고급 Database Issue:
Data Warehouse, Data Mining, Big Data

권동섭

Data Warehouse & Data Mining

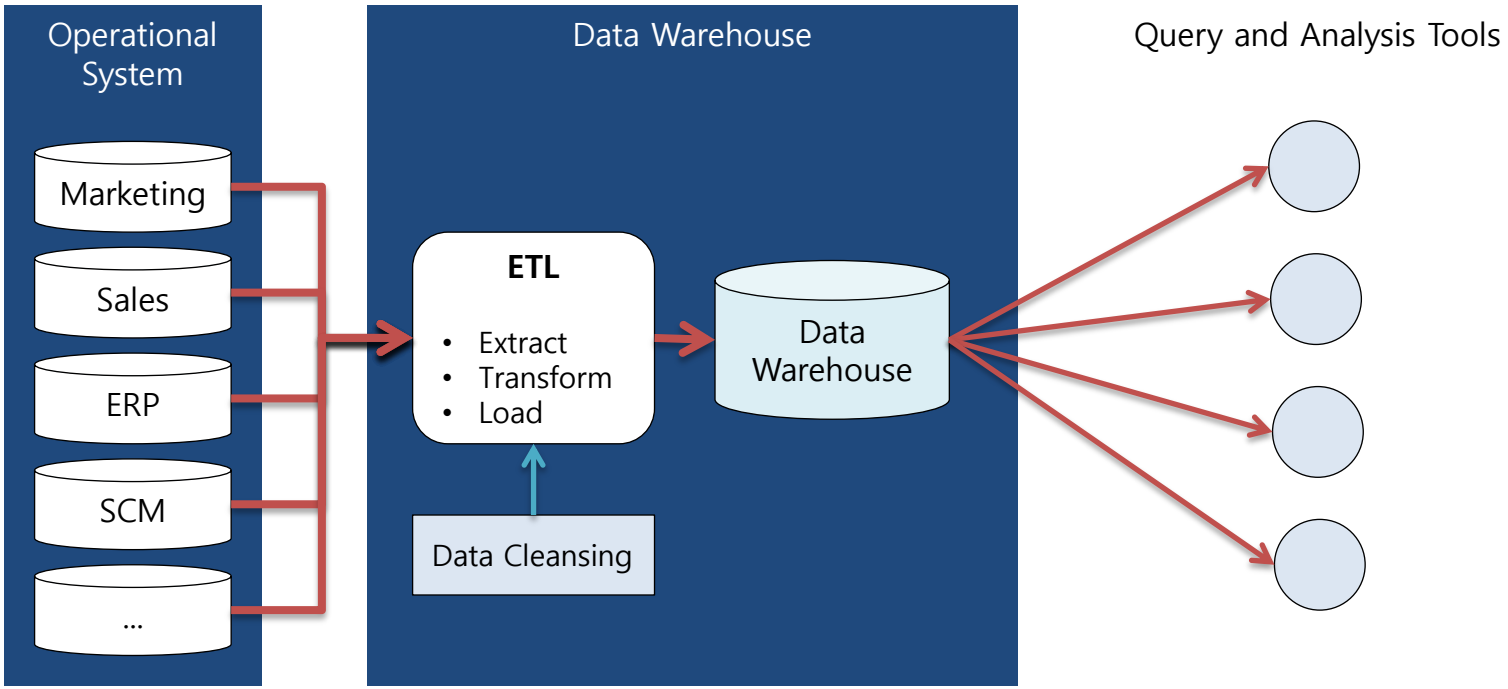
DATA WAREHOUSE

Database 작업의 두 종류

	OLTP	OLAP
정의	Online Transaction Processing	Online Analytical Processing
트랜잭션 길이	짧다	길다
트랜잭션 수준	높다	낮다
데이터 변경	빈번	드묾
질의	간단	복잡
동시사용자	다수	소수
데이터 접근 양	소량	대량
예	은행, POS, 전자상거래	데이터 분석, 의사결정시스템

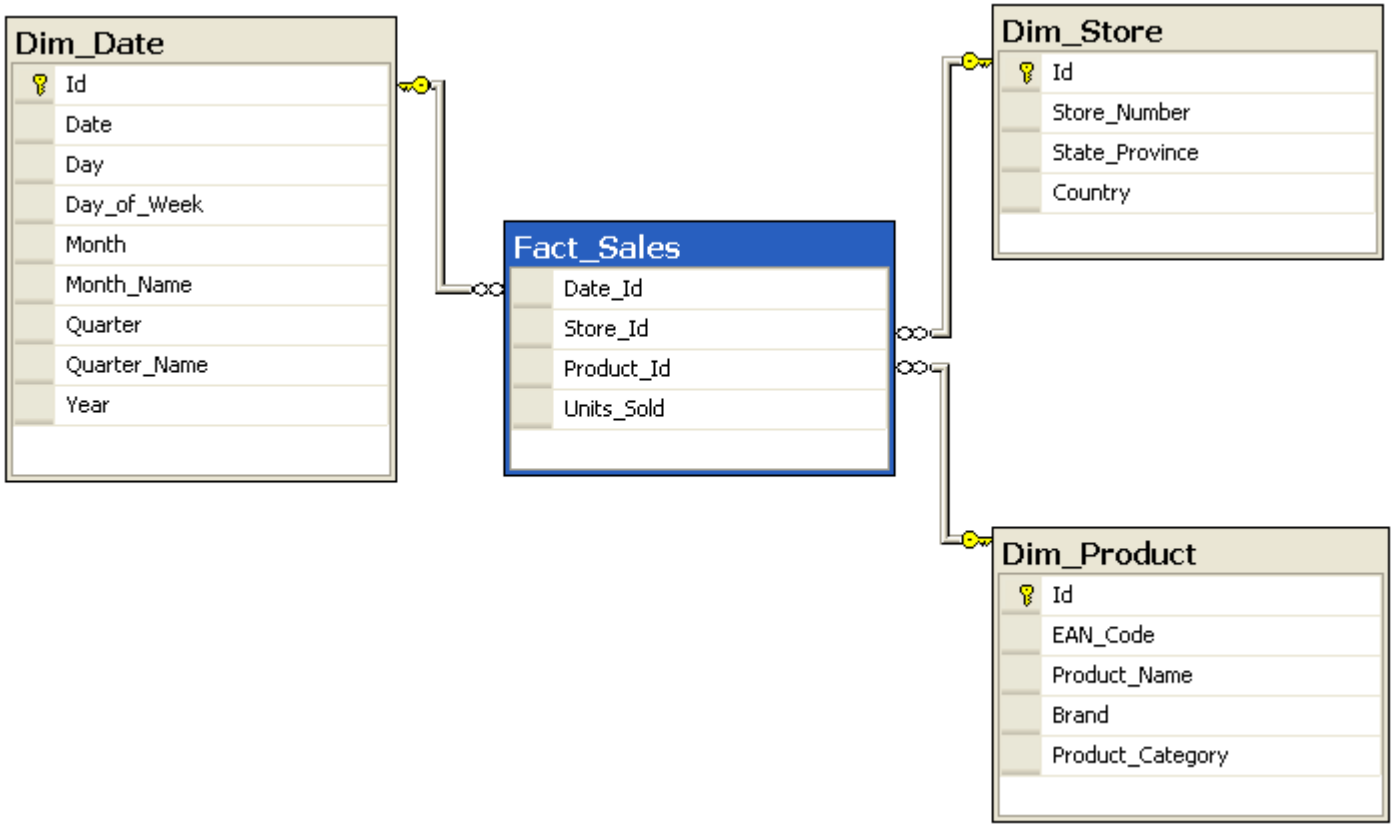
Data Warehouse

- 데이터 **분석/보고**를 위하여 **다수의 데이터 소스**의 데이터를 **통합** 하여 저장하는 **데이터 저장소**



Star 스키마

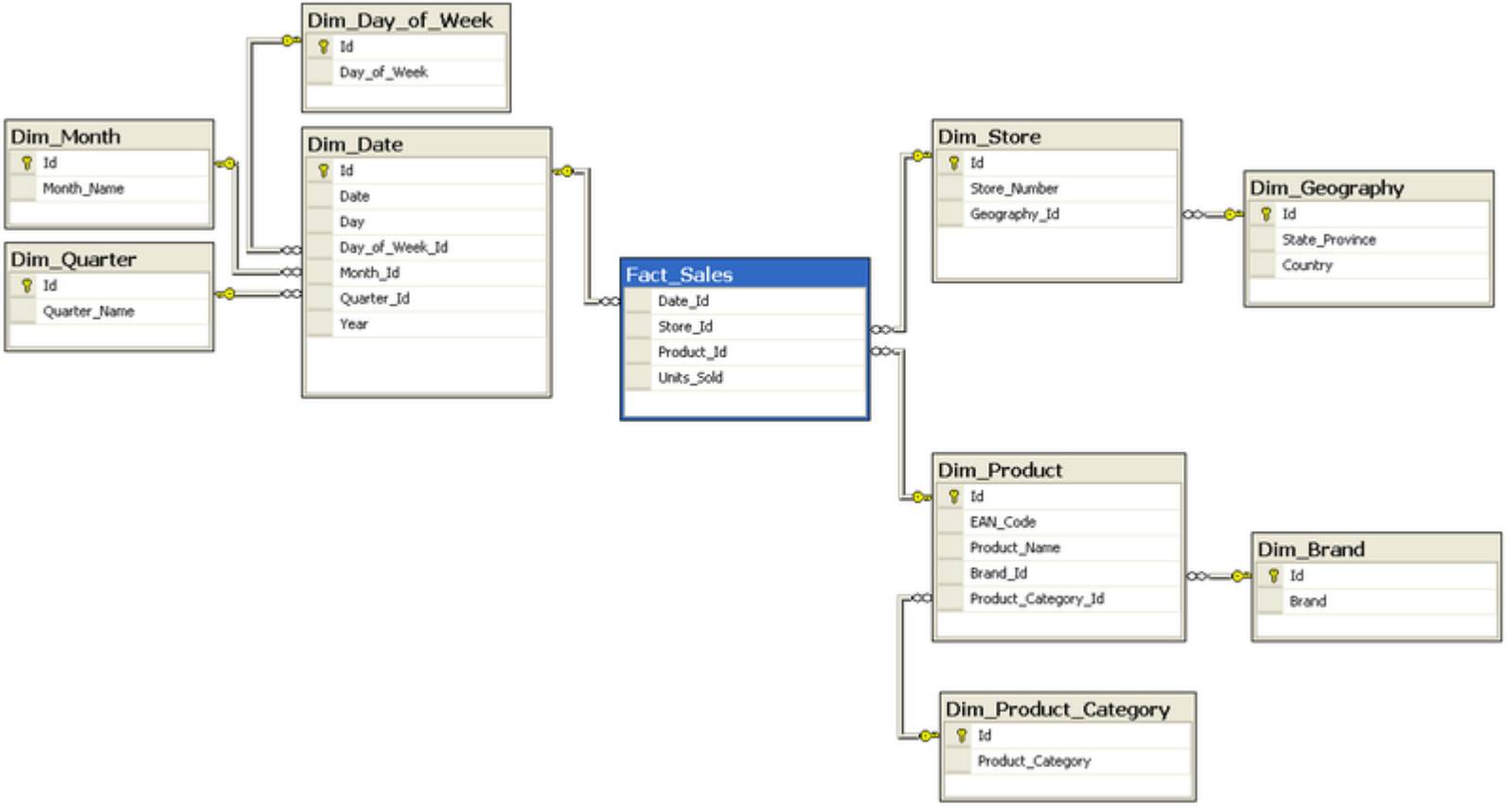
- **Fact** Table + **Dimensional** Table



from http://en.wikipedia.org/wiki/File:Star-schema-example.png

Snowflake 스키마

- Fact Table + **Multiple levels** of Dimensional Tables



from http://en.wikipedia.org/wiki/File:Snowflake-schema-example.png

예

- Star 스키마

```
Sales(storeID, itemID, custID, qty, price)
Store(storeID, city, state)
Item(itemID, category, brand, color, size)
Customer(custID, name, address)
```

- OLAP 질의

```
Select state, brand, Sum(qty*price)
From Sales F, Store S, Item I
Where F.storeID = S.storeID And F.itemID = I.itemID
Group By state, brand
```

예: Drill-Down / Roll-Up

```
Select state, category, Sum(qty*price)
From Sales F, Store S, Item I
Where F.storeID = S.storeID And F.itemID = I.itemID
Group By state, category
```


```
Select state, category, brand, Sum(qty*price)
From Sales F, Store S, Item I
Where F.storeID = S.storeID And F.itemID = I.itemID
Group By state, category, brand
```

예: Pivot

```
Select state, category, Sum(qty*price)
From Sales F, Store S, Item I
Where F.storeID = S.storeID And F.itemID = I.itemID
Group By state, category
```


```
Select state, address, brand, Sum(qty*price)
From Sales F, Store S, Customer C
Where F.storeID = S.storeID And F.custID = C.custID
Group By state, address
```

예: WITH CUBE, WITH ROLLUP

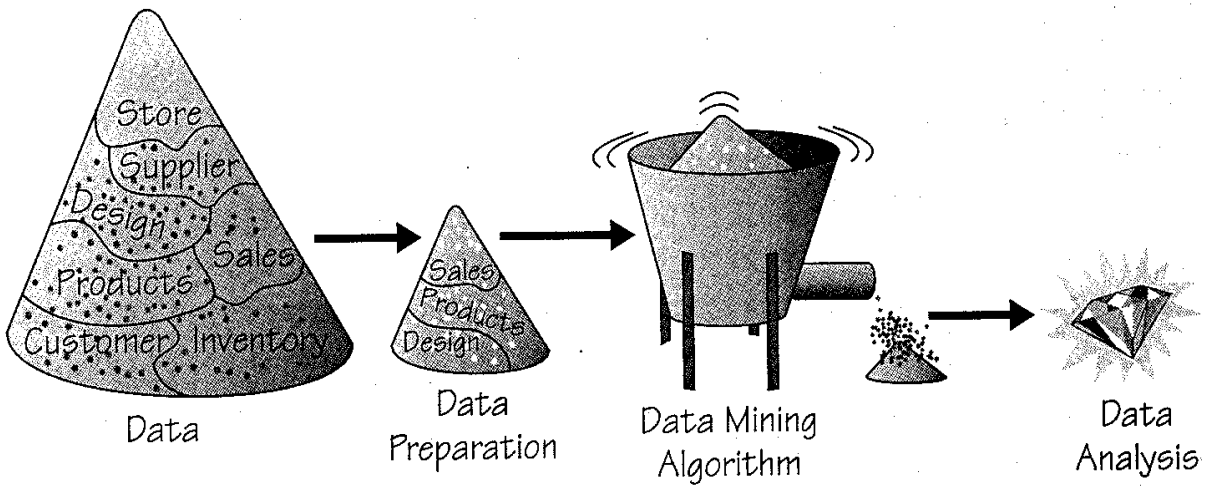
```
Select state, category, brand, Sum(qty*price)
From Sales F, Store S, Item I
Where F.storeID = S.storeID And F.itemID = I.itemID
Group By state, category, brand WITH CUBE
```

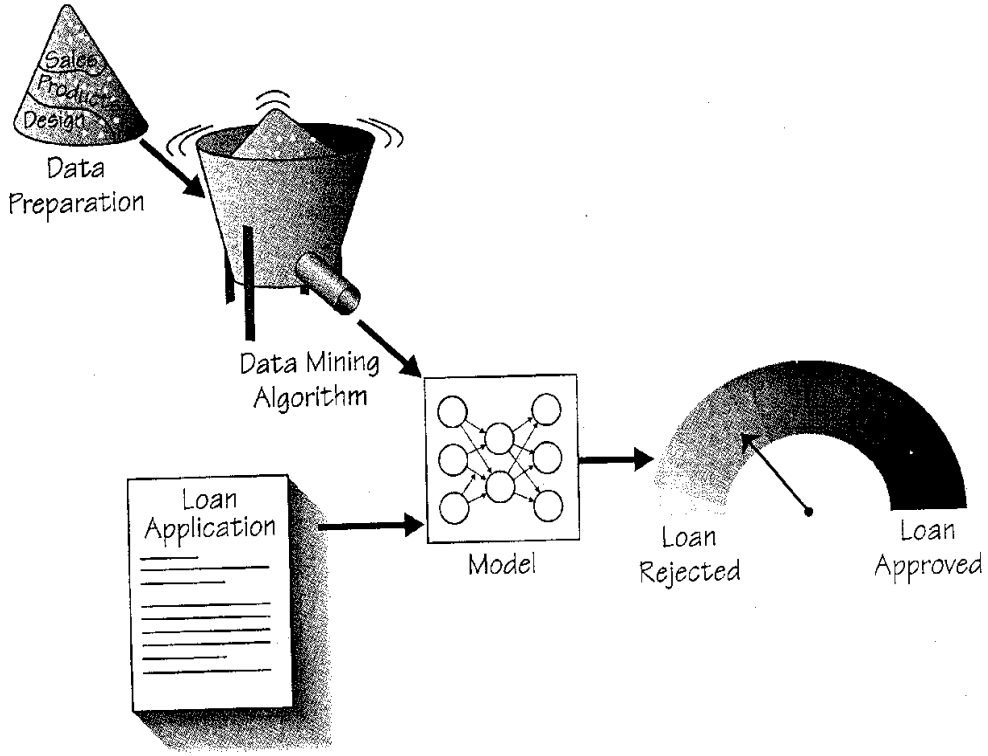
```
Select state, category, brand, Sum(qty*price)
From Sales F, Store S, Item I
Where F.storeID = S.storeID And F.itemID = I.itemID
Group By state, category, brand WITH ROLLUP
```

DATA MINING

Data Mining?

- 대규모의 데이터로부터 데이터의 패턴이나 규칙 등의 유용한 정보를 자동으로 발견하는 응용 및 방법 (Knowledge Discovery)
- 관련분야: 통계학, 인공지능, 데이터베이스 …
- 대표적인 분류
  - Association Rules
  - Clustering
  - Classification
  - Regression
- 참고:
  - Online book: Introduction to data mining, Saed Sayad, U. Toronto,


Store
Supplier
Design
Sales
Products
Customer
Inventory
Data
Sales
Products
Design
Data
Preparation
Data Mining
Algorithm
Data
Analysis


Sales
Products
Design
Data
Preparation
Data Mining
Algorithm
Loan
Application
Model
Loan
Rejected
Loan
Approved

Association Rules

- 연관규칙: 자주 함께 나타나는 item을 찾기
- 예: POS 판매 데이터 분석
  - {양파, 감자}를 사는 사람은 {햄버거}도 함께 사더라.
  - {기저귀}를 사는 사람은 {맥주}도 사더라.
- X가 일어나면 Y가 일어나더라.

$$Rule: \quad X \Rightarrow Y$$

$$Support = \frac{frq(X,Y)}{N}$$

$$Confidence = \frac{frq(X,Y)}{frq(X)}$$

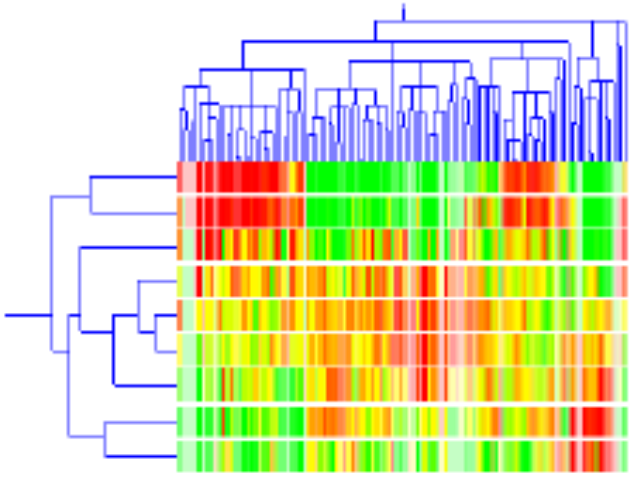
- 대표적인 알고리즘: Apriori

from http://chem-eng.utoronto.ca/~datamining/dmc/data_mining_map.htm

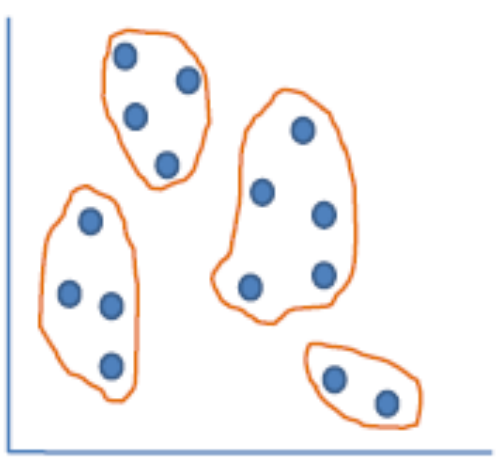
Clustering

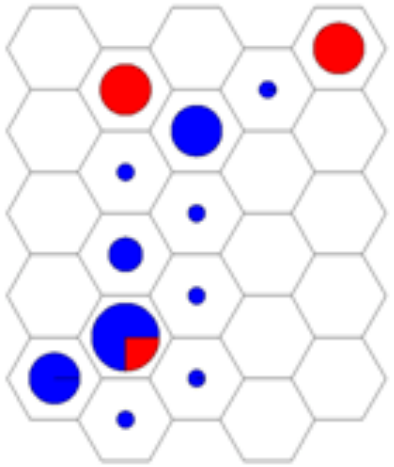
- 군집: 주어진 데이터를 비슷한 것끼리 나누시오.
- Unsupervised Learning: 학습용 데이터가 없음
- 종류
  - Hierarchical Clustering
  - Partitive Clustering

Hierarchical Clustering



Partitive Clustering

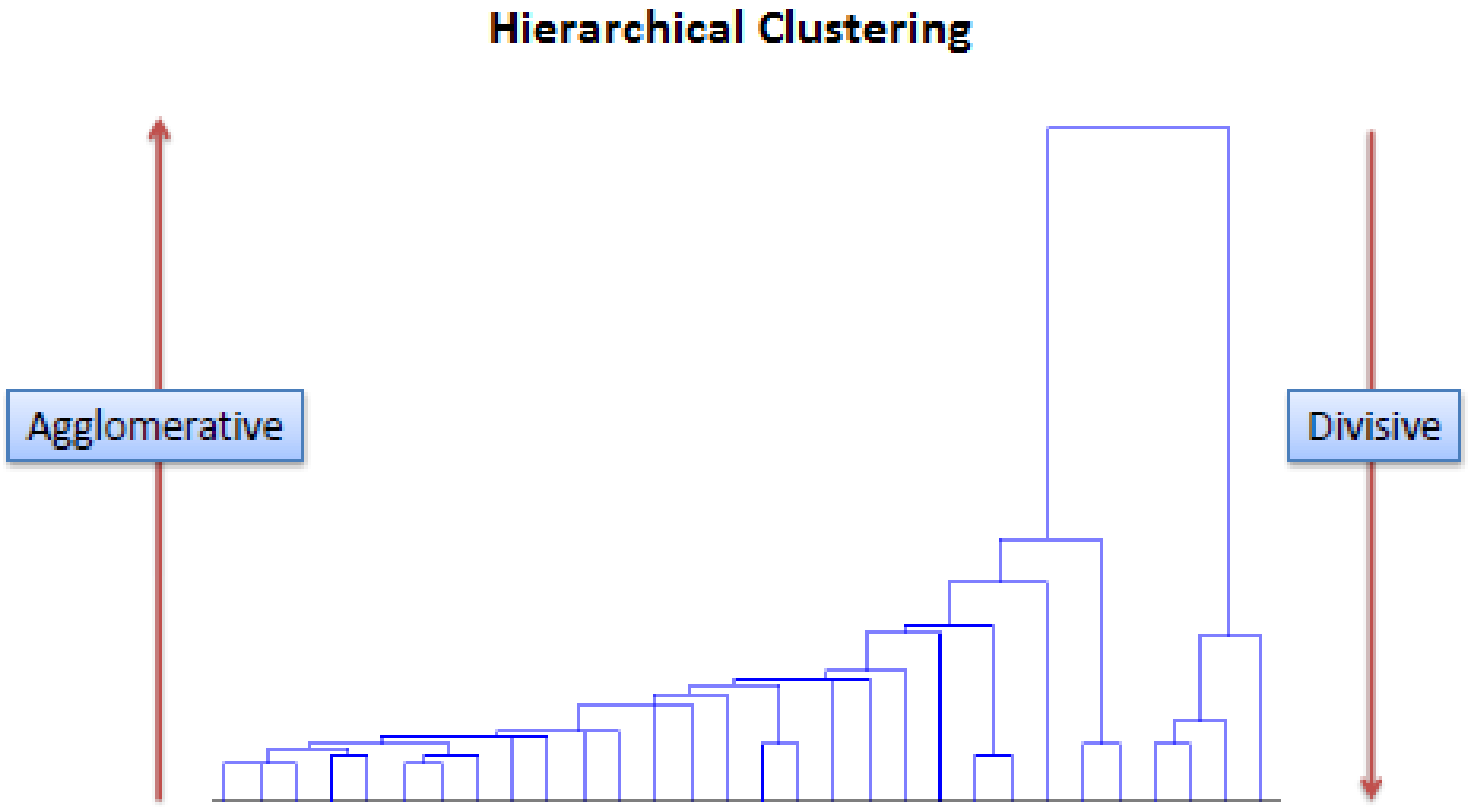




from http://chem-eng.utoronto.ca/~datamining/dmc/data_mining_map.htm

Clustering: Hierarchical Clustering

- Agglomerative: Bottom-Up
- Divisive: Top-Down

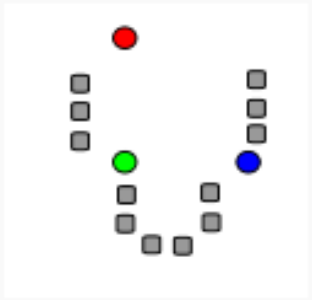


from http://chem-eng.utoronto.ca/~datamining/dmc/data_mining_map.htm

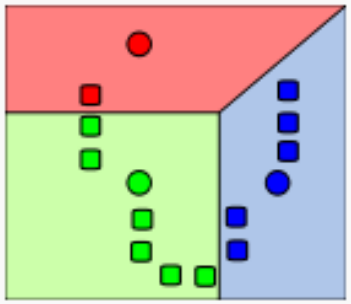
Clustering: Partitive Clustering

- K-Means Clustering

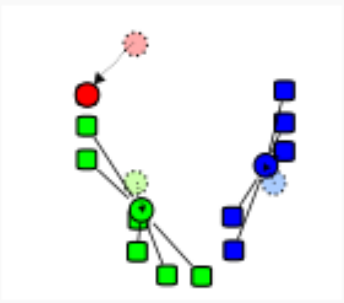
Demonstration of the standard algorithm



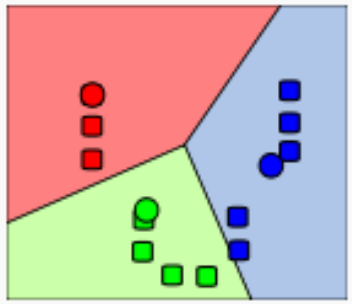
1) *k* initial "means" (in this case *k*=3) are randomly selected from the data set (shown in color).



2) *k* clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.



3) The centroid of each of the *k* clusters becomes the new means.

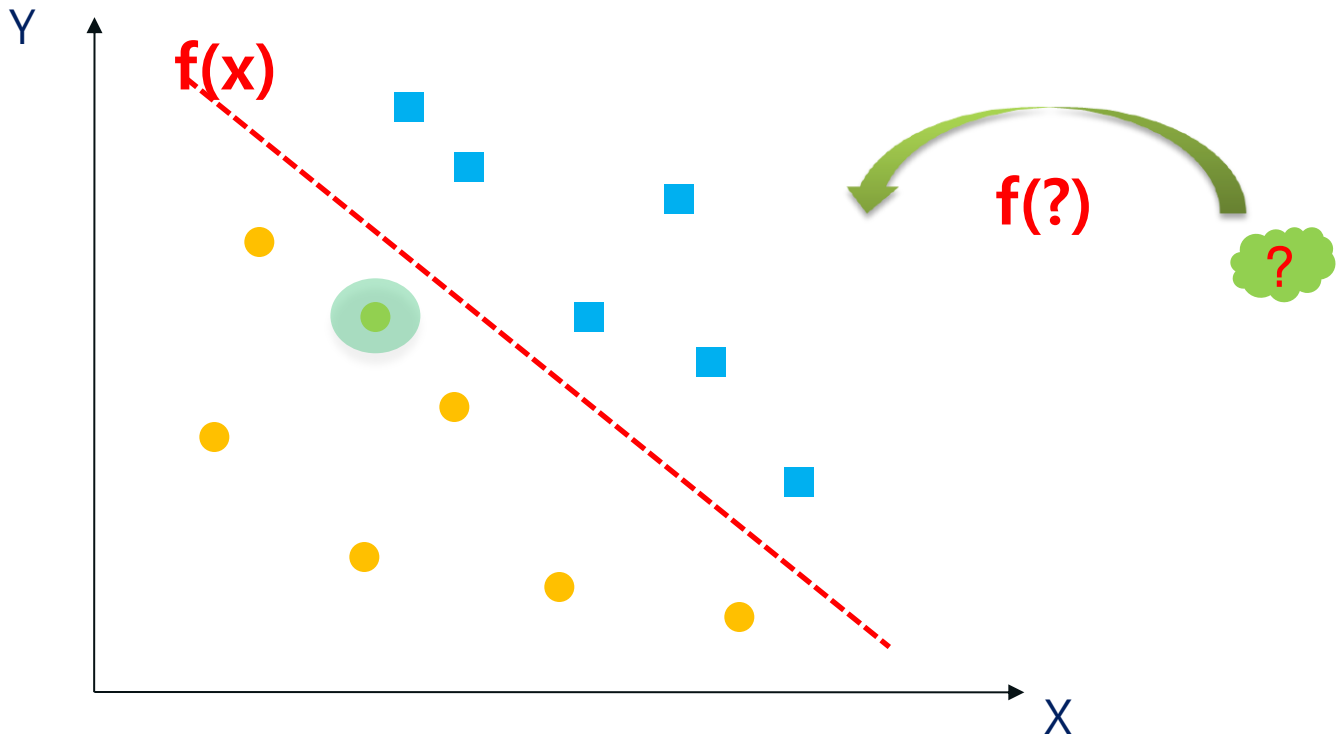


4) Steps 2 and 3 are repeated until convergence has been reached.

from http://en.wikipedia.org/wiki/K-means_clustering

Classification

- 분류: 주어진 변수에 대한 category (분류) 찾기
- 예: 이런 증상의 환자는 암에 걸렸는가? (YES/NO)
- Supervised Learning: training data가 필요



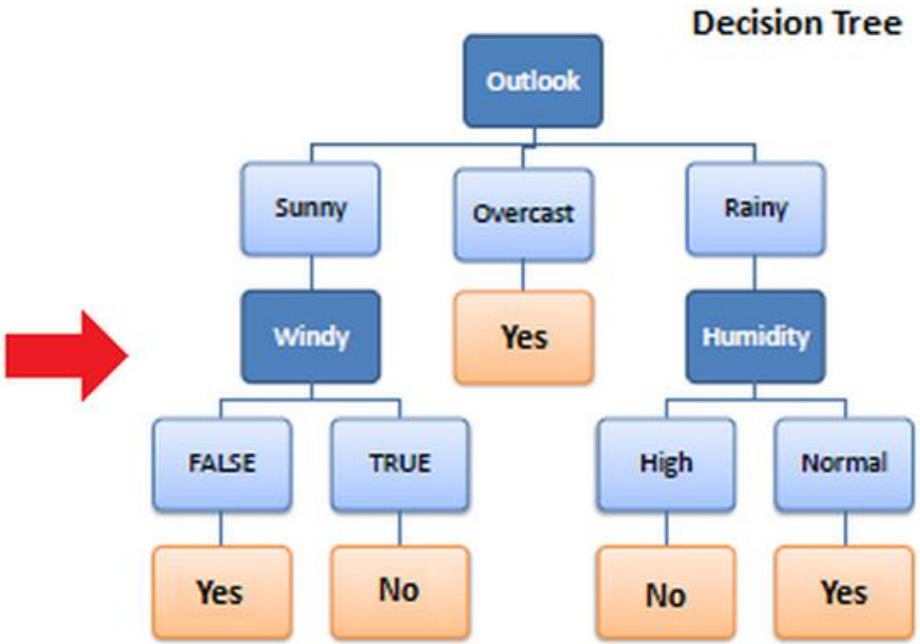
대표적 Classification 알고리즘

- **Decision Tree**
- k-Nearest Neighbors
  - 가장 가까운 값을 가진 데이터와 동일하게 분류
- Artificial Neural Network
- Naive Bayes
- **Support Vector Machines**

Decision Tree (의사결정트리)

- 예) ID3: Information Gain이 큰 것을 우선으로 분류
  - Information Gain: Entropy의 감소량
  - Entropy: 데이터의 무질서도

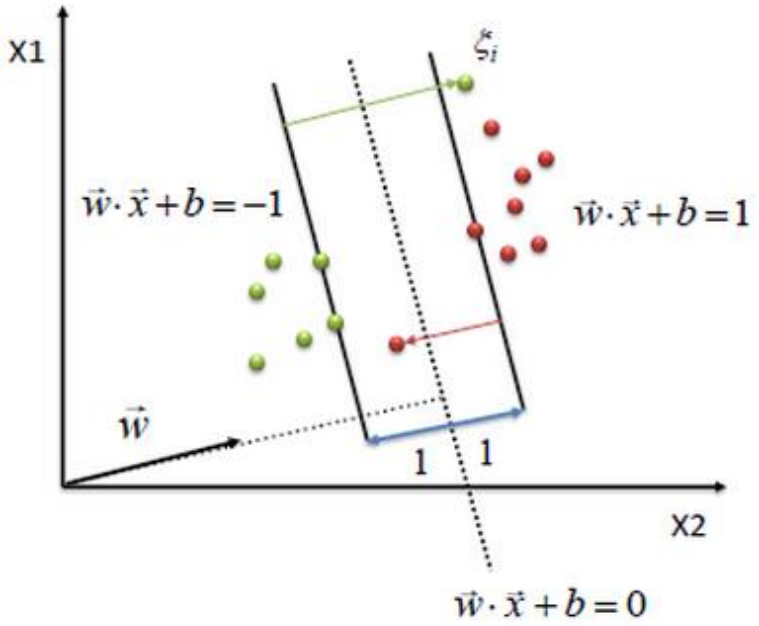
Predictors				Target
Outlook	Temp.	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

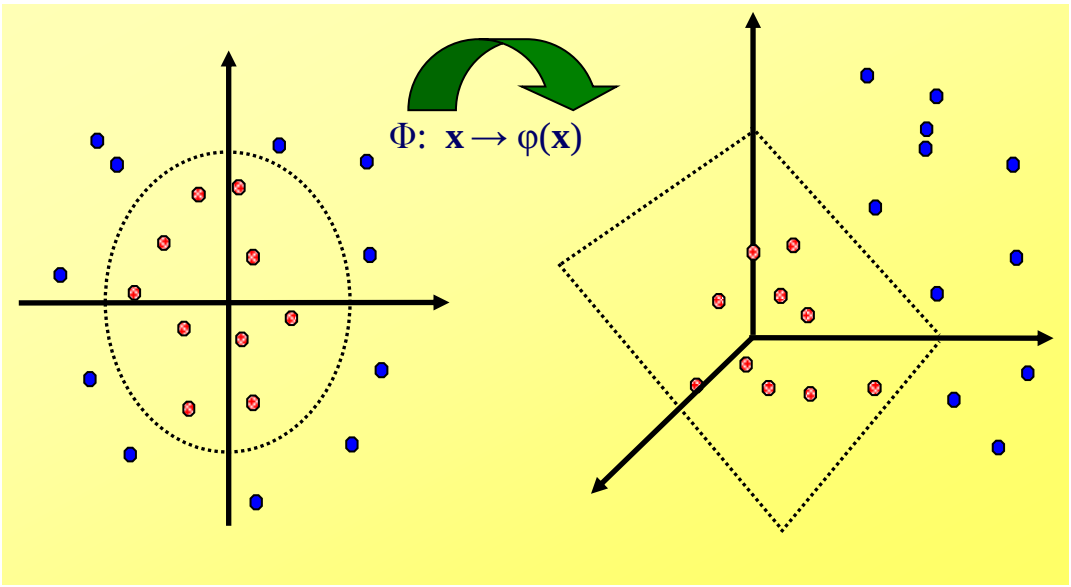


from http://chem-eng.utoronto.ca/~datamining/dmc/data_mining_map.htm

Support Vector Machine (SVM)

- Linear SVM: Define an optimal hyperplane - **maximize margin**
- Soft margin: have a penalty term for **misclassifications**.
- Nonlinear: using **Kernel** function (map data to feature space)
- Multiclass SVM: using many binary-SVMs (one-against-all/pairwise)

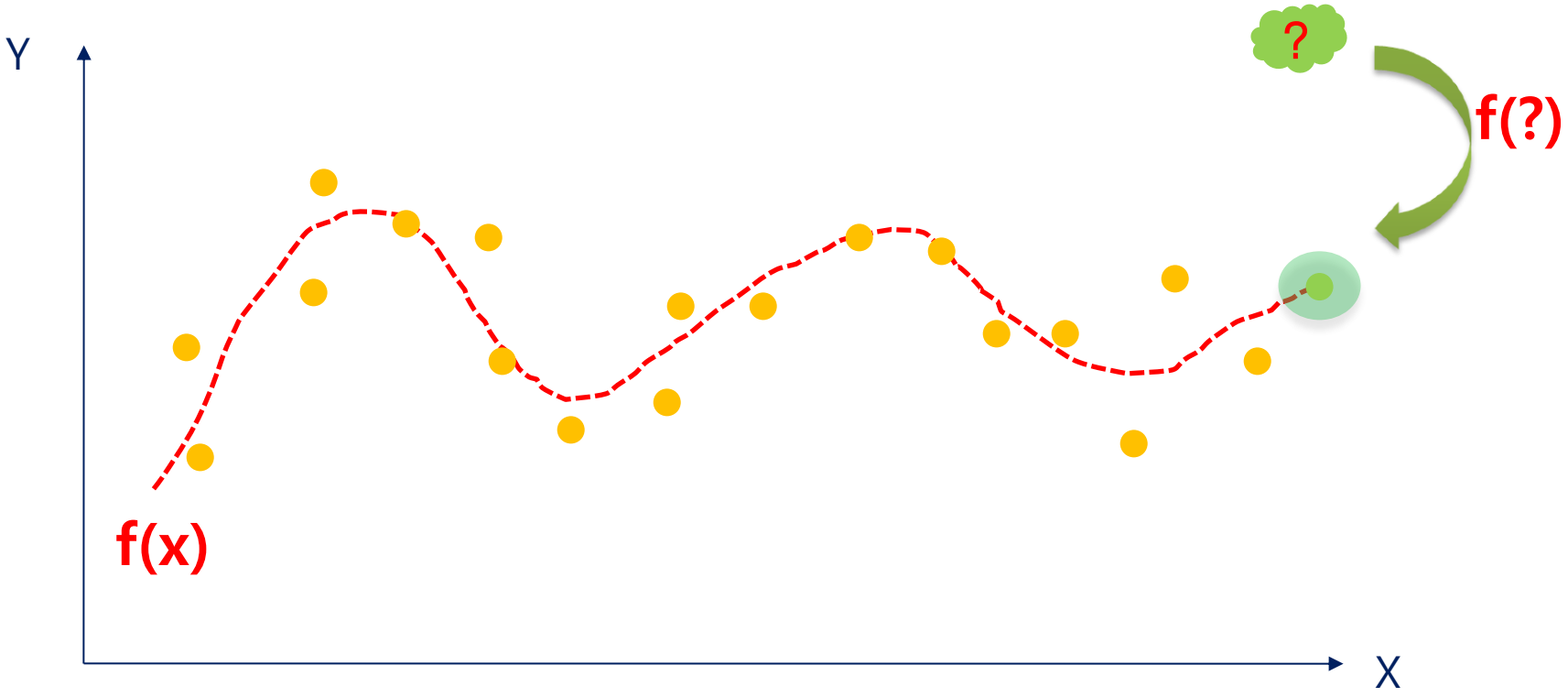




from http://chem-eng.utoronto.ca/~datamining/dmc/data_mining_map.htm

Regression

- 회귀: 주어진 변수에 대한 값(numeric) 예측
- 예: 이런 특징을 가진 환자의 기대 수명은?
- Supervised Learning: training data가 필요



Classification vs. Regression

- Supervised Learning:
  - 학습 데이터가 있음 (정답 y를 아는 데이터가 있음)
  - 1. 학습을 통하여 모델 f를 구축
  - 2. 이를 이용하여 새로운 X가 들어오면 y를 예측함.

$$y = f(X)$$

- Classification과 Regression의 차이점
  - Classification: y가 categorical value (예: YES/NO)
  - Regression: y가 numerical value (예: 임의의 실수)

Confusion Matrix

		Condition (as determined by "Gold standard")		
		Positive	*Negative*	
Test outcome	*Positive*	True Positive	False Positive (Type I error)	→ Positive predictive value
	Negative	False Negative (Type II error)	True Negative	→ Negative predictive value
		↓ Sensitivity	↓ Specificity	

Precision = TP / (TP+FP) P라고 예측한 것 중 맞은 비율

Sensitivity / Recall = TP / P (TP+FN) 실제 P중에 맞게 예측한 비율

Specificity = TN / (TN+FP) 실제 N중에 맞게 예측한 비율

Accuracy = (TP+TN) / (P+N) 전체 예측 중에 맞은 비율

from http://en.wikipedia.org/wiki/Sensitivity_and_specificity

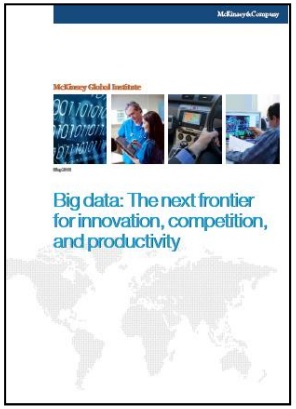
BIG DATA

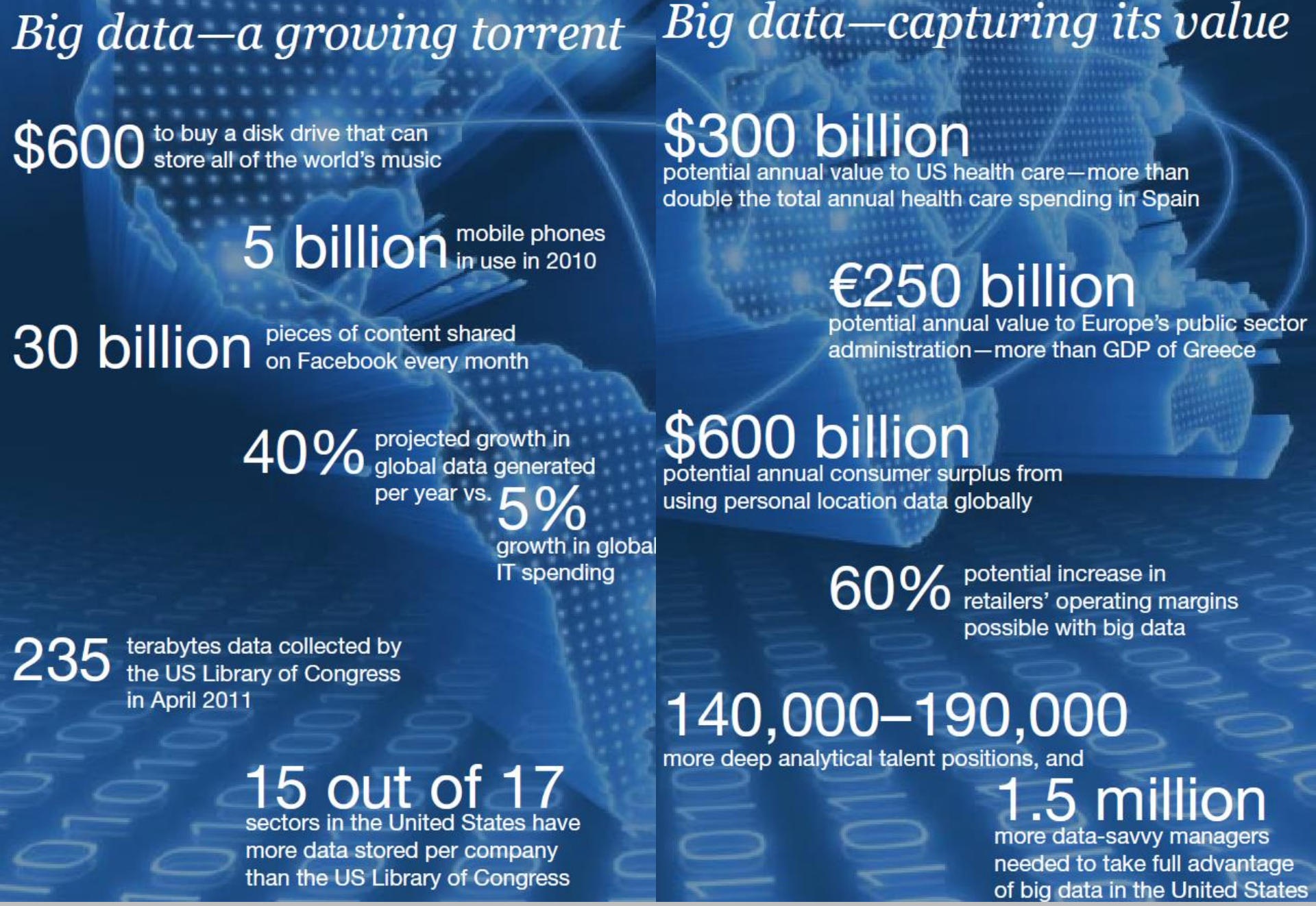
Outline

- **Background**
  - **Big Data Era**
  - **One size doesn't fit all.**
- NoSQL
- NewSQL

Big Data

- 데이터의 폭발적 증가
- 기존의 방식으로 저장/관리/분석하기 어려울 정도로 큰 규모의 Data
- 참고
  - Big data: The next frontier for innovation, competition, and productivity, McKinsey, 2011.05
    http://www.mckinsey.com/mgi/publications/big_data/pdfs/MGI_big_data_full_report.pdf
  - 정보홍수 속에서 금맥 찾기: 빅 데이터(Big Data) 분석과 활용, SERI, 2011.02
    http://seri.org/db/dbReptV.html?s_menu=0212&pubkey=db20110210001
  - Big Data, 미래를 여는 비밀 열쇠, KT 경제경영연구소, 2011.07
    http://www.digieco.co.kr/KTFront/report/report_strategy_view.action?board_seq=5520&board_id=strategy





Big data: The next frontier for innovation, competition, and productivity, McKinsey, 2011.05

ONE SIZE DOES NOT FIT ALL

- 전통적인 RDBMS
  - 관계형 데이터 모델, ER-Diagram, SQL
  - 스키마, 정규화, 데이터 무결성 …
  - 트랜잭션, ACID 속성 …
  - 병행처리(Concurrency control), 2PLP…
  - 회복, 로그 …

- 전통적 DBMS의 문제점
  - Scalability: 오라클을 10,000대에 설치/관리할 수 있나??
  - Performance: 오라클에서 초당 만건 이상의 변경을 처리할 수 있나?
  - Schema: 정형화된 스키마가 없으면?
  - Reliability는 필요 없으니 더 빠를 수는 없나?
  - Persistent는 필요 없으니 더 쉬울 수는 없나?
  - 복잡한 데이터 모델은 필요 없으니 더 간단할 수 없나?

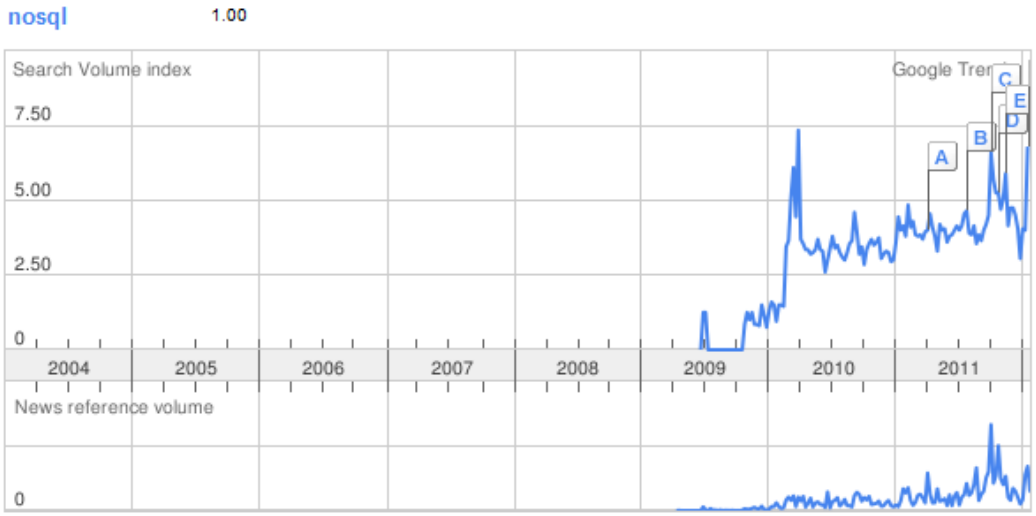
Outline

- Background
- **NoSQL**
  - **Not Only SQL**
  - **MapReduce**
  - **Key/Value**
  - **Document**
  - **Graph**
- NewSQL

NoSQL

- "Not only SQL"
- 전통적인 RDBMS와는 다른 종류의 새로운 데이터 저장/관리 시스템

- Google trends Jan. 2012



A Microsoft Researchers: NoSQL Needs Standardization
PCWorld - Apr 5 2011

B CouchBase, SQLite Launch Unified NoSQL Query Language
PCWorld - Jul 29 2011

C Oracle does about-face on NoSQL
Computerworld - Oct 4 2011

D Neo Technology Gains Significant Momentum Driving Enterprise NOSQL
MarketWatch - Oct 25 2011

E First Look: Oracle NoSQL Database
Slashdot - Nov 16 2011

F Amazon Web Services launches NoSQL effort: Here comes some disruption
ZDNet - Jan 18 2012

NoSQL의 장/단점

- 장점
  - 유연한 스키마
  - 쉽고 빠른 설치/관리
  - Massive Scalability
  - 완화된 일관성(consistency)→ High Performance & Availability

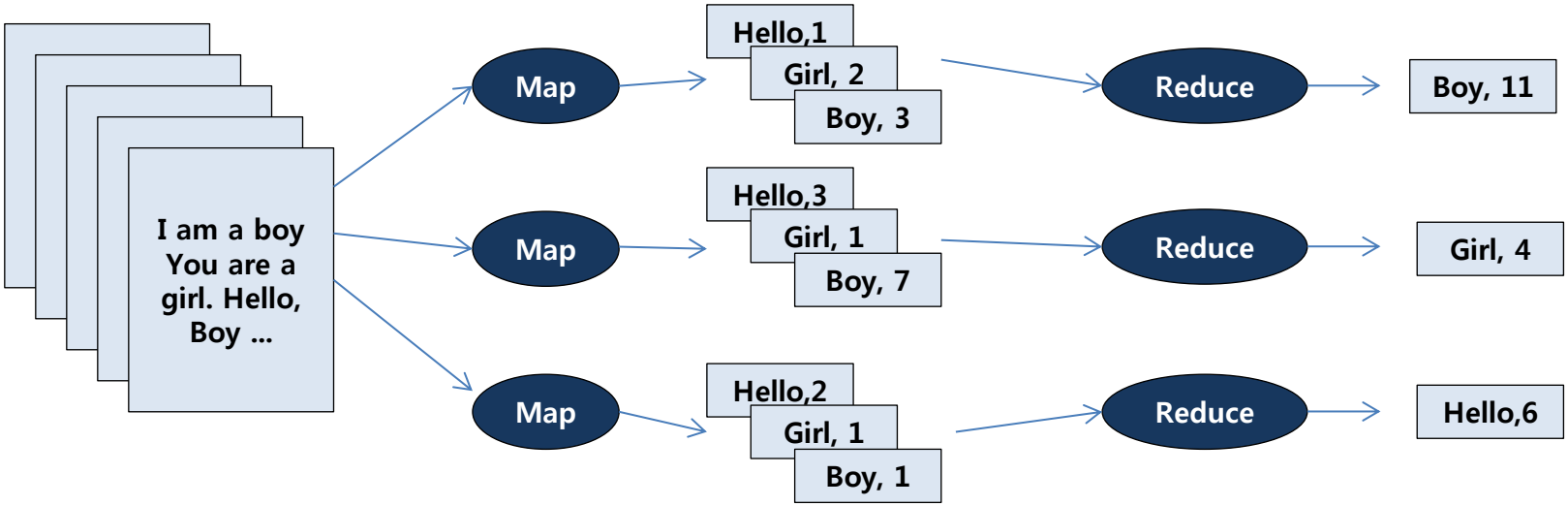
- 단점
  - SQL 과 같은 표준 질의 언어 부족 → 프로그래밍 모델 도입
  - 완화된 일관성 → ACID가 보장 안됨

NoSQL 종류

- MapReduce / **Hadoop**
  - 대용량 데이터 분석 (OLAP)
  - Google: MapReduce + Google File System
  - Apache: Hadoop + Hadoop Distributed File System
- Key-Value
  - High Performance 병렬 해시
  - 잦은 변경, 빠른 반응 속도 (OLTP)
  - Dynamo (Amazon), **Cassandra** (Facebook/Apache), BigTable (Google), **HBase** (Facebook/Apache)
- Document
  - Key + 문서 (XML, JSON 과 같은 반구조화/비구조화 자료구조)
  - **MongoDB**, **CouchDB** (Apache), SimpleDB (Amazon)
- Graph
  - Node/Edge, RDF, Semantic Web
  - 예: Neo4j, Allegro

MapReduce

- Map: 문제를 sub-problem으로 나누어 분산 해결
  - **map(item) → <key, value>**
- Reduce: sub-problem 별로 결과를 취합
  - **reduce(key, <list of values>) → value**
- 예) 책 한 권에서 단어별 출연 빈도수 세기
  - Map: 아이들 각각에게 한 페이지씩 나누어주고, <단어, 빈도수>를 각각 포스트잇에 적어 보고하도록 함.
  - Reduce: A/B/C/D/E... 별로 포스트잇을 취합하여 각각 숫자를 더함



MapReduce 프레임워크

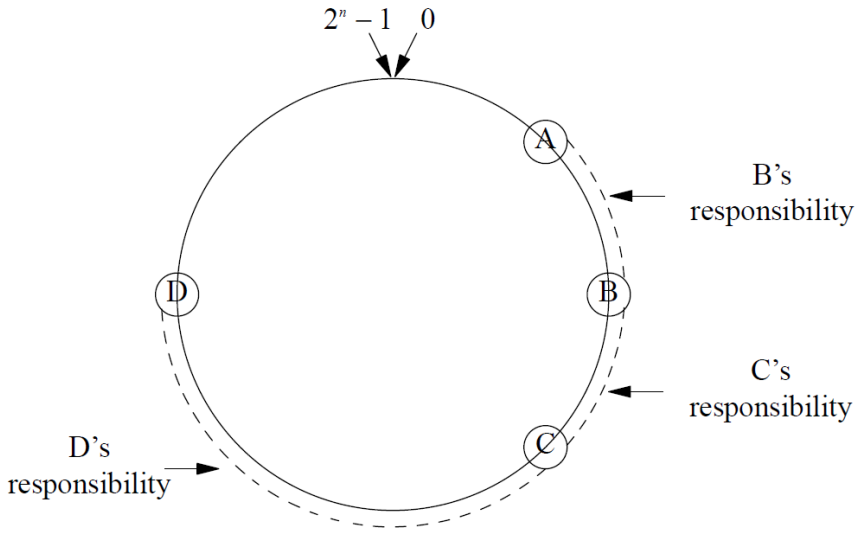
- Apache Hive (Facebook/Netflix)
  - Hadoop + Data warehouse
  - HiveQL(SQL 유사 질의 언어) 지원
- Apache Pig (Yahoo)
  - Pig Latin: High-level language for Hadoop
- Apache ZooKeeper
  - 분산환경에서 설정 공유, 이벤트 처리, 분산 관리 등
  - open source centralized configuration service and naming registry for large distributed systems

Key/Value

- 단순한 데이터 모델: (key, value)
- 단순한 연산: put, get, update, delete
- 장점
  - Efficiency: 빠른 처리 속도
  - Scalability: 필요에 따른 손쉬운 서버 확장
  - Fault-tolerance: 데이터 복제

DHT: Distributed Hash Tables

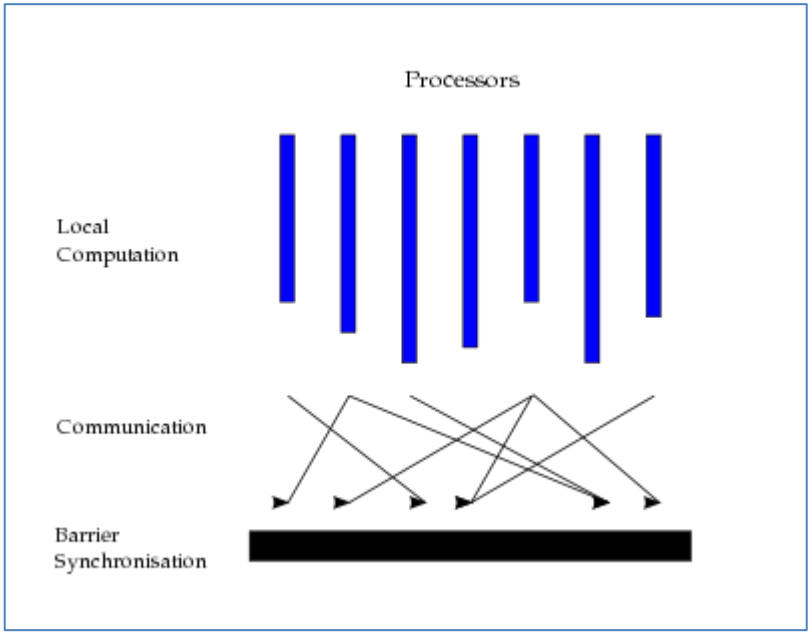
- Hash
  - put(key, value)
  - value ← get(key)
- Distributed
  - 임의의 노드에 분산 저장
  - 노드의 추가/삭제가 자유로움
- 대표적인 구현 방법
  - Chord(1):
    - Ring형태 구성
    - m비트의 키와 노드 ID사용
    - O(log N)의 라우팅 테이블 크기
    - O(log N) 홉 만에 데이터 도달 보장



참고문헌
(1) Stoica, Ion et al. (2001). "Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications". Proceedings of SIGCOMM'01

BSP (Bulk Synchronous Parallel)

- 병렬 계산을 위한 계산 모델



- Pregel: A System for Large-Scale Graph Processing, SIGMOD 2010 by Google
- Apache Hama: http://hama.apache.org/
- Apache Giraph: http://incubator.apache.org/giraph/

Outline

- Background
- NoSQL
- **NewSQL**
  - **New approaches**
  - **Analytic DBMS**

NewSQL (?)

- VoltDB by **Michael Stonebraker**
  - Automatic partitioning across a shared nothing server cluster
  - Main memory data architecture
  - Elimination of multi-threading and locking overhead
  - Built-in High Availability using synchronous multi-master replication
  - Review of VoltDB's stored procedure interface

- Vertica (HP) by **Michael Stonebraker**
  - Real-Time Loading & Querying
  - Advanced In-Database Analytics
  - Columnar Storage & Execution
  - Aggressive Data Compression
  - Scale-Out MPP Architecture
  - Automatic High Availability
  - Optimizer, Execution Engine & Workload Management
  - Native BI, ETL, & Hadoop/MapReduce Integration

TPC-H (OLAP Benchmark) 결과 1

100 GB Results

Rank	Company	System	QphH	Price/QphH	Watts/KQphH	System Availability	Database	Operating
1	DELL	Dell PowerEdge R710 using EXASolution 4.0	1,112,401	.12 USD	NR	10/01/11	EXASOL EXASolution 4.0	EXASOL EXACluster OS 4.
2	DELL	Dell PowerEdge R710 using EXASolution 4.0	377,012	.22 USD	NR	10/01/11	EXASOL EXASolution 4.0	EXASOL EXACluster OS 4.
3	DELL	Dell PowerEdge R610 using VectorWise	303,289	.16 USD	1.28	06/30/11	VectorWise 1.6	RedHat Enterprise Linux.6
4	Actian	HP ProLiant DL380 G7	251,561	.38 USD	NR	03/31/11	VectorWise 1.5	RedHat Enterprise Linux.6
5	CPI server&storage	CPI Phoenix IQ-201	209,298	1.25 USD	NR	01/14/08	EXASOL EXASolution 2.0	EXASOL EXACluster OS 1.
6	hp invent	HP ProLiant DL380 G7	73,974	.58 USD	5.93	07/02/10	Microsoft SQL Server 2008 R2 Enterprise Edition	Microsoft Windows Server Edition
7	hp invent	HP ProLiant DL385 G7	71,438	.51 USD	6.48	07/14/10	Microsoft SQL Server 2008 R2 Enterprise Edition	Microsoft Windows Server Edition
8	SYBASE An SAP Company	Sun Fire X4270	53,501	1.14 USD	NR	12/04/09	Sybase IQ Single Application Server Edition v.15.1 ESD #1	Sun Solaris 10
9	hp invent	HP ProLiant DL380 G6	51,422	1.07 USD	NR	09/14/09	Microsoft SQL Server 2008 Enterprise x64 Edt SP1	Microsoft Windows Server Edt SP2
10	hp invent	HP ProLiant DL380 G6	51,085	1.09 USD	NR	10/05/09	Microsoft SQL Server 2008 Enterprise x64 Edt SP1	Microsoft Windows Server Edt SP2

TPC-H (OLAP Benchmark) 결과 2

10,000 GB Results

Rank	Company	System	QphH	Price/QphH	Watts/KQphH	System Availability	Database
1	DELL	Dell PowerEdge R710 using EXASolution 4.0	7,128,255	.53 USD	NR	10/01/11	EXASOL EXASolution 4.0
2	IBM	IBM System p 570	343,551	32.89 USD	NR	04/15/08	IBM DB2 Warehouse 9.5
3	HP invent	HP Integrity Superdome/Dual-Core Itanium/1.6 GHz	208,457	27.97 USD	NR	09/10/08	Oracle Database 11g Enterprise Edition
4	IBM	IBM System p5 575 with DB2 UDB 8.2	180,108	47.00 USD	NR	08/30/06	IBM DB2 UDB 8.2
5	HP invent	HP Integrity Superdome-DC Itanium2/1.6GHz/64p/128c	171,380	32.91 USD	NR	04/01/07	Oracle Database 10g R2 Enterprise Edt w/Partitioning
6	HP invent	HP Integrity Superdome - Itanium2/1.5 GHz-128p/128	86,282	161.24 USD	NR	04/06/05	Oracle Database 10g Enterprise Edition
7	UNISYS	Unisys ES7000 Model 7600R Enterprise Server(16s)	80,172	18.95 USD	NR	02/17/09	Microsoft SQL Server 2008 Enterprise x64 Edition
8	HP invent	HP Integrity Superdome	63,650	38.54 USD	NR	08/30/08	Microsoft SQL Server 2008 Enterprise Editio
9	HP invent	HP Integrity Superdome - Itanium2/1.5 GHz-64p/64c	49,104	118.13 USD	NR	03/25/04	Oracle Database 10g Enterprise Edition

분석 전용 DBMS

- 기존 DBMS
  - Concurrency Control
  - Row-기반 저장구조
  - 중앙 집중 서버
  - 고수준의 트랜잭션 레벨
  - 디스크 기반
- 분석 전용 DBMS
  - No concurrency
  - Column-기반 저장구조
  - 병렬/분산 처리
  - 약화된 트랜잭션 레벨
  - 메인 메모리 기반

참고: Key Features of EXASOL

- Relational database management system
- Standard hardware cluster
- In-memory query processing
- Massively parallel data processing
- Column by column storage
- Intelligent and innovative compression algorithms
- Self-learning and self-optimising system
- Simple integration thanks to standard interfaces

Column-Oriented DBMS

- 컬럼 순서로 데이터 저장

고객ID	이름	주소	나이	전화번호
CE1	박현민	서울	24	....
CE2	이강선	대전	20	....
CE3	권동섭	서울	18	....

- 이점
  - 컬럼 단위의 대한 분석에 용이
  - 압축이 효과적 → 메모리 기반 처리 용이
  - 병렬 처리에 유리 (SIMD: Single Instruction, Multiple Data)