BUSINESS INTELLIGENCE

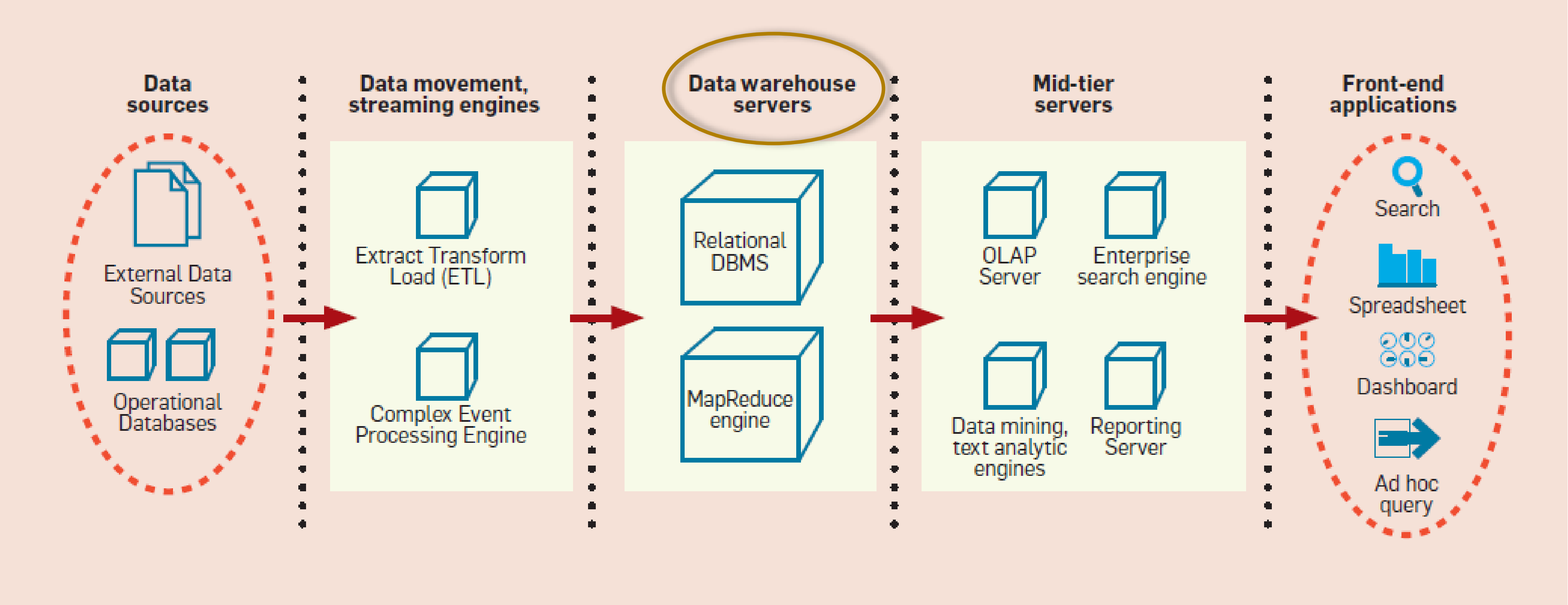
Reminds on Data Warehousing

(details at the Decision Support Database course)

Business Informatics Degree

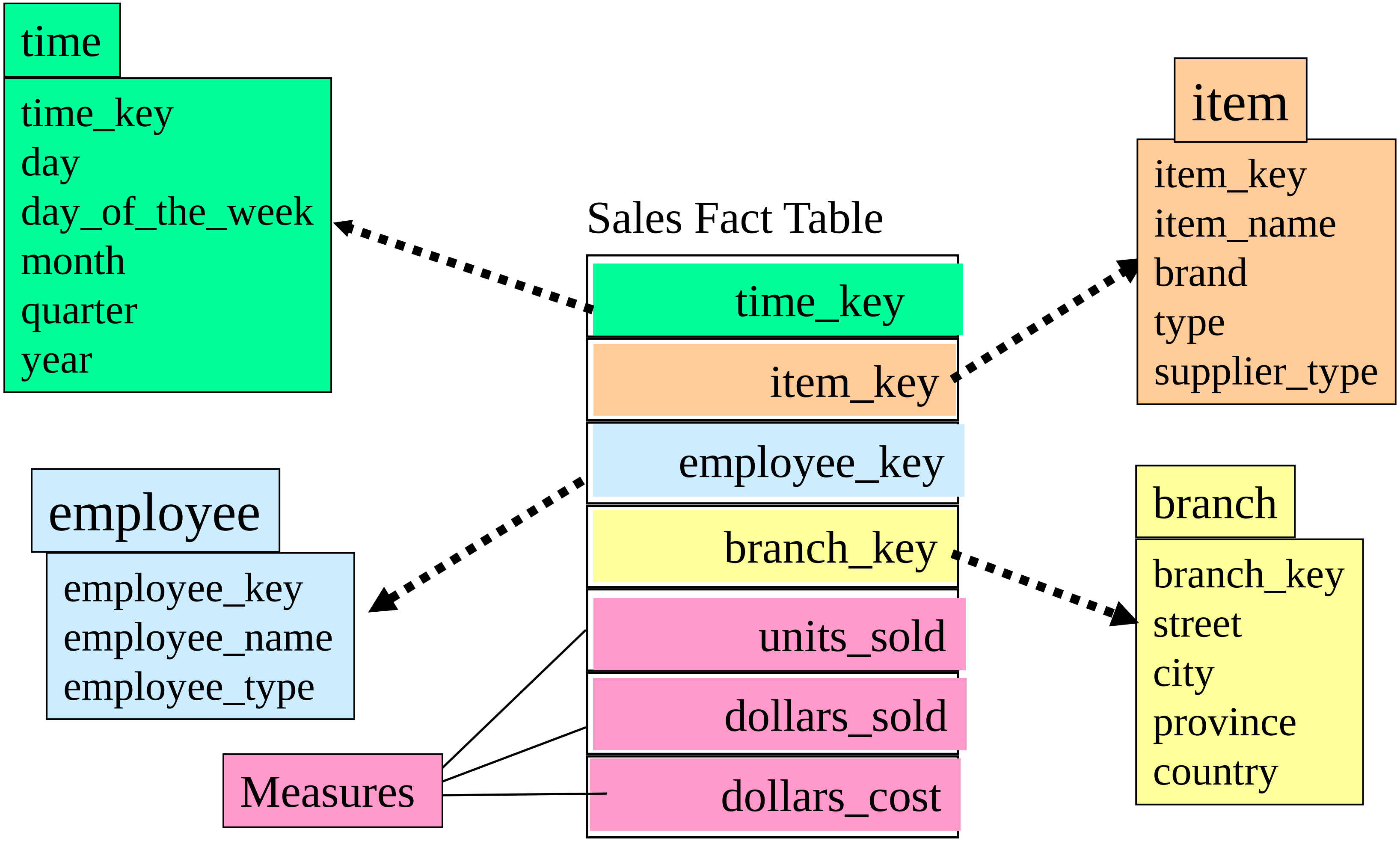
BI Architecture

Figure 1. Typical business intelligence architecture.



Star-schema datawarehouse

- A fact table with star-schema dimension tables only



time
- time_key
- day
- day_of_the_week
- month
- quarter
- year

Sales Fact Table
- time_key
- item_key
- employee_key
- branch_key
- units_sold
- dollars_sold
- dollars_cost

item
- item_key
- item_name
- brand
- type
- supplier_type

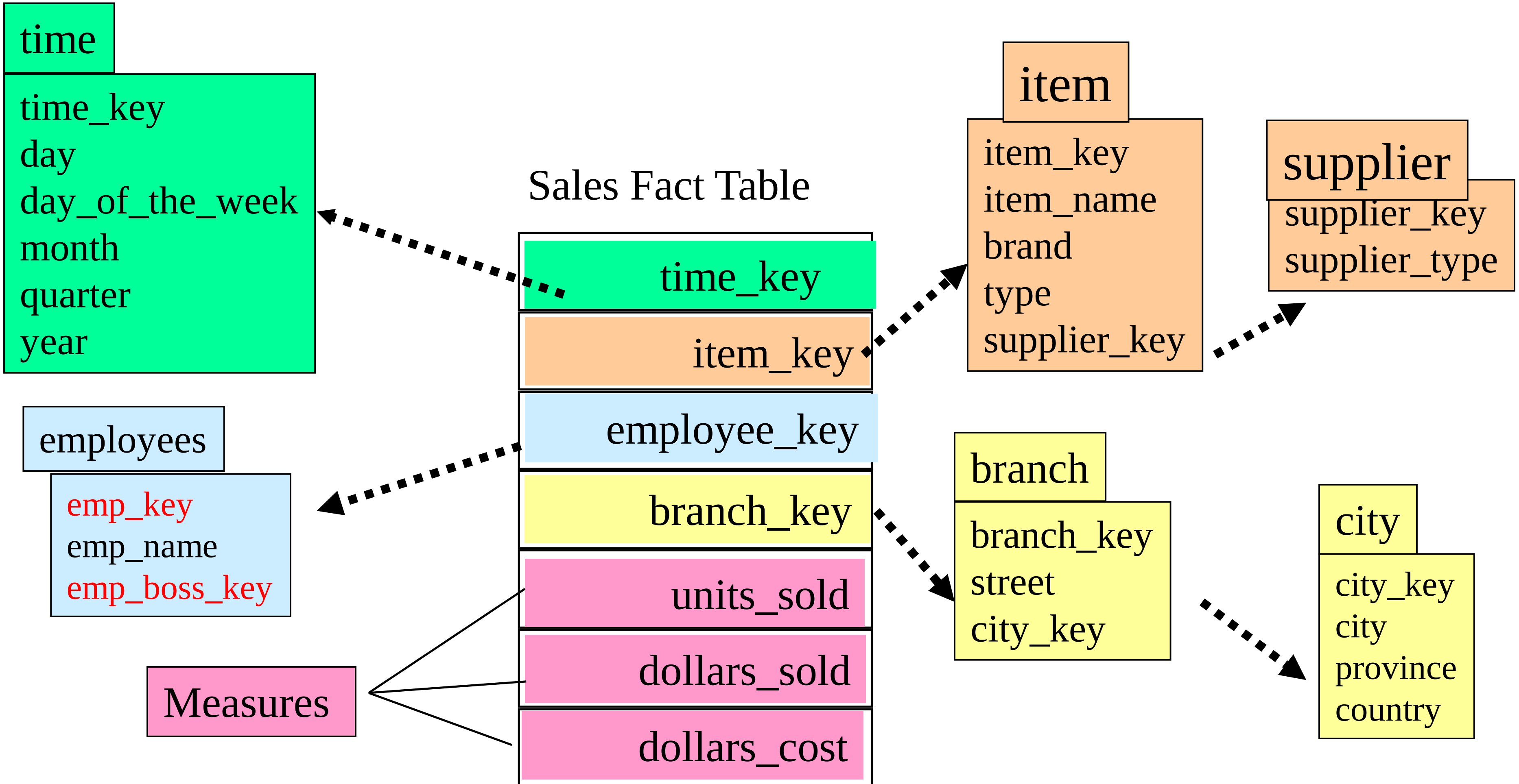
employee
- employee_key
- employee_name
- employee_type

branch
- branch_key
- street
- city
- province
- country

Measures

Snowflake-schema datawarehouse

- A fact table with star-schema, snowflake and parent-child dimension tables



time
time_key
day
day_of_the_week
month
quarter
year

Sales Fact Table
time_key
item_key
employee_key
branch_key
units_sold
dollars_sold
dollars_cost

item
item_key
item_name
brand
type
supplier_key

supplier
supplier_key
supplier_type

employees
emp_key
emp_name
emp_boss_key

branch
branch_key
street
city_key

city
city_key
city
province
country

Measures

Which DBMS technology for DW?

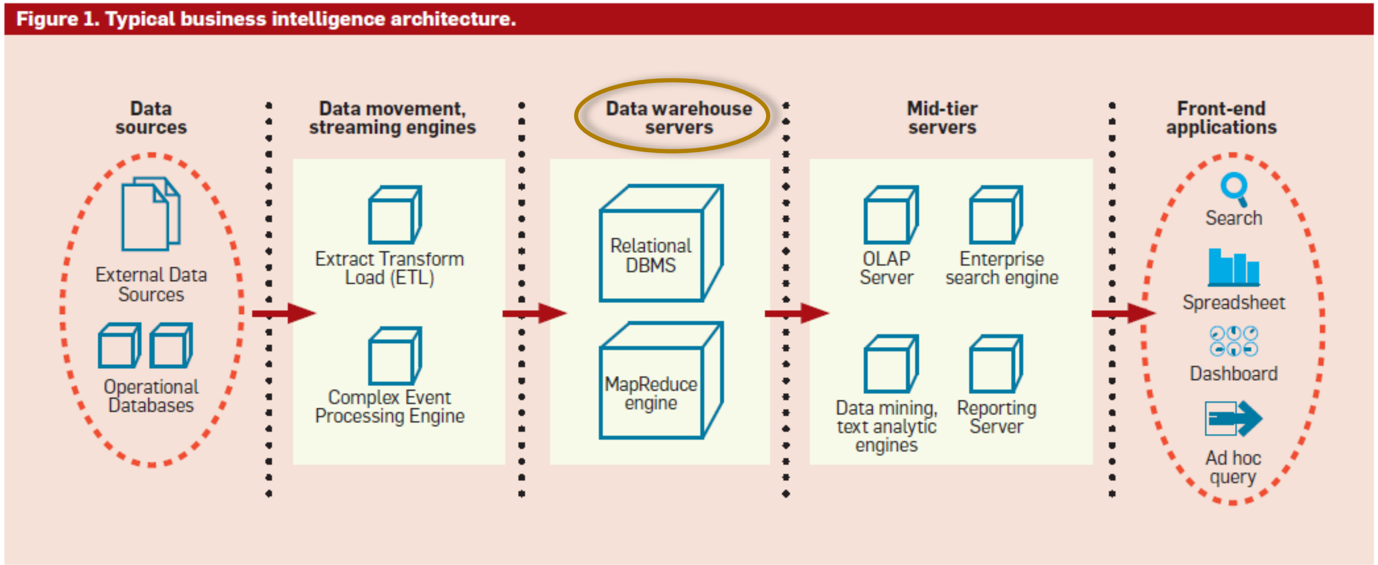
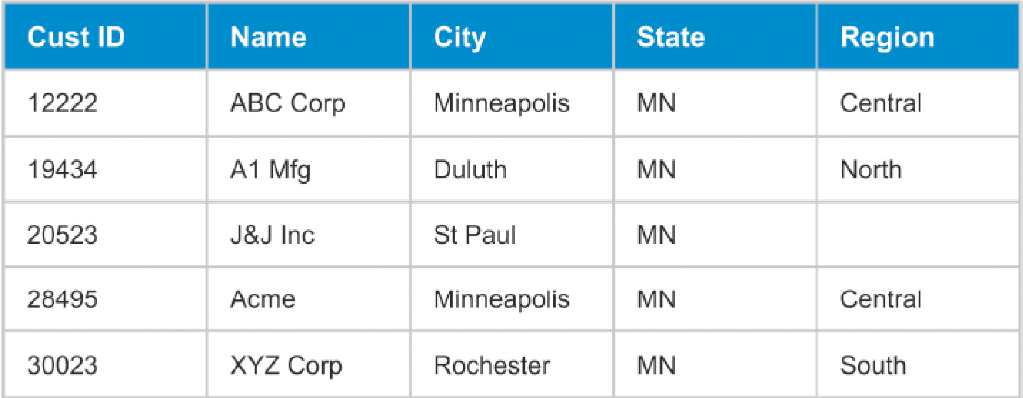


Figure 1. Typical business intelligence architecture.

- □ Storage technology
- □ Architecture

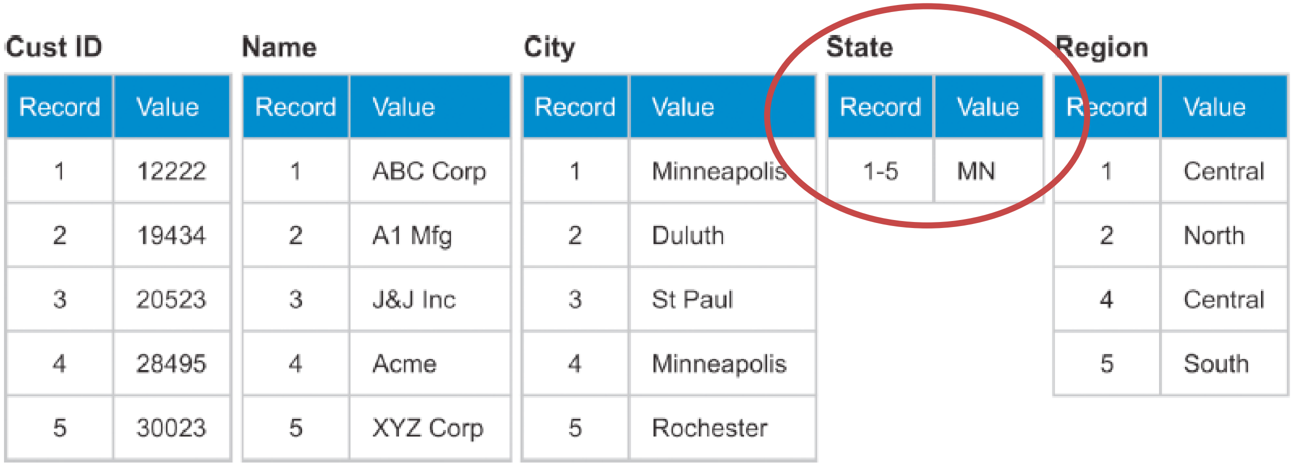
Storage

RDBMS: record oriented structure



Cust ID	Name	City	State	Region
12222	ABC Corp	Minneapolis	MN	Central
19434	A1 Mfg	Duluth	MN	North
20523	J&J Inc	St Paul	MN	
28495	Acme	Minneapolis	MN	Central
30023	XYZ Corp	Rochester	MN	South

Columnar: column oriented structure



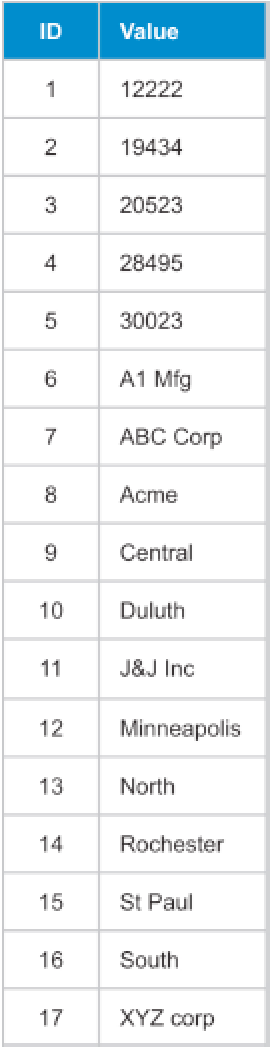
Cust ID		Name		City		State		Region	
Record	Value	Record	Value	Record	Value	Record	Value	Record	Value
1	12222	1	ABC Corp	1	Minneapolis	1-5	MN	1	Central
2	19434	2	A1 Mfg	2	Duluth			2	North
3	20523	3	J&J Inc	3	St Paul			4	Central
4	28495	4	Acme	4	Minneapolis			5	South
5	30023	5	XYZ Corp	5	Rochester				

Advantages:

- Faster Scan
- Data Compression (e.g. State)

Storage

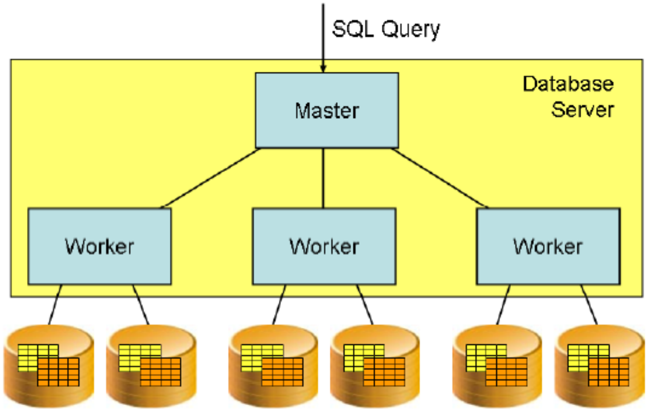
- Correlation value-based database
  - Data cells contain the index to an order-set value
- In-memory database
  - Data is stored in compressed format in main memory
- Extraction-based system
  - Storage of attribute extracted from continuous data flows (eg., web traffic, sensors)
- …

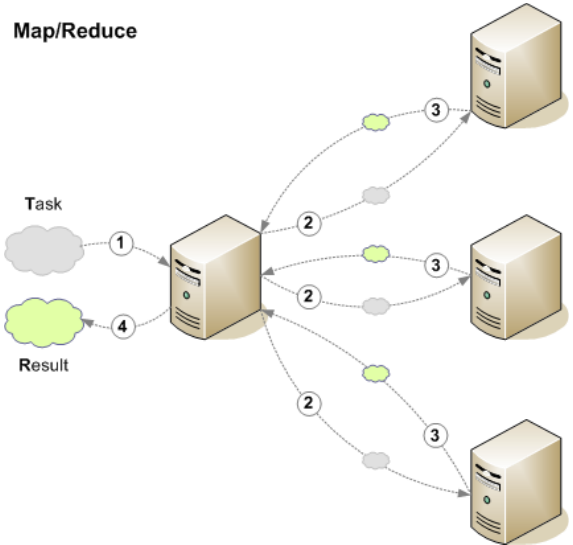


ID	Value
1	12222
2	19434
3	20523
4	28495
5	30023
6	A1 Mfg
7	ABC Corp
8	Acme
9	Central
10	Duluth
11	J&J Inc
12	Minneapolis
13	North
14	Rochester
15	St Paul
16	South
17	XYZ corp

Architecture

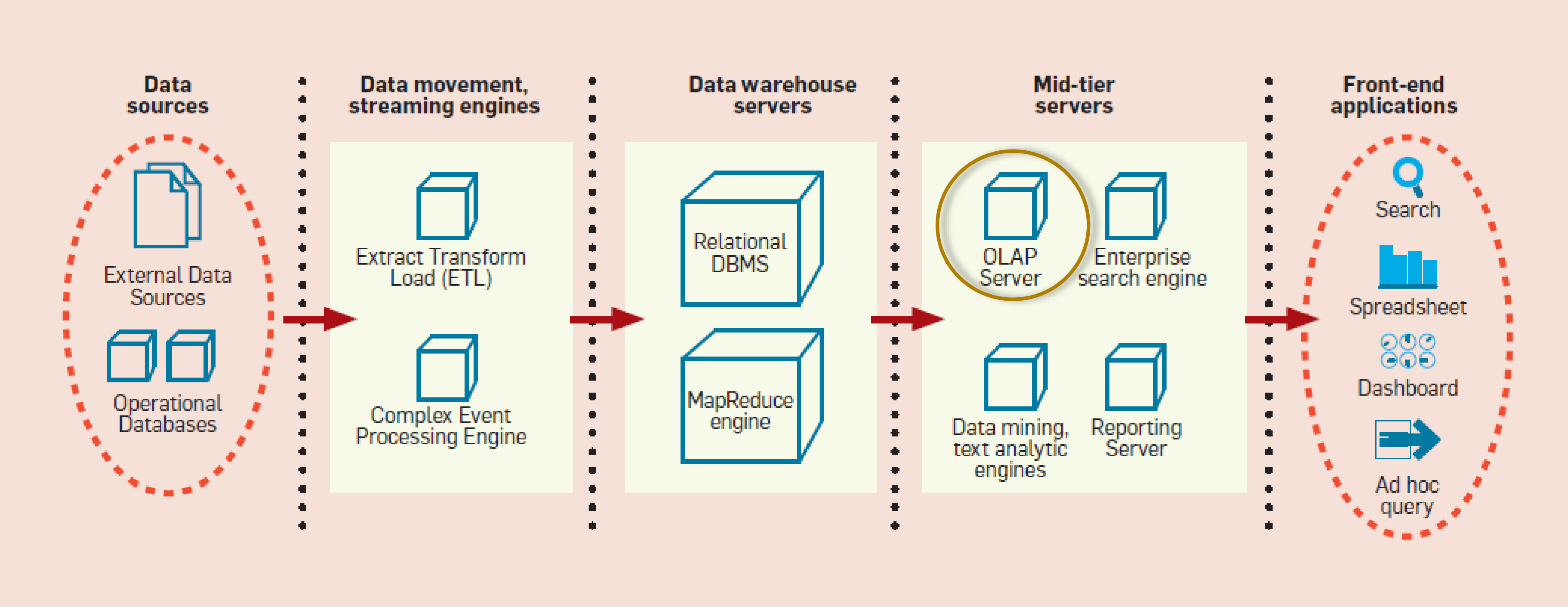
- Sequential
  - SQL query processing by a single processor
- Parallel
  - SQL query plan processing by a multi-processor machine, with shared memory
- Distributed (Map-reduce)
  - SQL query processing distributed to a set of independent machines
    - Teradata SQL-MR, Hadoop HiveQL



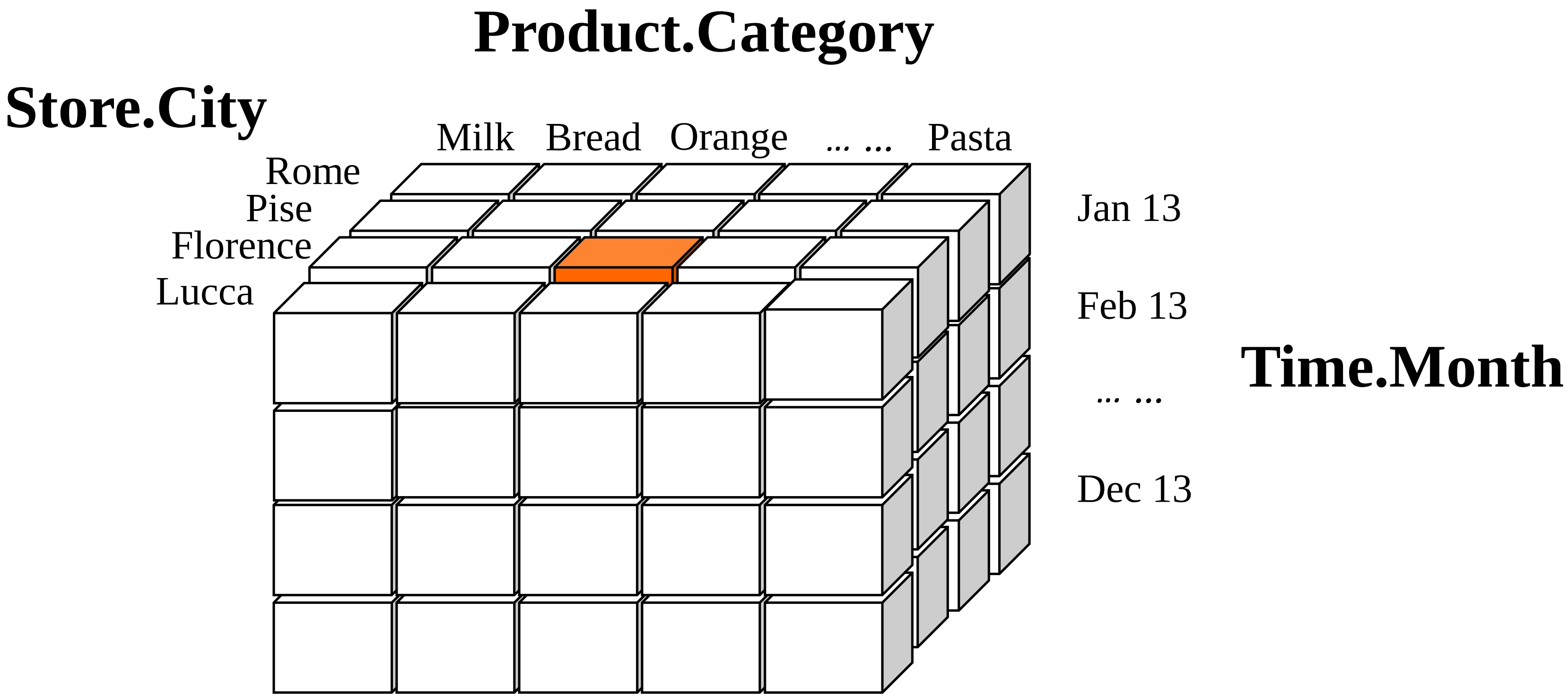


BI Architecture

Figure 1. Typical business intelligence architecture.

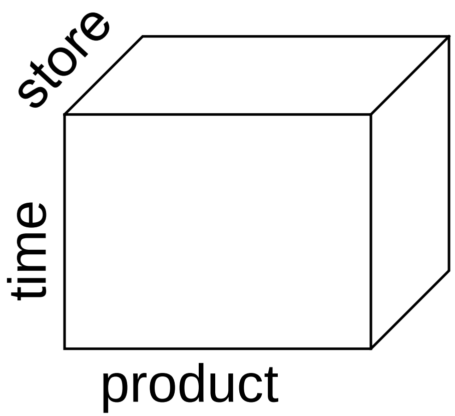


K-dimensional cuboid

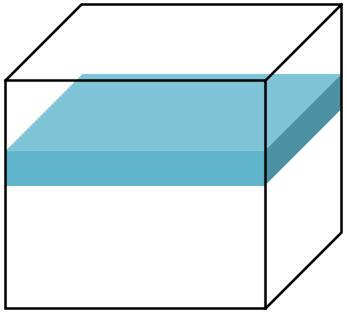


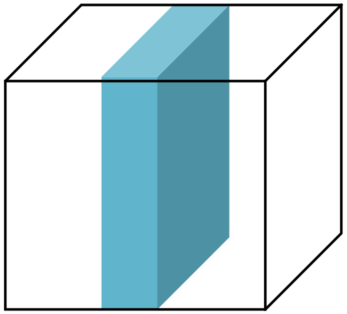
An hyper-cube with K axes, with a level of some hierarchy at each axis. A cell of the cuboid contains the values of metrics for the conditions given by the cell coordinates.

Cube navigation by different users



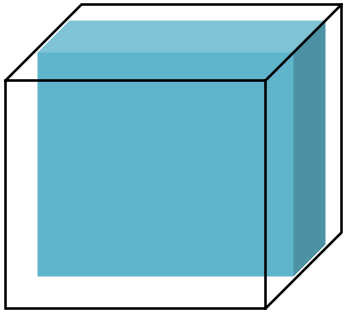
Finance manager look at sales
of a period compared to the previous period
for any product and any market





Product managers look at sales
of some products
in any period and in any market

Branch manager look at sales
of his/her stores
for any product and any period



Cuboids in SQL

Order or pivoting

Aggregate

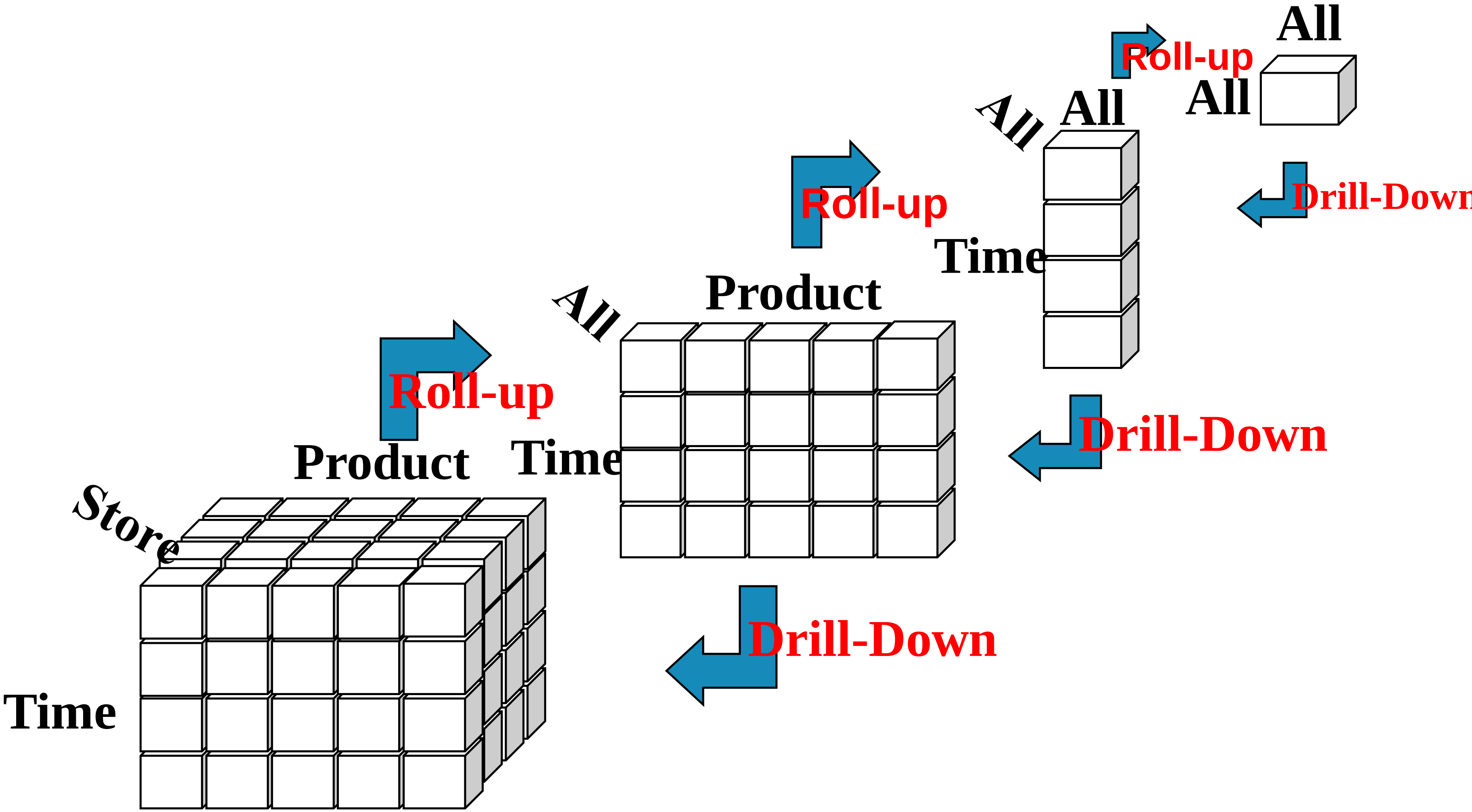
Measure

```
SELECT L.city, I.brand, T.month, SUM(dollars_sold)
 FROM fact AS F, location AS L, time AS T, item AS I
 WHERE F.location_key = L.location_key AND
   F.time_key = T.time_key AND
    F.item_key = I.item_key
GROUP BY L.city, I.brand, T.month
```

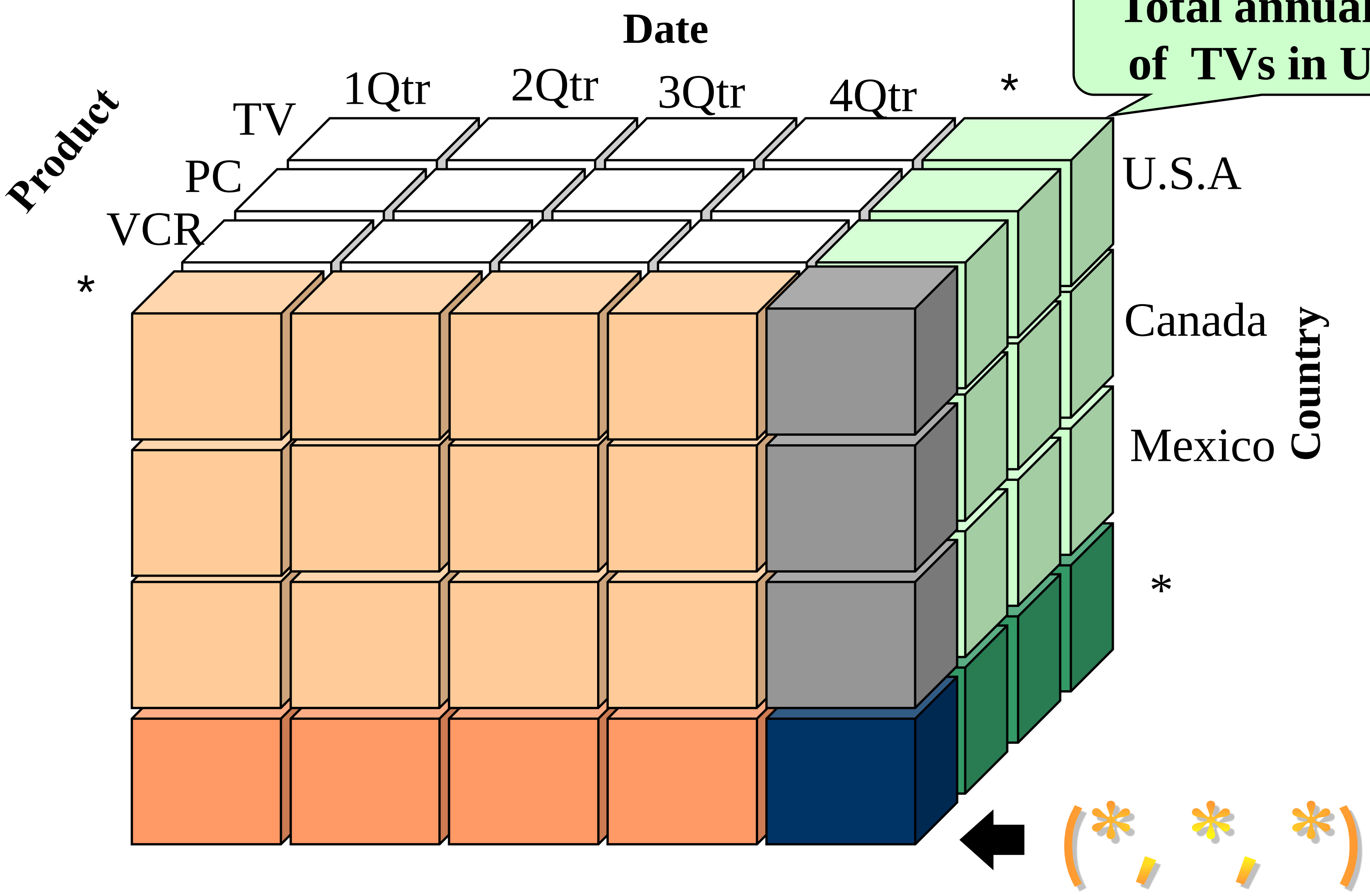
Star-Join

Hierarchy levels

How many cuboids?



Data Cube
(extended cube, hypercube)



Total annual sales of TVs in U.S.A.

Data cube in SQL Server

Order or pivoting

Aggregate

Measure

```
SELECT L.city, I.brand, T.month, SUM(dollars_sold)
 FROM fact AS F, location AS L, time AS T, item AS I
 WHERE F.location_key = L.location_key AND
   F.time_key = T.time_key AND F.item_key =
   I.item_key
GROUP BY CUBE(L.city, I.brand, T.month)
```

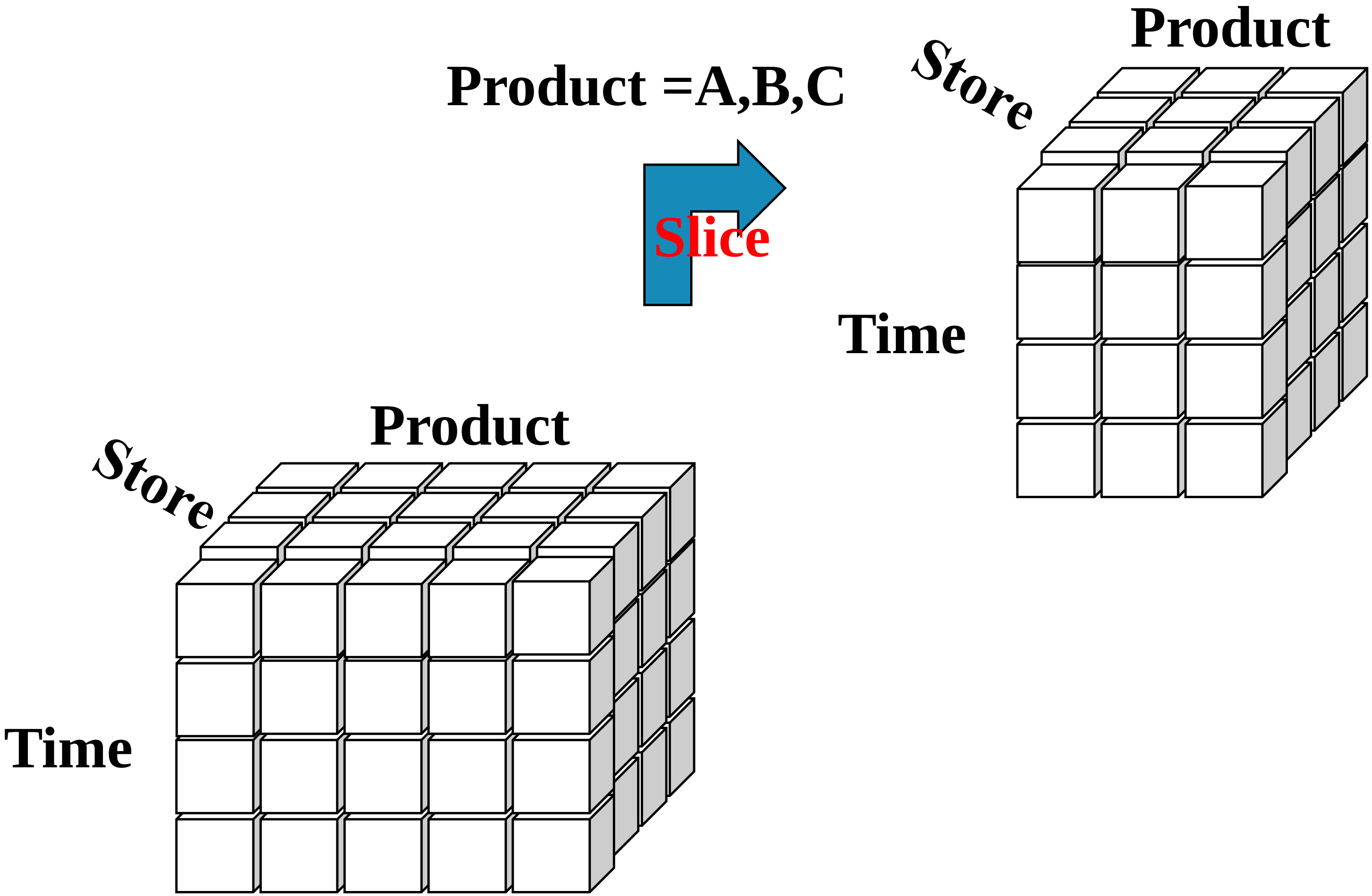
Star-Join

Hierarchy levels

GROUP BY ROLLUP(L.city, I.brand, T.month)

- all initial subsequences of the group-by attributes

Slice and Dice



Product =A,B,C

Slice

Store

Product

Time

Store

Product

Time

Slice in SQL Server

Order or pivoting

Aggregate

Measure

```
SELECT L.city, I.brand, T.month, SUM(dollars_sold)
 FROM fact AS F, location AS L, time AS T, item AS I
 WHERE F.location_key = L.location_key AND
    F.time_key = T.time_key AND
   F.item_key = I.item_key  AND
  T.year = 2016
GROUP BY CUBE(L.city, I.brand, T.month)
```

Star-Join

Slice

Hierarchy levels

Star-join executions in SQL Server

- Star-join optimization
  - automatically detected (vs to be setup in Oracle)
- Bitmap join indexes
  - not available (vs available in Oracle)
- Columnstore indexes (since SQL Server 2012)
  - see docs
    - http://msdn.microsoft.com/en-us/library/gg492088.aspx
  - Example (on a copy of sales_fact)
    - CREATE CLUSTERED COLUMNSTORE INDEX cci_sales ON sales_fact_copy

Materialized views in SQL Server

- Create a (normal) view

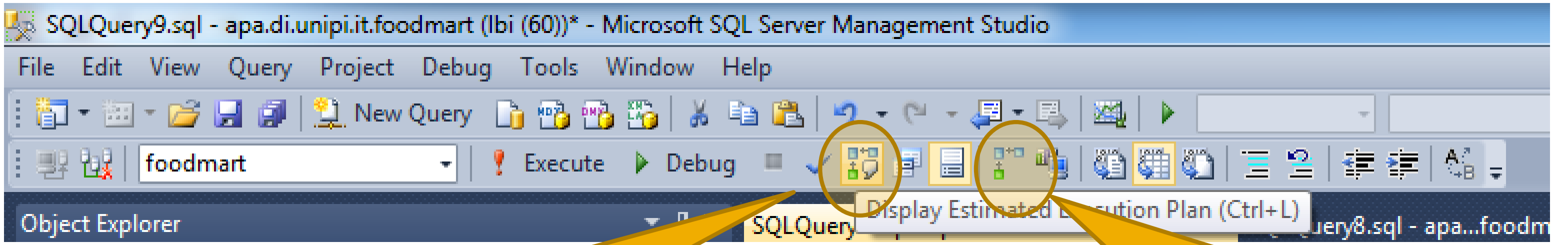
```
CREATE VIEW schema.view_name
 WITH SCHEMABINDING -- binds the reference tables schema
 AS SELECT …, COUNT_BIG(*) as n
 FROM …
 WHERE …
 GROUP BY …
```

- Make an index on it

```
CREATE UNIQUE CLUSTERED INDEX index_name
    ON schema.view_name (attributes);
```

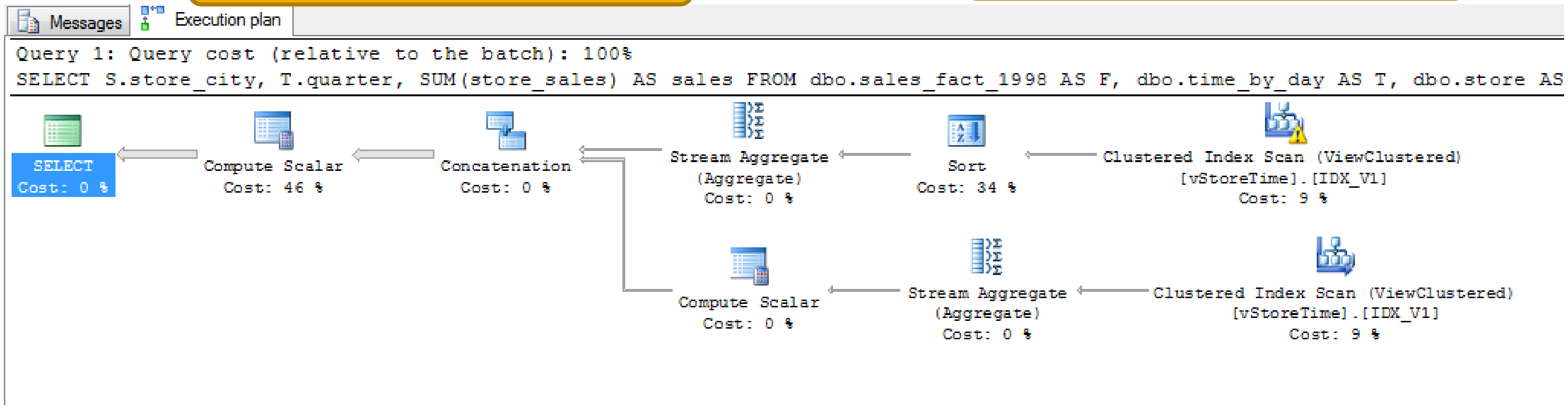
- They are called Indexed Views
  - http://msdn.microsoft.com/en-us/library/ms191432.aspx

Materialized views in SQL Server


SQLQuery9.sql - apa.di.unipi.it.foodmart (lbi (60))* - Microsoft SQL Server Management Studio
File Edit View Query Project Debug Tools Window Help
New Query
foodmart
Execute
Debug
Display Estimated Execution Plan (Ctrl+L)
Object Explorer
SQLQuery
uery8.sql - apa...foodm

Estimated execution plan

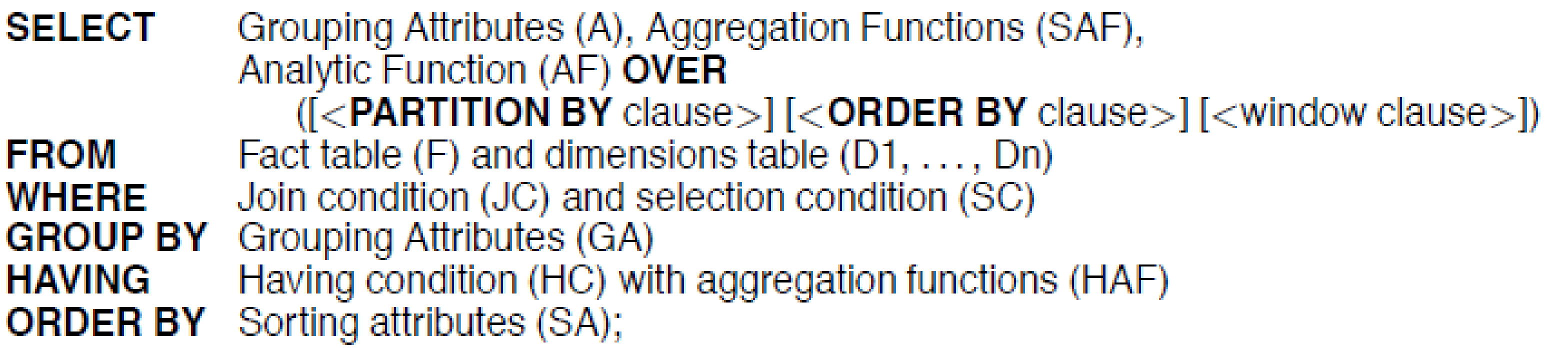
Actual execution plan



```
Query 1: Query cost (relative to the batch): 100%
SELECT S.store_city, T.quarter, SUM(store_sales) AS sales FROM dbo.sales_fact_1998 AS F, dbo.time_by_day AS T, dbo.store AS
```

Analytic SQL

- Aggregate functions
  - MIN, MAX, SUM, COUNT, AVG, VAR, VARP, STDEV, STDEVP
- Ranking functions
  - RANK, DENSE_RANK, NTILE, ROW_NUMBER
- Analytic functions (since SQL Server 2012)
  - CUME_DIST, LEAD, FIRST_VALUE, PERCENTILE_CONT, LAG, PERCENTILE_DISC, LAST_VALUE, PERCENT_RANK



```
SELECT      Grouping Attributes (A), Aggregation Functions (SAF),
            Analytic Function (AF) OVER
               ([<PARTITION BY clause>] [<ORDER BY clause>] [<window clause>])
FROM        Fact table (F) and dimensions table (D1, . . . , Dn)
WHERE       Join condition (JC) and selection condition (SC)
GROUP BY    Grouping Attributes (GA)
HAVING      Having condition (HC) with aggregation functions (HAF)
ORDER BY    Sorting attributes (SA);
```

Exercise

□ For each product family, store country, and quarter, show the percentage of sales over the total of the product family