

Data Warehousing Overview

CS245 Notes 11

Hector Garcia-Molina
Stanford University

Warehousing

- Growing industry: \$8 billion in 1998
- Range from desktop to huge:
 - ◆ Walmart: 900-CPU, 2,700 disk, 23TB Teradata system
- Lots of buzzwords, hype
 - ◆ slice & dice, rollup, MOLAP, pivot, ...

Outline

- What is a data warehouse?
- Why a warehouse?
- Models & operations
- Implementing a warehouse

What is a Warehouse?

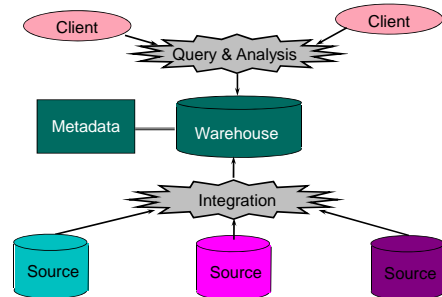
- Collection of diverse data
 - ◆ subject oriented
 - ◆ aimed at executive, decision maker
 - ◆ often a copy of operational data
 - ◆ with value-added data (e.g., summaries, history)
 - ◆ integrated
 - ◆ time-varying
 - ◆ non-volatile



What is a Warehouse?

- Collection of tools
 - ◆ gathering data
 - ◆ cleansing, integrating, ...
 - ◆ querying, reporting, analysis
 - ◆ data mining
 - ◆ monitoring, administering warehouse

Warehouse Architecture



Motivating Examples

- Forecasting
- Comparing performance of units
- Monitoring, detecting fraud
- Visualization

CS 245

Notes11

7

Why a Warehouse?

- Two Approaches:
 - ◆ Query-Driven (Lazy)
 - ◆ Warehouse (Eager)

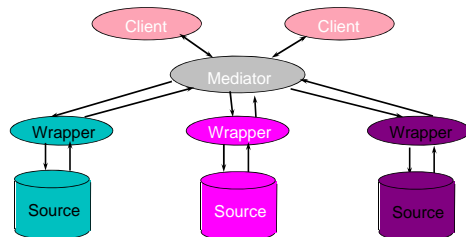


CS 245

Notes11

8

Query-Driven Approach



CS 245

Notes11

9

Advantages of Warehousing

- High query performance
- Queries not visible outside warehouse
- Local processing at sources unaffected
- Can operate when sources unavailable
- Can query data not stored in a DBMS
 - ◆ Modify, summarize (store aggregates)
 - ◆ Add historical information

CS 245

Notes11

10

Advantages of Query-Driven

- No need to copy data
 - ◆ less storage
 - ◆ no need to purchase data
- More up-to-date data
- Query needs can be unknown
- Only query interface needed at sources
- May be less draining on sources

CS 245

Notes11

11

OLTP vs. OLAP

- OLTP: On Line Transaction Processing
 - ◆ Describes processing at operational sites
- OLAP: On Line Analytical Processing
 - ◆ Describes processing at warehouse

CS 245

Notes11

12

OLTP vs. OLAP

OLTP

- Mostly updates
- Many small transactions
- Mb-Tb of data
- Raw data
- Clerical users
- Up-to-date data
- Consistency, recoverability critical

OLAP

- Mostly reads
- Queries long, complex
- Gb-Tb of data
- Summarized, consolidated data
- Decision-makers, analysts as users

CS 245

Notes11

13

Warehouse Models & Operators

• Data Models

- ◆ relations
- ◆ stars & snowflakes
- ◆ cubes

• Operators

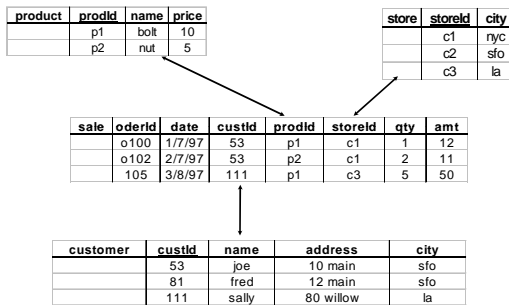
- ◆ slice & dice
- ◆ roll-up, drill down
- ◆ pivoting
- ◆ other

CS 245

Notes11

14

Star

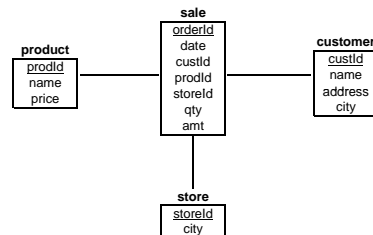


CS 245

Notes11

15

Star Schema



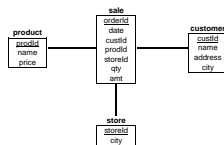
CS 245

Notes11

16

Terms

- Fact table
- Dimension tables
- Measures

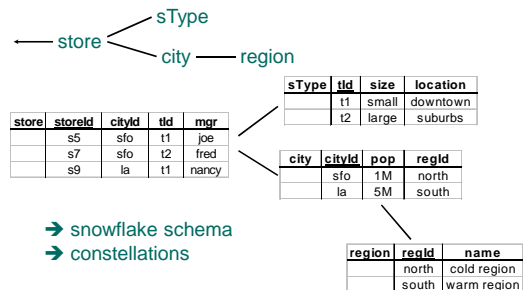


CS 245

Notes11

17

Dimension Hierarchies



CS 245

Notes11

18

Cube

Fact table view:

sale	proldd	storeld	amt
p1	c1	1	12
p2	c1	1	11
p1	c3	1	50
p2	c2	1	8

Multi-dimensional cube:

	c1	c2	c3
p1	12		50
p2	11	8	

dimensions = 2

CS 245

Notes11

19

3-D Cube

Fact table view:

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

Multi-dimensional cube:

	c1	c2	c3
day 2	p1 44	p2 4	
day 1	p1 12	p2 8	p1 50
	p2 11		

dimensions = 3

CS 245

Notes11

20

ROLAP vs. MOLAP

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

CS 245

Notes11

21

Aggregates

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

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

→ 81

CS 245

Notes11

22

Aggregates

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

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

ans	date	sum
	1	81
	2	48

CS 245

Notes11

23

Another Example

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

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

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

→ rollup →

← drill-down ←

CS 245

Notes11

24

Aggregates

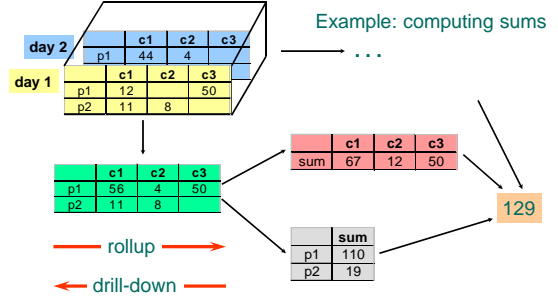
- Operators: sum, count, max, min, median, ave
- "Having" clause
- Using dimension hierarchy
 - ◆ average by region (within store)
 - ◆ maximum by month (within date)

CS 245

Notes11

25

Cube Aggregation

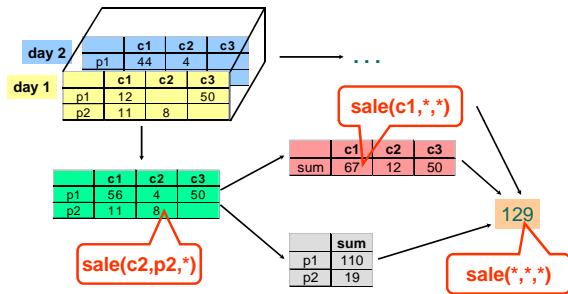


CS 245

Notes11

26

Cube Operators

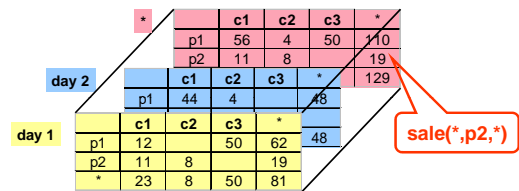


CS 245

Notes11

27

Extended Cube

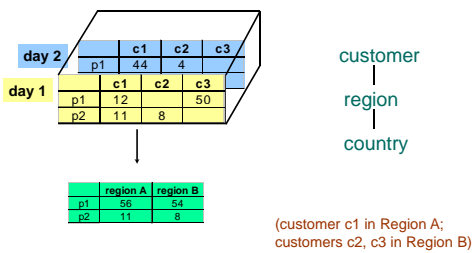


CS 245

Notes11

28

Aggregation Using Hierarchies

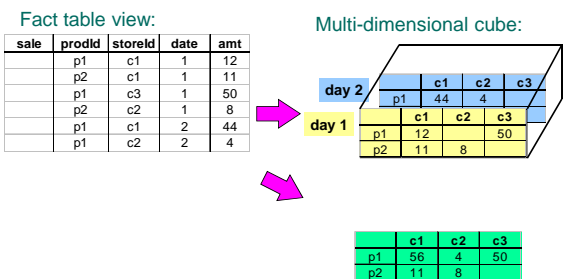


CS 245

Notes11

29

Pivoting



CS 245

Notes11

30

Query & Analysis Tools

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

CS 245

Notes11

31

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

CS 245

Notes11

32

Data Mining

- Decision Trees
- Clustering
- Association Rules

CS 245

Notes11

33

Decision Trees

Example:

- Conducted survey to see what customers were interested in new model car
- Want to select customers for advertising campaign

sale	custId	car	age	city	newCar
	c1	taurus	27	sf	yes
	c2	van	35	la	yes
	c3	van	40	sf	yes
	c4	taurus	22	sf	yes
	c5	merc	50	la	no
	c6	taurus	25	la	no

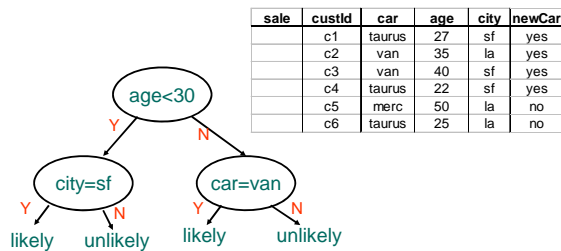
training set

CS 245

Notes11

34

One Possibility

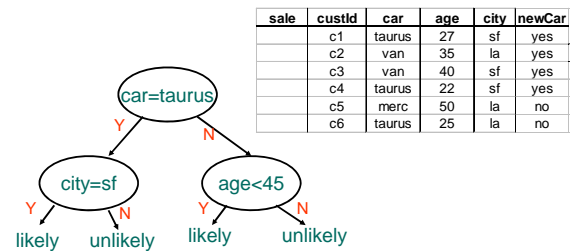


CS 245

Notes11

35

Another Possibility



CS 245

Notes11

36

Issues

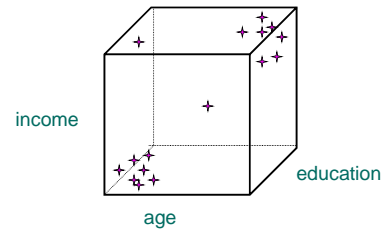
- Decision tree cannot be “too deep”
 - would not have statistically significant amounts of data for lower decisions
- Need to select tree that most reliably predicts outcomes

CS 245

Notes11

37

Clustering

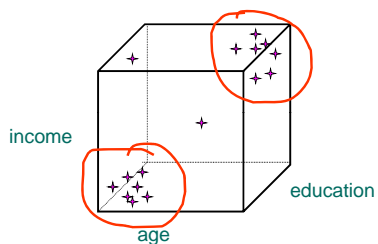


CS 245

Notes11

38

Clustering



CS 245

Notes11

39

Another Example: Text

- Each document is a vector
 - ◆ e.g., <100110...> contains words 1,4,5,...
- Clusters contain “similar” documents
- Useful for understanding, searching documents



CS 245

Notes11

40

Issues

- Given desired number of clusters?
- Finding “best” clusters
- Are clusters semantically meaningful?
 - ◆ e.g., “yuppies” cluster?
- Using clusters for disk storage

CS 245

Notes11

41

Association Rule Mining

sales records:

transaction id	customer id	products bought
tran1	cust33	p2, p5, p8
tran2	cust45	p5, p8, p11
tran3	cust12	p1, p9
tran4	cust40	p5, p8, p11
tran5	cust12	p2, p9
tran6	cust12	p9

market-basket data

- Trend: Products p5, p8 often bought together
- Trend: Customer 12 likes product p9

CS 245

Notes11

42

Association Rule

- Rule: $\{p_1, p_3, p_8\}$
- Support: number of baskets where these products appear
- High-support set: support \geq threshold s
- Problem: find all high support sets

CS 245

Notes11

43

Finding High-Support Pairs

- Baskets(basket, item)
- `SELECT I.item, J.item, COUNT(I.basket)`
`FROM Baskets I, Baskets J`
`WHERE I.basket = J.basket AND`
`I.item < J.item`
`GROUP BY I.item, J.item`
`HAVING COUNT(I.basket) >= s;`

CS 245

Notes11

44

Finding High-Support Pairs

- Baskets(basket, item)
- `SELECT I.item, J.item, COUNT(I.basket)`
`FROM Baskets I, Baskets J`
`WHERE I.basket = J.basket AND`
`I.item < J.item`
`GROUP BY I.item, J.item`
`HAVING COUNT(I.basket) >= s;`

WHY?

CS 245

Notes11

45

Example

basket	item
t1	p2
t1	p5
t1	p8
t2	p5
t2	p8
t2	p11
...	...

→

basket	item1	item2
t1	p2	p5
t1	p2	p8
t1	p5	p8
t2	p5	p8
t2	p5	p11
t2	p8	p11
...

CS 245

Notes11

46

Example

basket	item
t1	p2
t1	p5
t1	p8
t2	p5
t2	p8
t2	p11
...	...

→

basket	item1	item2
t1	p2	p5
t1	p2	p8
t1	p5	p8
t2	p5	p8
t2	p5	p11
t2	p8	p11
...

check if count \geq s

CS 245

Notes11

47

Issues

- Performance for size 2 rules

basket	item
t1	p2
t1	p5
t1	p8
t2	p5
t2	p8
t2	p11
...	...

big ↓

basket	item1	item2
t1	p2	p5
t1	p2	p8
t1	p5	p8
t2	p5	p8
t2	p5	p11
t2	p8	p11
...

even bigger! ↓

- Performance for size k rules

CS 245

Notes11

48

Implementing a Warehouse

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

CS 245

Notes11

49

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

CS 245

Notes11

50

Monitoring Techniques

- Periodic snapshots
- Database triggers
- Log shipping
- Data shipping (replication service)
- Transaction shipping
- Polling (queries to source)
- Screen scraping
- Application level monitoring

Advantages & Disadvantages!!

CS 245

Notes11

51

Monitoring Issues

- Frequency
 - ◆ periodic: daily, weekly, ...
 - ◆ triggered: on "big" change, lots of changes, ...
- Data transformation
 - ◆ convert data to uniform format
 - ◆ remove & add fields (e.g., add date to get history)
- Standards (e.g., ODBC)
- Gateways

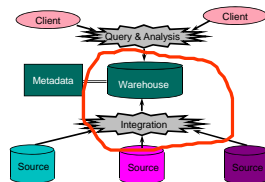
CS 245

Notes11

52

Integration

- Data Cleaning
- Data Loading
- Derived Data



CS 245

Notes11

53

Data Cleaning

- Migration (e.g., yen \leftrightarrow dollars)
- Scrubbing: use domain-specific knowledge (e.g., social security numbers)
- Fusion (e.g., mail list, customer merging)
 - billing DB \rightarrow customer1(Joe) \rightarrow merged_customer(Joe)
 - service DB \rightarrow customer2(Joe) \rightarrow merged_customer(Joe)
- Auditing: discover rules & relationships (like data mining)

CS 245

Notes11

54

Loading Data

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

CS 245

Notes11

55

Derived Data

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

CS 245

Notes11

56

Materialized Views

- Define new warehouse relations using SQL expressions

sale	prold	storeld	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	

product	id	name	price
p1	bolt	10	
p2	nut	5	

joinTb	prold	name	price	storeld	date	amt
	p1	bolt	10	c1	1	12
	p2	nut	5	c1	1	11
	p1	bolt	10	c3	1	50
	p2	nut	5	c2	1	8
	p1	bolt	10	c1	2	44
	p1	bolt	10	c2	2	4

does not exist at any source

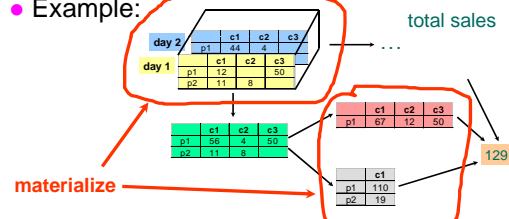
CS 245

Notes11

57

What to Materialize?

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



CS 245

Notes11

58

Materialization Factors

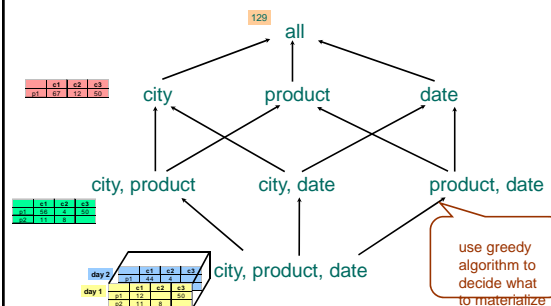
- Type/frequency of queries
- Query response time
- Storage cost
- Update cost

CS 245

Notes11

59

Cube Aggregates Lattice



CS 245

Notes11

60

Dimension Hierarchies

all

↓

state

↓

city

cities	city	state
	c1	CA
	c2	NY

CS 245
Notes11
61

Dimension Hierarchies

not all arcs shown...

CS 245
Notes11
62

Interesting Hierarchy

all

↙ ↘

weeks years

↘ ↙

days quarters

↙ ↘

 months

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

conceptual dimension table

CS 245
Notes11
63

Algorithms

- Query Optimization
- Parallel Processing
- Data Mining

CS 245
Notes11
64

Example: Association Rules

- How do we perform rule mining efficiently?

CS 245
Notes11
65

Example: Association Rules

- How do we perform rule mining efficiently?
- Observation: If set X has support t , then each X subset must have at least support t

CS 245
Notes11
66

Example: Association Rules

- How do we perform rule mining efficiently?
- Observation: If set X has support t , then each X subset must have at least support t
- For 2-sets:
 - if we need support s for $\{i, j\}$
 - then each i, j must appear in at least s baskets

CS 245

Notes11

67

Algorithm for 2-Sets

- Find OK products
 - those appearing in s or more baskets
- Find high-support pairs using only OK products

CS 245

Notes11

68

Algorithm for 2-Sets

- INSERT INTO okBaskets(basket, item)


```
SELECT basket, item
FROM Baskets
GROUP BY item
HAVING COUNT(basket) >= s;
```

CS 245

Notes11

69

Algorithm for 2-Sets

- INSERT INTO okBaskets(basket, item)


```
SELECT basket, item
FROM Baskets
GROUP BY item
HAVING COUNT(basket) >= s;
```
- Perform mining on okBaskets


```
SELECT I.item, J.item, COUNT(I.basket)
FROM okBaskets I, okBaskets J
WHERE I.basket = J.basket AND
      I.item < J.item
GROUP BY I.item, J.item
HAVING COUNT(I.basket) >= s;
```

CS 245

Notes11

70

Counting Efficiently

- One way: threshold = 3

basket	I.item	J.item
t1	p5	p8
t2	p5	p8
t2	p8	p11
t3	p2	p3
t3	p5	p8
t3	p2	p8
...

CS 245

Notes11

71

Counting Efficiently

- One way: threshold = 3

basket	I.item	J.item
t1	p5	p8
t2	p5	p8
t2	p8	p11
t3	p2	p3
t3	p5	p8
t3	p2	p8
...

sort →

basket	I.item	J.item
t3	p2	p3
t3	p2	p8
t1	p5	p8
t2	p5	p8
t3	p5	p8
t2	p8	p11
...

CS 245

Notes11

72

Counting Efficiently

- One way: threshold = 3

basket	I.item	J.item
t1	p5	p8
t2	p5	p8
t2	p8	p11
t3	p2	p3
t3	p5	p8
t3	p2	p8
...

sort

basket	I.item	J.item
t3	p2	p3
t1	p5	p8
t2	p5	p8
t3	p5	p8
t2	p8	p11
...

count & remove

count	I.item	J.item
3	p5	p8
5	p12	p18
...

CS 245 Notes11 73

Counting Efficiently

- Another way: threshold = 3

basket	I.item	J.item
t1	p5	p8
t2	p5	p8
t2	p8	p11
t3	p2	p3
t3	p5	p8
t3	p2	p8
...

CS 245 Notes11 74

Counting Efficiently

- Another way: threshold = 3

basket	I.item	J.item
t1	p5	p8
t2	p5	p8
t2	p8	p11
t3	p2	p3
t3	p5	p8
t3	p2	p8
...

scan & count

count	I.item	J.item
1	p2	p3
2	p2	p8
3	p5	p8
5	p12	p18
1	p21	p22
2	p21	p23
...

keep counter array in memory

CS 245 Notes11 75

Counting Efficiently

- Another way: threshold = 3

basket	I.item	J.item
t1	p5	p8
t2	p5	p8
t2	p8	p11
t3	p2	p3
t3	p5	p8
t3	p2	p8
...

scan & count

count	I.item	J.item
1	p2	p3
2	p2	p8
3	p5	p8
5	p12	p18
1	p21	p22
2	p21	p23
...

remove

count	I.item	J.item
3	p5	p8
5	p12	p18
...

keep counter array in memory

CS 245 Notes11 76

Counting Efficiently

- Another way: threshold = 3

basket	I.item	J.item
t1	p5	p8
t2	p5	p8
t2	p8	p11
t3	p2	p3
t3	p5	p8
t3	p2	p8
...

scan & count

count	I.item	J.item
1	p2	p3
2	p2	p8
3	p5	p8
5	p12	p18
1	p21	p22
2	p21	p23
...

remove

count	I.item	J.item
3	p5	p8
5	p12	p18
...

~~keep counter array in memory~~

CS 245 Notes11 77

Yet Another Way

- (1) scan & hash & count threshold = 3

basket	I.item	J.item
t1	p5	p8
t2	p5	p8
t2	p8	p11
t3	p2	p3
t3	p5	p8
t3	p2	p8
...

(1) scan & hash & count

count	bucket
1	A
5	B
2	C
1	D
8	E
1	F
...	...

in-memory hash table

CS 245 Notes11 78

Yet Another Way

basket	I.item	J.item
t1	p5	p8
t2	p5	p8
t2	p8	p11
t3	p2	p3
t3	p5	p8
t3	p2	p8
...

(1) scan & hash & count

count	bucket
1	A
5	B
2	C
1	D
8	E
1	F
...	...

in-memory hash table

threshold = 3

basket	I.item	J.item
t1	p5	p8
t2	p5	p8
t2	p8	p11
t3	p5	p8
t5	p12	p18
t8	p12	p18
...

(2) scan & remove

CS 245 Notes11 79

Yet Another Way

basket	I.item	J.item
t1	p5	p8
t2	p5	p8
t2	p8	p11
t3	p2	p3
t3	p5	p8
t3	p2	p8
...

(1) scan & hash & count

count	bucket
1	A
5	B
2	C
1	D
8	E
1	F
...	...

in-memory hash table

threshold = 3

basket	I.item	J.item
t1	p5	p8
t2	p5	p8
t2	p8	p11
t3	p5	p8
t5	p12	p18
t8	p12	p18
...

(2) scan & remove

CS 245 Notes11 80

Yet Another Way

basket	I.item	J.item
t1	p5	p8
t2	p5	p8
t2	p8	p11
t3	p2	p3
t3	p5	p8
t3	p2	p8
...

(1) scan & hash & count

count	bucket
1	A
5	B
2	C
1	D
8	E
1	F
...	...

in-memory hash table

threshold = 3

basket	I.item	J.item
t1	p5	p8
t2	p5	p8
t2	p8	p11
t3	p5	p8
t5	p12	p18
t8	p12	p18
...

(2) scan & remove

CS 245 Notes11 81

Yet Another Way

basket	I.item	J.item
t1	p5	p8
t2	p5	p8
t2	p8	p11
t3	p2	p3
t3	p5	p8
t3	p2	p8
...

(1) scan & hash & count

count	bucket
1	A
5	B
2	C
1	D
8	E
1	F
...	...

in-memory hash table

threshold = 3

basket	I.item	J.item
t1	p5	p8
t2	p5	p8
t2	p8	p11
t3	p5	p8
t5	p12	p18
t8	p12	p18
...

(2) scan & remove

CS 245 Notes11 82

Discussion

- Hashing scheme: 2 (or 3) scans of data
- Sorting scheme: requires a sort!
- Hashing works well if few high-support pairs and many low-support ones

CS 245 Notes11 83

Discussion

- Hashing scheme: 2 (or 3) scans of data
- Sorting scheme: requires a sort!
- Hashing works well if few high-support pairs and many low-support ones

CS 245 Notes11 84

14

Conclusions

- Massive amounts of data and complexity of queries will push limits of current warehouses
- Need better systems:
 - ◆ easier to use
 - ◆ provide quality information