# Enterprise Software
## Managing Information in the Cloud

**Rakesh Ranjan**
**Rakesh.ranjan@sjsu.edu**

# Information management in the Cloud

- What are the enterprise information that need to be managed?
  - Transactional
  - Social
  - mobile
- What are the advantages of managing information in the cloud?
  - Flexibility, agility and speed
  - Cost
  - Secured?
- What to look for in the enterprise data management solution?
  - Ability to support massive data volumes
  - Enable big data initiatives
  - Able to address new formats
  - Regulatory compliant
  - Be able to save store, archive, protect and access broader and much more challenging class of information

# Advantages of managing information in the cloud

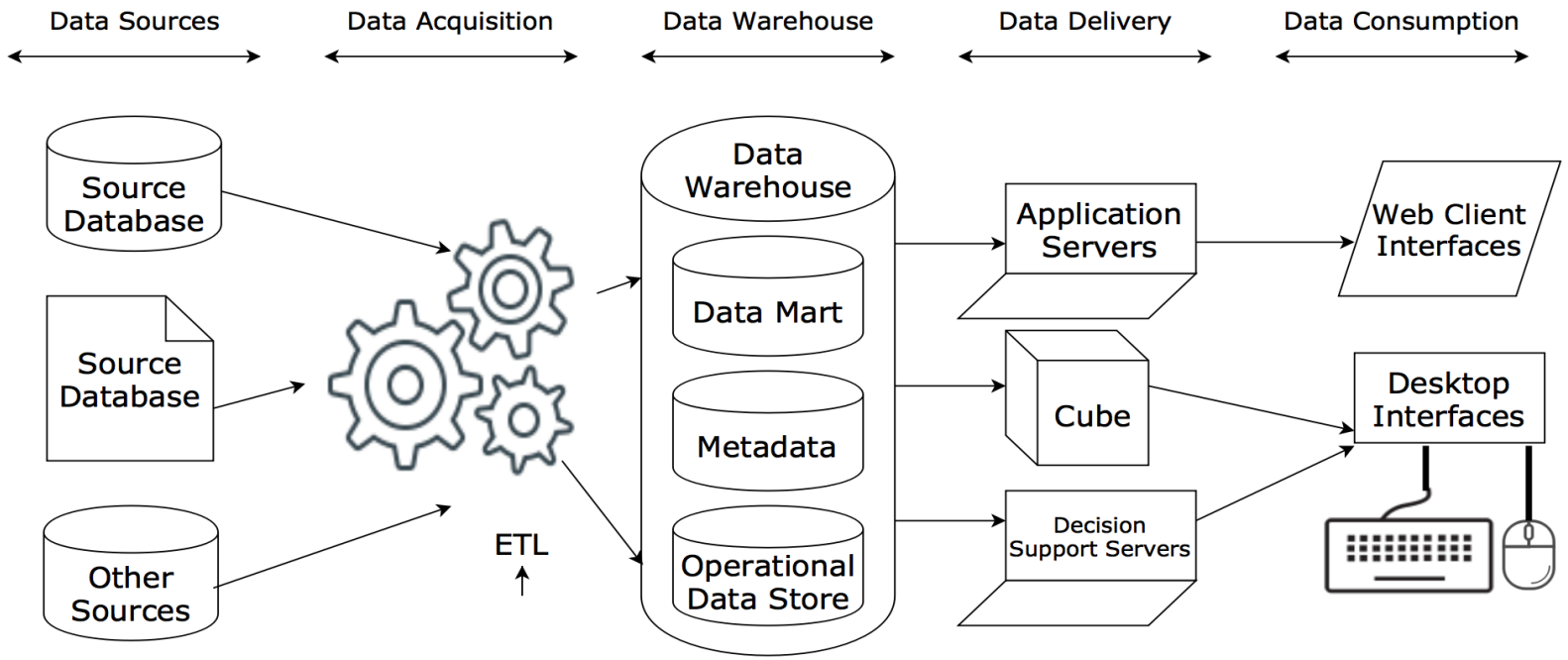
- Immediately upgrade their infrastructure
  - Without making up-front investments
- Lower total cost of ownership
- Eliminate the data silos
  - Silos are a roadblock to the consistent implementation of compliance processes across all regulated content
- Deploy a flexible solution
- Protect data in a secured environment

# What does a data management system do?

- Reduce redundancy
- Avoid inconsistency
- Share data
- Enforce standards
- Apply security restrictions
- Maintain integrity
- Balance conflicting requirements
- **Make data independent**

# What is a data warehouse?

**A data warehouse is used to support forecasting and decision-making processes across the enterprise. It acts as a centralized repository of an organization's data, ultimately providing a comprehensive and homogenized view of the organization.**

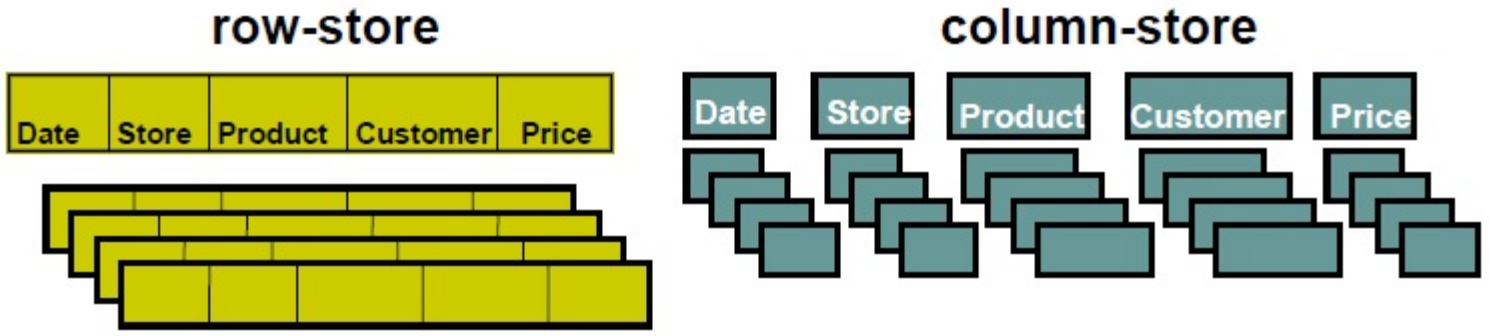


# Emerging Database Technologies

- Columnar Databases
- In-memory Databases
- GIS databases
- Genome Databases
- Temporal Databases
- Time Series database
- Graph Databases

# Columnar Database

- Stores content by columns rather than row.
- The 2-D data represented at conceptual level will be mapped to 1-D data structure at physical level.
- Row-by –Row approach keeps all the information about one entity together.
- Column – by –Column approach keeps all attribute information together.
- Column oriented databases handle fixed length data
- Columnar DBMS are special purpose databases and are not designed to replace general purpose RDBMS.
- Logical storing details of RDBMS vs. Columnar DBMS.



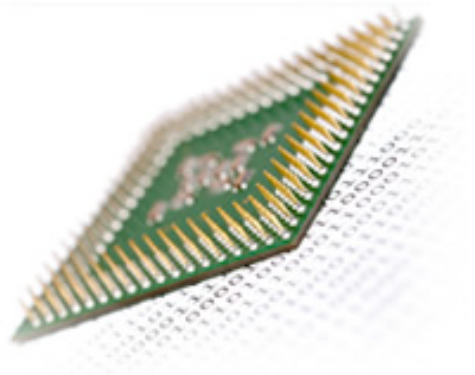
# Columnar Database- Benefits / Tradeoffs

- Most data warehousing applications make more number of reads and lesser number of writes.
- They mostly retrieve and analyze lesser number of columns compare to the several number of columns that actually exist.
- Row oriented databases have the overhead of seeking through all columns.

- Row oriented databases work well for granularity at the entity level.
- Column oriented databases work well for granularity at the attribute level.
- Row oriented – Optimal write time and abundant reading overhead for retrieval of subset queries.
- Column oriented – Optimal read time for subset retrieval queries, bad write performance.

# In-Memory Database

- Uses main memory to store data rather than disk
- Disk I/O eliminated – high transaction throughput
- Used in applications where response time is critical
  - Real time analytics
  - Measure user engagement and developing trends in real time
  - Real time data ingestion
  - Real time decision engine
- Apache Geode http://geode.apache.org/
- dashDB http://www.ibm.com/analytics/us/en/technology/cloud-data-services/dashdb/
- MemSQL http://www.memsql.com/
- Redis http://redis.io/
- VoltDB https://www.voltdb.com/
- SQLite https://www.sqlite.org/



# GIS Database

Geographic Information System (GIS) is a collection of computer hardware, software, and geographic data for capturing, storing, updating, manipulating, analyzing, and displaying all forms of geographically-referenced information.

- Use Cases: Finding patterns:
  - Identify areas with high crime rates to determine where to assign more resources.
  - Find where a disease outbreak is most concentrated.
  - Find hot spots of potential customers within a region for targeted marketing.
- Use Cases: Find suitable locations:
  - Find the best location for a new warehouse by identifying lots that are vacant, at least five acres, and within a mile of a freeway.
  - Identify areas that may be suitable habitat for a particular species by combining layers of vegetation, elevation, and distance from streams and lakes.

# ArcGIS for Developers

## Visualization

Create thematic interactive maps that allow your users to explore and understand their geographic data.

## Geocoding

Search for places and addresses and display them on your map.

## Directions

Generate directions, optimal routes and calculate drive time areas.

## Ready-to-use Content

Choose from a collection of ready-to-use basemaps, demographic maps, and imagery and make interactive maps with your data.

## GeoEnrichment

Enrich your existing hosted services with demographic variables for a given study area.

## Geotrigger℠ Service

Use the Esri Geotrigger Service to easily add location awareness to your apps.

## Spatial Analysis

Analyze your data spatially to detect patterns, assess trends, and make decisions.

## Real-time Processing

Connect to sensors such as GPS and mobile devices and process incoming data.

## Imagery

Access ArcGIS Online image services (basemap, multispectral, event and temporal) to visualize and analyze change.

## Data Storage

Create custom REST endpoints to store and visualize your content.

## Offline Editing

Take your maps and data offline to view, edit, search and find routes.

https://developers.arcgis.com/en/

# Temporal Database

- Allows you to query data in specific time
- Provides business time component along with system time
- Bi-temporal table combines both system time and application time
- Automatic versioning of rows, complete history of changes
- Query the database as of any point in time in the past, i.e. you can go back in time
- Represent your application's logical notion of time, such as the 'effective dates' of business events
- Support for time-based business queries:
  - Did Mrs Jones have insurance coverage on Nov 16, 2008?
  - How many contracts are going to expire next months?
- Support for time-based updates
  - Effective Feb 1, the interest rate on account YXZ will increase by 1%

  http://www.cs.arizona.edu/people/rts/tdbbook.pdf

# Time series database (TSDB)

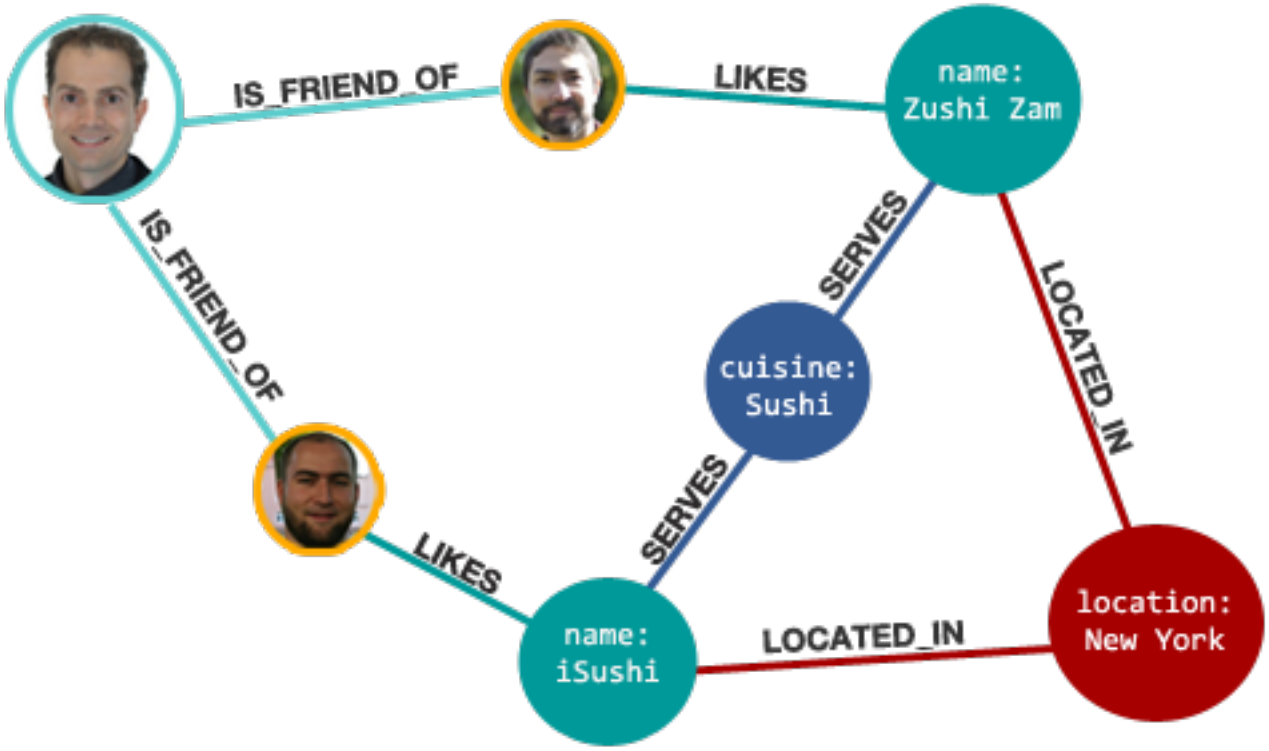
A **time series** is a sequence taken at successive equally spaced points in **time**. Thus it is a sequence of discrete-**time data**. Examples of **time series** are heights of ocean tides, counts of sunspots, and the daily closing value of the Dow Jones Industrial Average.

- IoT/Sensor/Device data
  - Utility and energy companies store data from devices like utility meters and digital thermostats
  - Weather company uses TSDB to store a variety of data from over 130,000 data sources including satellites, radars, forecast models, users, and weather stations worldwide.
- Metrics/log analytics
  - fast range queries and aggregation of vast amounts of gaming TS data including game performance, game statistics, and player activities
- Edge device analytics
  - ensure patient and provider records are always available for hospital and clinic edge analytics.
  - fast storage and analysis of device or system logs at the edge and aggregate them over time
  - real-time analysis of system performance due to its speed and resiliency in TELCO
- Time stamp data feeds
  - store and analyze sale price and volume of a traded stock (e.g. market indices)
  - store and analyze total value and delivery location of an order over time
-

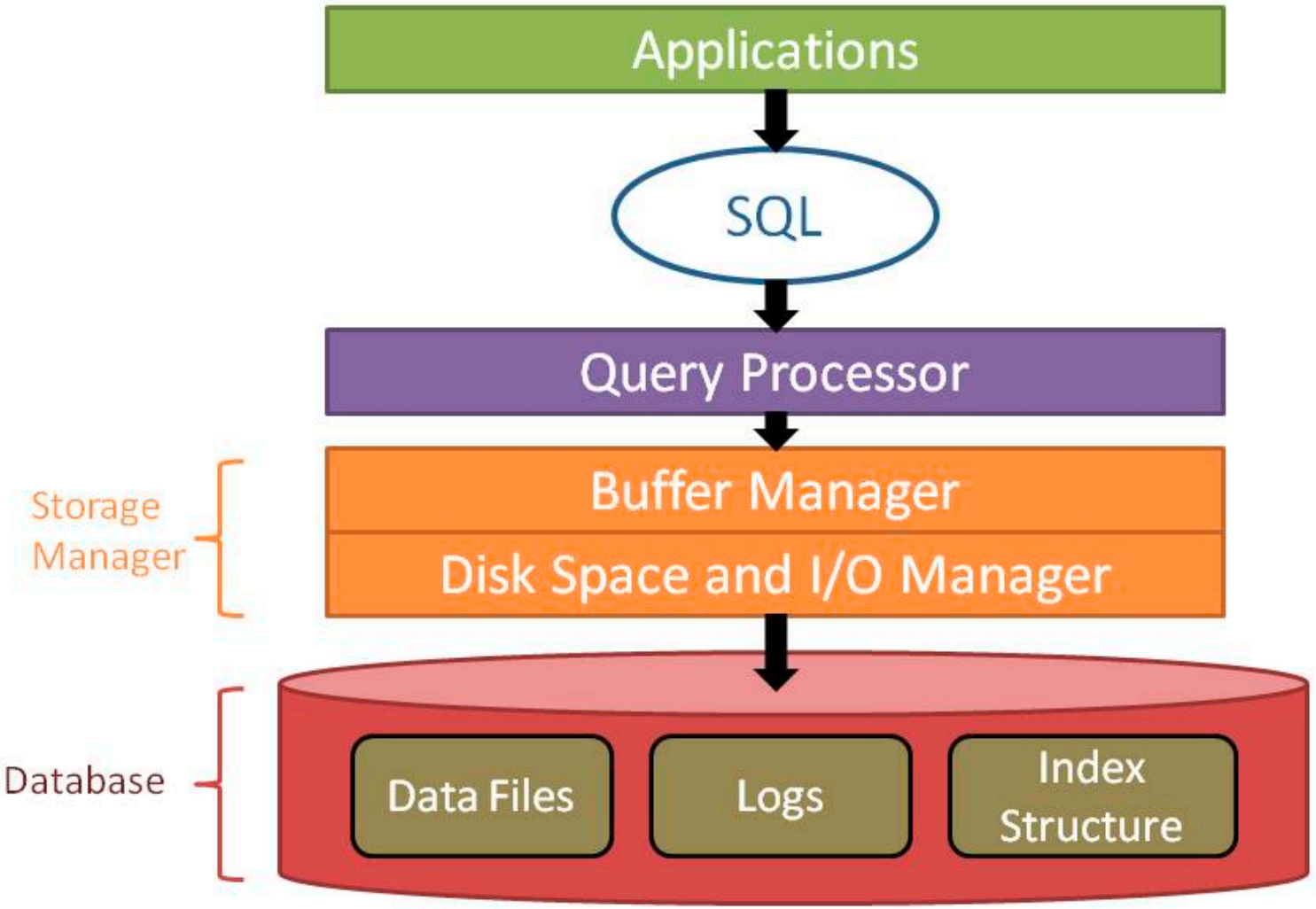
# Graph Databases

There are no classical indexes for Graph DBs. rather, each object stored is mapped with “nodes” and “edges”.

- Node has at least one or many named properties
- Edges define the relationships between nodes

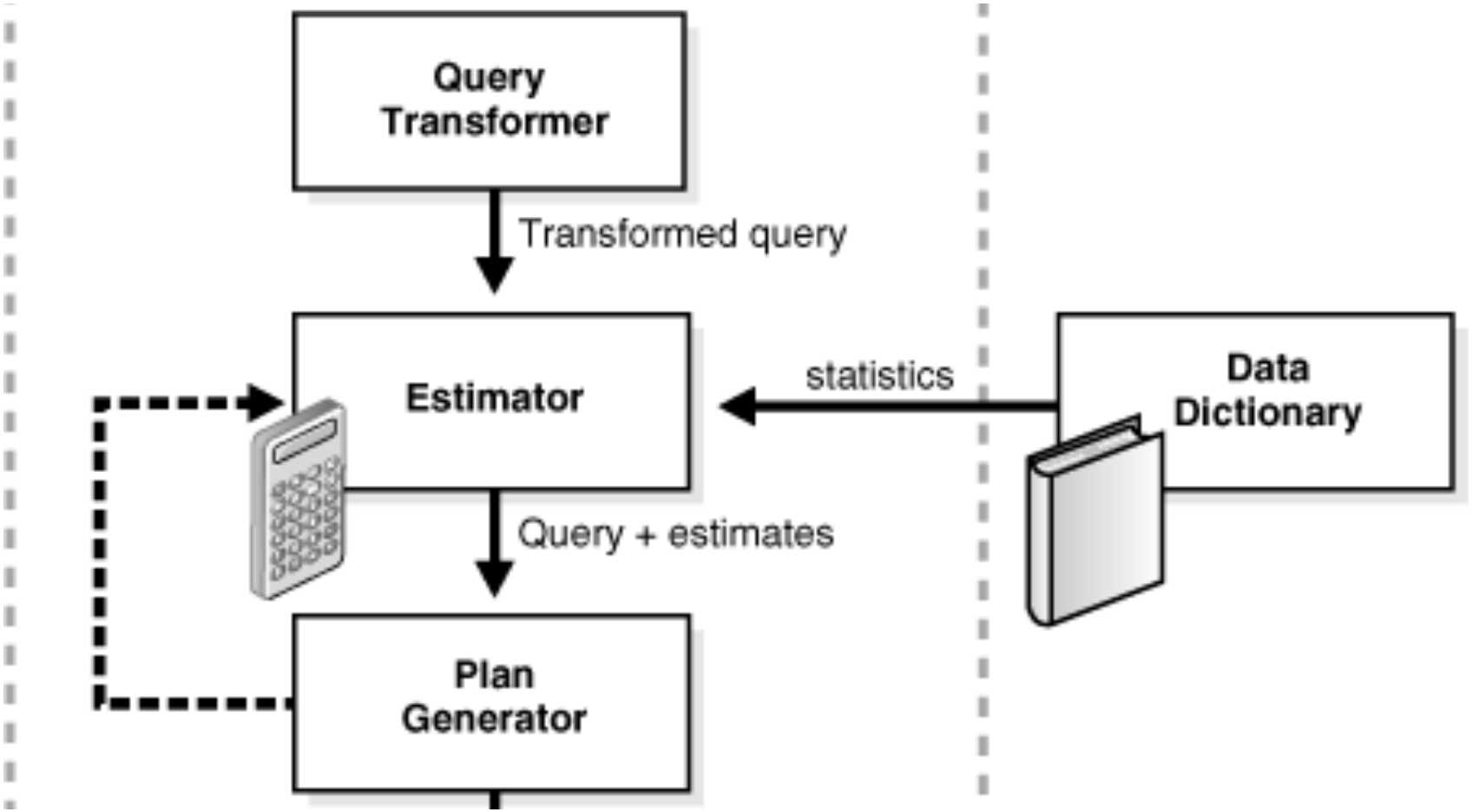


# Architecture of a DBMS

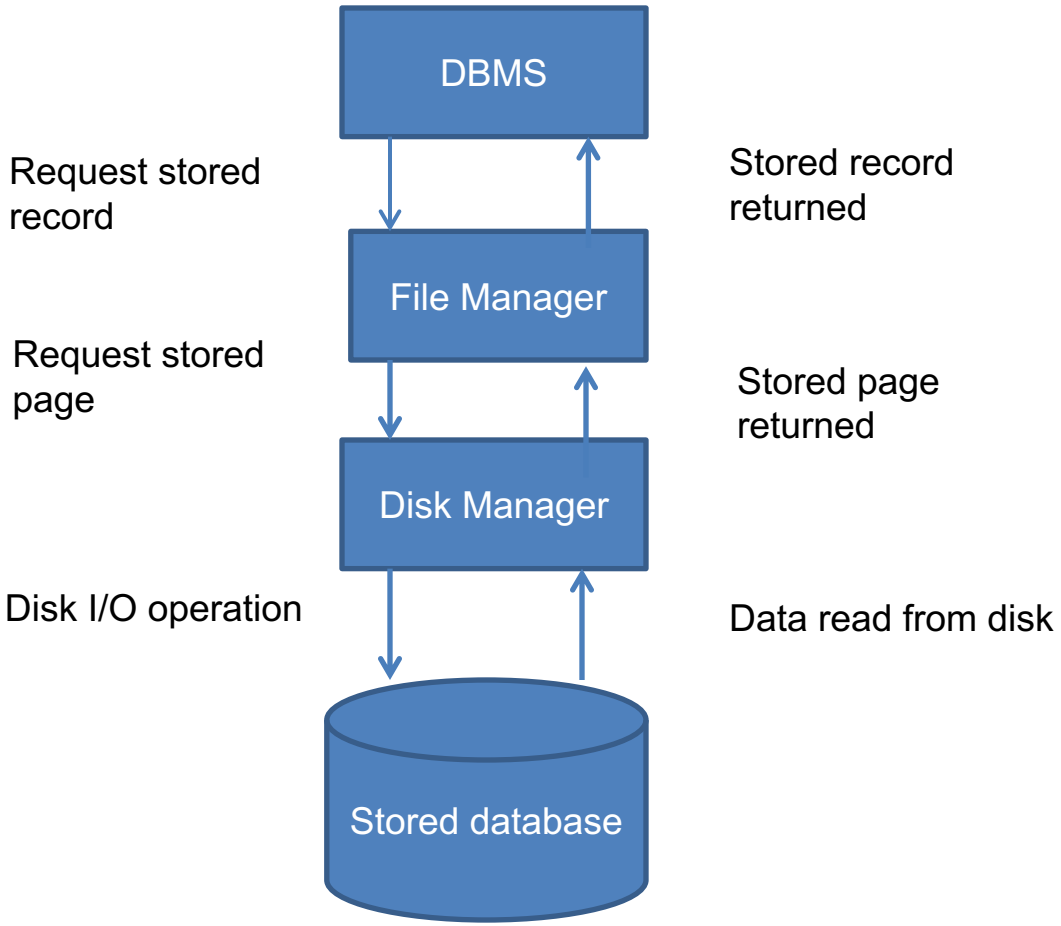



Reference: http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=/com.ibm.db2.luw.admin.perf.doc/doc/c0005418.html

# Query Optimizer



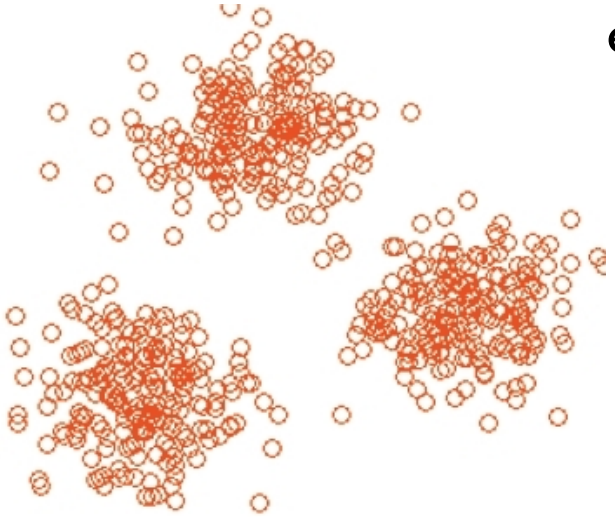
# Data Access – an overview

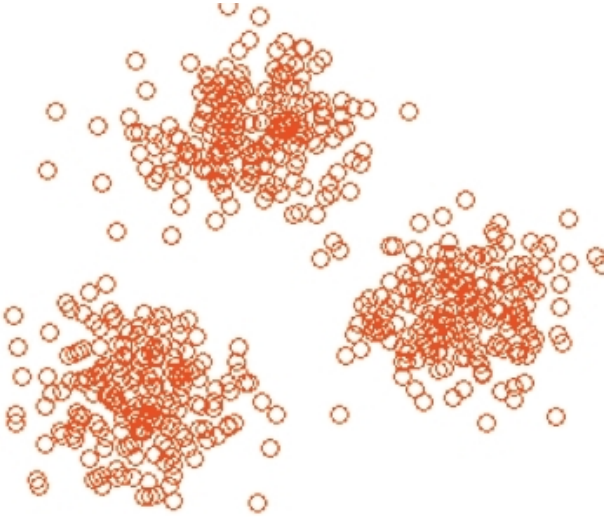


# Data Clustering

- Store records that are logically related (and are therefore frequently used together) and physically close together on disk
- Physical data clustering is extremely important in warehouses where high query performance is needed
- Clustering greatly reduces physical I/O
- Reduces seek time involve in I/O for adjacent pages

e zero if pages are i





# Indexing

City data (Index)

| Athens | |
|---|---|
| London | |
| London | |
| Paris | |
| Paris | |

Supplier data (Table)

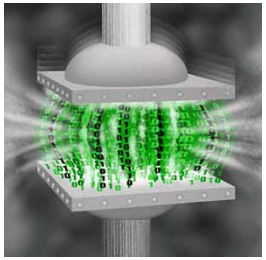
| s1 | Smith | 20 | London |
|---|---|---|---|
| s2 | Jones | 10 | Paris |
| s3 | Blake | 30 | Paris |
| s4 | Clark | 20 | London |
| s5 | Adams | 30 | Athens |

Indexing the supplier data on CITY

# How indexes are used

- Indexes are used two different ways:
  - Sequential:
    - sequence of values in the index file
    - Also works for range queries (find suppliers whose city is in alphabetical range)
  - Direct:
    - Direct access to the index value supplied as query parameter
    - Also used for list queries (find suppliers whose city is in the specified list)

- Indexes can be created on group of columns (also called as multi-column Index)
  - Find supplier in Paris with status 30
  - Single scan of a single index
  - If 2 separate indexes exists, two scans will be needed and DBMS has to decide which scan to be used first
  - Performance characteristics could differ

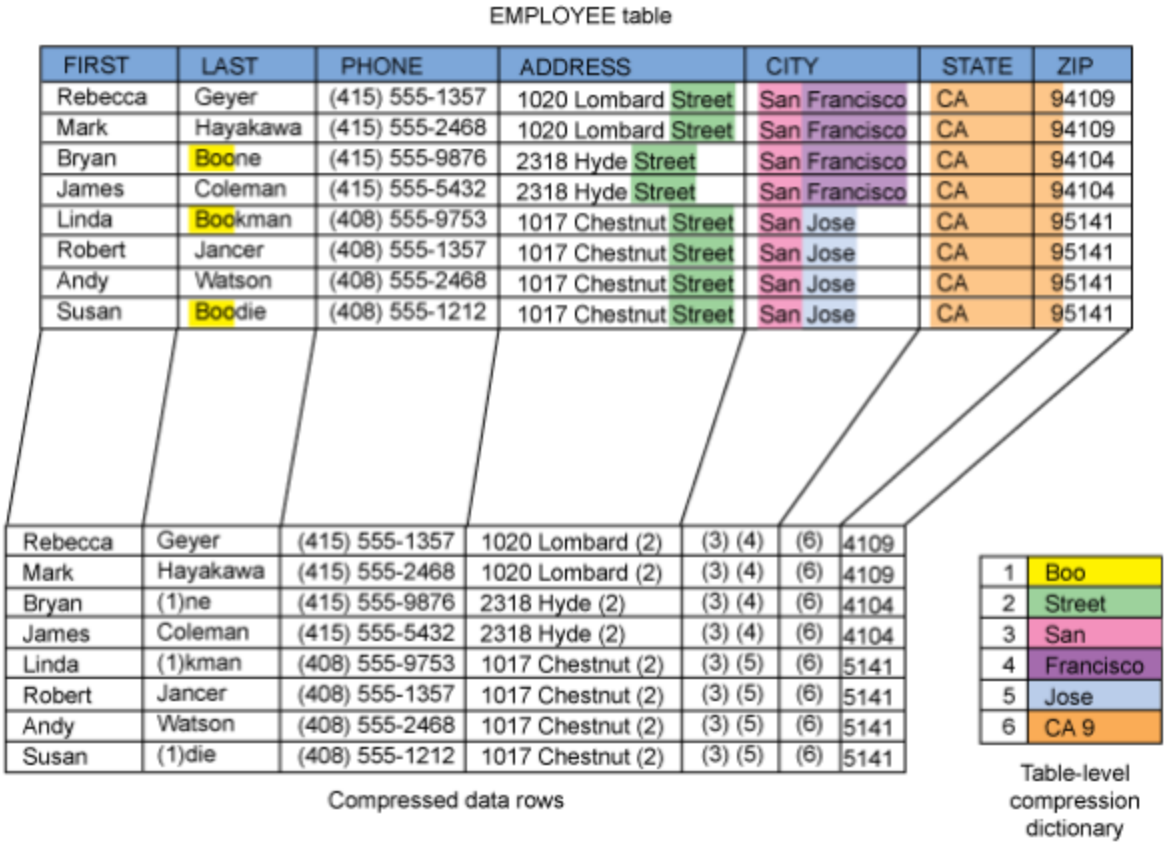


# Data Compression

- Volume of data increases -> cardinality drops
- Only 78,800 unique last names in 300 million people (US census 2010)
- 6,600 unique first names (4,400 for females and 2,200 for males)
- Other names like cities, states, addresses also tend to be highly redundant with low cardinality
- Repeating patterns are replaced with a 12 bit symbols
- These symbols are stored along with the corresponding patterns in an object called dictionary
- Dictionary itself is stored on the database pages
- Loaded into memory along with the compressed data when the data in the table is accessed
- Data is compressed when storage saving is realized, not just because repeating patterns are found

# Table level compression

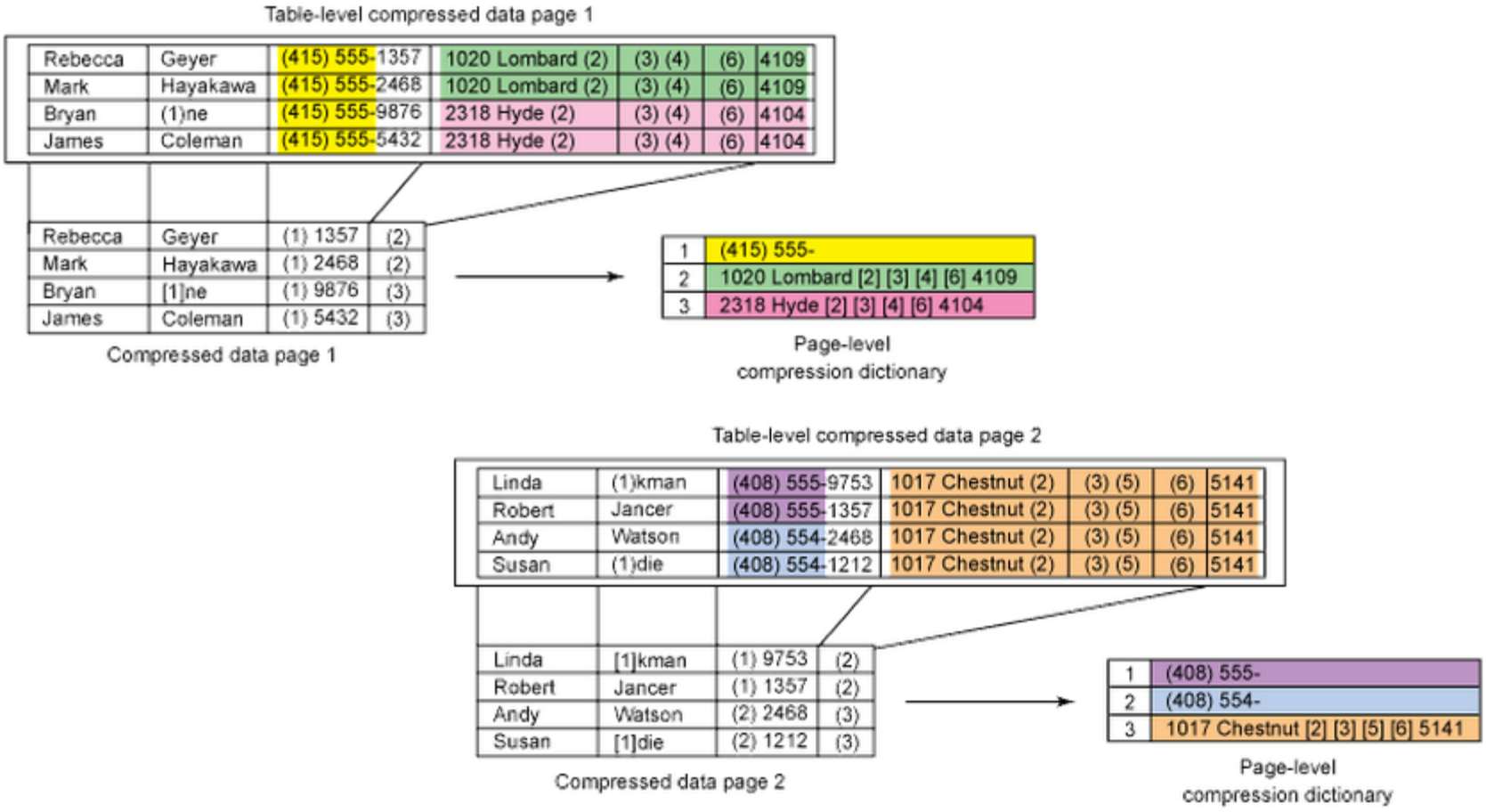
- Dictionary based compression algorithm
- One compression dictionary for each table




http://www.ibm.com/developerworks/data/library/techarticle/dm-1205db210compression/

# Adaptive compression

- Dictionary based compression algorithm
- One compression dictionary for each page (can be multiple for a table)



# Query Optimization

- Query optimization is one of the factors that affect application performance.
- SQL and Xquery compiler performs several steps to produce an access plan that can be executed.
- While compiling the statements the query optimizer estimates the execution cost of different ways of satisfying the query.
- Optimizer uses sorting when no index satisfies requested ordering of data or thinks its less expensive than index scan.
- Optimizer uses Group and sort pushdown when necessary:

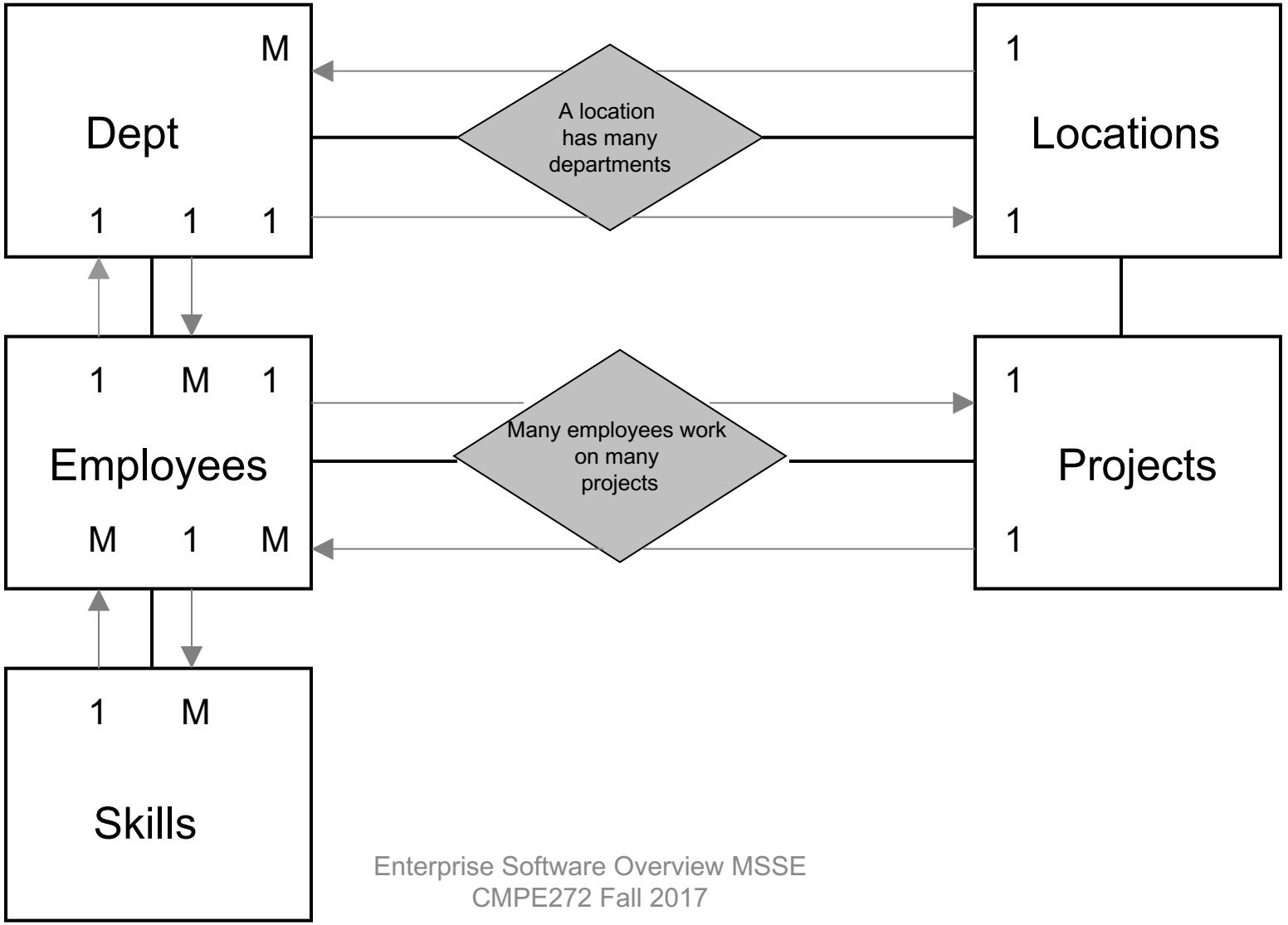
*select workdept, avg(salary) as avg_dept_salary from employee group by workdept*

- Improving Query performance using MQTs: **http://tinyurl.com/3swu6dr**
- **Required reading:** http://infolab.stanford.edu/~hyunjung/cs346/chaudhuri.pdf

# Business Modeling

- A formal representation of business information: its objects, the object's properties or attributes, and the relationships of one object to another.
- Serves as a verification of the users' view of the business before the database is even designed.
- Entity Relationship Diagram (ERD): a pictorial representation of the user's view of the business.
  - Business entity: something that is fundamental to the organization, and an individual instance or occurrence of this thing can be uniquely identified.

# Entity Relationship Diagram (ERD) Example



# Design with Normalization

- **Normalization**: the process of steps that will identify, for elimination, redundancies in a database design.
  - First Normal Form: eliminate repeating groups
  - Second Normal Form: eliminate columns (attributes) that depend only on part of the key
  - Third Normal Form: eliminate columns that don't depend on the key at all

# 1NF: eliminate repeating groups

Denormalized EMPL

| EMP NO | LAST NAME | WORK DEPT | DEPT NAME | SKILL 1 | SKILL2 | SKILLN… |
|---|---|---|---|---|---|---|
| 000030 | KWAN | GRE | OPERATIONS | 141 | | |
| 000250 | SMITH | BLU | PLANNING | 002 | 011 | 067 |
| 000270 | PEREZ | RED | MARKETING | 415 | 447 | |
| 000300 | SMITH | BLU | PLANNING | 011 | 032 | |

Normalized to INF - EMPL

| EMP NO | LAST NAME | WORK DEPT | DEPT NAME |
|---|---|---|---|
| 000030 | KWAN | GRE | OPERATIONS |
| 000250 | SMITH | BLU | PLANNING |
| 000270 | PEREZ | RED | MARKETING |
| 00030 | SMITH | BLU | PLANNING |

EMPL_SKILL TABLE

| EMP NO | SKILL | SKILL DESC |
|---|---|---|
| 000030 | 141 | RESEARCH |
| 000250 | 002 | BID PREP |
| 000250 | 011 | NEGOTIATION |
| 000250 | 067 | PROD SPEC |
| 000270 | 415 | BENEFITS ANL |
| 000270 | 447 | TESTING |
| 000300 | 011 | NEGOTIATION |
| 000300 | 032 | INV CONTROL |

# 2NF: eliminate columns that depend only on part of the key

Normalized – 1NF

EMPL_SKILL TABLE

| EMP NO | SKILL | SKILL DESC |
|---|---|---|
| 000030 | 141 | RESEARCH |
| 000250 | 002 | BID PREP |
| 000250 | 011 | NEGOTIATION |
| 000250 | 067 | PROD SPEC |
| 000270 | 415 | BENEFITS ANL |
| 000270 | 447 | TESTING |
| 000300 | 011 | NEGOTIATION |
| 000300 | 032 | INV CONTROL |

Normalized – 2NF

EMPL_SKILL TABLE

| EMP NO | SKILL | DATE CERT |
|---|---|---|
| 000030 | 141 | |
| 000250 | 002 | |
| 000250 | 011 | |
| 000250 | 067 | |
| 000270 | 415 | |
| 000270 | 447 | |
| 000300 | 011 | |
| 000300 | 032 | |

SKILL DESC TABLE

| SKILL | SKILL DESC |
|---|---|
| 141 | RESEARCH |
| 002 | BID PREP |
| 011 | NEGOTIATION |
| 067 | PROD SPEC |
| 415 | BENEFITS ANL |
| 447 | TESTING |
| 011 | NEGOTIATION |
| 032 | INV CONTROL |

**Question: What are the three problems with the 1NF EMPL_SKILLTABLE?**

# 3NF: eliminate columns that don't depend on the key at all

Denormalized EMPL

| EMP NO | LAST NAME | WORK DEPT | DEPT NAME | MGRNO | … |
|---|---|---|---|---|---|
| 000030 | KWAN | GRE | OPERATIONS | 000080 | |
| 000250 | SMITH | BLU | PLANNING | 000010 | |
| 000270 | PEREZ | RED | MARKETING | 000020 | |
| 000300 | SMITH | BLU | PLANNING | 000010 | |

Normalized to 3NF

EMPL

| EMP NO | LAST NAME | WORK DEPT | … |
|---|---|---|---|
| 000030 | KWAN | GRE | |
| 000250 | SMITH | BLU | |
| 000270 | PEREZ | RED | |
| 00030 | SMITH | BLU | |

DEPT

| WORK DEPT | DEPT NAME | MGRNO |
|---|---|---|
| GRE | OPERATIONS | 000080 |
| BLU | PLANNING | 000010 |
| RED | MARKETING | 000020 |
| BLU | PLANNING | 000010 |

# Normalization/De-normalization Example

EMPL

(EMPNO, DEPT, LAST, MI, FIRST, JOB)
1,000,000 rows
30 chars / row

DEPT

(DEPT, DEPTNAME, MGRNO)
10,000 rows
25 chars / row

Transaction rate 20,000 per day, two tables accessed.

```
SELECT LAST, MI, FIRST, MGRNO
FROM EMPL A, DEPT B
WHERE A.DEPT = B.DEPT AND
EMPNO = '000010'
```

# De-normalization Example

EMPL

(EMPNO, DEPT, LAST, MI, FIRST, MGRNO)
1,000,000 rows
30 chars / row

DEPT

(DEPT, DEPTNAME, MGRNO)
10,000 rows
25 chars / row

Transaction rate 20,000 per day, one table accessed.

```
SELECT LAST, MI, FIRST, MGRNO
FROM EMPL
WHERE EMPNO = '000010'
```

# Cost & Benefits of Normalization

- Cost
  - Storage

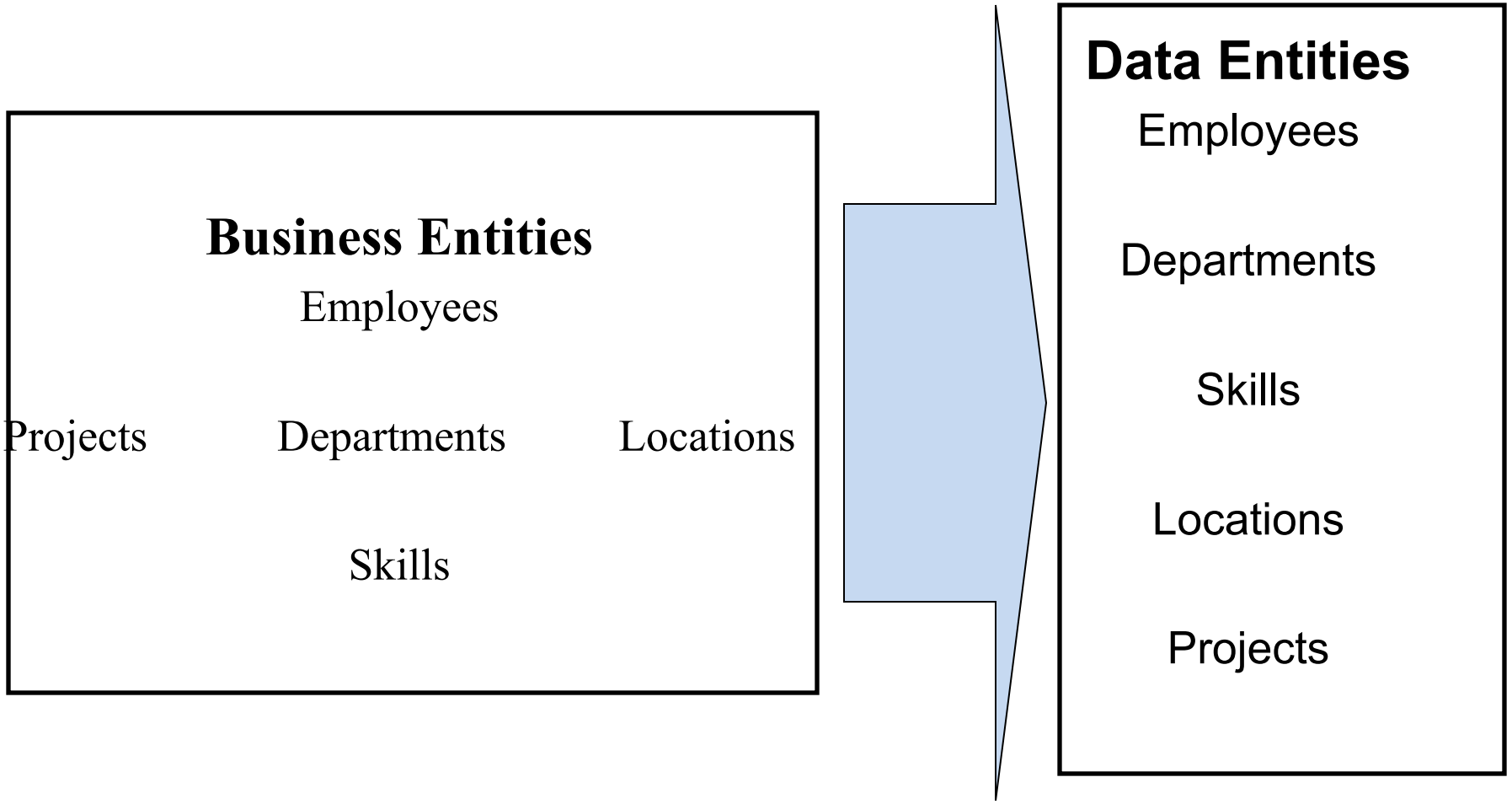
    1,000,000 rows

    X 3 characters

    `3,000,000 characters (Approx 900 pages)`

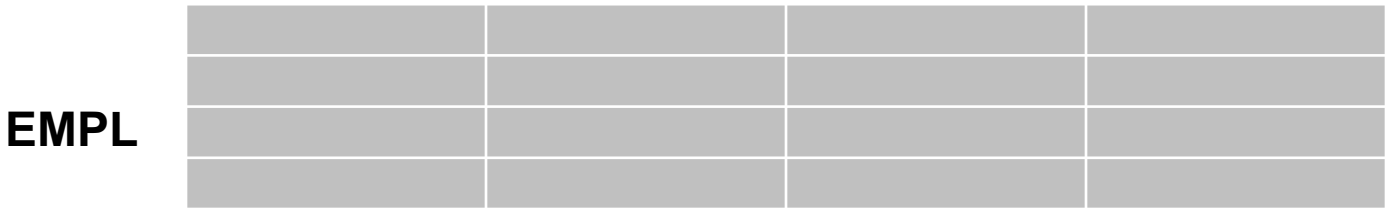
  - Additional updates if MGRNO changes
    - Save 20000 accesses per day

# Data Modeling

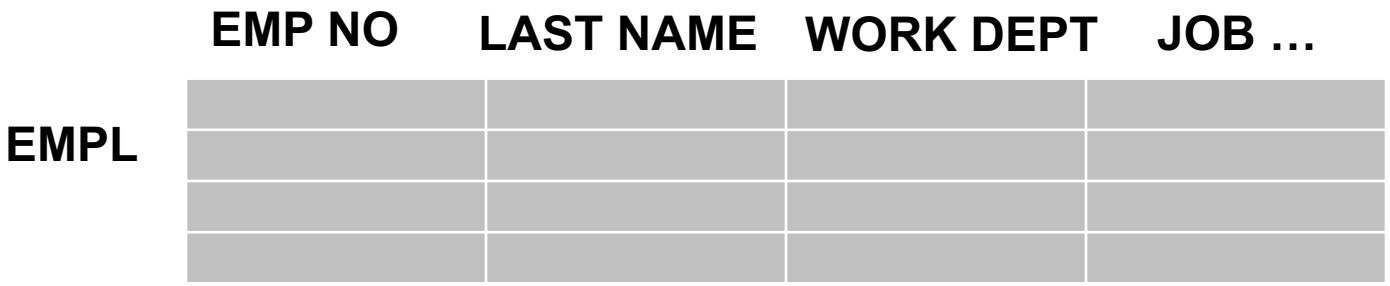


# Tables and Columns

- One Table per Entity




- One Column for Each Data Element(how you decide the granularity?)




- Each occurrence of entity must be uniquely identified. These unique identifiers will be the primary keys in the table.

# Data Modeling exercise

- Parts of Data Modeling:
  - Defining a table's columns (entity's data elements)
  - Defining its primary and foreign keys
  - Resolving many-to-many relationships
  - Identifying and removing redundancies (normalization)
- Data Modeling Case Study
  - Credit card Fraud detection
  - Insurance Claim Fraud detection
  - Industrial damage detection

# NULL Characteristics

- NULLS: a column constraint telling the Database what it should do when the value for a column is unknown
- 3 null characteristics:
  - NOT NULL : Column must always have a value
  - NOT NULL WITH DEFAULT: Column must always have a value – if we don't supply one, Database does
  - "nullable": Column can be marked as having an 'unknown value'

# Related Data

- Data relationships are designed into our tables
- A row in one table may be associated with a row or rows in another table
- Related Tables
  - A row in one table can carry a value from some table's unique key
  - Example: Each employee row contains the value of the employee's department KEY
- Primary Key: best identifies the data being stored in the table
  - One per table
  - All columns must be NOT NULL

# Related Data continued..

- Unique Key: a column or set of columns that contain unique values
  - Contains unique values
  - Most all tables have at least one unique key
  - Without a unique key, you may not be able to find a specific row in the table
  - One of the unique keys can be defined to DBMS as being the primary key of the table
- Foreign Key: a column or set of columns that contains values from some table's unique key
  - Designed into tables to define relationships between rows

# Referential Integrity

- Referential Integrity: the automatic enforcement of referential constraints
- Referential constraints: the limiting of a set of foreign key values to a set of unique key values
- The Rules DBMS Enforces to Maintain Referential Integrity
  - A foreign key value must match a unique key value or be null
  - Primary key values cannot be null
  - All non-null values must match a value in the referenced column (column set)
- Processing Rules Enforced to Maintain Referential Integrity
  - Every unique key value remains unique and is not null
  - Every foreign key value matches a unique key value or is null

# Referential Integrity- Delete rules

## - ON DELATE CASCADE

| DEP | DEPNAME | MANAGER | DIVISION |
|---|---|---|---|
| BLU | PLANNING | 000020 | EASTERN |
| GRE | OPERATIONS | 000080 | WESTERN |
| RED | MARKETING | 000010 | EASTERN |

| EMPNO | LASTNAME | FIRSTNAME | DEP | GOVT_ID | SALARY |
|---|---|---|---|---|---|
| 000010 | HAAS | CHRISTINE | RED | 888-88-2794 | 52750.00 |
| 000020 | THOMPSON | MICHAEL | BLU | 888-89-4261 | 31000.00 |
| 000030 | KWAN | SALLY | GRE | 888-88-9456 | 33000.00 |

| DEP | DEPNAME | MANAGER | DIVISION |
|---|---|---|---|
| GRE | OPERATIONS | 000080 | WESTERN |
| RED | MARKETING | 000010 | EASTERN |

| EMPNO | LASTNAME | FIRSTNAME | DEP | GOVT_ID | SALARY |
|---|---|---|---|---|---|
| 000010 | HAAS | CHRISTINE | RED | 888-88-2794 | 52750.00 |
| 000030 | KWAN | SALLY | GRE | 888-88-9456 | 33000.00 |

# Referential Integrity- Delete rules

## - ON DELATE SET NULL

| DEP | DEPNAME | MANAGER | DIVISION |
|---|---|---|---|
| BLU | PLANNING | 000020 | EASTERN |
| GRE | OPERATIONS | 000080 | WESTERN |
| RED | MARKETING | 000010 | EASTERN |

| EMPNO | LASTNAME | FIRSTNAME | DEP | GOVT_ID | SALARY |
|---|---|---|---|---|---|
| 000010 | HAAS | CHRISTINE | RED | 888-88-2794 | 52750.00 |
| 000020 | THOMPSON | MICHAEL | BLU | 888-89-4261 | 31000.00 |
| 000030 | KWAN | SALLY | GRE | 888-88-9456 | 33000.00 |

| DEP | DEPNAME | MANAGER | DIVISION |
|---|---|---|---|
| GRE | OPERATIONS | 000080 | WESTERN |
| RED | MARKETING | 000010 | EASTERN |

| EMPNO | LASTNAME | FIRSTNAME | DEP | GOVT_ID | SALARY |
|---|---|---|---|---|---|
| 000010 | HAAS | CHRISTINE | RED | 888-88-2794 | 52750.00 |
| 000020 | THOMPSON | MICHAEL | Unknown | 888-89-4261 | 31000.00 |
| 000030 | KWAN | SALLY | GRE | 888-88-9456 | 33000.00 |

# Referential Integrity- Delete rules

## - ON DELATE RESTRICT

| DEP | DEPNAME | MANAGER | DIVISION |
|---|---|---|---|
| BLU | PLANNING | 000020 | EASTERN |
| GRE | OPERATIONS | 000080 | WESTERN |
| RED | MARKETING | 000010 | EASTERN |

| EMPNO | LASTNAME | FIRSTNAME | DEP | GOVT_ID | SALARY |
|---|---|---|---|---|---|
| 000010 | HAAS | CHRISTINE | RED | 888-88-2794 | 52750.00 |
| 000020 | THOMPSON | MICHAEL | BLU | 888-89-4261 | 31000.00 |
| 000030 | KWAN | SALLY | GRE | 888-88-9456 | 33000.00 |

| DEP | DEPNAME | MANAGER | DIVISION |
|---|---|---|---|
| BLU | PLANNING | 000020 | EASTERN |
| GRE | OPERATIONS | 000080 | WESTERN |
| RED | MARKETING | 000010 | EASTERN |

| EMPNO | LASTNAME | FIRSTNAME | DEP | SOC_SEC | SALARY |
|---|---|---|---|---|---|
| 000010 | HAAS | CHRISTINE | RED | 888-88-2794 | 52750.00 |
| 000020 | THOMPSON | MICHAEL | | 888-89-4261 | 31000.00 |
| 000030 | KWAN | SALLY | GRE | 888-88-9456 | 33000.00 |

# A Business Intelligence System


**A Business Intelligence System**

by H. P. Luhn

An automatic system is being developed to disseminate information to the various sections of any industrial, scientific or government organization. This intelligence system will utilize data-processing machines for auto-abstracting and auto-encoding of documents and for creating interest profiles for each of the "action points" in an organization. Both incoming and internally generated documents are automatically abstracted, characterized by a word pattern, and sent automatically to appropriate action points. This paper shows the flexibility of such a system in identifying known information, in finding who needs to know it and in disseminating it efficiently either in abstract form or as a complete document.

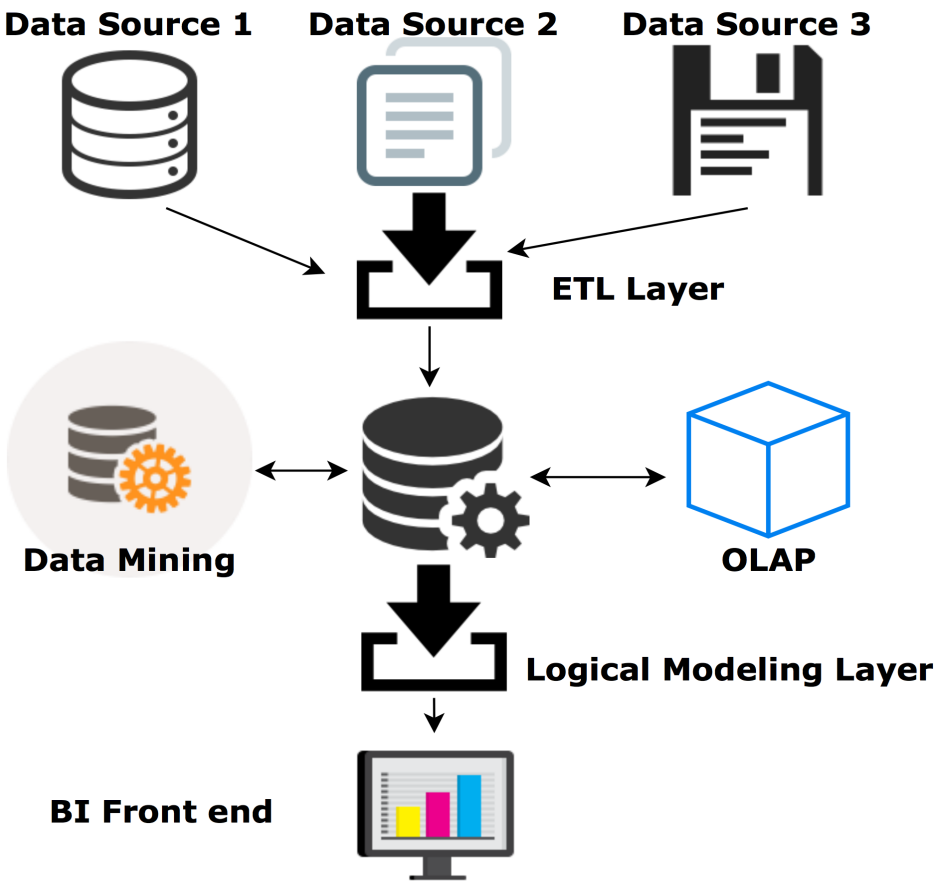
http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5392644

# Luhn's vision -> current state of BI

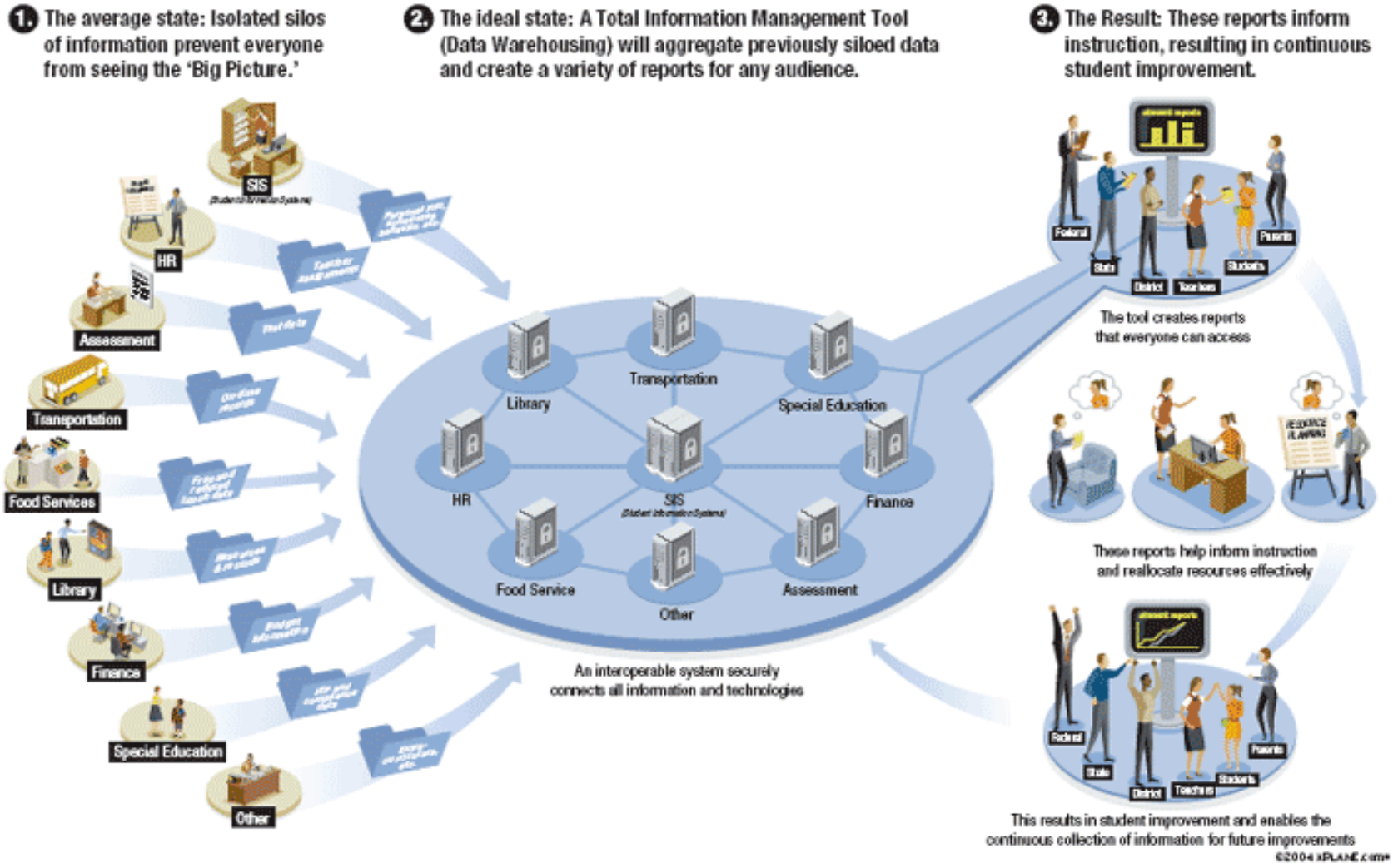
- 1970 - E. F. Codd first proposed the relational model for data
- Mid-1970 - IBM had a working prototype of a relational database management system (RDBMS)
- 1980 - RDBMS's use was proliferating.
- 1983 - Teradata sold the first relational database designed specifically for decision support to Wells Fargo.
- 1986 - Ralph Kimball founded Red Brick Systems(part of IBM now) to build databases for the same market.
- 1991 - Bill Inmon's *Building the Data Warehouse* (Wiley) was published. Inmon advocated creating Enterprise data model.
- 1995 – Inmon's book was a big hit. TDWI was formed.
- 1996 – Ralph Kimball published The Data warehouse Toolkit, challenged the EDM and advocated the data mart model
- 1997 – Microsoft researchers introduced cube (multi dimensional data model)
- 2000 – BI Platforms emerging
- 2005 – BI appliances maximizing ROI
- 2009 – Smarter Analytics platform

# A simple BI Platform

- Heterogeneous data sources
- ETL or ELT
- Integrated mining and Cubing services
- Reporting services
- Ease of use/ latency
- Suitable for Data mart



# BI at your university



# Information Governance

- What information do you have?
- Where it is stored?
- How is the information encrypted?
- How it is updated?
- What is its level of accuracy?
- How does your business use it, and for what purpose?
- What is the actual value of this information?
- What information do you keep, and for how long?
- What information do you retire, and when do you do it?

# Information Governance Infrastructure

- Information Integration
  - Data warehouse + ERP + CRM
- Master data management
  - Consolidation of master data (employee, supplier, product, customer etc..)
- Enterprise content management
  - eDiscovery, business transformation
- Enterprise data management
  - Data server and tools (DB2/Oracle, MySQL, Postgres etc)
- Business intelligence and performance management
  - What happened and why
  - Measure actuals against forecast
- Metadata management
  - Ensure consistency, completeness and context of data

# Mini Projects

**Mini Projects:**

**SQLite**

- Install SQLite Add-ons for Firefox from: https://addons.mozilla.org/en-us/firefox/addon/sqlite-manager/
- Design a database for Purchase Order Management System.
- Create a sample schema with necessary tables from previous step.
- Insert sample data
- Try different queries learnt in this chapter

**DB2 Express C**

- Download DB2 express C for your operating system http://www-03.ibm.com/software/products/en/db2expressc
- Create Sample database (use: db2sampl command)
- Run a sample query (use where clause and Group by)
- Generate query explain plan (use: db2exfmt tool)
- Post the query and db2exfmt output snapshot in a pdf document

**Graph Data store**

- Sign up for IBM Bluemix at www.bluemix.net
- Navigate the catalog for Data and Analytics section
- Click on IBM Graph service, create the service and follow the documentation to create a sample graph application using the API documentation: https://ibm-graph-docs.ng.bluemix.net/api.html