Database Design and Implementation

CS 645

OLAP

context

- In the late 80s and early 90s, companies began to use their DBMSs for complex, interactive, exploratory analysis of historical data.

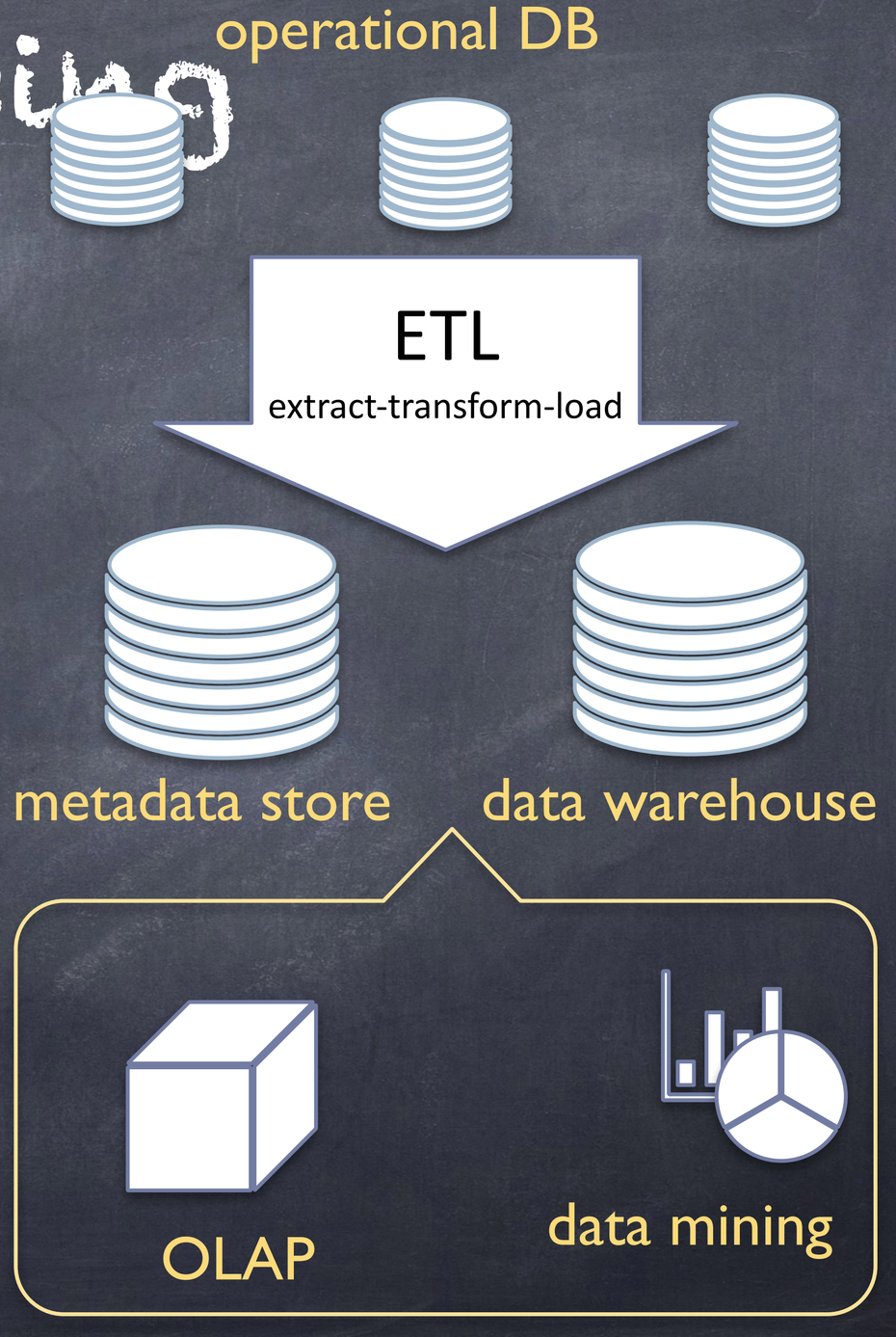
store A store B store C

Operational data:
purchase transactions:store, customers, products, sales, etc

Decision making:
- how much of which products to order for which store?
- when to deliver the products?
- benefits of promotional offers?

data warehousing

- Data:
  - Integrated data spanning long time periods, often augmented with summary information.
  - Large volumes: several terabytes to petabytes common.
- Queries:
  - Interactive response time expected for complex queries
  - Ad-hoc updates uncommon



data warehouse

- An organization-wide repository for decision-making
  - An integrated enterprise warehouse collects info about all subjects, e.g. customers, products, sales, assets, personnel.
  - The data is used to assist in decision making
    - e.g., how much of which products to order for which stores, when to deliver the products, the benefits of various promotional offers, etc.
  - Analytics is called On-Line Analytic Processing (OLAP).
  - OLAP tasks slowed down the normal operation of the company, called On-Line Transaction Processing (OLTP), leading to separation of DWs from operational DBs.

OLTP vs OLAP databases

OLTP/operational/production	OLAP/data warehouse/DSS
operate the business	diagnose the business
short queries, little data	large queries, large data
current data	current and historical data
queries change data e.g., order entry	queries are mostly read-only e.g., OLAP, statistics
often distributed	often integrated and centralized

multi-dimensional data

- To support OLAP, warehouse data is often structured multidimensionally, as measures and dimensions.
  - Measure: numeric attribute, e.g. sales amount
  - Dimension: attribute categorizing the measure, e.g. product, store, date of sale.
- Star schema:
  - The central fact table contains a foreign key for each dimension, plus an attribute for each measure.
  - There will also be a dimension table for each dimension.

examples

- Purchase (ProductID, StoreID, DateID, Amt)
  Product(ID, SKU, size, brand)
  Store(ID, Address, Sales District, Region, Manager)
  Date (ID, Week, Month, Holiday, Promotion)

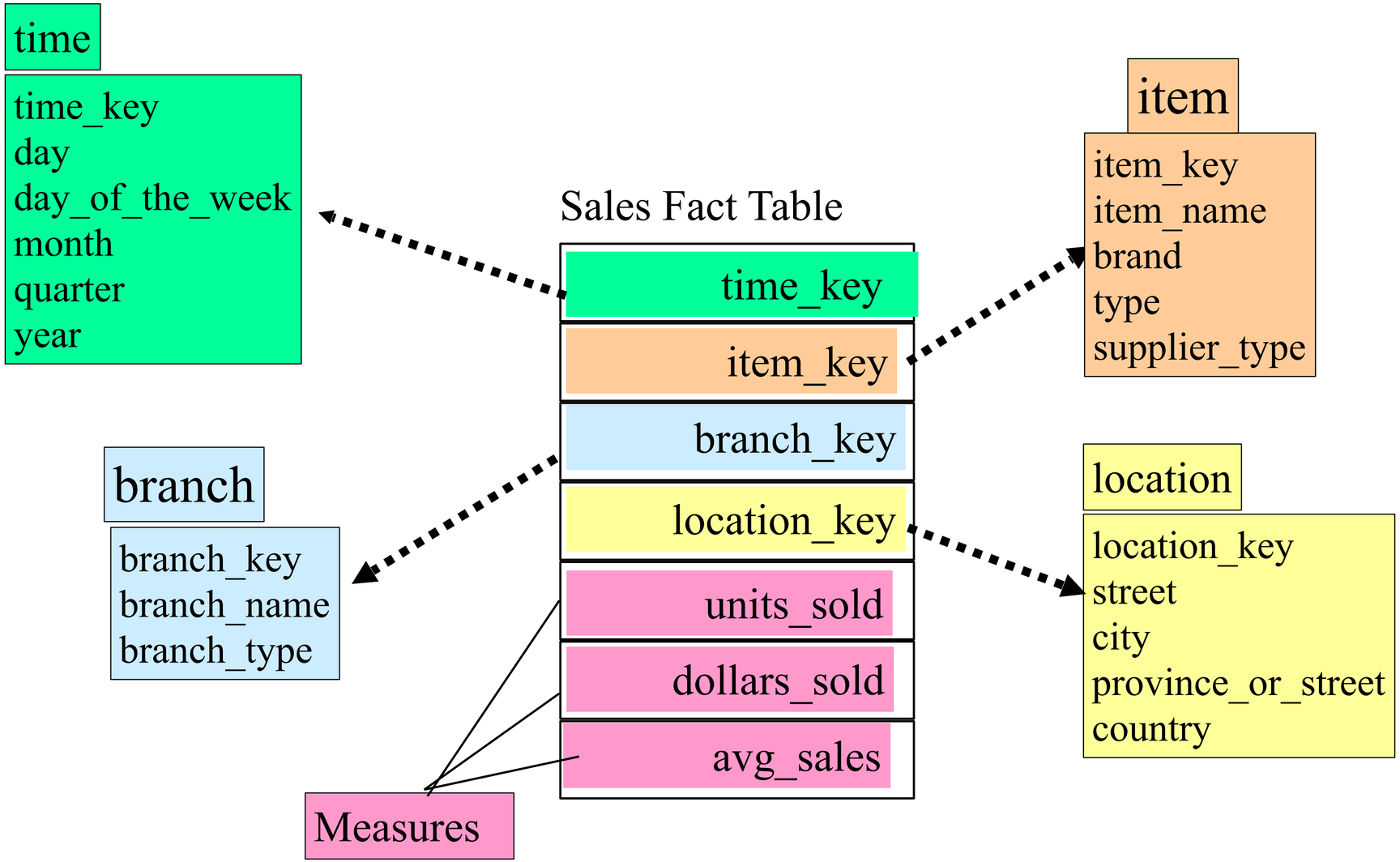
- Claims (ProvID, MembID, ProcedureID, DateID, Cost)
  Providers(ID, Practice, Address, ZIP, City, State)
  Members(ID, Contract, Name, Address)
  Procedure (ID, Name, Type)

- Telecomm (CustID, SalesRepID, ServiceID, DateID)
  SalesRep(ID, Address, Sales District, Region, Manager)
  Service(ID, Name, Category)
  ...

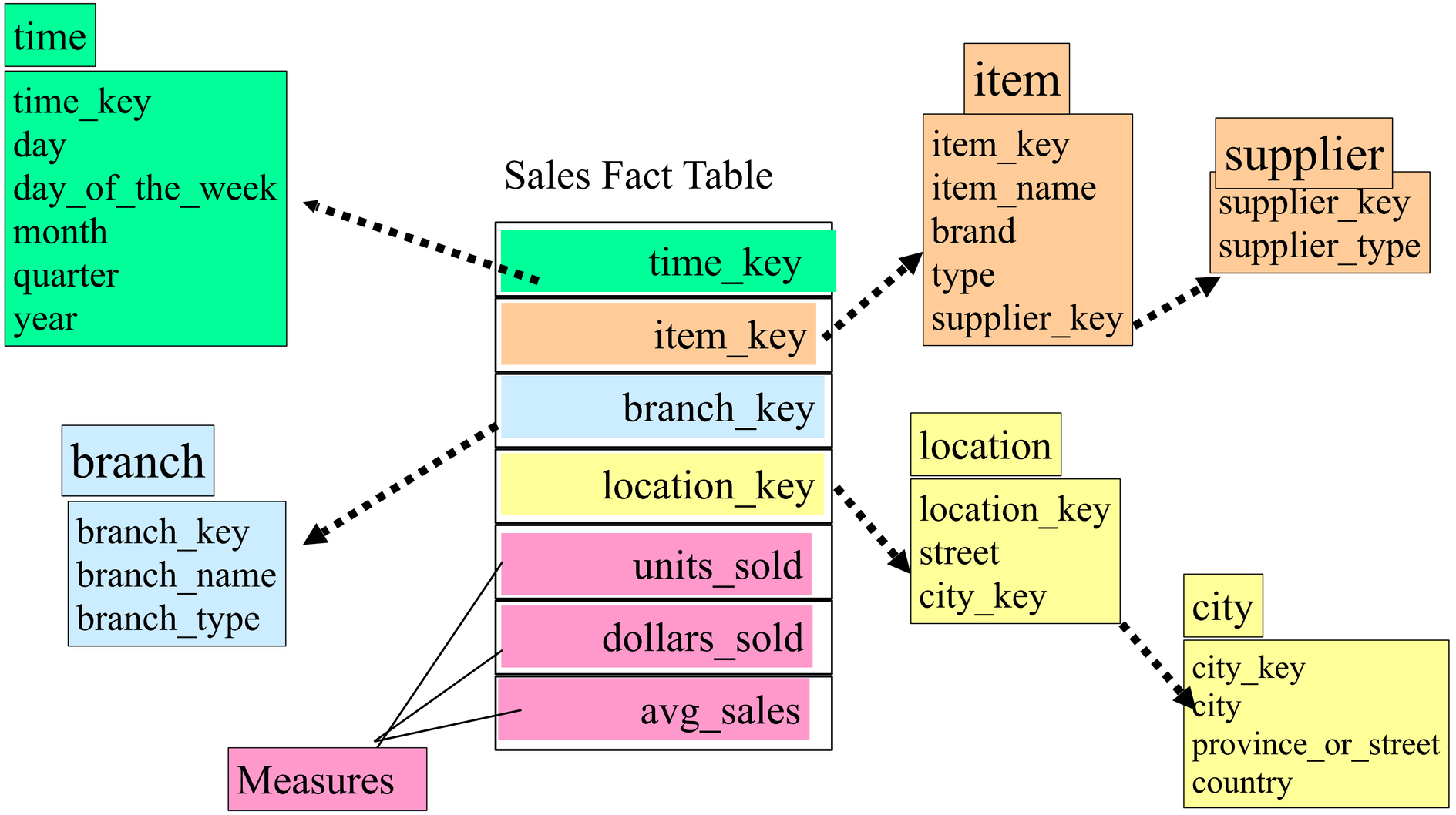
Conceptual Modeling of Data Warehouses

- Modeling data warehouses: dimensions & measures
  - Star schema: A fact table in the middle connected to a set of dimension tables
  - Snowflake schema: A refinement of star schema where some dimensional hierarchy is normalized into a set of smaller dimension tables, forming a shape similar to snowflake
  - Fact constellations: Multiple fact tables share dimension tables, viewed as a collection of stars, therefore called galaxy schema or fact constellation

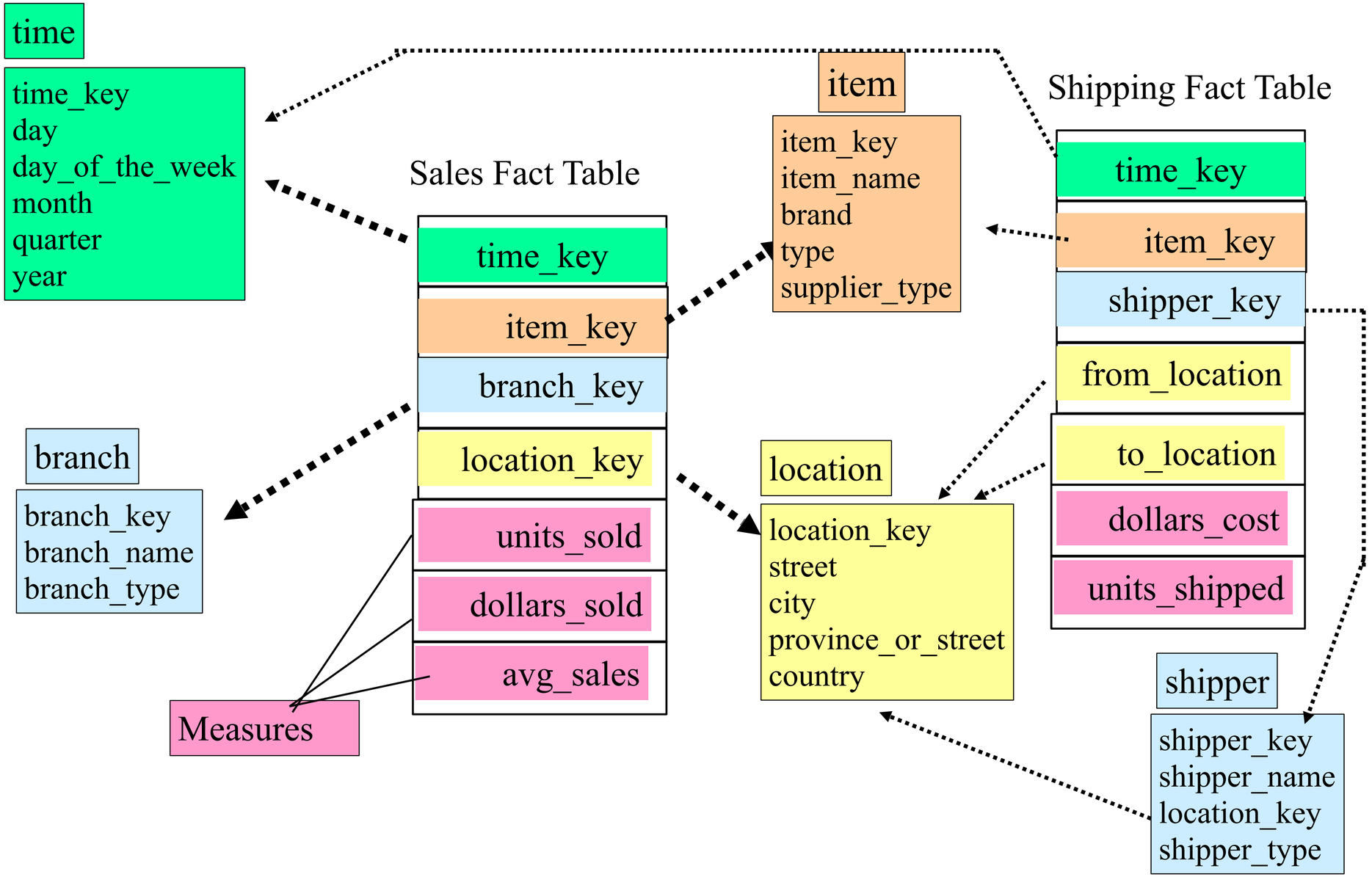
Example of Star Schema



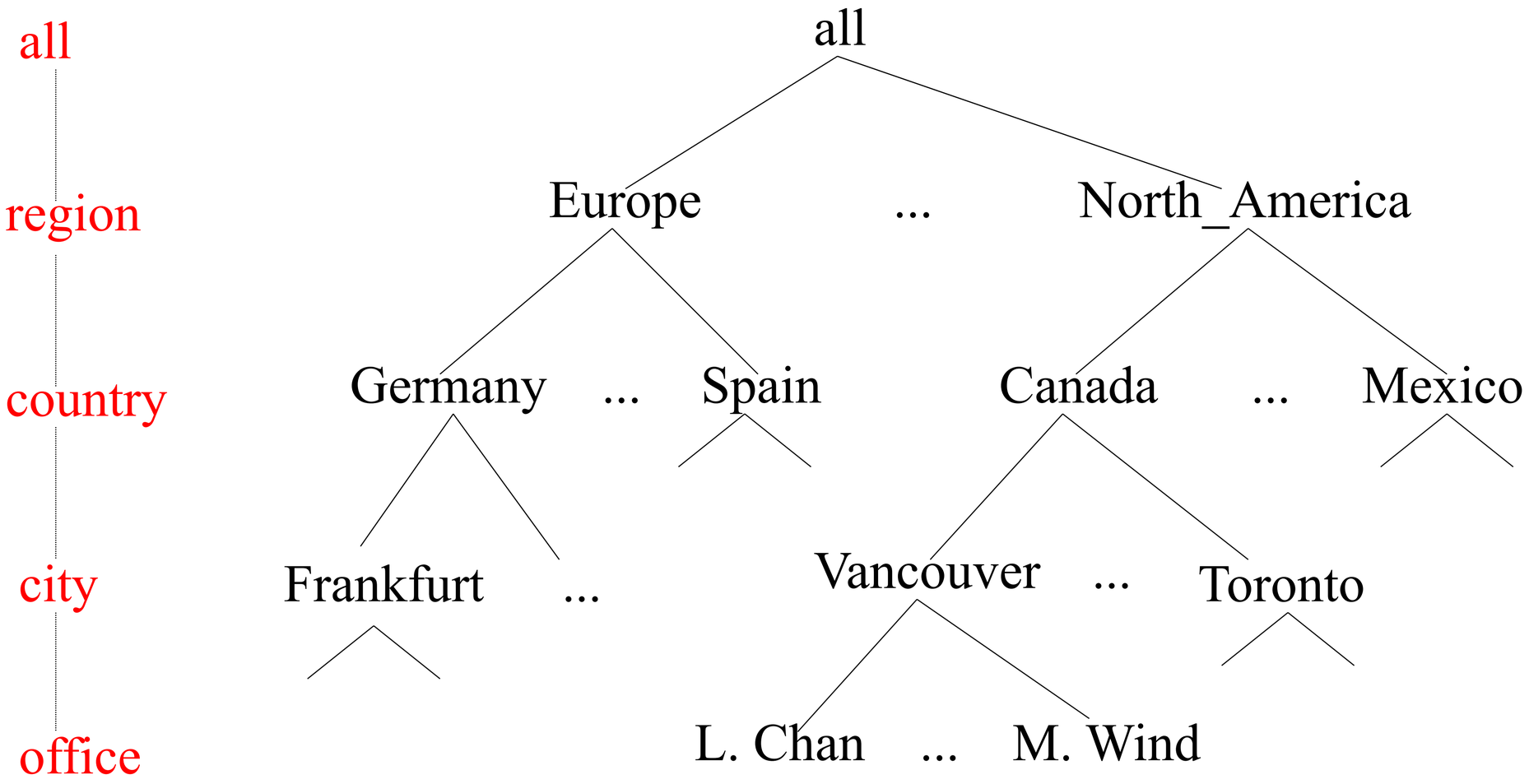
Example of Snowflake Schema



Example of Fact Constellation



A Concept Hierarchy: Dimension (location)



For each dimension, some of the attributes may be organized in a hierarchy

star / snowflake schemas

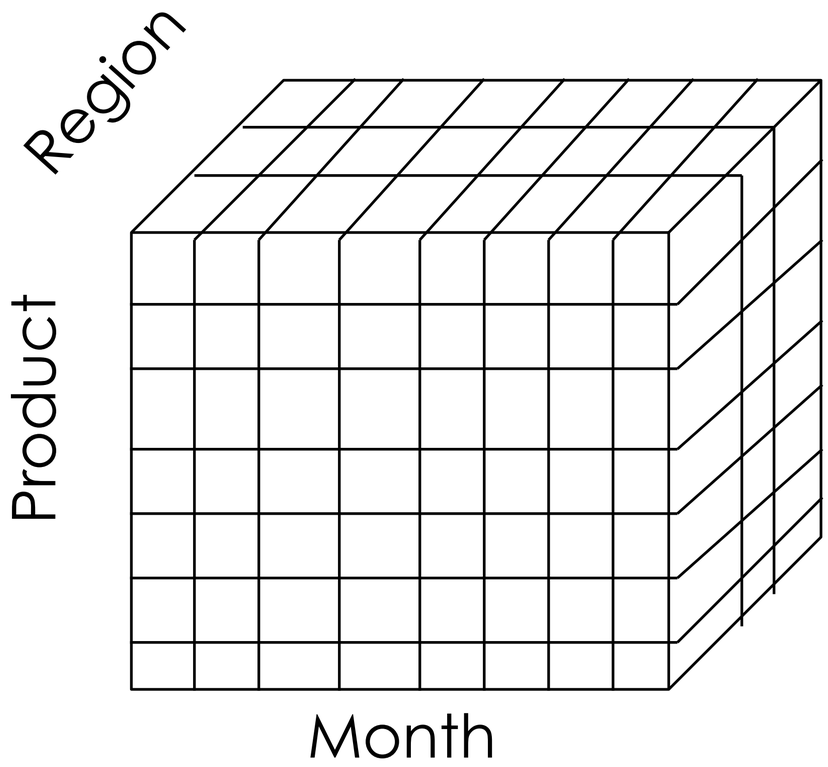
- Why normalize?
  - Save space
  - Remove store redundancy and anomalies
- Why denormalize?
  - Performance benefits, e.g., avoiding joins
- Which is more important in Data Warehouses?
- If fully normalized, it is a snowflake schema

MOLAP vs ROLAP

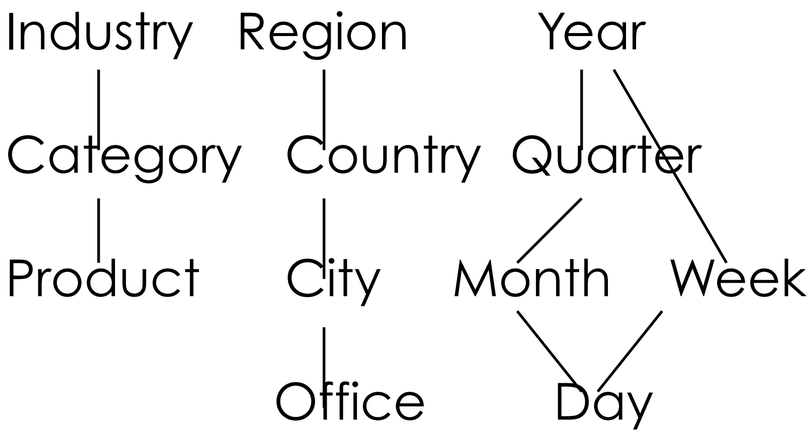
- MOLAP systems: store multidimensional data in a disk-resident, persistent array.
- ROLAP systems: store data as a relation
  - The main relation, which relates dimensions to a measure (e.g., sales), is the fact table.
  - Each dimension has additional attributes in the dimension table.
  - Fact tables are much larger than dimension tables.

Multidimensional Data

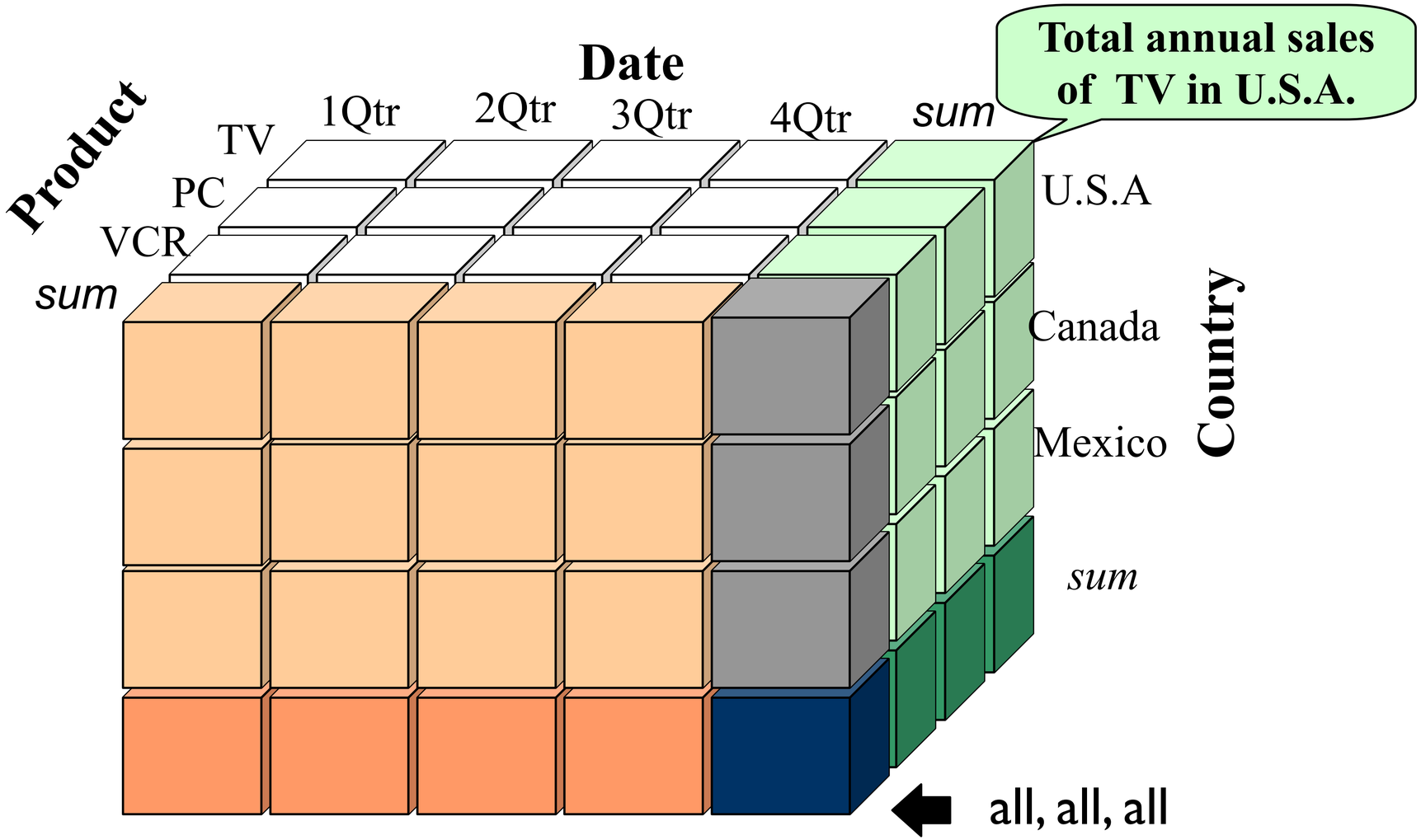
- Numeric measures, which depend on a set of dimensions
- Sales volume as a function of product, month, and region



Dimensions: Product, Location, Time
Hierarchical summarization paths



sample data cube



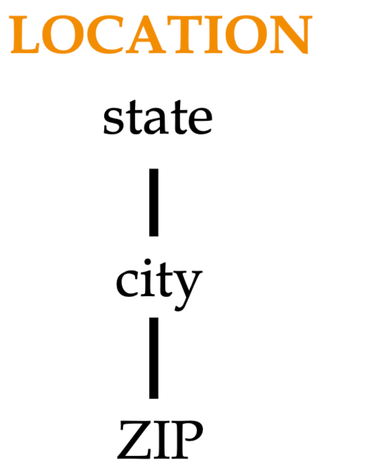
OLAP queries

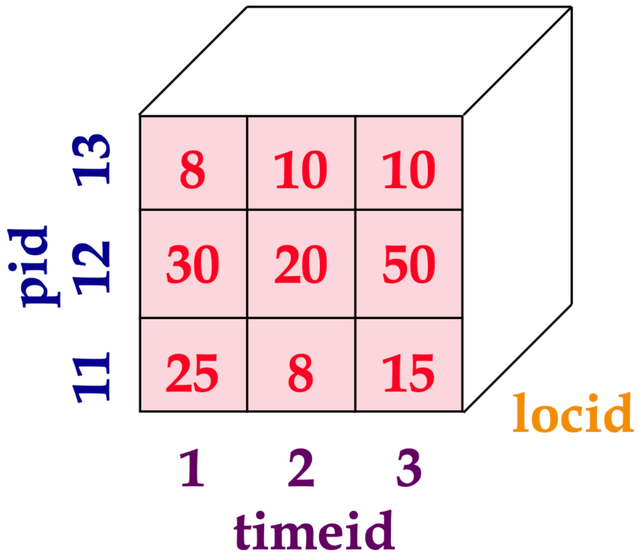
- Influenced by SQL and spreadsheets
- Common operation: aggregate a measure over one or more dimensions.
  - find total sales
  - find total sales for each city
  - find top five products ranked by total sales

OLAP queries

- Roll-up: aggregates at increasingly coarser levels of a dimension hierarchy.

Given total sales by zip, we can roll-up to get total sales by city, and then by state





pid	timed	locid	amt
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

OLAP queries

Drill-down: the reverse of roll-up

Given total sales by state, we can drill-down to get total sales by city

Can also drill-down on a different dimension to get total sales for each state by product

LOCATION

state

|

city

|

ZIP

PRODUCT

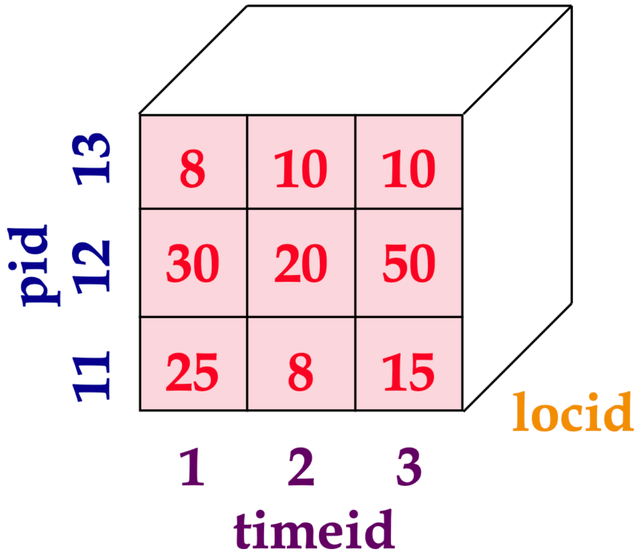
category

|

pname

|

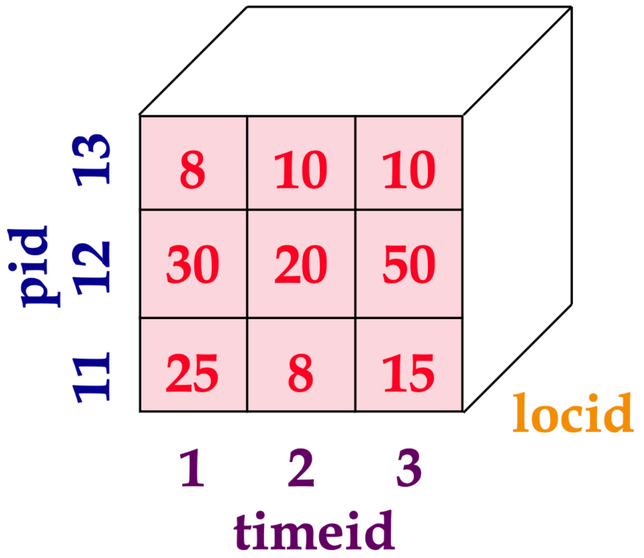
PID



pid	timed	locid	amt
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

OLAP queries

- Slicing and dicing: equality and range selection in one or more dimensions



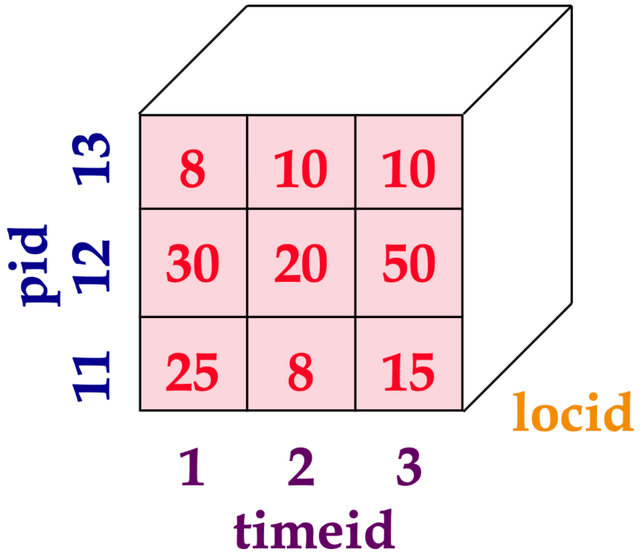
pid	timed	locid	amt
11	1	1	25
11	2	1	8
11	3	1	15
12	1	1	30
12	2	1	20
12	3	1	50
13	1	1	8
13	2	1	10
13	3	1	10
11	1	2	35

OLAP queries

- Pivoting: aggregates on selected dimensions

Pivoting on state and year yields the cross-tabulation below

	OR	CA	Total
2007	63	81	144
2008	38	107	145
2009	75	35	110
Total	176	223	339



pid	timed	locid	amt
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

comparison with SQL

- Can compute cross-tabulation with SQL queries

pid	timed	locid	amt
11	1	1	25
11	2	1	8
11	3	1	15
12	1	1	30

	OR	CA	Total
2007	63	81	144
2008	38	107	145
2009	75	35	110
Total	176	223	339

```
SELECT T.year, L.state, SUM(S.amt)
FROM   Sales S, Times T, Locations L
WHERE  S.timeid=T.timeid AND S.locid=L.locid
GROUP BY T.year, L.state
```

```
SELECT T.year, SUM(S.amt)
FROM   Sales S, Times T
WHERE  S.timeid=T.timeid
GROUP BY T.year
```

```
SELECT L.state,SUM(S.amt)
FROM   Sales S, Location L
WHERE  S.locid=L.locid
GROUP BY L.state
```

the CUBE operator

- With k dimensions, we have 2^k possible SQL GROUP BY queries that can be generated through pivoting on a subset of dimensions.
- GROUP BY CUBE(pid, locid, timeid)

- Equivalent to rolling up Sales on all eight subsets of the set {pid, locid, timeid}

```
SELECT grouping-list, SUM(S.amt)
FROM   Sales S
GROUP BY CUBE(grouping-list)
ORDER BY grouping-list
```

the CUBE operator

- GROUP BY CUBE(pid, locid, timeid)

```
SELECT grouping-list, SUM(S.amt)
FROM   Sales S
GROUP BY CUBE(grouping-list)
ORDER BY grouping-list
```

- Equivalent to rolling up Sales on all eight subsets of the set {pid, locid, timeid}
- Each roll-up amounts to a query of the form:

```
SELECT SUM(S.amt)
FROM   Sales S
GROUP BY grouping-list
```

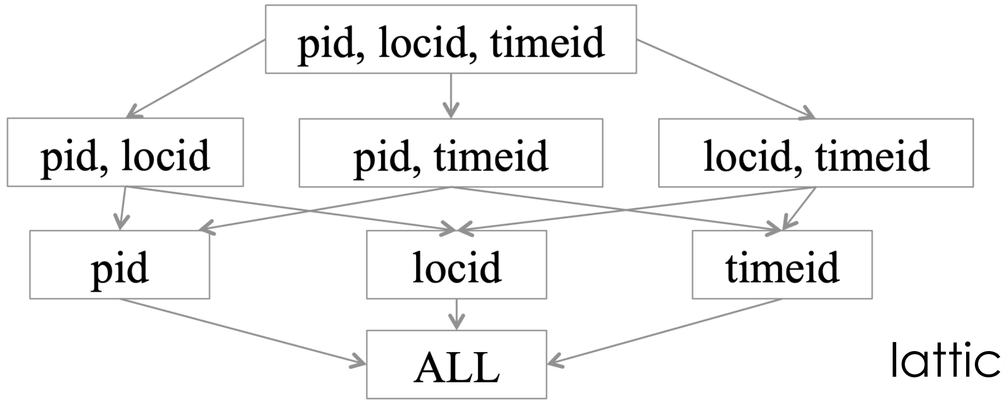
the CUBE operator

- GROUP BY CUBE(pid, locid, timeid)

```
SELECT grouping-list, SUM(S.amt)
FROM   Sales S
GROUP BY CUBE(grouping-list)
ORDER BY grouping-list
```

- Equivalent to rolling up Sales on all eight subsets of the set {pid, locid, timeid}
- Each roll-up amounts to a query of the form:

```
SELECT SUM(S.amt)
FROM   Sales S
GROUP BY grouping-list
```



lattice of group-by operations

views and decision support

- In large databases, precomputation is necessary for fast response times
- Example: Precompute daily sums for the cube.
- These precomputed queries are called Materialized Views