TÉCNICO LISBOA

Data Analysis and Integration

OLAP operations

Data Warehousing

Data warehousing

- What is a data warehouse?

A **Data Warehouse** is a category of software technology that enables analysts, managers and executives to gain insight into data through fast, consistent, interactive access to a wide variety of possible views of information which has been transformed from raw data to reflect the real dimensionality of the enterprise as understood by the user

Bill Inmon, 1992

Data warehouse is the consolidated, historical, and summarised data of an organisation to support business decisions

A data warehouse is a database that stores historical data in a convenient schema for multidimensional analysis using OLAP operations

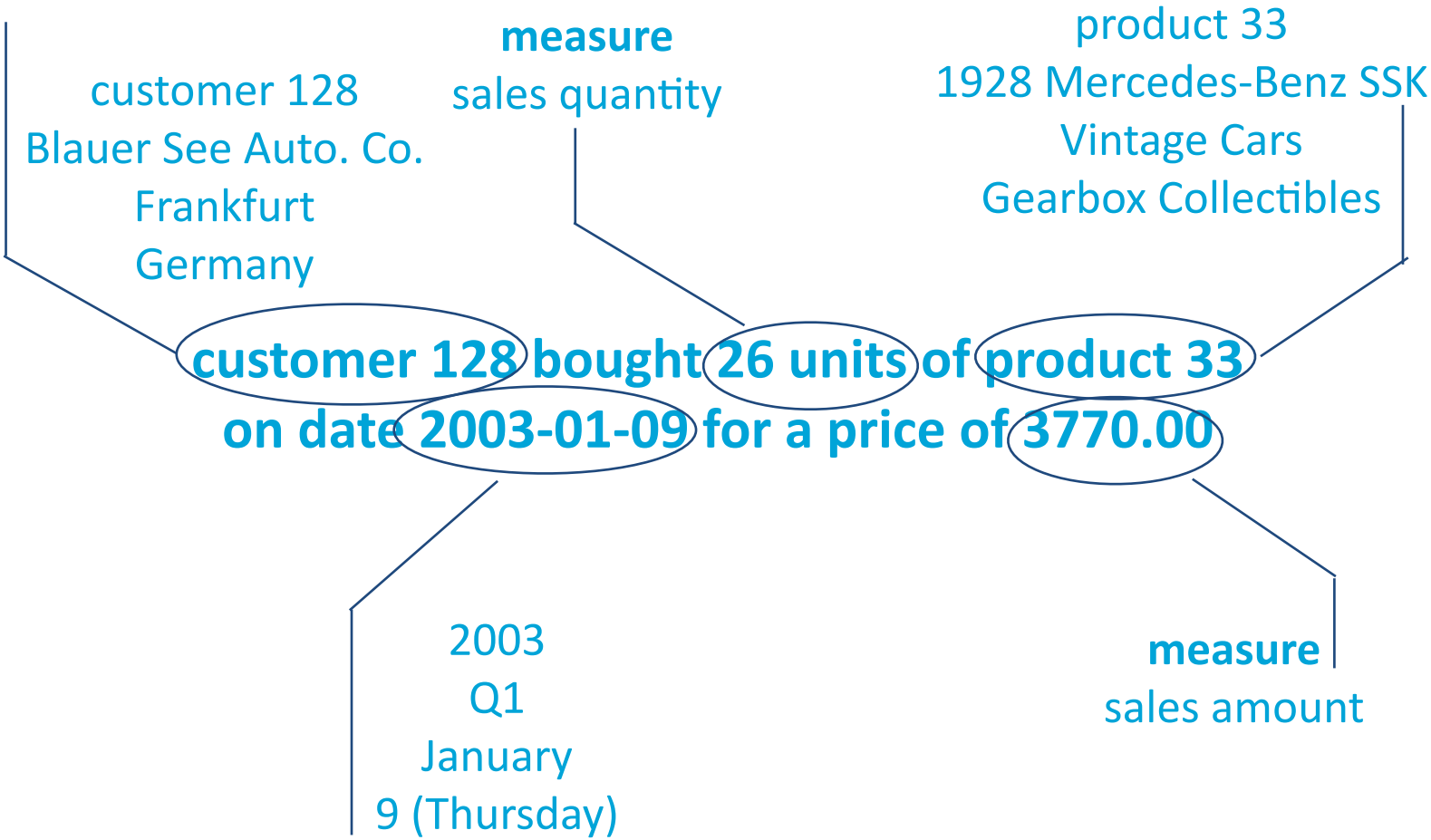
Data warehousing

- A data warehouse stores **facts**

Customer 128 bought 26 units of product 33 on date 2003-01-09 for a price of $3770.00

Data warehousing

- Facts have associated **measures**



Data warehousing

- **Facts** define the possible analysis dimensions
  - customer c bought product p (2D)
  - customer c bought product p on date d (3D)
  - customer c bought product p on date d in store s (4D)
  - customer c bought product p on date d in store s using payment method m (5D)
    - m may be 'cash', 'credit card', etc.
  - customer c bought product p on date d in store s using payment method m through sales representative e (6D)
    - e is an employee number
  - etc.

Data warehousing

- **Measures** define the quantity being analyzed
  - sales
    - quantity sold
    - sales amount
  - production
    - quantity produced
    - production cost
  - logistics
    - distance traveled
    - transported weight
  - etc.

Data warehousing

- **Facts** can be **grouped** by dimensions
  - examples
    - by customer (1D)
    - by product (1D)
    - by time (1D)
    - by customer and product (2D)
    - by customer and time (2D)
    - by product and time (2D)
    - by customer, product and time (3D)

Data warehousing

- As **facts** are **grouped**, their **measures** are **aggregated**
  - examples
    - sales amount by customer (1D)
    - sales amount by product (1D)
    - sales amount by time (1D)
    - sales amount by customer and product (2D)
    - sales amount by customer and time (2D)
    - sales amount by product and time (2D)
    - sales amount by customer, product and time (3D)

Data warehousing

- **Measures** are **aggregated** at a certain **level** of detail
  - examples
    - sales amount by customer country
    - sales amount by customer city
    - sales amount by customer country and product line
    - sales amount by customer city and product vendor
    - sales amount by customer country, product line and year
    - sales amount by customer city, product vendor and month

Data warehousing

- **Levels** are organized into **hierarchies**
  - examples
    - country, state, city
    - year, month, day
- A **dimension** can have one or more **hierarchies**
  - example: time dimension
    - year, month, day (for calendar year, from Jan to Dec)
    - year, semester, period (for school year, from Sep to Aug)

Dimension levels

- Each **dimension** can have different **levels**
  - customer (city; country)
  - product (productline; productvendor)
  - time (year; quarter; month)

OLAP Cubes & Information Visualisation

2D Matrix Representation

Consider a collection of measurements dependent on dimensions:

- **Measurements:** Sales
- **Dimensions**: Supplier Key (SK), Product Key (PK), Location (LOC), e Date (DATE)

SK = Supplier Key
PK = Product Key

SK	PK	QTY
S1	P1	300
S1	P2	200
S2	P1	300
S2	P2	400
S2	P2	200
S4	P2	200

Table

Aggregation Function

SUM

SK	P1	P2
S1	300	200
S2	300	600
S4	NULL	200

PK

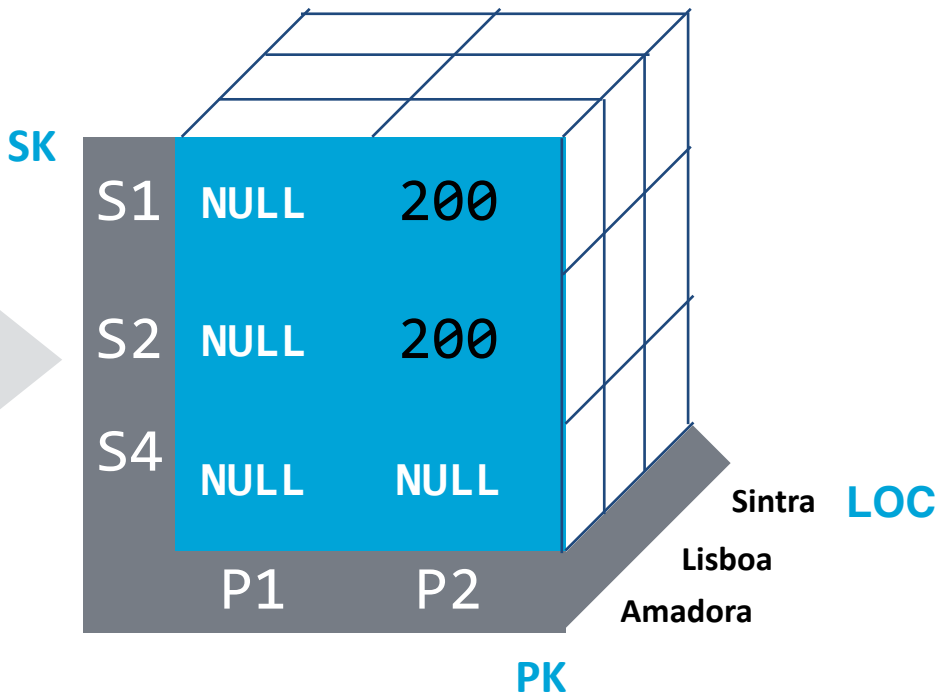
Analysis Matrix

3D Cube Representation

SK	PK	LOC	QTY
S1	P1	Lisboa	300
S1	P2	Amadora	200
S2	P1	Lisboa	300
S2	P2	Lisboa	400
S2	P2	Amadora	200
S4	P2	Sintra	200

Table

SUM



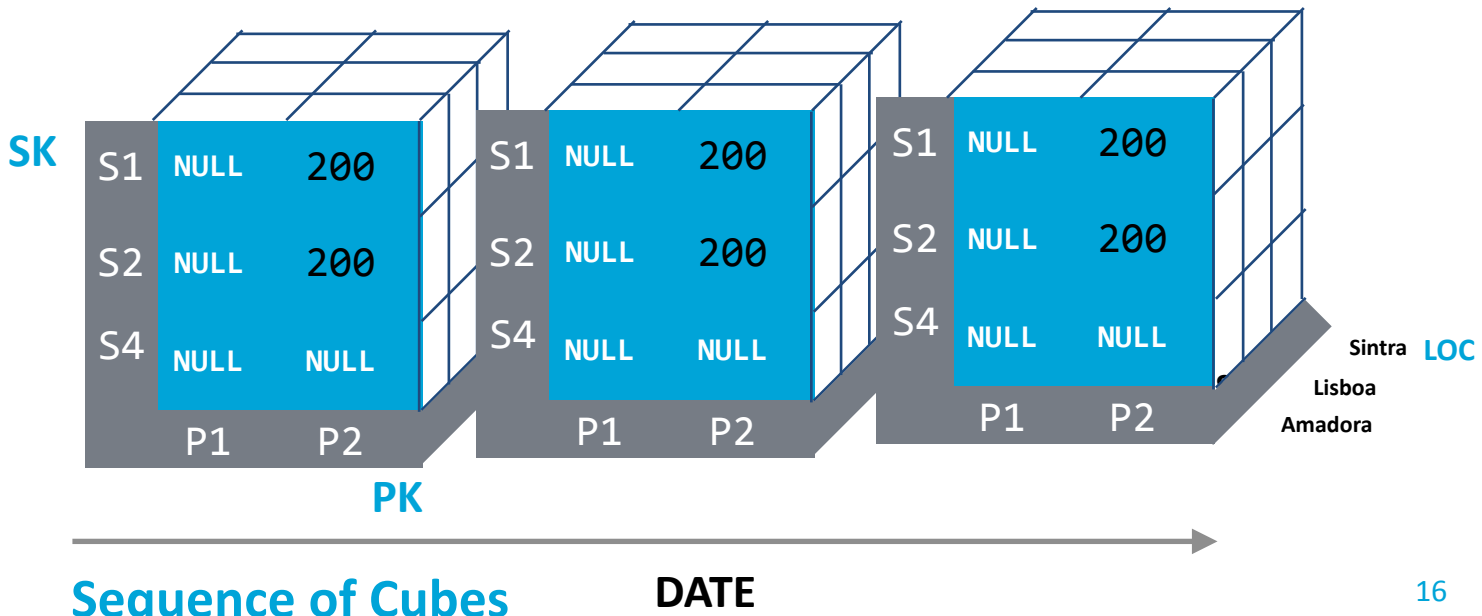
Analysis Cube

nD Hypercube Representation

SK	PK	LOC	DATE	QTY
S1	P1	Lisboa	15 Jan 2015	300
S1	P2	Amadora	15 Fev 2015	200
S2	P1	Lisboa	15 Mar 2015	300
S2	P2	Lisboa	15 Jan 2016	400
S2	P2	Amadora	15 Fev 2016	200
S4	P2	Sintra	15 Mar 2015	200

Table

SUM

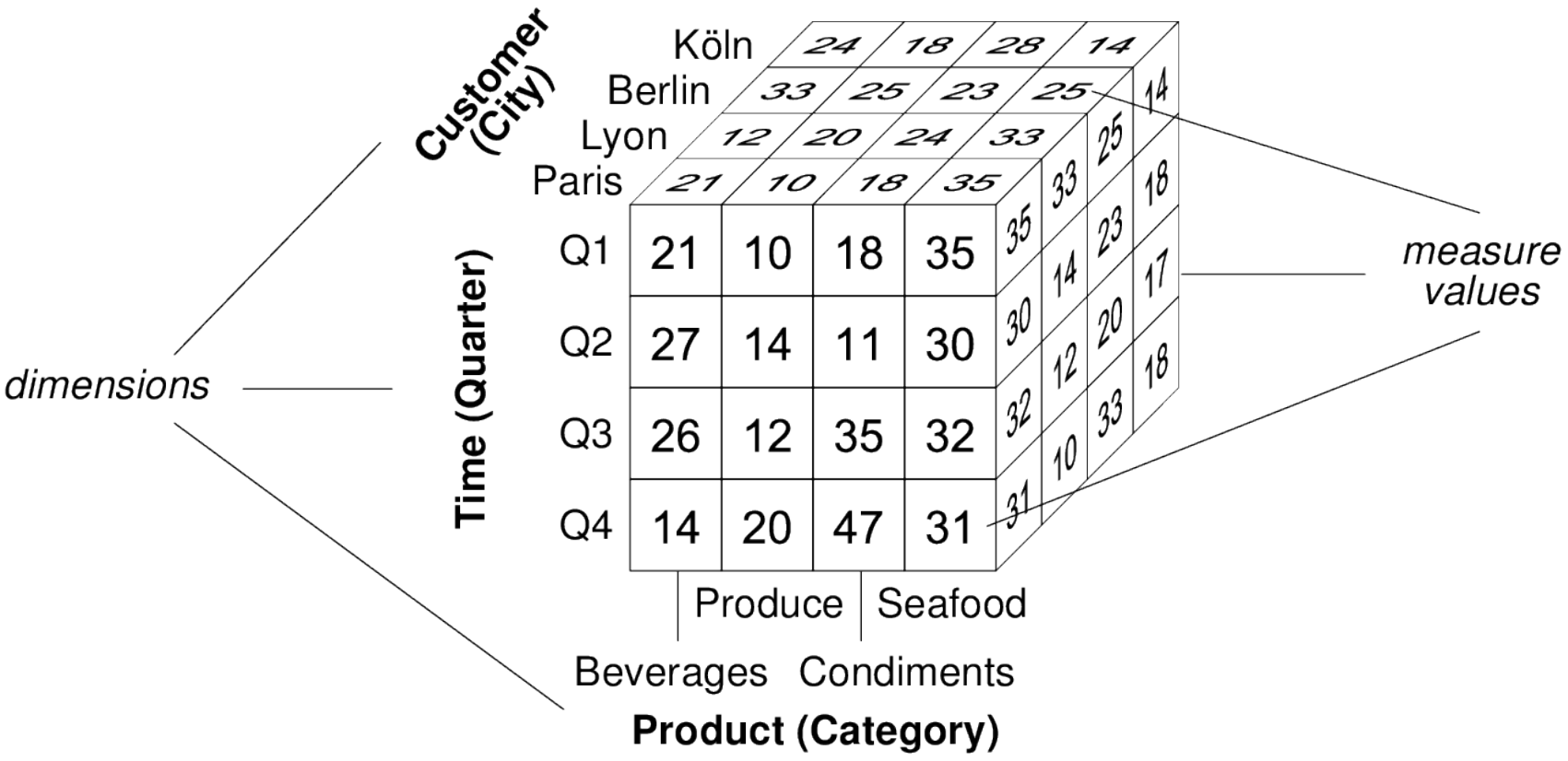


Sequence of Cubes

High Level OLAP Operations

Multidimensional model

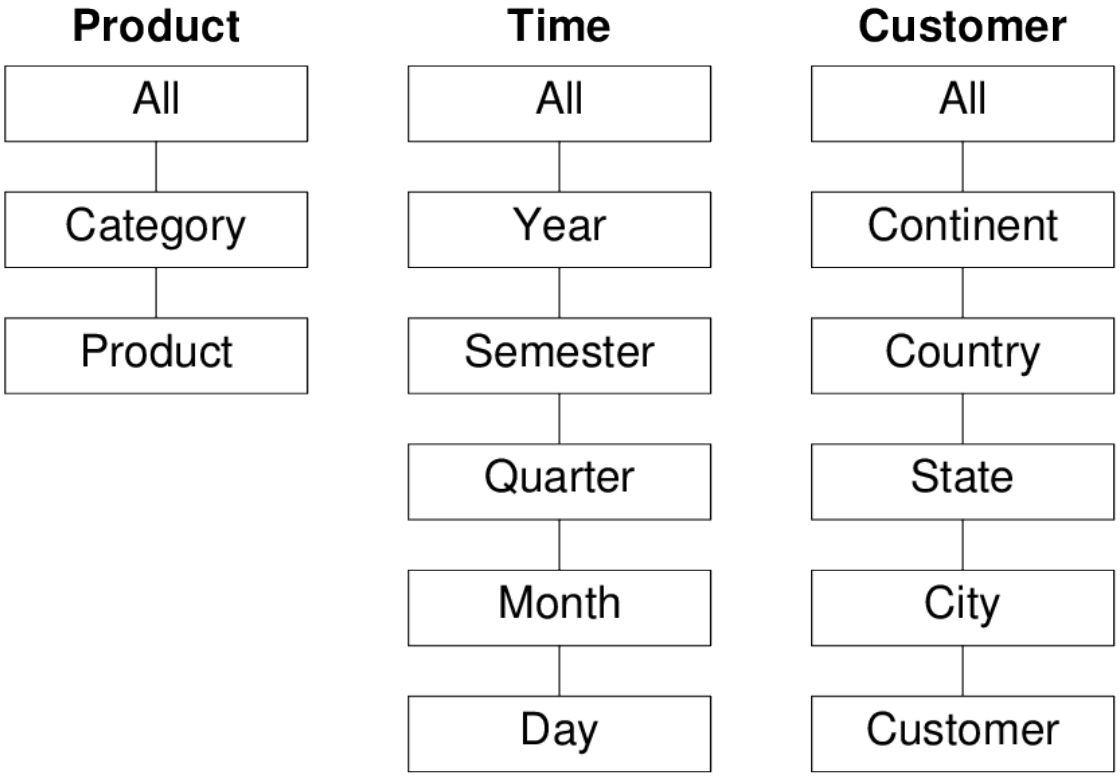
- Data can be viewed as a **cube**



A. Vaisman, E. Zimányi, *Data Warehouse Systems: Design and Implementation*, Springer, 2014

Hierarchies

- Each **dimension** has multiple **levels** (**hierarchy**)

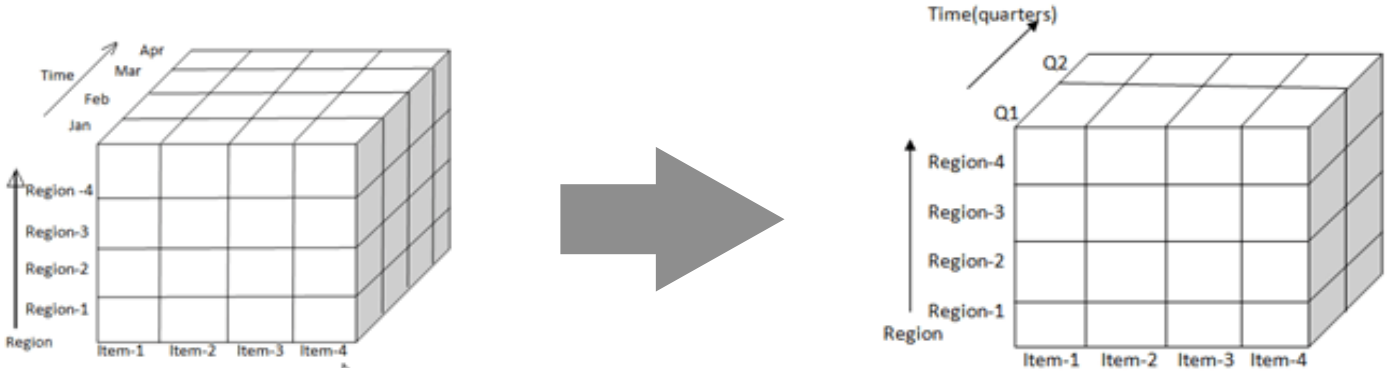


OLAP Operations

- High-level operations on Data Cubes that interactively formulate queries on multi-dimensional model data without requiring knowledge of SQL
- Influenced by SQL and by the Spreadsheet paradigms
- **Aggregating** ou **Detailing** a measurement according to one or more dimensions
  - Obtaining the sales total…
  - Obtaining the sales total for each city and for each region…
  - Obtaining the 5 top products in sales volume…

Roll-up

- Increase the level of aggregation
- Aggregates the fact values one further level (on one or more of the dimensions.)
- Equivalent to doing GROUP BY dimension by using attribute hierarchy

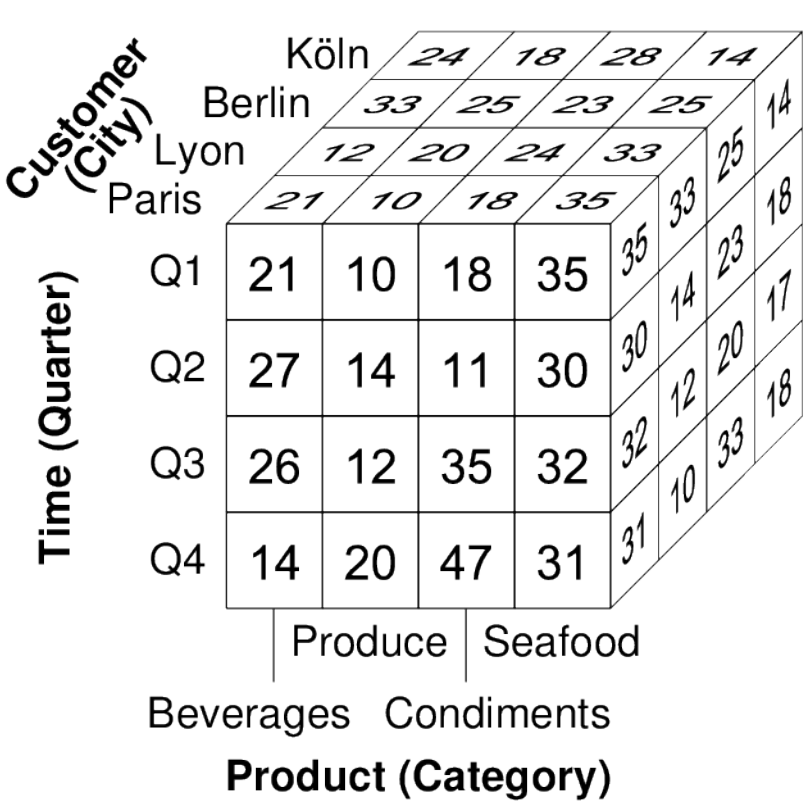


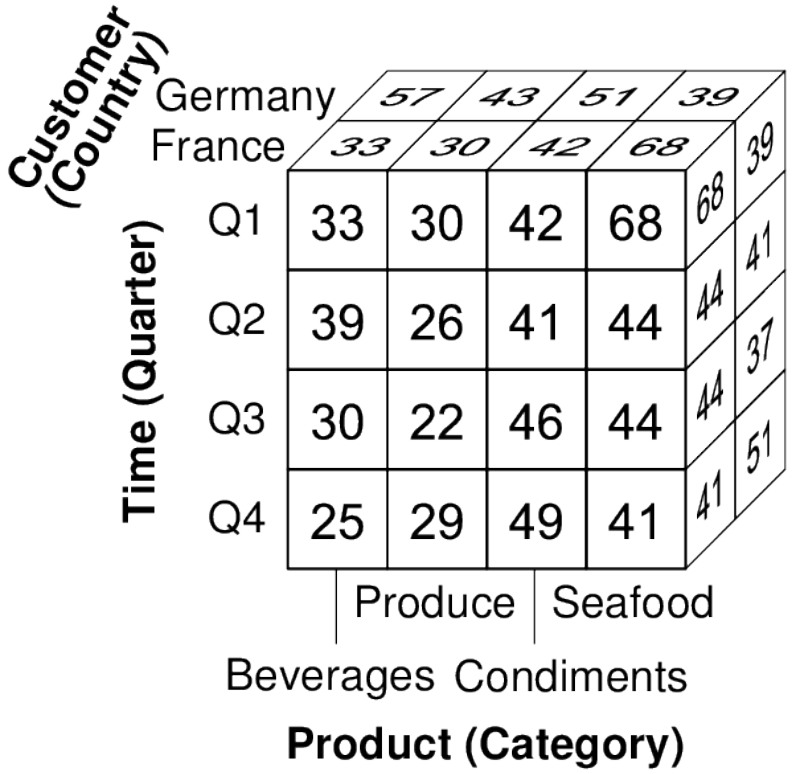
Months to Quarters

Credit: Figure from http://www.teradatapoint.com/data-warehousing-

OLAP operations

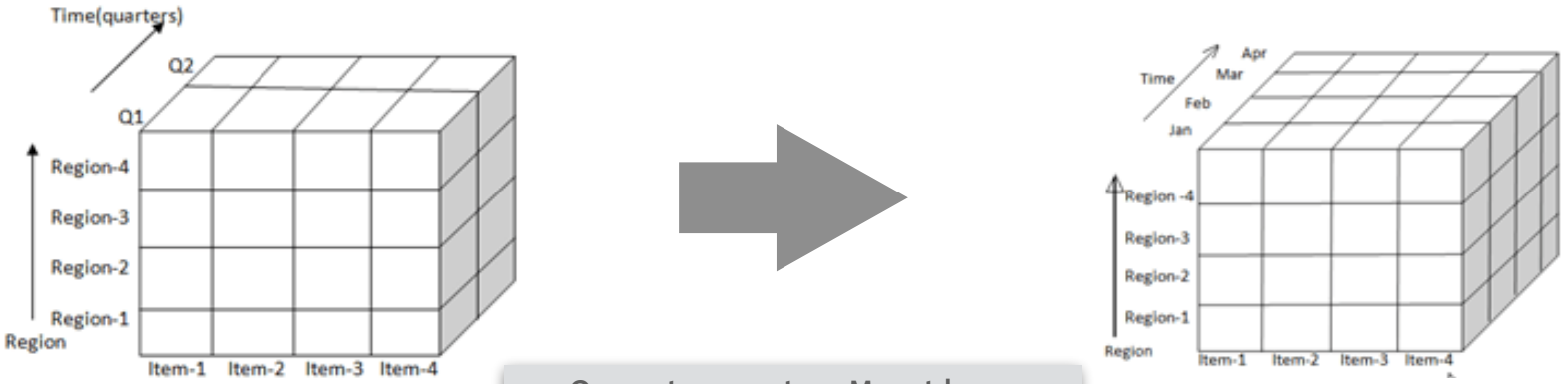
- **Roll-up** to country





Drill-down

- Decreases the level of aggregation, i.e., increases the detail along one or more dimension hierarchies
- Aggregates data at a lower level of granularity dimension hierarchy, thereby viewing data in a more specialised level within a dimension.
- Opposite of Roll-up

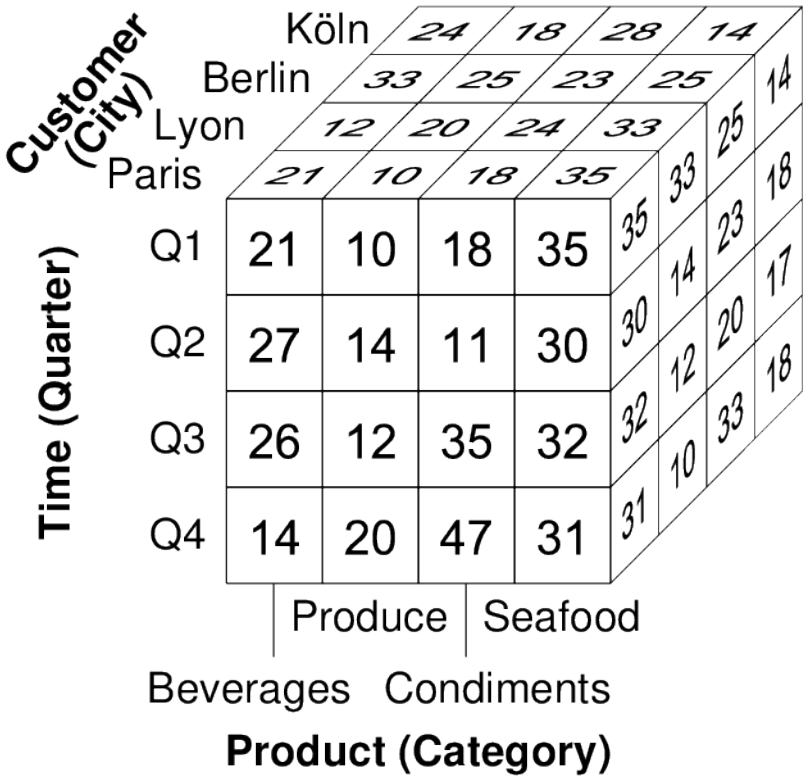


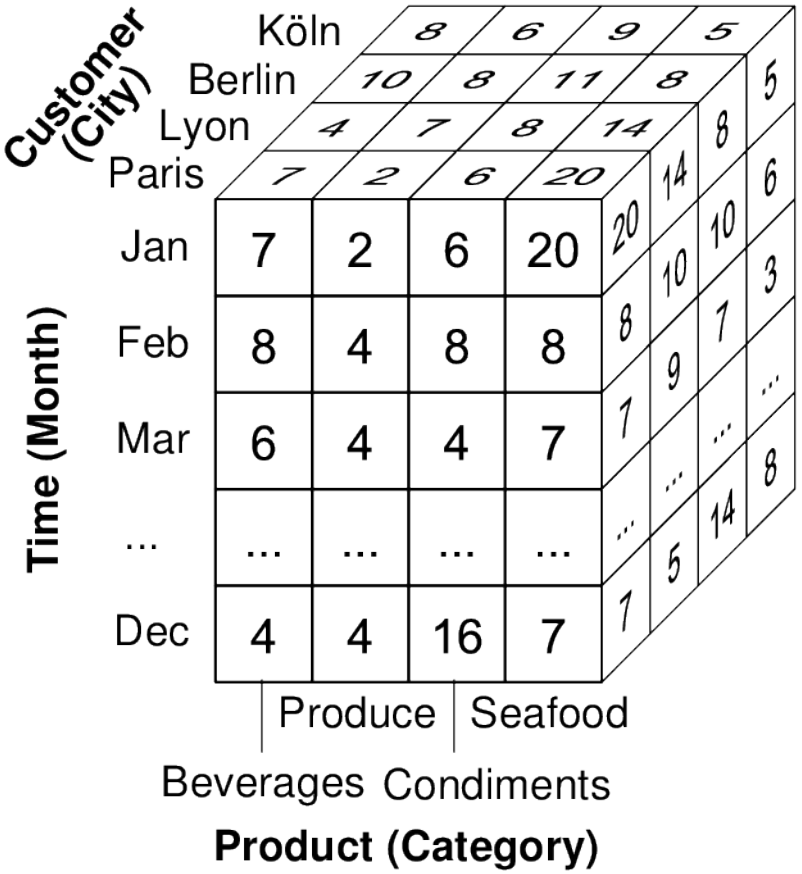
Quarters to Months

Credit: Figure from http://www.teradatapoint.com/data-warehousing-olap

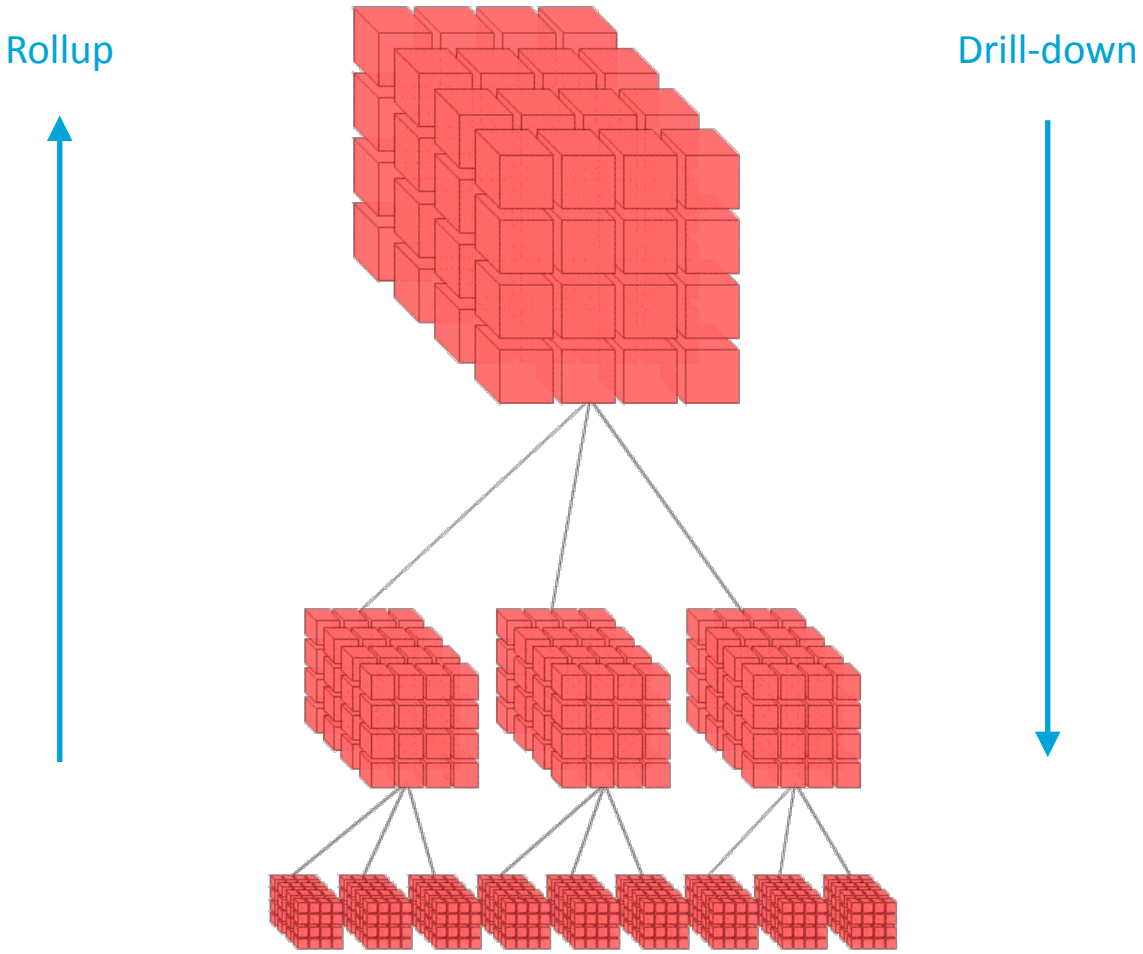
OLAP operations

- **Drill-down** to month





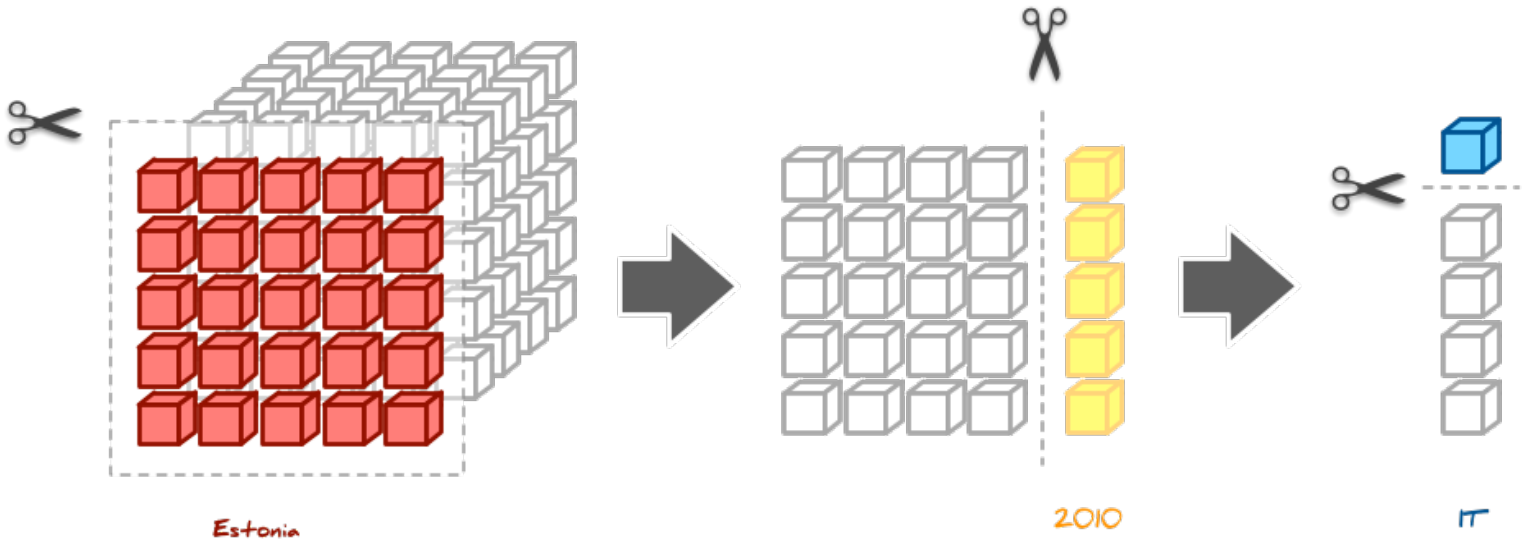
Roll-up vs Drill-down



Credit: Figure from https://visibledata.wordpress.com/data/datacloud/datacube/

Slice

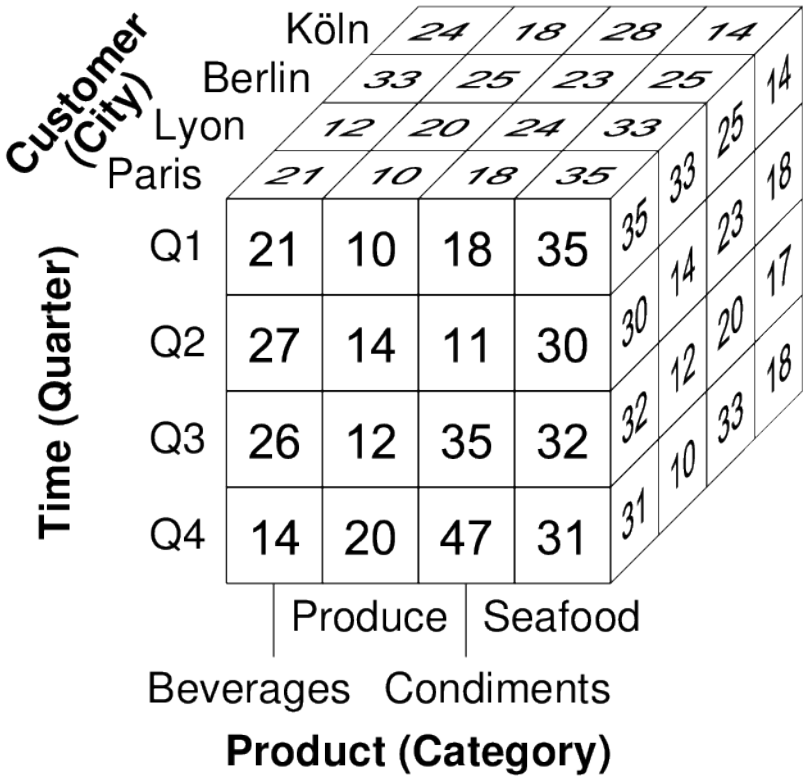
- Projects one (or more) dimensions of the given cube, resulting in a sub-cube along given dimensions
- Reduces the dimensionality of the cubes

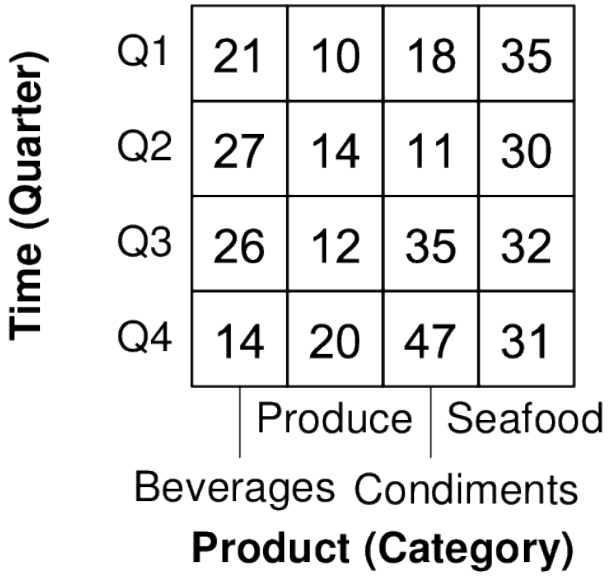


Credit: Figure from http://okfnlabs.org/blog/2014/01/20/olap-cubes-and-logical-model.html

OLAP operations

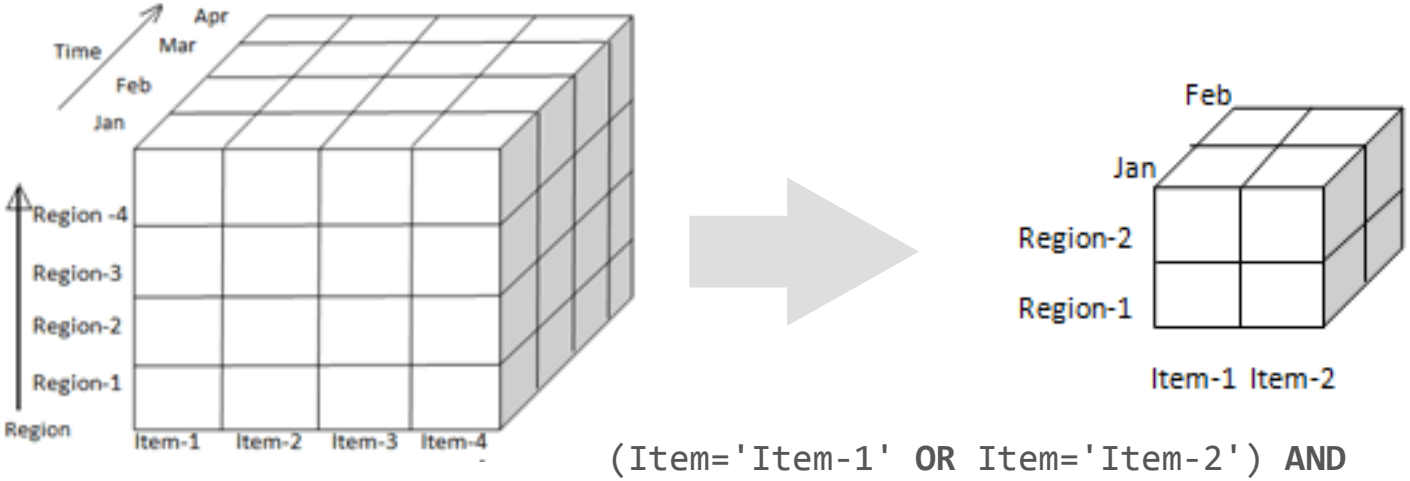
- **Slice** on city (Paris)





Dice

- Selects a sub-cube by performing by constraining the range of values over one or more dimensions.
- Reduces the number of member values of one or more dimensions.

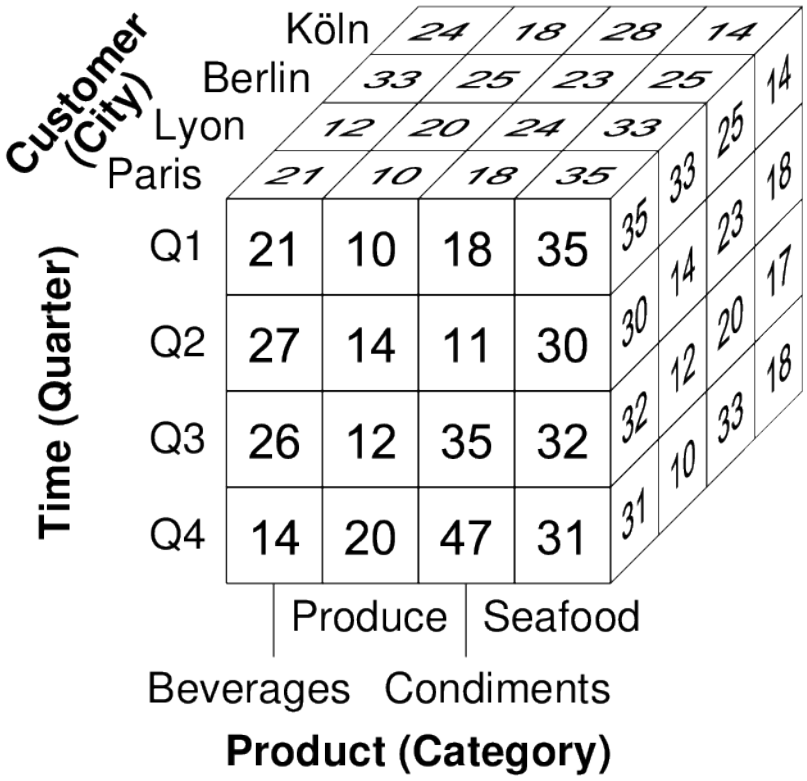


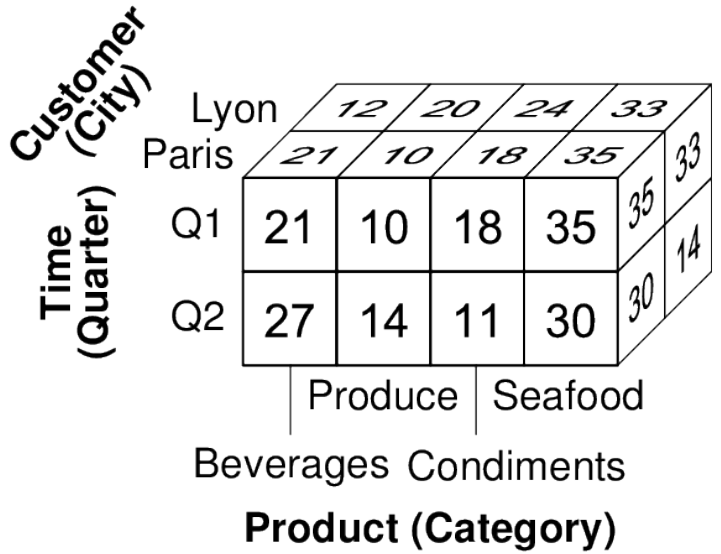

```
(Item='Item-1' OR Item='Item-2') AND
(Region='Region-1' OR Region='Region-2') AND
(Month='Jan' OR Month='Feb')
```

Credit: Figure from http://www.teradatapoint.com/data-warehousing-olap

OLAP operations

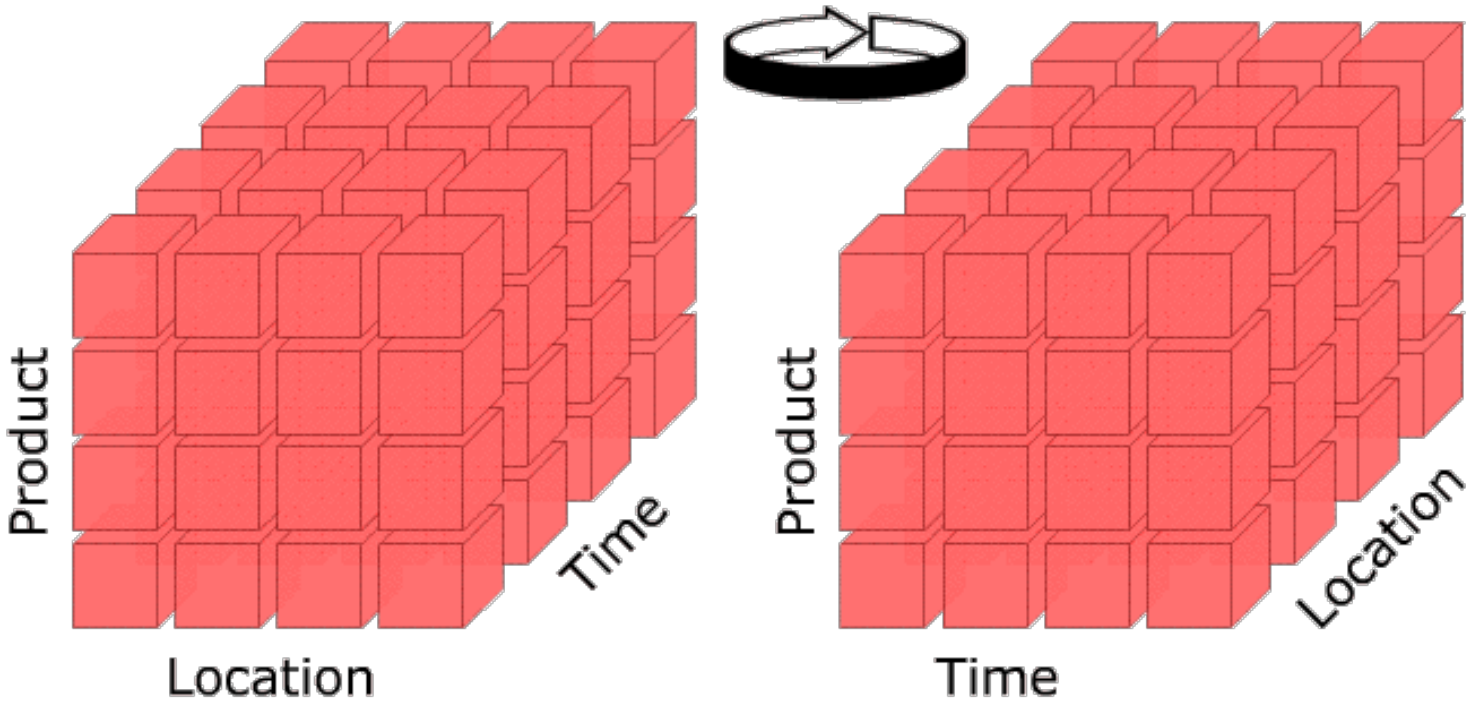
- **Dice** on city {Paris, Lyon} and quarter {Q1, Q2}





Pivot

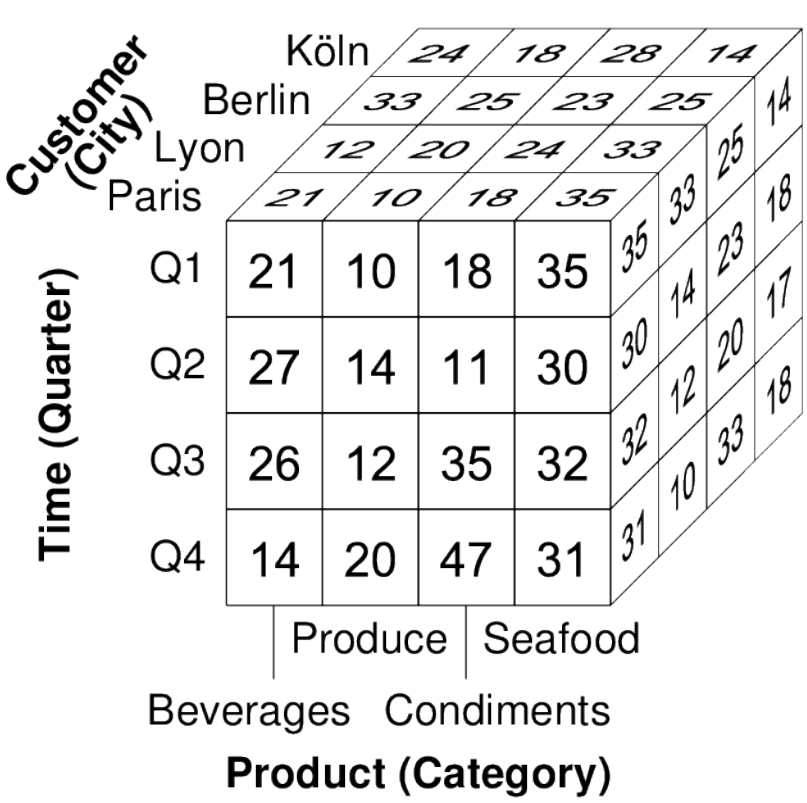
- Rotates the data axes to view the data from different perspectives.
- Re-orienting the multidimensional view of data

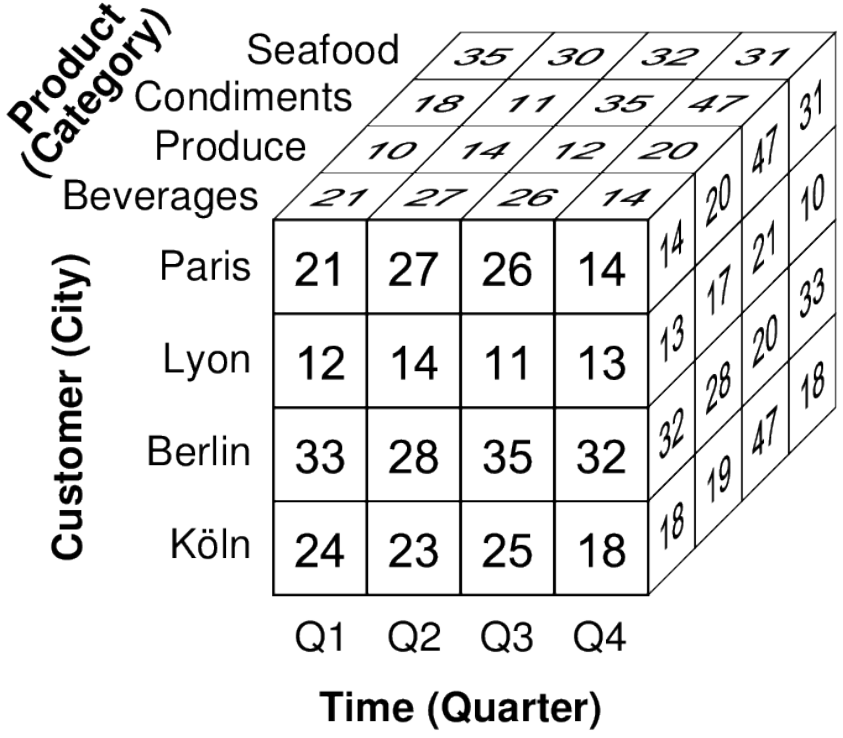


Credit: Figure from https://visibledata.wordpress.com/data/datacloud/datacube/

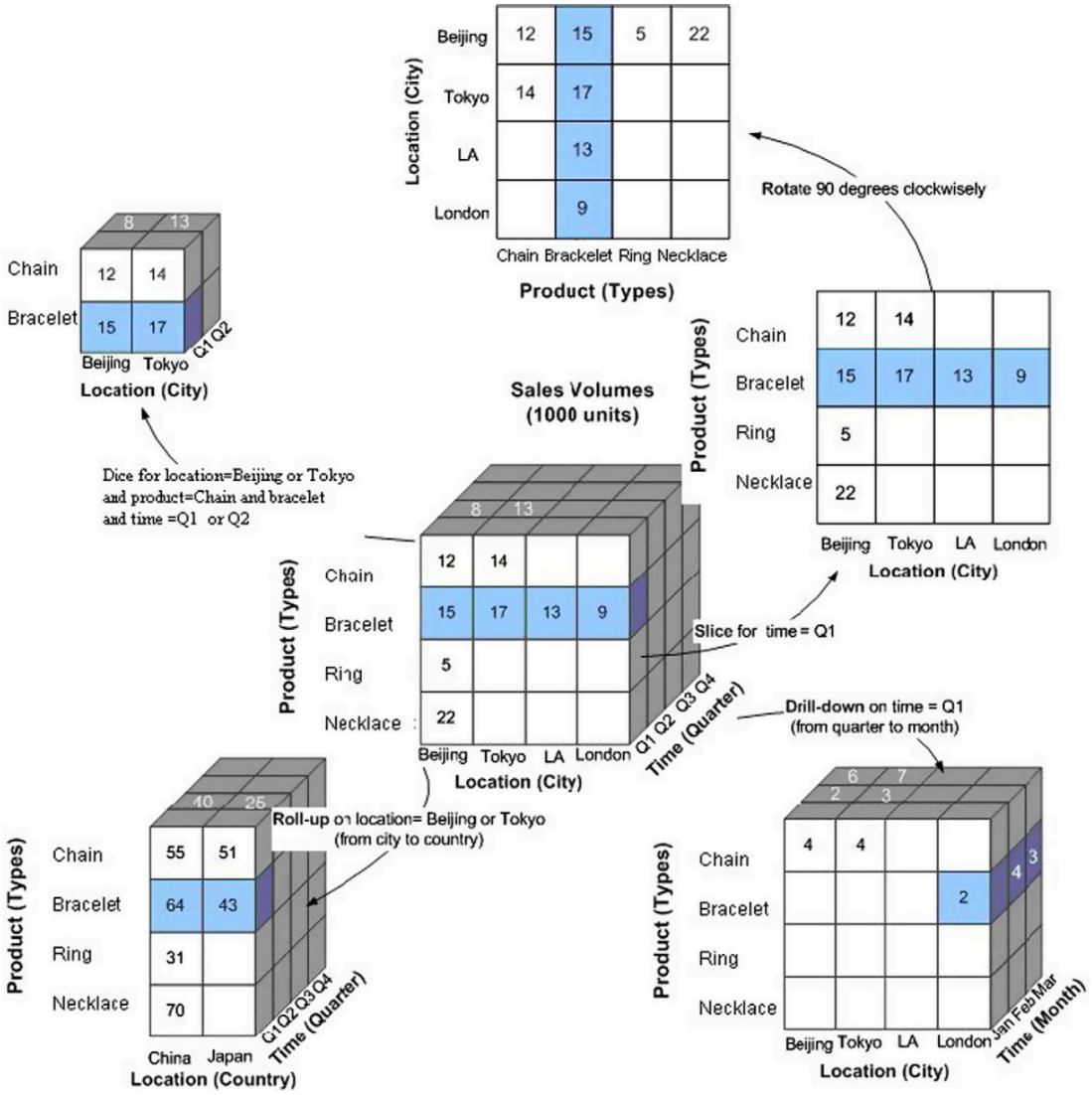
OLAP operations

- **Pivot** (rotate) dimensions





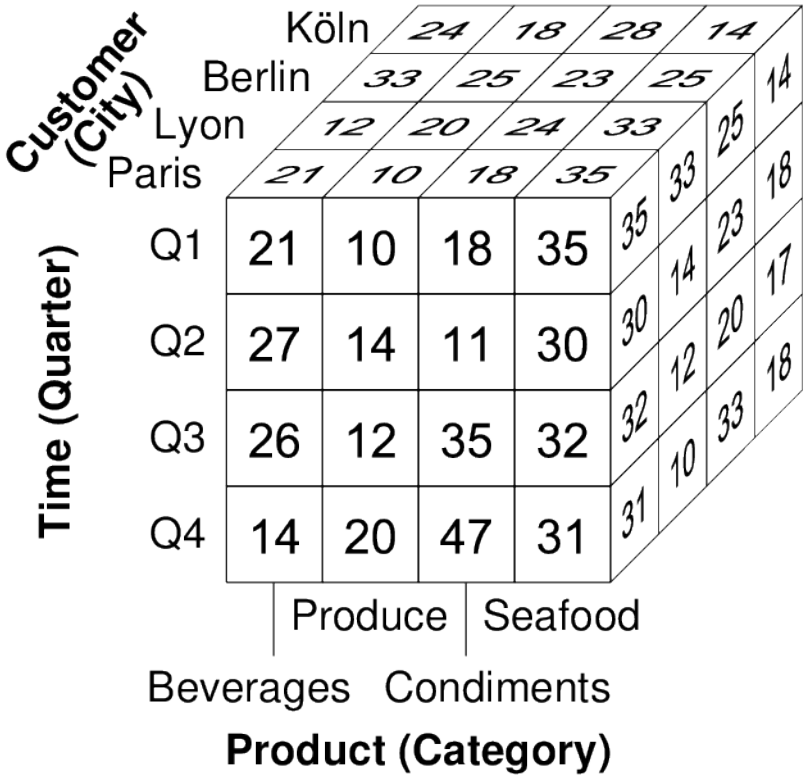
Combining Operations

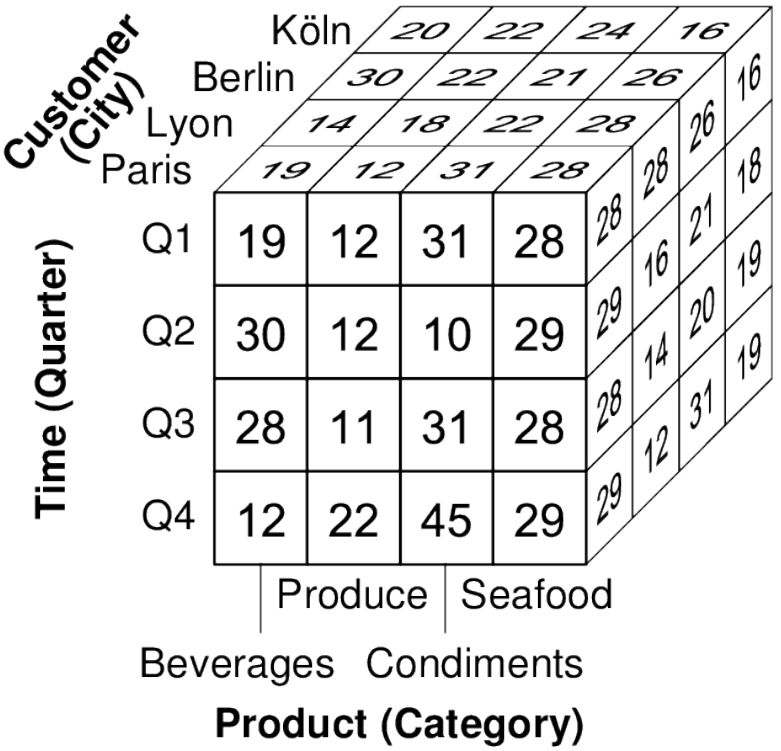


Credit: Figure from C.K.M. Lee et al. / Expert Systems with Applications 38 (2011)

Other OLAP operations

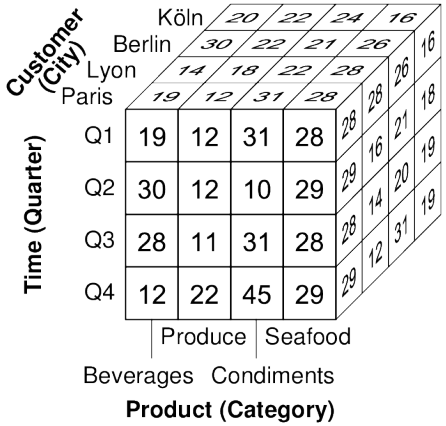
- Cube for 2012 vs. cube for 2011 (previous year)

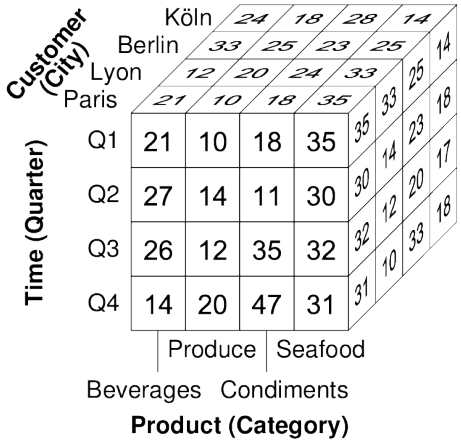


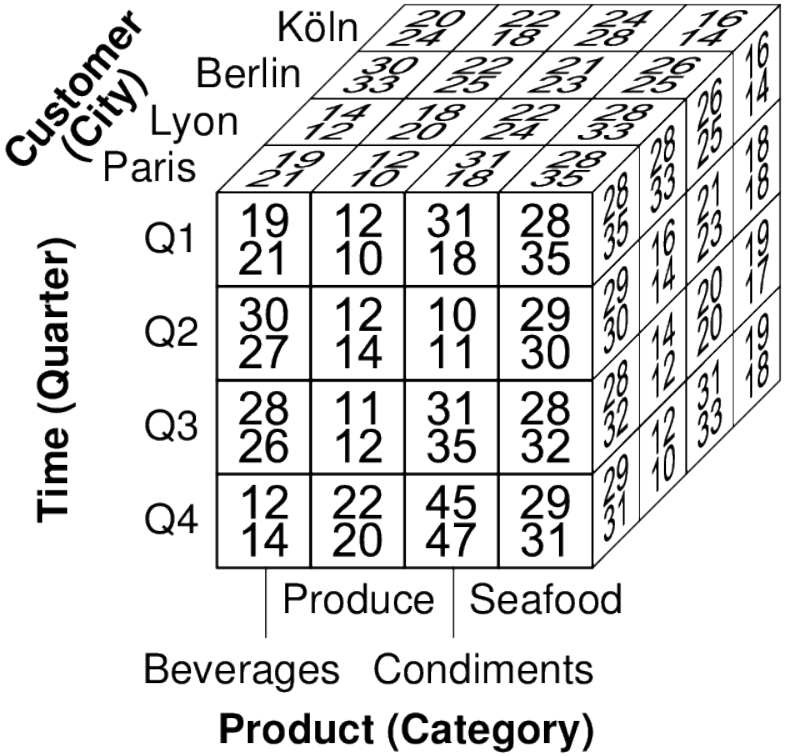


Other OLAP operations

- Drill-across 2011 and 2012

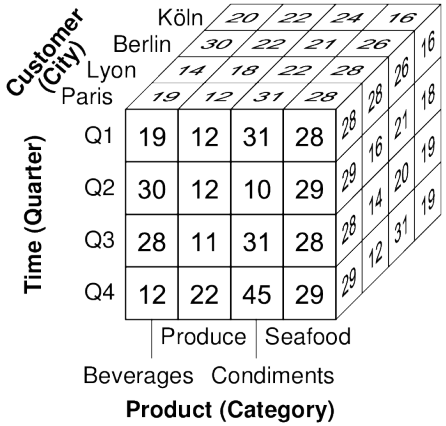


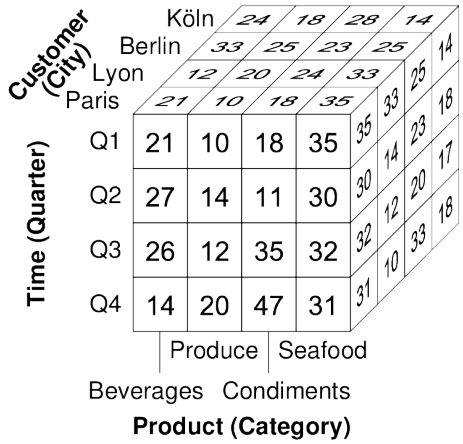


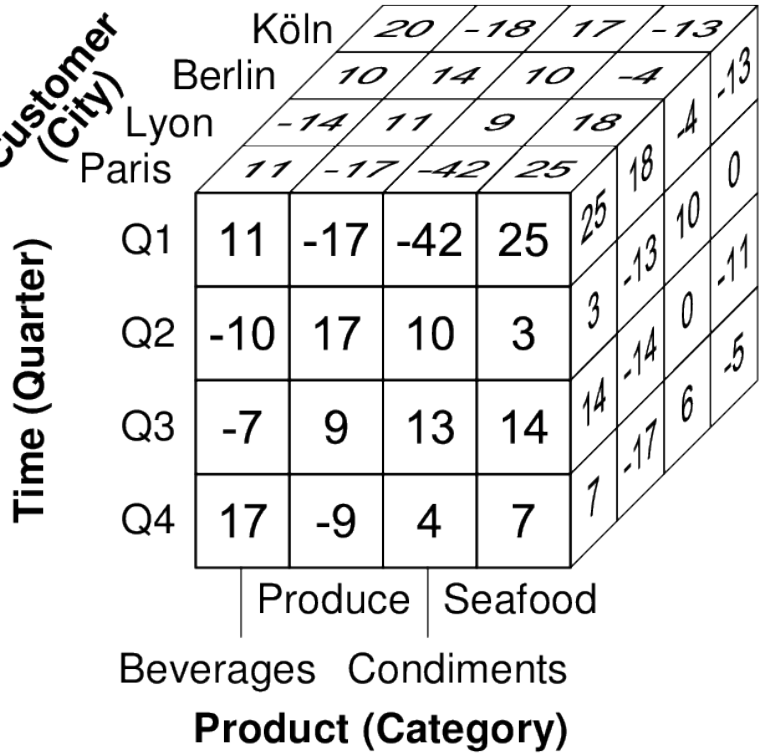


Other OLAP operations

- Percentage change between 2011 and 2012



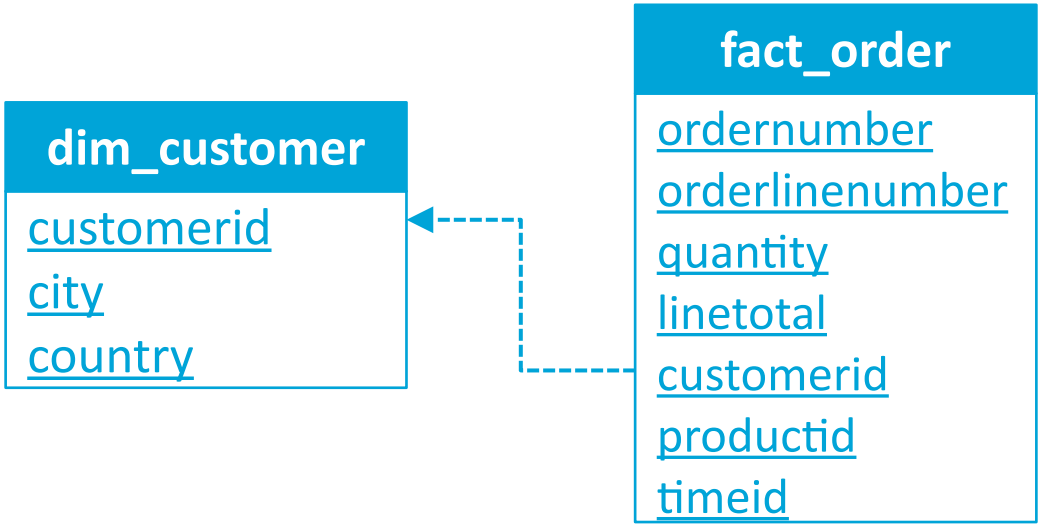




OLAP Operations in SQL

Data Warehousing

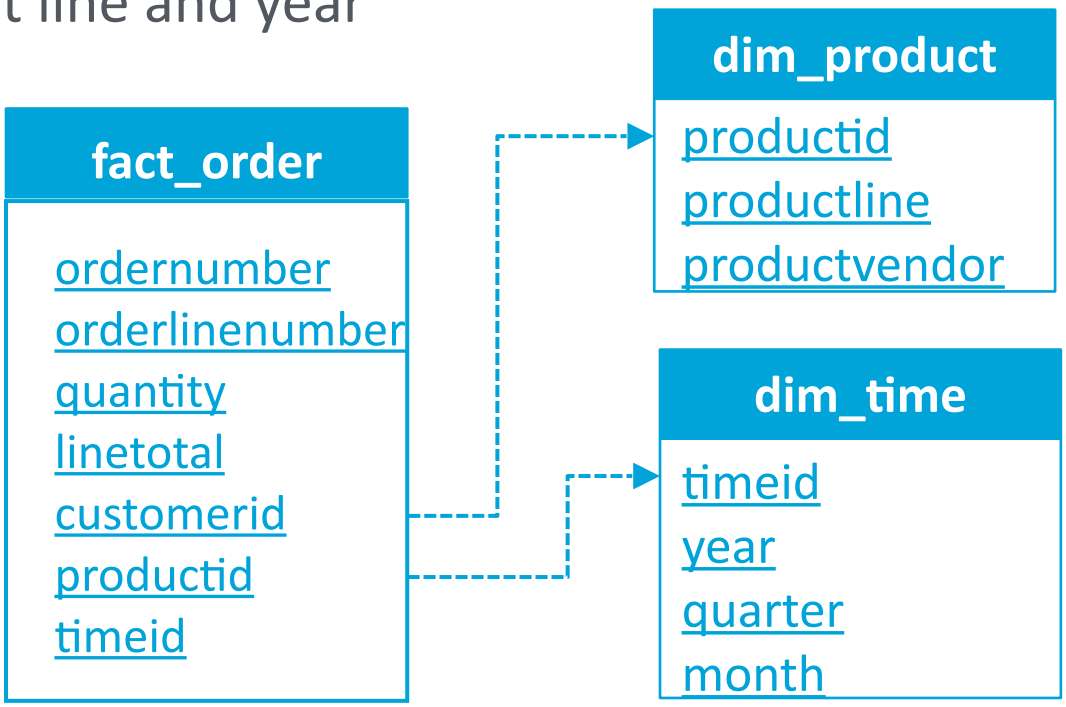
- Sales by customer country




```
select b.country, sum(a.linetotal) as sales
from fact_order as a, dim_customer as b
where a.customerid = b.customerid
group by b.country;
```

Data Warehousing

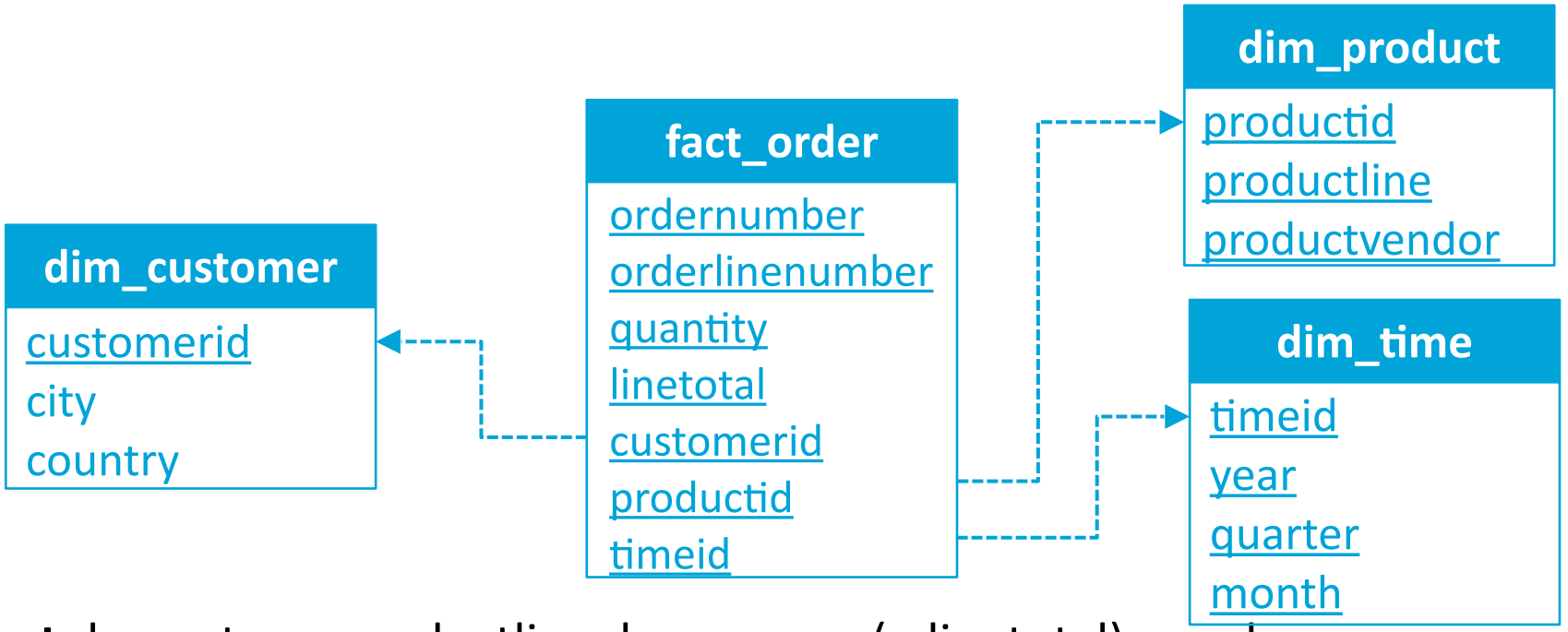
- Sales by product line and year




```
select c.productline, b.year, sum(a.linetotal) as sales
from fact_order as a, dim_time as b, dim_product as c
where a.timeid = b.timeid and a.productid = c.productid
group by c.productline, b.year;
```

Data Warehousing

- Sales by customer country, product line and year




```
select d.country, c.productline, b.year, sum(a.linetotal) as sales
from fact_order as a, dim_time as b, dim_product as c, dim_customer as d
where a.timeid = b.timeid and a.productid = c.productid and
        a.customerid = d.customerid
group by d.country, c.productline, b.year;
```

Drill-down

- Going from a higher to a lower level is called **drill-down**
  - e.g. sales by customer country → sales by customer city

```
select country, sum(linetotal) as sales
from fact_order natural join dim_customer
group by country;
```

Something missing?

```
select city, sum(linetotal) as sales
from fact_order natural join dim_customer
group by city;
```

Drill-down

- Going from a higher to a lower level is called **drill-down**
  - e.g. sales by customer country → sales by customer city

```
select country, sum(linetotal) as sales
from fact_order natural join dim_customer
group by country;
```

```
select country, city, sum(linetotal) as sales
from fact_order natural join dim_customer
group by country, city;
```

Drill-down

- Going from a higher to a lower level is called **drill-down**
  - e.g. sales by customer country ⟶ sales by customer city

```
+-------------+---------+
| country     | sales   |
+-------------+---------+
| Australia   |  630638 |
| Austria     |  202089 |
| Belgium     |  108485 |
| Canada      |  224085 |
| Denmark     |  245582 |
| Finland     |  329472 |
| France      | 1111022 |
| Germany     |  220354 |
| Hong Kong   |   48766 |
| Ireland     |   57788 |
| Italy       |  403696 |
| Japan       |  188212 |
| ...         |     ... |
+-------------+---------+
21 rows in set (0.02 sec)
```

```
+-------------+--------------------+---------+
| country     | city               | sales   |
+-------------+--------------------+---------+
| Australia   | Chatswood          |  151631 |
| Australia   | Glen Waverly       |   64621 |
| Australia   | Melbourne          |  200845 |
| Australia   | North Sydney       |  154070 |
| Australia   | South Brisbane     |   59471 |
| Austria     | Graz               |   52218 |
| Austria     | Salzburg           |  149871 |
| Belgium     | Bruxelles          |   75037 |
| Belgium     | Charleroi          |   33448 |
| Canada      | Montréal           |   74224 |
| Canada      | Tsawassen          |   74665 |
| Canada      | Vancouver          |   75196 |
| ...         | ...                |     ... |
+-------------+--------------------+---------+
81 rows in set (0.03 sec)
```

Drill-down

- Another example of **drill-down**
  - e.g. sales by year → sales by month

```
select year, sum(linetotal) as sales
from fact_order natural join dim_time
group by year;
```

```
select year, month, sum(linetotal) as sales
from fact_order natural join dim_time
group by year, month;
```

Drill-down

- Another example of **drill-down**
  - e.g. sales by year ⟶ sales by month

```
+------+---------+
| year | sales   |
+------+---------+
| 2003 | 4312435 |
| 2004 | 4987780 |
| 2005 | 1980850 |
+------+---------+
3 rows in set (0.02 sec)
```

```
+------+-------+---------+
| year | month | sales   |
+------+-------+---------+
| 2003 |     1 |  764883 |
| 2003 |     2 |  140920 |
| 2003 |     3 |  174467 |
| 2003 |     4 |  201557 |
| 2003 |     5 |  192785 |
| 2003 |     6 |  170533 |
| 2003 |     7 |  225638 |
| 2003 |     8 |  197822 |
| 2003 |     9 |  263836 |
| 2003 |    10 |  589773 |
| 2003 |    11 | 1086757 |
| 2003 |    12 |  303464 |
| 2004 |     1 |  316662 |
| 2004 |     2 |  318663 |
|  ... |   ... |     ... |
+------+-------+---------+
29 rows in set (0.01 sec)
```

Roll-up

- Going from a lower to a higher level is called **roll-up**
  - e.g. sales by customer city → sales by customer country

```
select country, city, sum(linetotal) as sales
from fact_order natural join dim_customer
group by country, city;
```

```
select country, sum(linetotal) as sales
from fact_order natural join dim_customer
group by country;
```

Roll-up

- Going from a lower to a higher level is called **roll-up**
  - e.g. sales by customer city ⟶ sales by customer country

country	city	sales
Australia	Chatswood	151631
Australia	Glen Waverly	64621
Australia	Melbourne	200845
Australia	North Sydney	154070
Australia	South Brisbane	59471
Austria	Graz	52218
Austria	Salzburg	149871
Belgium	Bruxelles	75037
Belgium	Charleroi	33448
Canada	Montréal	74224
Canada	Tsawassen	74665
Canada	Vancouver	75196
...	...	...

81 rows in set (0.03 sec)

country	sales
Australia	630638
Austria	202089
Belgium	108485
Canada	224085
Denmark	245582
Finland	329472
France	1111022
Germany	220354
Hong Kong	48766
Ireland	57788
Italy	403696
Japan	188212
...	...

21 rows in set (0.02 sec)

Roll-up

- Another example of **roll-up**
  - e.g. sales by month → sales by year

```
select year, month, sum(linetotal) as sales
from fact_order natural join dim_time
group by year, month;
```

```
select year, sum(linetotal) as sales
from fact_order natural join dim_time
group by year;
```

Roll-up

- Another example of **roll-up**
  - e.g. sales by month → sales by year

```
+------+-------+---------+
| year | month | sales   |
+------+-------+---------+
| 2003 |     1 |  764883 |
| 2003 |     2 |  140920 |
| 2003 |     3 |  174467 |
| 2003 |     4 |  201557 |
| 2003 |     5 |  192785 |
| 2003 |     6 |  170533 |
| 2003 |     7 |  225638 |
| 2003 |     8 |  197822 |
| 2003 |     9 |  263836 |
| 2003 |    10 |  589773 |
| 2003 |    11 | 1086757 |
| 2003 |    12 |  303464 |
| 2004 |     1 |  316662 |
| 2004 |     2 |  318663 |
|  ... |   ... |     ... |
+------+-------+---------+
29 rows in set (0.01 sec)
```

```
+------+---------+
| year | sales   |
+------+---------+
| 2003 | 4312435 |
| 2004 | 4987780 |
| 2005 | 1980850 |
+------+---------+
3 rows in set (0.02 sec)
```

Slice

- Selecting a particular value of dimension level is a **slice**
  - e.g. sales by product line in 2003

```
select productline, year, sum(linetotal) as sales
from fact_order natural join dim_product natural join dim_time
group by productline, year;
```

```
select productline, year, sum(linetotal) as sales
from fact_order natural join dim_product natural join dim_time
group by productline, year
having year = 2003;
```

Slice

- Selecting a particular value of dimension level is a **slice**
  - e.g. sales by product line in 2003

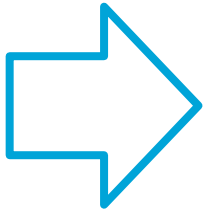
```
select productline, year, sum(linetotal) as sales
from fact_order natural join dim_product natural join dim_time
group by productline, year;
```

```
select productline, year, sum(linetotal) as sales
from fact_order natural join dim_product natural join dim_time
where year = 2003
group by productline, year;
```

Slice

- Selecting a particular value of dimension level is a **slice**
  - e.g. sales by product line in 2003

```
+------------------+------+---------+
| productline      | year | sales   |
+------------------+------+---------+
| Classic Cars     | 2003 | 1513998 |
| Classic Cars     | 2004 | 1837904 |
| Classic Cars     | 2005 |  738587 |
| Motorcycles      | 2003 |  397392 |
| Motorcycles      | 2004 |  590632 |
| Motorcycles      | 2005 |  286327 |
| Planes           | 2003 |  347924 |
| Planes           | 2004 |  529129 |
| Planes           | 2005 |  200077 |
| Ships            | 2003 |  244652 |
| Ships            | 2004 |  375498 |
| Ships            | 2005 |  128219 |
| Trains           | 2003 |   72857 |
| Trains           | 2004 |  124885 |
| Trains           | 2005 |   36920 |
| ...              |  ... |     ... |
+------------------+------+---------+
21 rows in set (0.04 sec)
```



```
+------------------+------+---------+
| productline      | year | sales   |
+------------------+------+---------+
| Classic Cars     | 2003 | 1513998 |
| Motorcycles      | 2003 |  397392 |
| Planes           | 2003 |  347924 |
| Ships            | 2003 |  244652 |
| Trains           | 2003 |   72857 |
| Trucks and Buses | 2003 |  420523 |
| Vintage Cars     | 2003 | 1315089 |
+------------------+------+---------+
7 rows in set (0.01 sec)
```

Dice

- Applying multiple slicing conditions is called a **dice**
  - e.g. sales of Motorcycles in 2003 and 2004

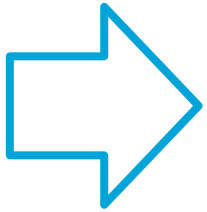
```
select productline, year, sum(linetotal) as sales
from fact_order natural join dim_product natural join dim_time
group by productline, year;
```

```
select productline, year, sum(linetotal) as sales
from fact_order natural join dim_product natural join dim_time
where productline = 'Motorcycles' and year in (2003, 2004)
group by productline, year;
```

Dice

- Applying multiple slicing conditions is called a **dice**
  - e.g. sales of Motorcycles in 2003 and 2004

```
+------------------+------+---------+
| productline      | year | sales   |
+------------------+------+---------+
| Classic Cars     | 2003 | 1513998 |
| Classic Cars     | 2004 | 1837904 |
| Classic Cars     | 2005 |  738587 |
| Motorcycles      | 2003 |  397392 |
| Motorcycles      | 2004 |  590632 |
| Motorcycles      | 2005 |  286327 |
| Planes           | 2003 |  347924 |
| Planes           | 2004 |  529129 |
| Planes           | 2005 |  200077 |
| Ships            | 2003 |  244652 |
| Ships            | 2004 |  375498 |
| Ships            | 2005 |  128219 |
| Trains           | 2003 |   72857 |
| Trains           | 2004 |  124885 |
| Trains           | 2005 |   36920 |
| ...              |  ... |     ... |
+------------------+------+---------+
21 rows in set (0.04 sec)
```



```
+-------------+------+--------+
| productline | year | sales  |
+-------------+------+--------+
| Motorcycles | 2003 | 397392 |
| Motorcycles | 2004 | 590632 |
+-------------+------+--------+
2 rows in set (0.01 sec)
```

Pivot

- Changing the order of dimensions is called **pivot**
  - e.g. sales by product line, year ⟶ sales by year, product line

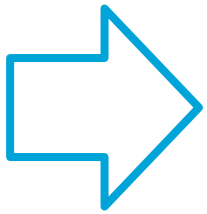
```
select productline, year, sum(linetotal) as sales
from fact_order natural join dim_product natural join dim_time
group by productline, year;
```

```
select year, productline, sum(linetotal) as sales
from fact_order natural join dim_product natural join dim_time
group by year, productline;
```

Pivot

- Changing the order of dimensions is called **pivot**
  - e.g. sales by product line, year ⟶ sales by year, product line

```
+------------------+------+---------+
| productline      | year | sales   |
+------------------+------+---------+
| Classic Cars     | 2003 | 1513998 |
| Classic Cars     | 2004 | 1837904 |
| Classic Cars     | 2005 |  738587 |
| Motorcycles      | 2003 |  397392 |
| Motorcycles      | 2004 |  590632 |
| Motorcycles      | 2005 |  286327 |
| Planes           | 2003 |  347924 |
| Planes           | 2004 |  529129 |
| Planes           | 2005 |  200077 |
| Ships            | 2003 |  244652 |
| Ships            | 2004 |  375498 |
| Ships            | 2005 |  128219 |
| Trains           | 2003 |   72857 |
| Trains           | 2004 |  124885 |
| Trains           | 2005 |   36920 |
| ...              |  ... |     ... |
+------------------+------+---------+
21 rows in set (0.04 sec)
```



```
+------+------------------+---------+
| year | productline      | sales   |
+------+------------------+---------+
| 2003 | Classic Cars     | 1513998 |
| 2003 | Motorcycles      |  397392 |
| 2003 | Planes           |  347924 |
| 2003 | Ships            |  244652 |
| 2003 | Trains           |   72857 |
| 2003 | Trucks and Buses |  420523 |
| 2003 | Vintage Cars     | 1315089 |
| 2004 | Classic Cars     | 1837904 |
| 2004 | Motorcycles      |  590632 |
| 2004 | Planes           |  529129 |
| 2004 | Ships            |  375498 |
| 2004 | Trains           |  124885 |
| 2004 | Trucks and Buses |  532024 |
| 2004 | Vintage Cars     |  997708 |
| 2005 | Classic Cars     |  738587 |
|  ... | ...              |     ... |
+------+------------------+---------+
21 rows in set (0.04 sec)
```

OLAP operations

- Typical analytical operations
  - **drill-down** and **roll-up** between levels
  - **slice** and **dice** to select particular values
  - **pivot** to rotate dimensions
  - etc.