

What if indices are not enough for decent online performance?

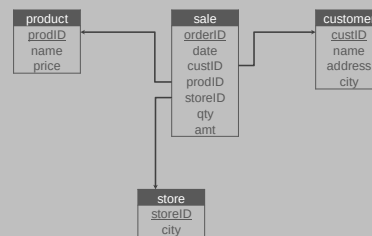
- Buy RAM
- Use a column database
In analytics queries can give a 10x easily
- Scalable, parallel processing
Mostly via noSQL
- **Precompute and use selected views**
Cheap, top results when applicable
Intricate maintenance
Applicability depends on schema and queries the user may formulate via the analytics dashboard
Star schemas and data cube analysis are a good (but not the only) target of precomputation

Last exercise on indexing: Which indices would boost Assignment 1 performance?

Star

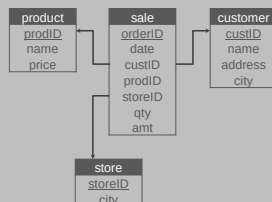
product				store		storeID	city
	prodID	name	price				
	p1	bolt	10			c1	nyc
	p2	nut	5			c2	sfo
						c3	la
sale	orderID	date	custID	prodID	storeID	qty	amt
	o100	1/7/97	53	p1	c1	1	12
	o102	2/7/97	53	p2	c1	2	11
	105	3/8/97	111	p1	c3	5	50
customer							
	custID	name	address	city			
	53	joe	10 main	sfo			
	81	fred	12 main	sfo			
	111	sally	80 willow	la			

Star Schema

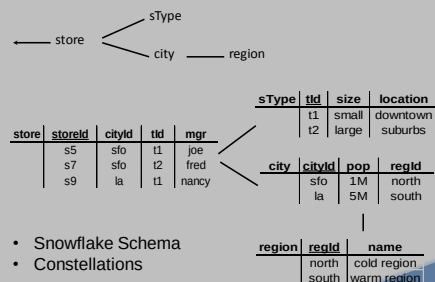


Terms

- Fact table
- Dimension tables
- Measures



Dimension Hierarchies



- Snowflake Schema
- Constellations

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

	c1	c2	c3
p1	44	4	
p2	12	8	50

dimensions = 3

ROLAP vs MOLAP

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

Aggregates

- Add up amounts for day 1
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
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
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

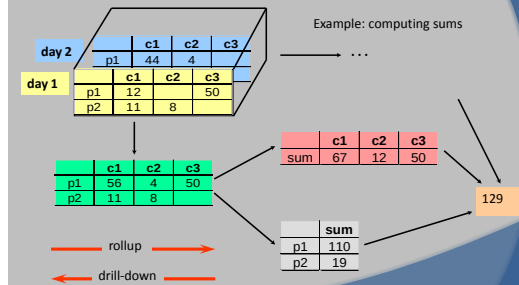
sale	prodId	date	amt
	p1	1	62
	p2	1	19
	p1	2	48

rollup →
← drill-down

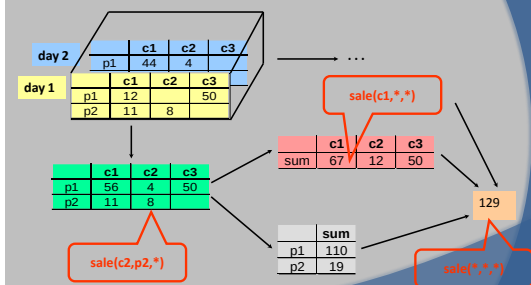
Aggregates

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

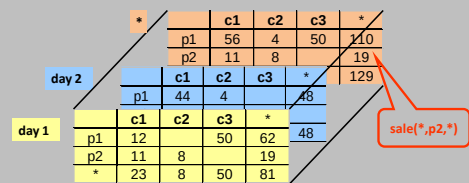
Cube Aggregation



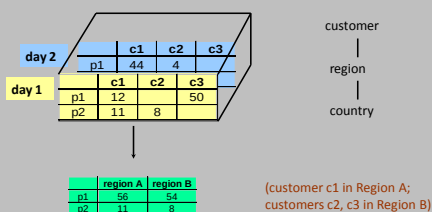
Cube Operators



Extended Cube



Aggregation Using Hierarchies

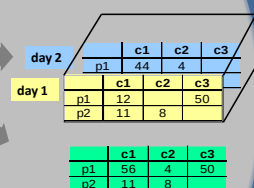


Pivoting

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



- Define new relations using SQL expressions

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

- Consider V1(Product,Customer,Sales) and V2(Product,Customer,Date,Sales) are precomputed
- ProductSales is not precomputed
- You need to answer the query

```
SELECT product, SUM(amt)
FROM SALES
GROUPBY product
```
- Write it in an alternate way, using one of the views in the most efficient way

Cube Aggregates Lattice: Choose the best nodes to materialize

View Maintenance (insert, delete cases)