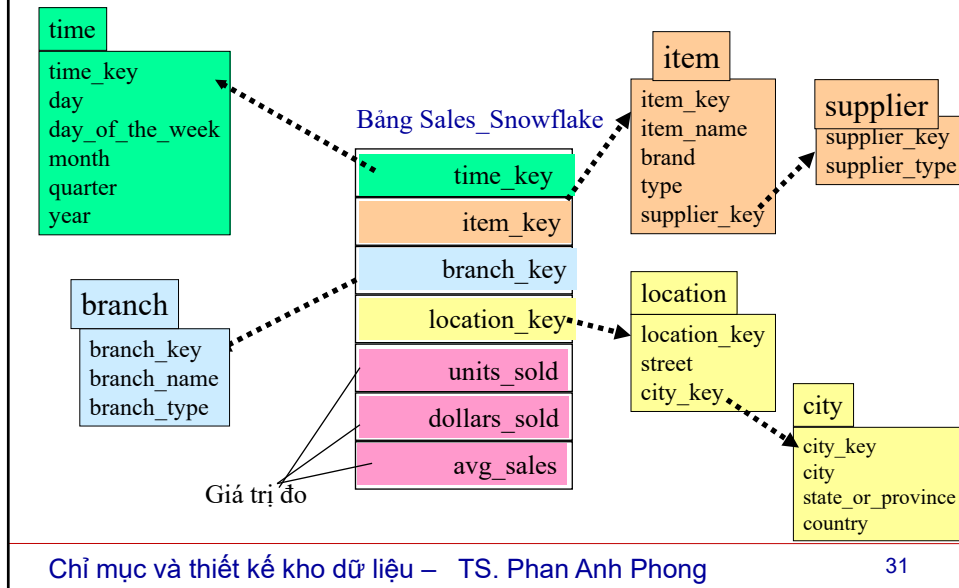
### ■ Các sơ đồ

- **Sơ đồ hình sao (star schema)**: là sơ đồ gồm một bảng trung tâm kết nối với các bảng chiều. Bảng trung tâm được gọi là bảng dữ kiện (fact table)
- **Sơ đồ hình bông tuyết (Snowflake schema)**: là sự mở rộng sơ đồ hình sao, trong đó **một số bảng chiều có các bảng phân cấp khái niệm của chiều đó**
- **Sơ đồ chòm sao (Fact constellations schema)**: là sơ đồ trong đó một số **bảng chiều kết nối với nhiều hơn một bảng dữ kiện** (bảng chiều dùng chung cho nhiều bảng dữ kiện)

## Ví dụ về sơ đồ hình sao



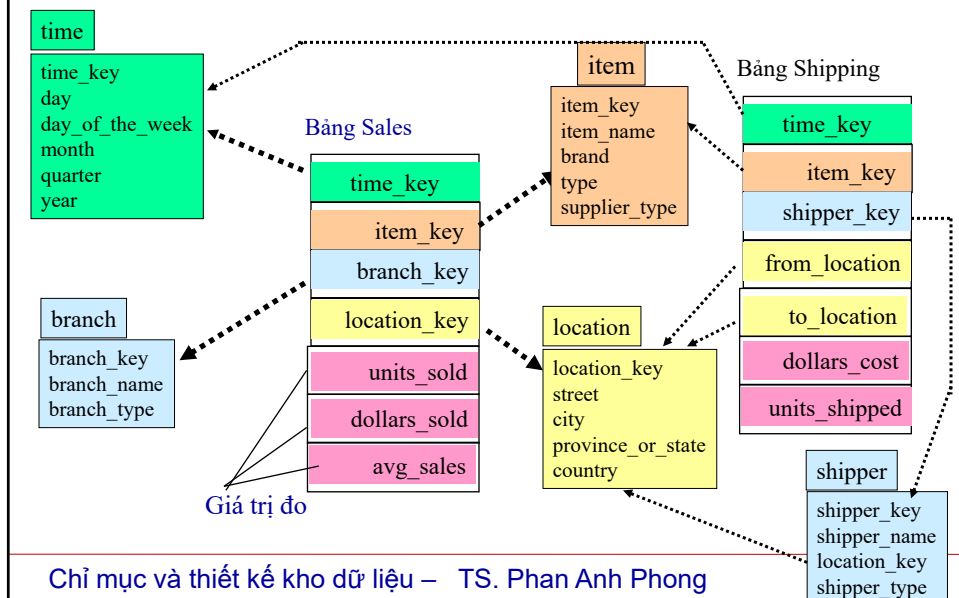
## Ví dụ về sơ đồ bông tuyết



31

31

## Ví dụ về sơ đồ chòm sao



32



## Định nghĩa sơ đồ kho dữ liệu



- Định nghĩa khối - (Bảng sự kiện)
  - Tên khối
  - Các chiều
  - Các độ đo
- Định nghĩa các chiều
  - Tên chiều
  - Các thuộc tính
  - Phân cấp chiều (**subdimension\_list**)
- Chú ý: Dựa vào sơ đồ để định nghĩa

## Định nghĩa sơ đồ kho dữ liệu



- Định nghĩa khối - (Bảng sự kiện)
  - Cú pháp:  
**define cube** <cube\_name> [<dimension\_list>]:  
    <measure\_list>  
Trong đó, **measure\_list** được định nghĩa theo cú pháp  
    **measure\_name** = phép đo()
    - Ví dụ:  
**define cube** sales\_star [time, item, branch, location]:  
    dollars\_sold = sum(sales\_in\_dollars),  
    avg\_sales = avg(sales\_in\_dollars),  
    units\_sold = count(\*)

## Định nghĩa sơ đồ kho dữ liệu



- Định nghĩa các chiều (Dimension Table)
  - Cú pháp:  
`define dimension <dimension_name> as (<attribute_or_subdimension_list>)`
  - Ví dụ:  
`define dimension item as (item_key, item_name, brand, type, supplier_type)`
  - Chú ý: khi có phân cấp chiều `subdimension_list`:  
`define dimension item as (item_key, item_name, brand, type, supplier(supplier_key, supplier_type))`

## Định nghĩa sơ đồ kho dữ liệu



- Trường hợp đặc biệt khi **bảng chiều dùng chung**
    - Trước tiên định nghĩa khối như ở phần trên
    - Sau đó định nghĩa chiều trong khối
- Cú pháp:
- ```
define dimension <dimension_name> as  
<dimension_name_first_time> in cube  
<cube_name_first_time>
```
- Ví dụ:
- ```
define dimension item as item in cube sales
```
- Sử dụng khi định nghĩa trong sơ đồ chòm sao

## Định nghĩa sơ đồ hình sao – ví dụ



- **define cube** sales\_star [time, item, branch, location]:  
    dollars\_sold = sum(sales\_in\_dollars),  
    avg\_sales = avg(sales\_in\_dollars), units\_sold = count(\*)
- **define dimension** time **as** (time\_key, day, day\_of\_week, month, quarter, year)
- **define dimension** item **as** (item\_key, item\_name, brand, type, supplier\_type)
- **define dimension** branch **as** (branch\_key, branch\_name, branch\_type)
- **define dimension** location **as** (location\_key, street, city, province\_or\_state, country)

Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

37

37

## Định nghĩa sơ đồ bông tuyết – ví dụ



```
define cube sales_snowflake [time, item, branch, location]:  
    dollars_sold = sum(sales_in_dollars), avg_sales =  
        avg(sales_in_dollars), units_sold = count(*)  
define dimension time as (time_key, day, day_of_week, month,  
    quarter, year)  
define dimension item as (item_key, item_name, brand, type,  
    supplier(supplier_key, supplier_type))  
define dimension branch as (branch_key, branch_name,  
    branch_type)  
define dimension location as (location_key, street, city(city_key,  
    province_or_state, country))
```

Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

38

38

## Định nghĩa sơ đồ chòm sao – ví dụ



```
define cube sales [time, item, branch, location]:
    dollars_sold = sum(sales_in_dollars), avg_sales =
        avg(sales_in_dollars), units_sold = count(*)
define dimension time as (time_key, day, day_of_week, month, quarter, year)
define dimension item as (item_key, item_name, brand, type, supplier_type)
define dimension branch as (branch_key, branch_name, branch_type)
define dimension location as (location_key, street, city, province_or_state,
    country)
define cube shipping [time, item, shipper, from_location, to_location]:
    dollar_cost = sum(cost_in_dollars), unit_shipped = count(*)
define dimension time as time in cube sales
define dimension item as item in cube sales
define dimension shipper as (shipper_key, shipper_name, location as location
    in cube sales, shipper_type)
define dimension from_location as location in cube sales
define dimension to_location as location in cube sales
```

Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

39

39

## Giá trị đo trong khối



- **Giá trị đo (measure):** tập các giá trị được tính toán dựa vào các cột, hoặc các dòng trong bảng sự kiện của khối đó, thông thường chúng là các giá trị số. Tập giá trị này là rất quan trọng trong việc xử lý, tổng hợp và phân tích dữ liệu trên khối
- Các giá trị đo hay dùng:  
count(), sum(), min(), max(),  
avg(), standard\_deviation(),  
median(), rank()...

Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

40

40

## Tính các giá trị đo trong khối



- Ví dụ: cho sơ đồ dữ liệu OLAP như sau:  
time (time\_key, day, day\_of\_week, month, quarter, year);  
item (item\_key, item\_name, brand, type,  
supplier(supplier\_key, supplier\_type));  
branch (branch\_key, branch\_name, branch\_type) ;  
location (location\_key, street, city, province\_or\_state,  
country)  
sales (time\_key, item\_key, branch\_key, location\_key,  
number\_of\_unit\_sold, price)
- Hãy cho biết tổng tiền và tổng các mặt hàng đã bán theo  
time, item, branch và location?

Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

41

41

## Tính các giá trị đo trong khối



- Hãy cho biết tổng tiền và tổng các mặt hàng đã bán theo time, item,  
branch và location?
- ```
select s.time_key, s.item_key, s.branch_key, s.location_key,  
sum(s.number_of_units_sold*s.price), sum(s.number_of_units_sold)  
from time t, item i, branch b, location l, sales s  
where s.time_key = t.time_key  
and s.item_key = i.item_key  
and s.branch_key = b.branch_key  
and s.location_key = l.location_key  
group by CUBE (s.time_key, s.item_key, s.branch_key, s.location_key)
```

Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

42

42

## ETL - Khái quát hóa giá trị thuộc tính



- Các bước:
  - Bước 1. Tuyển tập các dữ liệu liên quan đến công việc (quan hệ khởi tạo) bằng cách sử dụng các truy vấn CSDL thông thường
  - Bước 2. Loại bỏ các thuộc tính không cần thiết
  - Bước 3. Khái quát các thuộc tính
  - Bước 4. Kết nhập theo các độ đo tương ứng, sử dụng các phép toán OLAP: pivot, count, sum, roll-up, cube...
  - Bước 5. Tương tác với người dùng để cải thiện các yêu cầu và trực quan hóa kết quả, đưa ra các luật...

## Ví dụ về khái quát hóa dữ liệu



Ví dụ: Hãy mô tả đặc điểm chung của các **học viên sau đại học của một trường đại học**

- Bước 1. Thu thập dữ liệu để đưa ra *quan hệ khởi tạo* bằng câu lệnh SQL thông thường, chẳng hạn:  

```
Select name, gender, major, birth_place, birth_date,  
residence, phone#, gpa  
from students  
where student_status in {"Msc", "PhD" }
```
- Bước 2, 3: Thực hiện quy nạp theo thuộc tính
- Bước 4. Trình bày kết quả trong quan hệ tổng quát, bảng tổng hợp (cross-tab), hoặc dạng luật
- Bước 5. ...

## Ví dụ về khái quát hóa dữ liệu



Quan hệ khởi tạo

| Name           | Gender  | Major         | Birth-Place             | Birth_date  | Residence                | Phone #  | GPA           |
|----------------|---------|---------------|-------------------------|-------------|--------------------------|----------|---------------|
| Jim Woodman    | M       | CS            | Vancouver, BC, Canada   | 8-12-76     | 3511 Main St., Richmond  | 687-4598 | 3.67          |
| Scott Lachance | M       | CS            | Montreal, Que, Canada   | 28-7-75     | 345 1st Ave., Richmond   | 253-9106 | 3.70          |
| Laura Lee      | F       | Physics       | Seattle, WA, USA        | 25-8-70     | 125 Austin Ave., Burnaby | 420-5232 | 3.83          |
| ...            | ...     | ...           | ...                     | ...         | ...                      | ...      | ...           |
| Loại           | Giữ lại | Sci, Eng, Bus | Khái quát theo quốc gia | Khoảng tuổi | Thành phố                | Loại     | Excl, VG, ... |

Quan hệ khái quát cơ bản

| Gender | Major   | Birth_region | Age_range | Residence | GPA       | Count |
|--------|---------|--------------|-----------|-----------|-----------|-------|
| M      | Science | Canada       | 20-25     | Richmond  | Very-good | 16    |
| F      | Science | Foreign      | 25-30     | Burnaby   | Excellent | 22    |
| ...    | ...     | ...          | ...       | ...       | ...       | ...   |

Bảng tổng hợp theo Gender, Birth\_region

Bảng tổng hợp theo các thuộc tính khác (có thể có)

| Birth_Region | Canada | Foreign | Total |
|--------------|--------|---------|-------|
| Gender       |        |         |       |
| M            | 16     | 14      | 30    |
| F            | 10     | 22      | 32    |
| Total        | 26     | 36      | 62    |

Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

45

45

## Nguyên tắc quy nạp theo thuộc tính



- Tập trung dữ liệu: thu thập các dữ liệu liên quan, các chiều và đầu ra của bước này là *quan hệ khởi tạo*
- Loại các thuộc tính: loại thuộc tính A nếu thuộc tính này có nhiều giá trị phân biệt nếu (1) không có thao tác tổng quát trên A, hoặc (2) A đã được diễn tả ở một thuộc tính khác
- Khái quát hóa thuộc tính: nếu thuộc tính A có nhiều giá trị phân biệt nhưng có một thao tác tổng quát trên A khi đó ta lựa chọn thao tác đó để khái quát hóa thuộc tính A
- Điều chỉnh số thuộc tính của quan hệ khái quát cơ bản: thường là từ 2 đến 8 (gợi ý)
- Tạo lập quan hệ kết quả theo thực tế sử dụng

Chỉ mục và thiết kế kho dữ liệu – TS. Phan Anh Phong

46

46