## Data Warehousing and Decision Support (mostly using Relational Databases) CS634 Class 20

Slides based on "Database Management Systems" 3rd ed, Ramakrishnan and Gehrke, Chapter 25

#### Introduction

- Increasingly, organizations are analyzing current and historical data to identify useful patterns and support business strategies.
- Emphasis is on complex, interactive, exploratory analysis of very large datasets created by integrating data from across all parts of an enterprise
  - Contrast such Data Warehousing and On-Line Analytic Processing (OLAP) with traditional On-line Transaction Processing (OLTP): mostly long queries, instead of the short update Xacts of OLTP.
  - In past, both were using "structured data" that can be fairly easily loaded into a database
  - Today, businesses also monitor social media, web clicks, etc., which are not properly structured, hard to put in RDB.

#### Structured vs. Unstructured Data

- So far, we have been working with structured data
- Structured data:
  - Entities with attributes, each fitting a SQL data type
  - Relationships
  - Each row of data is precious
  - Loads into relational tables, long-term storage
  - Can be huge
- Unstructured data, realm of "big data"
  - Often doesn't fit into E/R model, too sloppy
  - Each piece of data is not precious—it's statistical
  - Sometimes just processed and thrown away
  - No permanent specialized repository, maybe saved in files
  - Can be really huge

# Data Warehouses using RDB vs. Data Lakes using Hadoop

- Both are ways to hold huge amounts of data
- Data lakes hold "big data", use big data techniques to query and analyze data. Hadoop provides a high-availability scalable distributed systems.
- Big data can be original, uncleaned data, vs. cleaned data for RDB systems.
- A data lake can hold both original and cleaned data. Term "data lake" was invented in 2011, i.e., around same time as release of Hadoop.
- RDB Data warehouse technology ends up with data in a form easily understood by business people
- Big data is not there yet: usually need "data scientists" to interpret the data, write the queries, or at least new queries.
- Of course this is changing...
- Many big businesses have both a traditional data warehouse and a data lake, load some of same data in both <u>Datamation article</u>
- We may reserve "data warehouse" without adjective to encompass both RDB data warehouses and big-data warehouses that provide user-friendly access methods

### **Bigness of Data**

Huge Data warehouses, all on Teradata systems (hard to find current sizes)

See <u>article</u>

- Biggest DW: Walmart, passed 1TB in 1992, 2.8 PB (petabytes) = 2800 TB in 2008, 30 PB in 2014, 40+ PB in 2017, processing 2.5PB/hour, growing...
- eBay: 9 PB DW in 2013, also has 40 PB of big data, uses Hadoop, etc.
- Apple: multiple-PB DW
- Big data:
  - Usually over 50TB, can't fit on one machine
  - Is judged by "velocity" as well as size
  - Google: processed 24 PB of data per day in 2009, invented Map-Reduce, published 2004

#### Teradata

- Teradata provides a relational database with ANSI compliant SQL, targeted to data warehouses
- Proprietary, expensive (\$millions)
- Uses a shared-nothing architecture on many independent nodes
- Partitioning by rows or (more recently) columns
- Scales up well: add node, add network bandwidth for it
- Now supports Hadoop as well as RDBMS: Teradata Appliance for Hadoop

#### Three Complementary Trends

- Data Warehousing: Consolidate data from many sources in one large repository (relational database or data lake).
  - Loading, periodic synchronization of data.
  - Semantic integration, Data cleaning of data on way in (RDB only so far)
  - Both simple and complex queries and views. (SQL or programmed)
  - Note: SQL is available on top of big data in most systems
- OLAP/Multidimensional Analysis
  - Queries based on spreadsheet-style operations and "multidimensional" view of data. Interactive queries. Look at data from different directions, granularity, etc.
  - Big Data Example: Apache Kylin, originally from eBay, available 2017

Data Mining: Exploratory search for interesting trends and anomalies.

Note: BI = Business intelligence, analysis of business information, includes OLAP and data mining

#### **EXTERNAL DATA SOURCES**

## Data Warehousing

- Integrated data spanning long time periods, often augmented with summary information.
- Several gigabytes to terabytes common, now petabytes too.
- Interactive response times expected for complex queries; ad-hoc updates uncommon.
- Read-mostly data



#### Warehousing Issues

- Semantic Integration: When getting data from multiple sources, must eliminate mismatches, e.g., different currencies, schemas.
- Heterogeneous Sources: Must access data from a variety of source formats and repositories.
  - Replication capabilities can be exploited here.
- Load, Refresh, Purge: Must load data, periodically refresh it, and purge too-old data.
- Metadata Management: Must keep track of source (lineage) loading time, and other information for all data in the warehouse.

#### OLAP: Multidimensional data model

- A way to make complex data understandable by business user, etc.
- Example: sales data
- Dimensions: Product, Location, Time
- A measure is a numeric value like sales we want to understand in terms of the dimensions
- Example measure: dollar sales value "sales"
- Example data point (one row of fact/cube table):
  - Sales = 25 for pid=1, timeid=1, locid=1 is the sum of sales for that day, in that location, for that product
  - Pid=1: details in Product table
  - Locid = 1: details in Location table
- Note aggregation here: sum of sales is most detailed data (but can have all sales data)

## Multidimensional Data Model

SalesCube(pid, timeid, locid, sales)

- Collection of numeric <u>measures</u>, which depend on a set of dimensions.
  - E.g., measure sales, dimensions Product (key: pid), Location (locid), and Time (timeid).
  - Full table, pg. 851

is shown:



	pid	timeid	locid	sales
-	11	1	1	25
	11	2	1	8
	11	3	1	15
	12	1	1	30
	12	2	1	20
	12	3	1	50
	13	1	1	8
	13	2	1	10
	13	3	1	10
	11	1	2	35

#### Granularity of Data

- Example of last slide uses time at granularity of days
- Individual transactions (sales at cashier) have been added together to make one row in this table
- Note: "measures" can always be aggregated
- Current hardware can handle more data
- Typical data warehouses hold the original transaction data
- So such a fact table has more columns, for example
- dateid, timeofday, prodid, storeid, txnid, clerkid, sales, ...

#### Data Warehouse vs. Data for OLAP

- Current DW fact table is huge, with individual transactions, large number of dimensions
- Can only use a subset of this for OLAP, because of explosion of cells
- Take DW fact table, roll up to days (say), drop less important columns, get much smaller data for OLAP
- Load data into OLAP, another tool.
- Table on pg. 851 is a cube table, not a DW fact table
- Can think of OLAP as a cache of most important aggregates of DW tables

#### Dimension Hierarchies: OLAP, DW

For each dimension, the set of values can be organized in a hierarchy:



## Schema underlying OLAP, used in RDB DW



- Fact/cube table in BCNF; dimension tables not normalized.
  - Dimension tables are small; updates/inserts/deletes are rare. So, anomalies less important than good query performance.
- This kind of schema is very common in DW and OLAP, and is called a star schema; computing the join of all these relations is called a star join.
- Note: in OLAP, this is not what the user sees, it's hidden underneath
- In DW, this is the basic setup, but usually with more dimensions
- Here only one measure, sales, but can have several

#### OLAP (and DW) Queries

- Influenced by SQL and by spreadsheets.
- A common operation is to <u>aggregate</u> a measure over one or more dimensions.
  - Find total sales.
  - Find total sales for each city, or for each state.
  - Find top five products ranked by total sales.
- <u>Roll-up</u>: Aggregating at different levels of a dimension hierarchy.
  - E.g., Given total sales by city, we can roll-up to get sales by state.

# OLAP Queries: MDX (**Multidimensional Expressions**)

- Originally a Microsoft SQL Server project, but now supported widely in the OLAP industry: Oracle, SAS, SAP, Teradata on server side, as well as Microsoft. Allows client programs to specify OLAP datasets.
- Example from Wikipedia

#### SELECT

- { [Measures].[Store Sales] } ON COLUMNS,
- { [Date].[2002], [Date].[2003] } ON ROWS

#### FROM Sales

```
WHERE ( [Store].[USA].[CA] )
```

- The SELECT clause sets the query axes as the Store Sales member of the Measures dimension, and the 2002 and 2003 members of the Date dimension.
- The FROM clause indicates that the data source is the Sales cube.
- The WHERE clause defines the "slicer axis" as the California member of the Store dimension.

#### **OLAP** Queries

- Drill-down: The inverse of roll-up: go from sum to details that were added up before
  - E.g., Given total sales by state, can drill-down to get total sales by county.
  - Drill down again, see total sales by city
  - E.g., Can also drill-down on different dimension to get total sales by product for each state.

#### OLAP Queries: cross-tabs

With relational DBs, we are used to tables with column names across the top, rows of data.

With OLAP, a spreadsheet-like representation is common,

Called a <u>cross-tabulation</u>:

- One dimension horizontally
- Another vertically

	WI	CA	Total
1995	63	81	144
1996	38	107	145
1997	75	35	110
Total	176	223	339

#### **OLAP Queries: Pivoting**

Example cross-tabulation:

	WI	CA	Tota	l
1995	63	81	144	
1996	38	107	145	
<b>1997</b>	75	35	110	
Total	176	223	339	

- Pivoting: switching dimensions on axes, or choosing what dimensions to show on axes
- Switching dimensions means pivoting around a point in the upper-left-hand corner
  - End up with "1995 1996 1997 Total" across top,
  - "WI CA Total" down the side

			WI	CA	Total
SQL Queries for cross-tak	entries	1995	63	81	144
The cross tabulation values can	1996	38	107	145	
using a collection of SQL quer	1997	75	35	110	
<b>SELECT SUM</b> (S.sales)		Total	176	223	339
<b>FROM</b> Sales S, Times T, Locations L <b>WHERE</b> S.timeid=T.timeid <b>AND</b> S.timeid=L.timeid <b>GROUP BY</b> T.year, L.state					
SELECT SUM(S.sales) FROM Sales S, Times T WHERE S.timeid=T.timeid	SELECT SUM(S.sales) FROM Sales S, Location L WHERE S.timeid=L.timeid CROUP BY L state			-	

## The CUBE Operator

- Generalizing the previous example, if there are k dimensions, we have 2^k possible SQL GROUP BY queries that can be generated through pivoting on a subset of dimensions.
- CUBE Query, pg. 857

SELECT T.year, L.state, SUM(S.sales)
FROM Sales S, Times T, Locations L
WHERE S.timeid = T.timeid and S.locid = L.locid
GROUP BY CUBE (T.year, L.state)

 Equivalent to rolling up Sales on all eight subsets of the set {pid, locid, timeid}; each roll-up corresponds to an SQL query of the form:

SELECT SUM(S.sales)FROMSales SGROUP BY grouping-list

Oracle 10+ Supports CUBE queries select t.year, s.store\_state, sum(dollar\_sales) from salesfact f, times t, store s where f.time\_key = t.time\_key and s.store\_key = f.store\_key group by cube(t.year, s.store state);

YEAR	STORE_STATE	SUM (DOLLAR_SALES)
	·	
		781403.59
	AZ	35684
	CA	77420.82
	со	38335.26 (some rows deleted)
	TX	40886.54
1994	WA	39540.16 396355.76
1994	AZ	17903.04
1994	CA	38966.54
1994	со	17870.33
1994	DC	20901.18 from dbs2 output

#### Oracle 11+ supports cross-tabs display

#### select cool, stars, count(\*) from yelp db.reviews Running on dbs3 (Oracle version 12): where stars in (2,3,4,5)SQL> select \* from ( group by cool, stars order by cool, stars; 2 select cool, stars from yelp db.review COOL STARS COUNT(\*) 3 ) pivot ( 4 count(stars) 0 2 323533 3 421229 0 5 for stars in (2, 3, 4, 5)0 4 787637 ) order by cool; 6 5 0 1516269 1 2 51358 Here is the output: ••• 2 3 5 COOL 4 323533 421229 787637 1516269 0 51358 88168 198705 300811 1 2 13812 27798 66019 84758 5116 11690 3 28468 31867 4 2455 5979 14690 15452

Same data, relationally:

... and so on ...

This says 323533 reviews awarded 2 stars but got no "cool" ratings

#### DW data $\rightarrow$ OLAP

- The CUBE query can do the roll-ups on DW data needed for OLAP
- Excel is the champ at OLAP queries
- Look at <u>video</u>
- This video shows pivot tables for a single Excel worksheet
- But Excel can work with database tables: see this longer video
- Pivot tables: drill down, roll up, pivot, ...