Data Warehousing

Roadmap

- What is a data warehouse, data mart
- Multi-dimensional data modeling
- Data warehouse design – schemas, indices
- The Data Cube operator – semantics and computation
- Aggregate View Selection

What is Data Warehouse?

- Definition: Collection of *decision support* technologies to enable *knowledge worker* (manager, analyst) to make better and faster decisions.
- 4.7 Billion market worldwide [2006 figure, olapreport.com]
  - Retail industries: user profiling, inventory management
  - Financial services: credit card analysis, fraud detection
  - Telecommunications: call analysis, fraud detection
- **Problems:**
  - Takes too long before anything is delivered (6-24 mo.)
  - Costly: hardware, software, manpower, training (>$1M)
- **Benefits:** Cleaning the muddy wind-shield of a car.

Data Warehouse Definition

- **Definition (W.H.Inmon):** "A data warehouse is a subject-oriented, integrated, time-variant, and nonvolatile collection of data in support of management's decision-making process."
- Subject Oriented: Organized around major subjects: *customer*, *product*, *sales*
  - Focus on modeling and analysis of data for decision makers, not on daily operations or transaction processing
- Integrated: Integrate multiple, heterogeneous data sources; exclude data that are not useful in the decision support process
- Time Variant: Needs large time horizon for trend analysis (current and past data)
- Non-Volatile: Physically separate non-volatile store from the operational environment

Data Marts

- A *subset* of what would be in a data warehouse
- Used for a single department, division or geographical location
- Much cheaper than implementing a complete data warehouse
- Can be used as "proof of concept"
- Data marts can co-exist with a data-warehouse
- Multiple data marts should be integrated to ensure consistency and synchronization

Why not Using Existing DB?

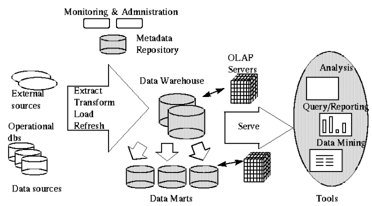
- DBMS is for On Line Transaction Processing (OLTP)
  - automate day-to-day operations (purchasing, banking etc)
- Data Warehouse is for On Line Analytical Processing (OLAP)
  - need historical data for trend analysis

OLTP vs. OLAP

	OLTP	OLAP
users	clerk, IT professional	knowledge worker
function	day to day operations	decision support
DB design	application-oriented	subject-oriented
data	current, up-to-date detailed, flat relational isolated	historical, summarized, multidimensional integrated, consolidated
usage	repetitive	ad-hoc
access	read/write index/hash on prim. key	lots of scans
unit of work	short, simple transaction	complex query
# records accessed	tens	millions
#users	thousands	hundreds
DB size	100MB-GB	100GB-TB
metric	transaction throughput	query throughput, response

Data Warehouse Architecture

- Extract data from operational data sources
  - clean, transform
- Bulk load/refresh
  - warehouse is offline
- OLAP-server provides multidimensional view



Examples of OLAP

- Comparisons (this period v.s. last period)
  - Show me the sales per store for this year and compare it to that of the previous year to identify discrepancies
- Ranking and statistical profiles (top N/bottom N)
  - Show me sales, profit and average call volume per day for my 10 most profitable salespeople
- Custom consolidation (market segments, ad hoc groups)
  - Show me an abbreviated income statement by quarter for the last four quarters for my northeast region operations

Roadmap

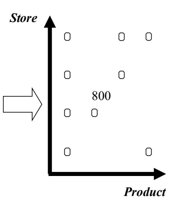
1. What is the data warehouse, data mart
2. Multi-dimensional data modeling
3. Data warehouse design – schemas, indices
4. The Data Cube operator – semantics and computation
5. Aggregate View Selection

Multidimensional Modeling

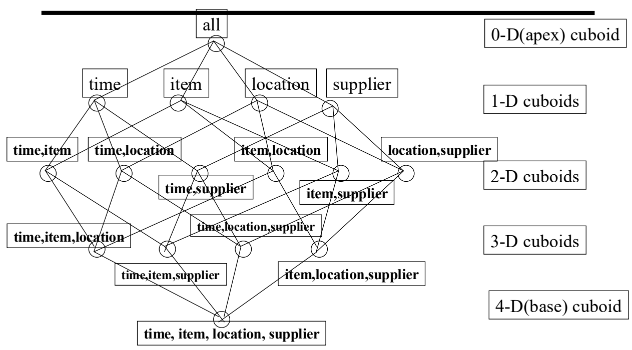
- Example: compute total *sales* volume per *product* and *store*

Store	Product	Total Sales
1	1	454
1	4	925
2	1	468
2	2	800
Etc.		

Total Sales		Product 1	2	3	4
Store	1	454	-	-	925
	2	468	800	-	-
	3	296	-	240	-
	4	652	-	540	745

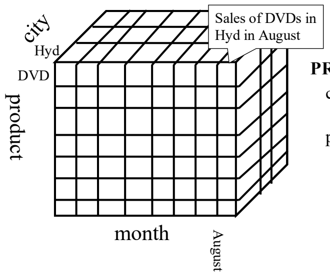


Cube: A Lattice of Cuboids



Dimensions and Hierarchies

- A cell in the cube may store values (measurements) relative to the combination of the labeled dimensions



DIMENSIONS

PRODUCT	**LOCATION**	**TIME**
category	region	year
product	country	quarter
	state	month, week
	city	day
	store	

Common OLAP Operations

- Roll-up: **move up the hierarchy**
  - **e.g given total sales per city, we can roll-up to get sales per state**
- **Drill-down: move down the hierarchy**
  - **more fine-grained aggregation**

PRODUCT	**LOCATION**	**TIME**
category	region	year
product	country	quarter
	state	month, week
	city	day
	store	

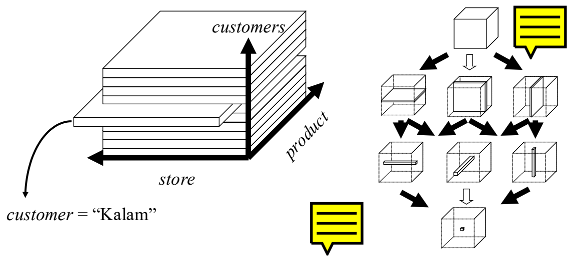
Pivoting

- Pivoting: aggregate on selected dimensions
  - usually 2 dims (cross-tabulation)

Sales		Product				
		1	2	3	4	ALL
Store	1	454	-	-	925	1379
	2	468	800	-	-	1268
	3	296	-	240	-	536
	4	652	-	540	745	1937
	ALL	1870	800	780	1670	5120

Slice and Dice Queries

- Slice and Dice: select and project on one or more dimensions



Roadmap

1. What is the data warehouse, data mart
2. Multi-dimensional data modeling
3. Data warehouse design – schemas, indices
4. The Data Cube operator – semantics and computation
5. Aggregate View Selection

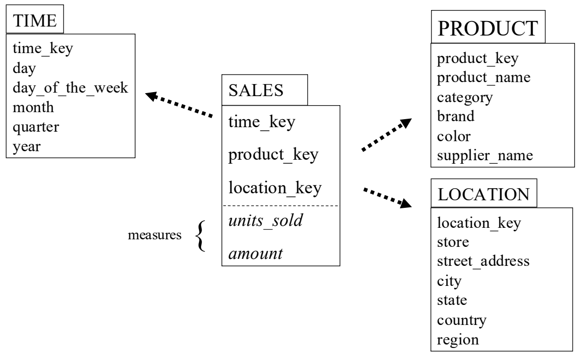
Data Warehouse Design

- ROLAP
  - Store data in RDBMS
  - Provide a multi-dimensional view of this data
  - Makes use of existing technology
  - Products: Redbrick, Informix, Sybase, SQL server
- MOLAP
  - Directly implement multi-dimensional model
  - Uses arrays
  - Lots of compression for sparse arrays
  - Products: Essbase, Oracle Express

ROLAP Schemas

- Most data warehouses adopt a *star schema* to represent the multidimensional model
- Each dimension is represented by a *dimension-table*
  - LOCATION(location_key,store,street_address,city,state,country,region)
  - dimension tables are not normalized
- Transactions are described through a *fact-table*
  - each tuple consists of a pointer to each of the dimension-tables (foreign-key) and a list of measures (e.g. sales)
- Snowflake Schema: Same as above, but dimension tables are normalized to provide explicit support for attribute hierarchies

The Star Schema



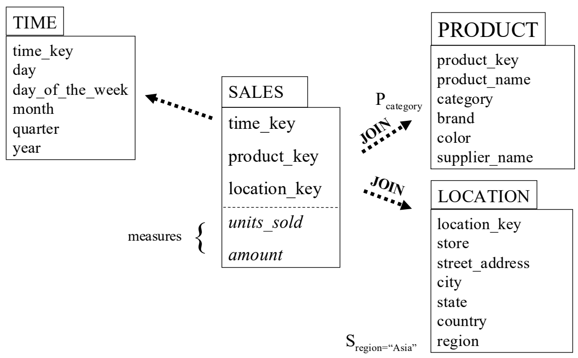
Advantages of Star Schema

- Facts and dimensions are clearly depicted
  - dimension tables are relatively static, data is loaded (append mostly) into fact table(s)
  - easy to comprehend (and write queries)

"Find total sales per product-category in our stores in Asia"

SELECT PRODUCT.category, SUM(SALES.amount)
FROM SALES, PRODUCT,LOCATION
WHERE SALES.product_key = PRODUCT.product_key
AND SALES.location_key = LOCATION.location_key
AND LOCATION.region="Asia"
GROUP BY PRODUCT.category

Star Schema Query Processing



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

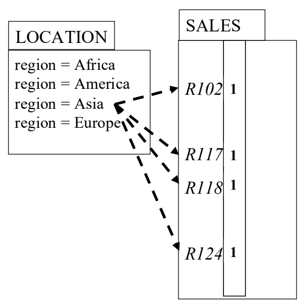
RecID	Asia	Europe	America
1	1	0	0
2	0	1	0
3	1	0	0
4	0	0	1
5	0	1	0

Index on Type

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

Join-Index

- Join index relates the values of the dimensions of a star schema to rows in the fact table.
  - a join index on *region* maintains for each distinct region a list of ROW-IDs of the tuples recording the sales in the region
- Join indices can span multiple dimensions OR
  - can be implemented as bitmap-indexes (per dimension)
  - use bit-op for multiple-joins



Unresolved: Coarse-grain Aggregations

- *"Find total sales per product-category in our stores in Asia"*
  - Join-index will prune ¾ of the data (uniform sales), but the remaining ¼ is still large (several millions transactions)
    - Index is unclustered
- High-level aggregations are expensive!!!!!

⇒Long Query Response Times

⇒Pre-computation is necessary

LOCATON

region
|
country
|
state
|
city
|
store

Unresolved: Multiple Simultaneous Aggregates

<u>4 Group-bys here:</u>
(store,product)
(store)
(product)
()
Need to write 4 queries!!!

Cross-Tabulation (products/store)

Sales		Product				
		1	2	3	4	ALL
Store	1	454	-	-	925	1379
	2	468	800	-	-	1268
	3	296	-	240	-	536
	4	652	-	540	745	1937
	ALL	1870	800	780	1670	5120

Sub-totals per store

Total sales

Sub-totals per product

Roadmap

1. What is the data warehouse, data mart
2. Multi-dimensional data modeling
3. Data warehouse design – schemas, indices
4. The Data Cube operator – semantics and computation
5. Aggregate View Selection

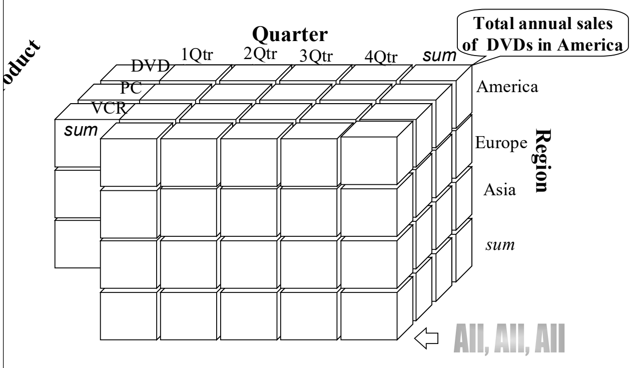
The Data Cube Operator (Gray et al)

- All previous aggregates in a single query:

SELECT LOCATION.store, SALES.product_key, SUM (amount)
FROM SALES, LOCATION
WHERE SALES.location_key=LOCATION.location_key
CUBE BY SALES.product_key, LOCATION.store
OR
CUBE product_key, store **BY** SUM(SALES.amount)

<u>Challenge</u>: Optimize Aggregate Computation

Data Cube: Multidimensional View



Relational View of Data Cube

Sales		Product				
		1	2	3	4	ALL
Store	1	454	-	-	925	1379
	2	468	800	-	-	1268
	3	296	-	240	-	536
	4	652	-	540	745	1937
	ALL	1870	800	780	1670	5120

SELECT LOCATION.store, SALES.product_key, SUM (amount)
FROM SALES, LOCATION
WHERE SALES.location_key=LOCATION.location_key
CUBE BY SALES.product_key, LOCATION.store

Store	Product_key	sum(amout)
1	1	454
1	4	925
2	1	468
2	2	800
3	1	296
3	3	240
4	1	625
4	3	240
4	4	745
1	ALL	1379
2	ALL	1268
3	ALL	536
4	ALL	1937
ALL	1	1870
ALL	2	800
ALL	3	780
ALL	4	1670
ALL	ALL	5120

Other Extensions to SQL

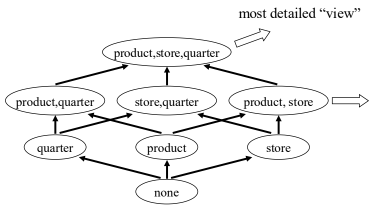
- Complex aggregation at multiple granularities (Ross et. all 1998)
  - Compute *multiple dependent* aggregates

```
SELECT LOCATION.store, SALES.product_key, SUM (amount)
FROM SALES, LOCATION
WHERE SALES.location_key=LOCATION.location_key
CUBE BY SALES.product_key, LOCATION.store: R
SUCH THAT R.amount = max(amount)
```

- Other proposals: the MD-join operator (Chatziantoniou et. all 1999]

Data Cube Computation

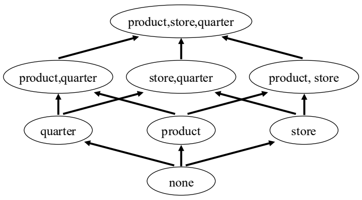
- Model dependencies among the aggregates:



can be computed from *view* (product,store,quarter) by summing-up all quarterly sales

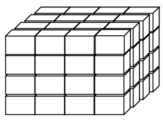
Computation Directives

- Hash/sort based methods (Agrawal et. al. VLDB'96)
  1. **Smallest-parent**
  2. **Cache-results**
  3. **Amortize-scans**
  4. **Share-sorts**
  5. **Share-partitions**



Alternative Array-based Approach

- Model data as a sparse multidimensional array
  - partition array into chunks (a small sub-cube which fits in memory).
  - fast addressing based on (chunk_id, offset)
- Compute aggregates in "multi-way" by visiting cube cells in the order which minimizes the # of times to visit each cell, and reduces memory access and storage cost.



What is the best traversing order to do multi-way aggregation?

Roadmap

- What is the data warehouse, data mart
- Multi-dimensional data modeling
- Data warehouse design
  - the star schema, bitmap indexes
- The Data Cube operator
  - semantics and computation
- Aggregate View Selection

Views and Decision Support

- OLAP queries are typically aggregate queries.
  - Pre-computation is essential for interactive response times.
  - The CUBE is in fact a collection of aggregate queries, and pre-computation is especially important: lots of work on what is best to pre-compute given a limited amount of space to store pre-computed results.
- Warehouses can be thought of as a collection of asynchronously replicated tables and periodically maintained views.
  - Has renewed interest in view maintenance!

View Modification (Evaluate On Demand)

View

```
CREATE VIEW RegionalSales(category,sales,state)
        AS SELECT P.category, S.sales, L.state
            FROM Products P, Sales S, Locations L
            WHERE P.pid=S.pid AND S.locid=L.locid
```

Query

```
SELECT R.category, R.state, SUM(R.sales)
FROM RegionalSales AS R GROUP BY R.category, R.state
```

Modified Query

```
SELECT R.category, R.state, SUM(R.sales)
FROM (SELECT P.category, S.sales, L.state
        FROM Products P, Sales S, Locations L
        WHERE P.pid=S.pid AND S.locid=L.locid) AS R
GROUP BY R.category, R.state
```


View Materialization (Pre-computation)

- Suppose we pre-compute RegionalSales and store it with a clustered B+ tree index on [category,state,sales].
  - Then, previous query can be answered by an index-only scan.

```
SELECT R.state, SUM(R.sales)
FROM RegionalSales R
WHERE R.category="Laptop"
GROUP BY R.state
```

Index on pre-computed view is great!

```
SELECT R.state, SUM(R.sales)
FROM RegionalSales R
WHERE R. state="Wisconsin"
GROUP BY R.category
```

Index is less useful (must scan entire leaf level).

Materialized Views

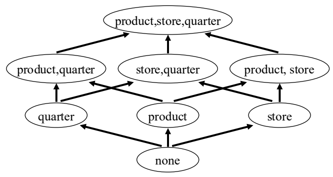
- A view whose tuples are stored in the database is said to be materialized.
  - Provides fast access, like a (very high-level) cache.
  - Need to maintain the view as the underlying tables change.
  - Ideally, we want incremental view maintenance algorithms.
- Close relationship to data warehousing, OLAP, (asynchronously) maintaining distributed databases, checking integrity constraints, and evaluating rules and triggers.

Issues in View Materialization

- Algorithm to maintain a materialized view?
- What views should we materialize, and what indexes should we build on the pre-computed results?
- Given a query and a set of materialized views (possibly with some indexes), can we use the materialized views to answer the query?

View Selection Problem

- Use some notion of *benefit* per view
- Limit: *disk space*

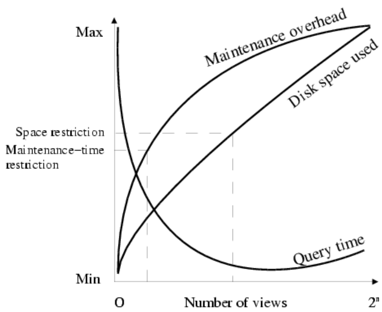


Hanarayan et al SIGMOD'96:

$$B(v,S)=\sum_{u: u\le v, C_v(u)<C_S(u)} (C_S(u)-C_v(u))$$

Pick views greedily until space is filled

Reality check:too many views!



[note: original graph at http://www.olapreport.com]

- 2^n views for n dimensions (no-hierarchies)
- Storage/update-time explosion
- More pre-computation doesn't mean better performance!!!!

Problem Generalization

- Materialize and maintain the right subset of views with respect to the workload and the available resources
- What is the workload?
  - “Farmers” v.s. “Explorers” [Inmon99]
  - Pre-compiled queries (report generating tools, data mining)
  - Ad-hoc analysis (unpredictable)
- What are the resources?
  - Disk space (getting cheaper)
  - Update window (getting smaller)

View Selection Problem

- Selection is based on a workload estimate (e.g. logs) and a given constraint (disk space or update window)
- NP-hard, optimal selection can not be computed > 4-5 dimensions
  - greedy algorithms (e.g. [Harinarayan96]) run at least in polynomial time in the number of views i.e exponential in the number of dimensions!!!
- Optimal selection can not be approximated [Karloff99]
  - greedy view selection can behave arbitrary bad
- Alternatives: use query result caching techniques and reuse prior computations

View Maintenance

- Two steps:
  - Propagate: Compute changes to view when data changes.
  - Refresh: Apply changes to the materialized view table.
- Maintenance policy: Controls when we do refresh.
  - Immediate: As part of the transaction that modifies the underlying data tables. (+ Materialized view is always consistent; - updates are slowed)
  - Deferred: Some time later, in a separate transaction. (- View becomes inconsistent; + can scale to maintain many views without slowing updates)

Deferred Maintenance

- Three flavors:
  - Lazy: Delay refresh until next query on view; then refresh before answering the query.
  - Periodic (Snapshot): Refresh periodically. Queries possibly answered using outdated version of view tuples. Widely used, especially for asynchronous replication in distributed databases, and for warehouse applications.
  - Event-based: E.g., Refresh after a fixed number of updates to underlying data tables.

Sources of Information - Books

1. The Data Warehouse Toolkit – Ralph Kimball ISBN 0-471-15337-0
2. The Data Warehouse Lifecycle Toolkit – Ralph Kimball, Laura Reeves, Warren Thornthwaite & Margy Ross ISBN 0-471-25547-5
3. Data Warehouse Design Solutions – Christopher Adamson & Michael Venerable ISBN 0-471-25195-X

Sources of Information – Web Sites

Technology Guides for Data Warehousing - www.techguide.com

Ralph Kimball Associates Articles - www.ralphkimball.com/html/articles.html

Data Warehousing - Data Warehousing Knowledge Center - www.datawarehousing.org

The Data Warehousing Information Center - www.dwinfocenter.org

Documenting data replication and data transformation sites on the Net - www.datawarehousing.com

DM Review Business Intelligence & Data Warehousing Enabling E-Business - www.dmreview.com/

The Data Warehousing Institute - www.dw-institute.com

Intelligent Enterprise Magazine - www.intelligententerprise.com/

DataWarehousing Forum - www.datawarehousing.com/forum/

Sources of Information - ListServer

Subscribing:
To subscribe to the data warehousing list server:

- Send an E-mail to dwlist-request@datawarehousing.com
- The first line of your message must be "subscribe".
- No subject line is required.

Unsubscribing:
To unsubscribe from the data warehousing list server:

- Send an E-mail to dwlist-request@datawarehousing.com
- The first line of your message must be "unsubscribe".
- No subject line is required.

Posting Messages:

- To post a message, send your message by email to dwlist@datawarehousing.com

The End

Thank you!