DATA WAREHOUSING

INTRODUCTION TO DATA SCIENCE

CARSTEN BINNIG
BROWN UNIVERSITY
**Definition:** A data warehouse is a database that is optimized for analytical workloads which integrates data from independent and heterogeneous data sources.
ENTERPRISE SCENARIO: WHOLEFOODS

Business Questions:
- What are the bestselling products?
- Is there difference between states?
- ...

Data Warehouse
OTHER APPLICATION DOMAINS

Restaurant Chains (McDonalds, etc.)

Retailers (Nike, ...)

Insurance Companies

Banks

...
HISTORY OF DATABASES

Age of Online Transaction Processing - OLTP (> 1970)
• **Goal**: have access to up-to-date business transactions
• 60s: IMS (hierarchical data model) => financial domain
• 80s: Oracle (relational data model) => most other domains (ERP, CRM)

Age of Online Analytical Processing - OLAP (> 1990)
• **Goal**: make business decisions
• 90s: Data-warehousing extensions to relational databases
• Recently: New systems like in-memory column stores
OLTP VS. OLAP

Online Transaction Processing (OLTP)

• Current state of data is important

• Queries read / update only few records; AKA point queries or CRUD workloads (Create, Read, Update, Delete)

• Data Modeling: Avoid redundancy, normalize schemas

Goal: High throughput of transactions (Oracle 1995)
EXAMPLE: CUSTOMER DATA (OLTP)

CREATE
READ
UPDATE
DELETE
OLTP VS. OLAP

Online Analytical Processing

• History of data is important (not only the current state)
• Big queries (aggregate data, joins);
  • No Updates, only bulk loads
  • Data freshness is not that important!
• Modeling: Redundancy is a feature (i.e., de-normalized schemas are preferred)

Goal: Low latency of “big” queries (<= 500ms)
EXAMPLE: REVENUE (OLAP)

Revenue By Product - Q4, 2014
- iPhone: 56%
- iPad: 13%
- Mac: 16%
- iPod: 1%
- iTunes/Software/Services: 11%
- Accessories: 3%

Revenue by region – Q4, 2014
- America: 48%
- China: 38%
- Europe: 8%
- Asia and Others: 6%
THE BIG PICTURE

Data Sources

Load

Business Intelligence and Analytics

Data Sources

ETL Extract Transform Load

Load

Data Warehouse

Business Intelligence and Analytics

Data Products

Business Intelligence

Analytics
FOR THE BUSINESS PERSON

Data Sources
• CSV files

ETL
• Copy and paste to Excel
• References + functions

Data Warehouse
• Excel Sheets

Business Intelligence and Analytics
• Excel functions
• Excel charts
FOR THE BUSINESS PERSON

Data Sources
• Web scraping, web services API
• Databases

ETL
• Visual transformation tools
• Informatica, IBM DataStage, Ab Initio, Talend

Data Warehouse
• Teradata, Oracle, IBM DB2, Microsoft SQL Server

Business Intelligence and Analytics
• Business Objects, Cognos, Microstrategy
• SAS, SPSS, R
FOR THE “HIP” WEB ENTERPRISE

Data Sources

• Logs from the services tier
• User clicks, user comments, web crawl data...

ETL

• Flume, Sqoop, Pig,/Crunch, Oozie (Workflow Scheduler)
• Hadoop/Hive, Spark/SparkSQL

Business Intelligence and Analytics

• Custom web-based dashboards
• R
DATA WAREHOUSING STEPS

Data Modeling → Data Integration → Querying
DATA WAREHOUSING STEPS

Data Modeling -> Data Integration -> Querying
A MORE TECHNICAL VIEW

Goal: Integration

ETL

Data Warehouse

One integrated schema

Goal: Performance

Prepare

Cubes

Derived analytical schemata

Multiple independent schemata
CUBE: MULTIDIMENSIONAL DATA MODEL

Product (Product)

Axis -> Dimensions

Balls

Beer

Data -> Key Figures

Data: 100 0 200

2010 2011 2012 2013

Customer
MULTIDIMENSIONAL DATA (CUBE)

Revenue for customer c3 in year 2013

Product

Date

Customer

<table>
<thead>
<tr>
<th></th>
<th>c1</th>
<th>c2</th>
<th>c3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Balls</td>
<td>100</td>
<td>0</td>
<td>200</td>
</tr>
<tr>
<td>Beer</td>
<td>0</td>
<td>1000</td>
<td>100</td>
</tr>
</tbody>
</table>

Revenue for customer c3 in year 2013
DATA OPERATIONS

**Slice:** Cut a slice out of the cube (e.g. `product=„Beer“`)

**Dice:** Cut a smaller cube out of the data (e.g. `product=„Beer“ and year=2013`)

**Drill-down:** Show details on the next level of detail (e.g., zoom into from sales per month to sales per week)

**Roll-up:** Aggregate data along a hierarchy (e.g., zoom out from sales per month to sales per year)
**RELATIONAL OLAP (ROLAP)**

**ROLAP:** Store multidimensional cube data in a *relational* database

**Star-Schema:**

**Fact table:** Store key figures (e.g., revenue, number of products sold, margin, ...)

**Dimension tables:** Store values on the axis!
STAR-SCHEMA (ROLAP)

- **Fact table (Sales)**
- **Dimension table (Customer)**
- **Dimension table (Product)**
- **Dimension table (PointOfSales)**
- **Dimension table (Time/Date)**
DIMENSION TABLE

Data in dimension tables:

• Distinct values of one axis of the cube (e.g. dates, product names, …)
• Many different data types (texts, dates, …)
• Often de-normalized (why?)

One dimension table is typically the **Time / Date table**

**Used to …**

• **Select data in fact table** (e.g. revenue in 2011): by joining dimension table with fact table
• **Group results** (e.g., revenue grouped by year)
## EXAMPLE: DIMENSION TABLE (CUSTOMER)

<table>
<thead>
<tr>
<th>custkey</th>
<th>lastName</th>
<th>firstName</th>
<th>city</th>
<th>Country</th>
<th>region</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Binnig</td>
<td>Carsten</td>
<td>Mannheim</td>
<td>Germany</td>
<td>Europe</td>
</tr>
<tr>
<td>2</td>
<td>Tellex</td>
<td>Stephanie</td>
<td>Providenc e</td>
<td>USA</td>
<td>North America</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

**PK**  
Attributes for selection and grouping
FACT TABLE

Data in fact tables

• Numeric key figures for aggregation e.g. revenue
• Foreign keys to dimensions (tables: customer, Product, date, ...)
• Mostly numeric data

Key figures are used for aggregations (e.g., total of orders, quantity of sales)

Data in fact table is constantly growing!

Primary key of fact table: Composed of all foreign keys
### EXAMPLE: FACT TABLE (SALES)

<table>
<thead>
<tr>
<th>custkey</th>
<th>productkey</th>
<th>datekey</th>
<th>...</th>
<th>revenue</th>
<th>quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td>1000</td>
<td>10</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>3</td>
<td></td>
<td>800</td>
<td>9</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td></td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Foreign Keys to dimension tables  Key figures
Dimensions often describe a hierarchy (i.e., 1:N relationships between entities)

**Static hierarchies:** Levels in hierarchy is fixed (e.g. Year->Month->Day or Region->Country->City)

**Flexible hierarchies:** Dynamic number of levels (e.g. management hierarchies, bill of materials – BOM)
**STATIC HIERARCHIES**

Levels of a hierarchy are represented by different columns

<table>
<thead>
<tr>
<th>City</th>
<th>Country</th>
<th>Region</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mannheim</td>
<td>Germany</td>
<td>Europe</td>
</tr>
<tr>
<td>Mosbach</td>
<td>Germany</td>
<td>Europe</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Year</th>
<th>Month</th>
<th>Day</th>
</tr>
</thead>
<tbody>
<tr>
<td>2012</td>
<td>01</td>
<td>1</td>
</tr>
<tr>
<td>2012</td>
<td>01</td>
<td>2</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>2012</td>
<td>01</td>
<td>31</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
FLEXIBLE HIERARCHIES

Jennifer Gray  
(President)

June White  
(Vice-president)

John Yellow  
(Buying)

William Red  
(Manufacturing)

Susan Black  
(Research)

Rachel Blue  
(Marketing)

Lizzie Brown  
(Public relations)

James Scarlet  
(Vice-president)

Joe Green  
(Customer service)
Levels of hierarchy are represented as recursive relationships (e.g., management hierarchy)

<table>
<thead>
<tr>
<th>empKey</th>
<th>lastName</th>
<th>bossKey</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>...</td>
<td>NULL</td>
</tr>
<tr>
<td>2</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>...</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>...</td>
<td>2</td>
</tr>
</tbody>
</table>
OTHER SCHEMATA: SNOWFLAKE

- **Customer**
- **Product**
- **Sales**
- **Country**
- **City**
- **Time**
- **Region**
OTHER SCHEMATA: GALAXY
An OLTP database tracks which user has borrowed which books for how long. We want to be able to answer questions like ‘who are the users with the longest lending (per book, per genre)?’

How should the fact table look like?

A) Lendings(bookId, genreId, userId, days)
B) Lendings(bookId, genreID, days)
C) Lendings(bookId, userId, days)
An OLTP database tracks which user has borrowed which books for how long. We want to be able to answer questions like ‘who are the users with the longest lending (per book, per genre)’?

How should the fact table look like?

A) Lendings(bookId, genreId, userId, days)
B) Lendings(bookId, genreID, days)
C) Lendings(bookId, userId, days)
DATA WAREHOUSING STEPS

Data Modeling

Data Integration

Querying
THE BIG PICTURE

- Application Database
- ETL (Extract, Transform, Load)
- Data Warehouse
- Data Products
- Business Intelligence
- Analytics
DATA INTEGRATION

Data integration is done by ETL Processes

- **Extract**: extract data out of an operational data source
- **Transform**: cleanse it and transform it into the target schema (e.g., split first and last names)
- **Load**: append it to the tables of a data warehouse

Operational Sources: files, databases, event logs, ...

Sink (Data Warehouse): RDBMS, specialized OLAP engines, ...

ETL WORKFLOWS

The ETL pipeline or workflow often consists of many sequential steps

- Often a mix of tools involved (Web-Service APIs, tools to reformat data, … )
- Analogy: Unix pipes and filters -> $ cat data_science.txt | wc | mail -s "word count" hammer@example.com

If the workflow is to be run more than once, it can be scheduled

- Scheduling can be time-based or event-based

Transformations are most complex Product (Separate slides on Data Integration!)
EXTRACT DATA

Database

Web-Service

HTML Files

SQL

HTTP

File-IO

SQL-Dump / CSV

JSON

HTML Tables
Typical Tasks

• Clean Data (e.g., add missing values, correct mis-spellings, …)
• Integrate Data when using multiple sources (e.g., schema matching)
• Execute relational transformations (e.g., joins)
LOAD DATA

Fact table (Lineitem)

Dimension table (Customer)

Dimension table (Region)

Dimension table (Part)

Dimension table (Time/Date)

Name | City | Zip
Carsten | CRANS | RI, 02905
Ugur | PVD | RI, 02902
Stan | BOS | MA, ...

Load into Warehouse
(e.g., generate keys)
DATA WAREHOUSING STEPS

Data Modeling → Data Integration → Querying
RECAP: STAR-SCHEMA (ROLAP)

- **Fact table (Sales)**
  - Dimension table (Customer)
    - 1
  - Dimension table (Product)
    - 1
  - Dimension table (Region)
    - 1
  - Dimension table (Time/Date)
    - 1
STAR-QUERY

Star query = typical query pattern for star schema

Example: Total revenue in a given year (e.g. 2013) per product

Join of multiple dimension tables with fact table +

• Selection (WHERE): on attributes in dimension tables
• Grouping (GROUP BY): on attributes in dimension tables
• Aggregation (SUM, AVG, COUNT, … and HAVING-clause): on attributes in fact table
EXAMPLE: STAR-QUERY

Total revenue in 2013 per product

```sql
select sum(revenue) as total, by p.ProductKey, p.name
from Linitem l, Customer c, Product p, Date d
where l.custKey = c.custKey
and l.ProductKey = p.ProductKey
and l.dateKey = d.dateKey
And d.year = 2013
group by p.ProductKey, p.name
```
The following star schema is used to track user who borrowed which books over time

**Dimensions:**

Book(bookId, title)
User(userId, name, DOB)
Genre(genreId, title)

**Fact Table:**

Lendings(bookId, userId, genreId, days)
Which SQL query returns the total number of books from the genre “Fantasy” for more than 90 days on average?

A) SELECT g.genre, COUNT(*)
    FROM BorrowedBooks bb, Books b, Genre g
    WHERE bb.bookID=b.bookID AND
    bb.genreID=g.genreID AND
    g.genre='Fantasy' AND
    bb.days > 90
    GROUP BY b.genre

B) SELECT genre, COUNT(*)
    FROM BorrowedBooks bb, Genre g
    WHERE bb.genreID=g.genreID AND
    g.genre='Fantasy'
    HAVING AVG(bb.days) > 90
Which SQL query returns the total number of books from the genre “Fantasy” for more than 90 days on average?

A) SELECT g.genre, COUNT(*)
   FROM BorrowedBooks bb, Books b, Genre g
   WHERE bb.bookID=b.bookID AND bb.genreID=g.genreID AND g.genre='Fantasy' AND bb.days > 90
   GROUP BY b.genre

B) SELECT genre, COUNT(*)
   FROM BorrowedBooks bb, Genre g
   WHERE bb.genreID=g.genreID AND g.genre='Fantasy'
   HAVING AVG(bb.days) > 90
SQL-EXTENSIONS

SQL has **different extensions** to support **analytical queries**

**Rollup (Grouping Sets)/ Cube:** special grouping by different sets of dimensional attributes

**Top(k)/Limit:** Top-k results ordered by a given key figure (e.g., top-10 customer which produced maximal total revenue)

**Skyline:** Finding optimal along multiple dimensions (e.g., hotels that are cheap and are close to the beach)
**ROLEUP**

**Rollup:** special grouping by different sets along a hierarchy

**Example (IBM DB2):**

```sql
select sum(revenue) as total, region, country, city
from Linitem l, Customer c
where l.custKey = c.custKey

group by rollup(region, country, city)
```

Query groups result by the following attribute sets: (region), (region, country) and (region, country, city)
## EXAMPLE: ROLLUP

<table>
<thead>
<tr>
<th>total</th>
<th>region</th>
<th>country</th>
<th>city</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,435,789</td>
<td>Europe</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>232,199</td>
<td>Europe</td>
<td>France</td>
<td>-</td>
</tr>
<tr>
<td>634,124</td>
<td>Europe</td>
<td>Germany</td>
<td>-</td>
</tr>
<tr>
<td>119,566</td>
<td>Europe</td>
<td>Germany</td>
<td>Munich</td>
</tr>
<tr>
<td>35,234</td>
<td>Europe</td>
<td>Germany</td>
<td>Mannheim</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>210,199</td>
<td>Europe</td>
<td>France</td>
<td>Paris</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
GROUPING SETS

Alternative for Rollup: Grouping sets define the set of group-by attributes explicitly

Example (Oracle):

select sum(total) as total, region, Country, city
from Linitem l, Customer c
where l.custKey = c.custKey

group by grouping sets(((region, country),
(region, country, city))

Query groups result by the following attribute combiCountry:
(region, Country) und (region, Country, city)
### EXAMPLE: GROUPING SETS

<table>
<thead>
<tr>
<th>total</th>
<th>region</th>
<th>Country</th>
<th>city</th>
</tr>
</thead>
<tbody>
<tr>
<td>232.199</td>
<td>Europe</td>
<td>France</td>
<td>-</td>
</tr>
<tr>
<td>634.124</td>
<td>Europe</td>
<td>Germany</td>
<td>-</td>
</tr>
<tr>
<td>119.566</td>
<td>Europe</td>
<td>Germany</td>
<td>Munich</td>
</tr>
<tr>
<td>35.234</td>
<td>Europe</td>
<td>Germany</td>
<td>Mannheim</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>210.199</td>
<td>Europe</td>
<td>France</td>
<td>Paris</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

(region, country)

(region, country, city)
TOP(K) OR LIMIT

**Top-k/ LIMIT functionality:**

- Sort aggregated result
- Limit result size by given k

**Example (PostgreSQL):**

```sql
select sum(total) as total, region
from Linitem l, Customer c
where l.custKey = c.custKey
group by region
order by total
limit 5;
```
Skyline is a multi-dimensional top(k)

- Skyline returns all tuples that are not dominated by any other point in the given set of dimensions
- Qualifying tuples also known as „Pareto-Optimum“

Example: Hotels low distance to beach + low price

```sql
select *
from hotels h
skyline of h.distance min, p.price min
```
EXAMPLE: SKYLINE (HOTELS)

Top (4) order by Price asc, Distance asc

Skyline (Pareto Curve)
SUMMARY

Data Modeling
• Multi-dimensional Model / Cube
• Star Schema / Snowflake Schema
• Hierarchies

ETL-Processes

SQL Extensions
• ROLLUP / GROUPING SETS
• TOP(k)
• SKYLINE
WHAT IS A GOOD DATA WAREHOUSE?

“A Data Warehouse is a
• subject-oriented,
• integrated,
• non-volatile and time-variant
collection of data in support of managements decisions”

(W. H. Inmon, Building the Data Warehouse, 1996)
SUBJECT ORIENTED DATABASE

Operational Databases:
- Are application oriented (e.g., bank accounts, loans, ...)
- Each DB manages only a subset-of the overall data

Data Warehouses:
- Global view on all data about a given subject / entity (e.g., customer)
- Not targeted towards one application
A data warehouse integrates (inconsistent) data coming from different sources in a consistent way

- DB 1 – m, f
- DB 2 – male, female
- DB 3 – 1,0 inconsistent

Data Warehouse

Periodic Data Loading
NON-VOLATILE AND TIME-VARIANT

Operational Databases represents the most up-to-date snapshot

Data Warehouses represents the history of all changes:
- New Data is only appended / never updated
- All entries have a timestamp
- Comparison over time are possible

*updated constantly*  *snapshotted data*