L07: SQL: Advanced & Practice

CS3200 Database design (fa18 s2)

https://northeastern-datalab.github.io/cs3200/

Version 9/27/2018

Announcements!

- Recall Exam 1 on THU Oct 4 (1h)
  - Topic: SQL on your own computer and PostgreSQL instance
  - look at HW10 on BB before
  - 0 point practice exam on MON Oct 1 (just practicing test modalities)
- Final exam scheduled for Dec 11, 8am-10am
- Use anonymous feedback option regularly. We have a structured feedback session in around 2 weeks.

- ? Excel and Databases

Practice: Conceptual evaluation strategy

R

a
1
2

S

a
1

U

a
2
3
4

What do these queries compute?

```
SELECT R.a, S.a
FROM   R, S
WHERE  R.a <> ALL
   (SELECT U.a
   FROM   U)
```

⇨

```
SELECT R.a, S.a
FROM   R, S, U
WHERE  R.a <> U.a
```

⇨

Practice: Conceptual evaluation strategy

R
a
1
2

S
a
1

U
a
2
3
4

What do these queries compute?

```
SELECT R.a, S.a
FROM   R, S
WHERE  R.a <> ALL
   (SELECT U.a
   FROM   U)
```

⇨

R.a	S.A
1	1

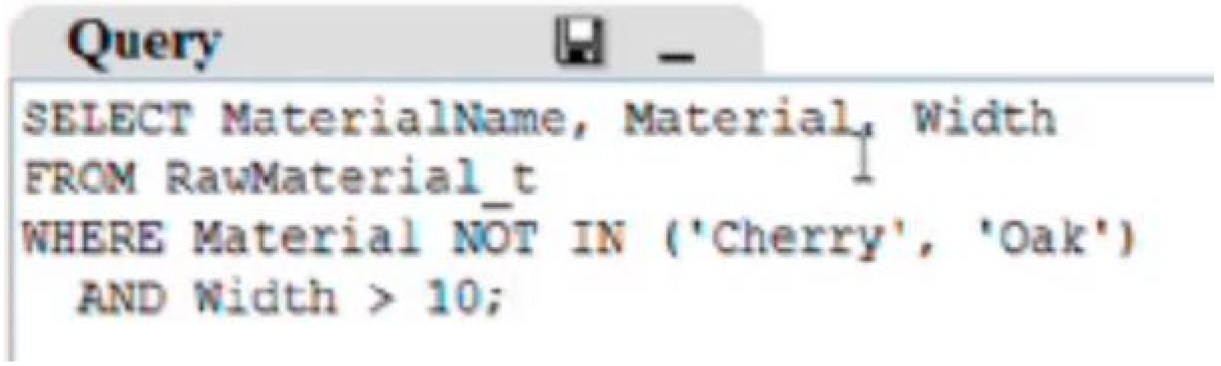
```
SELECT R.a, S.a
FROM   R, S, U
WHERE  R.a <> U.a
```

⇨

R.a	S.A
1	1
1	1
1	1
2	1
2	1

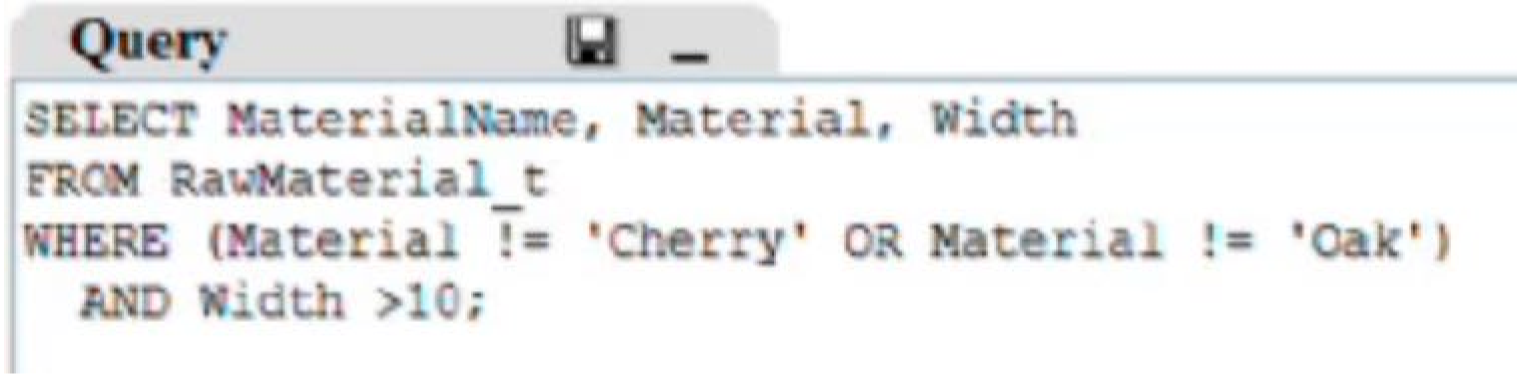
Small Detail on the Hoffer Video for Ch06

Are these two queries equivalent?
Hoffer video for Ch6 at around 20:20



Query

```
SELECT MaterialName, Material, Width
FROM RawMaterial_t
WHERE Material NOT IN ('Cherry', 'Oak')
  AND Width > 10;
```

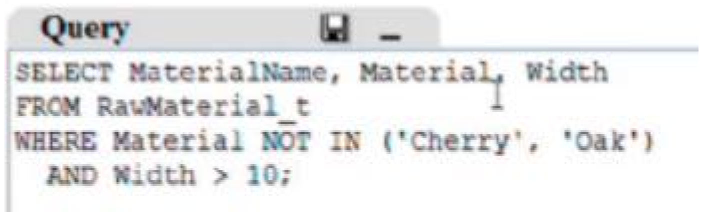


Query

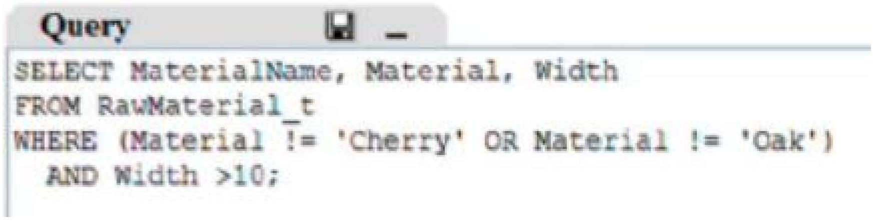
```
SELECT MaterialName, Material, Width
FROM RawMaterial_t
WHERE (Material != 'Cherry' OR Material != 'Oak')
  AND Width >10;
```

Source: http://mediaplayer.pearsoncmg.com/_ph_bp2_cc_set.title.Chapter_6%26%2358%3B_Writing_SQL_Queries%26%2358%3B_Part_1_%26%23150%3B_Single_Table_Queries__/bp_hoffer_mdm_10/hoffer10e_ch6_video.m4v

Venn Diagram

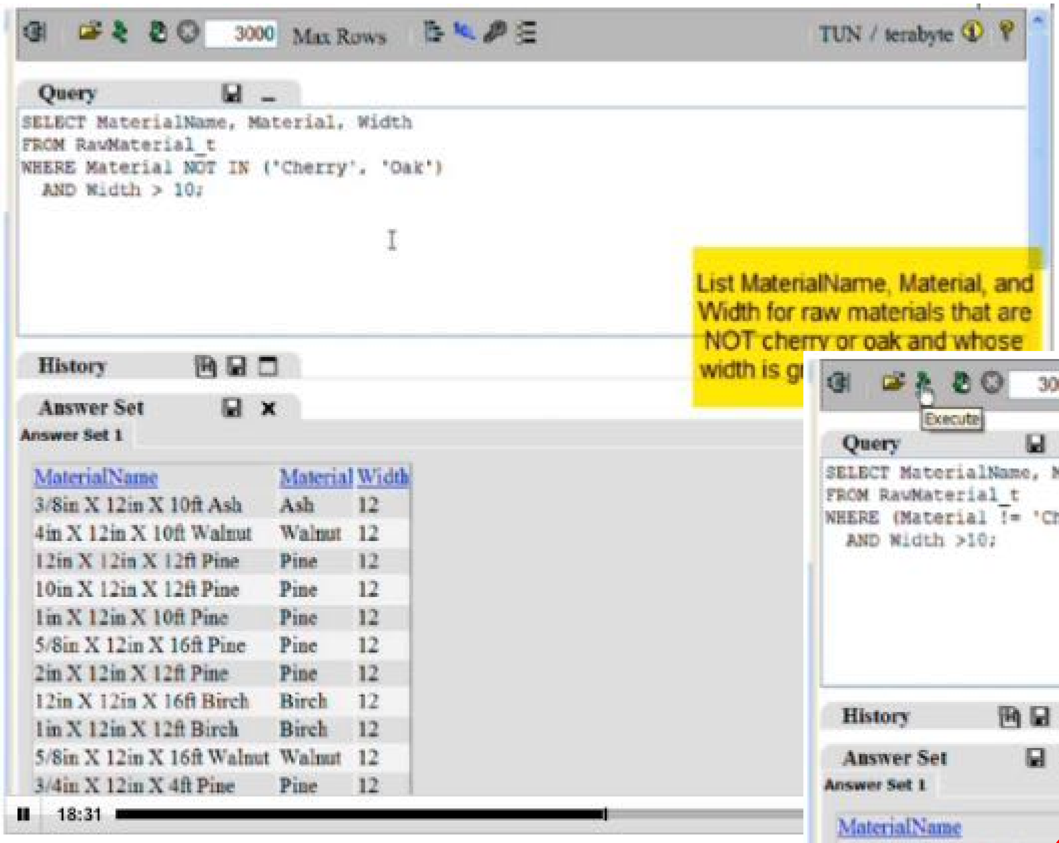

Query

```
SELECT MaterialName, Material, Width
FROM RawMaterial_t
WHERE Material NOT IN ('Cherry', 'Oak')
  AND Width > 10;
```


Query

```
SELECT MaterialName, Material, Width
FROM RawMaterial_t
WHERE (Material != 'Cherry' OR Material != 'Oak')
  AND Width >10;
```

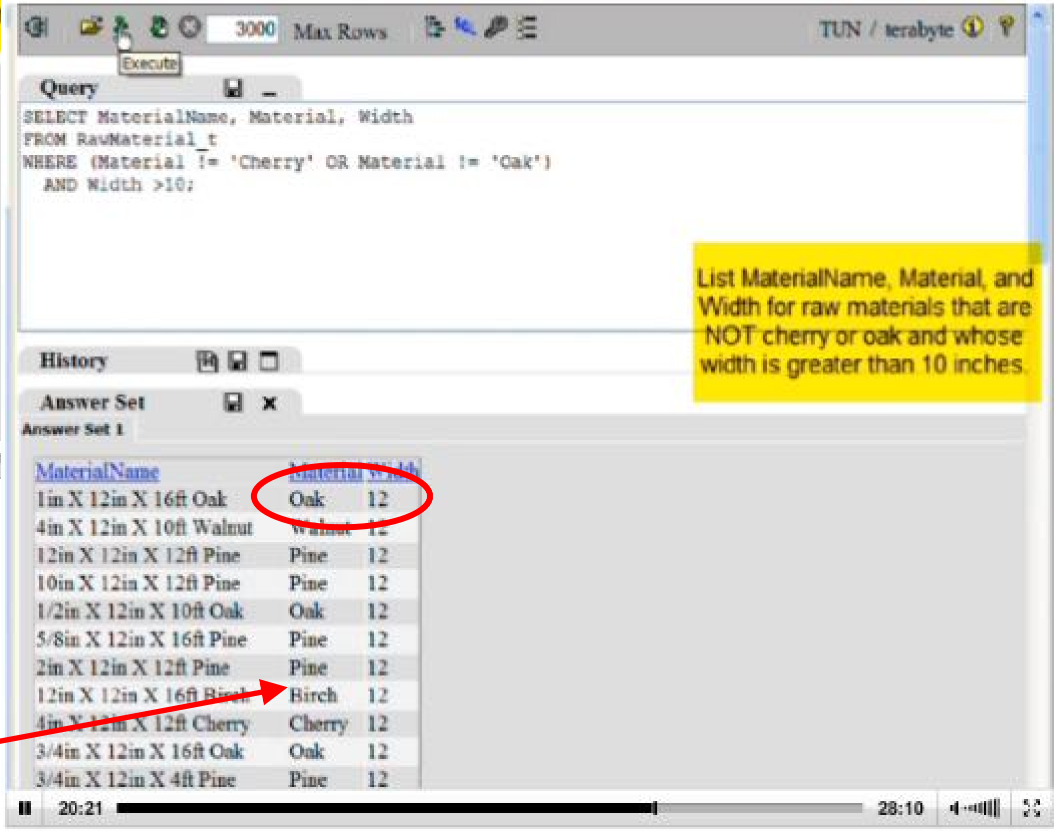
Are these two queries equivalent?



```
SELECT MaterialName, Material, Width
FROM RawMaterial_t
WHERE Material NOT IN ('Cherry', 'Oak')
  AND Width > 10;
```

List MaterialName, Material, and Width for raw materials that are NOT cherry or oak and whose width is g

MaterialName	Material	Width
3/8in X 12in X 10ft Ash	Ash	12
4in X 12in X 10ft Walnut	Walnut	12
12in X 12in X 12ft Pine	Pine	12
10in X 12in X 12ft Pine	Pine	12
1in X 12in X 10ft Pine	Pine	12
5/8in X 12in X 16ft Pine	Pine	12
2in X 12in X 12ft Pine	Pine	12
12in X 12in X 16ft Birch	Birch	12
1in X 12in X 12ft Birch	Birch	12
5/8in X 12in X 16ft Walnut	Walnut	12
3/4in X 12in X 4ft Pine	Pine	12



```
SELECT MaterialName, Material, Width
FROM RawMaterial_t
WHERE (Material != 'Cherry' OR Material != 'Oak')
  AND Width >10;
```

List MaterialName, Material, and Width for raw materials that are NOT cherry or oak and whose width is greater than 10 inches.

MaterialName	Material	Width
1in X 12in X 16ft Oak	Oak	12
4in X 12in X 10ft Walnut	Walnut	12
12in X 12in X 12ft Pine	Pine	12
10in X 12in X 12ft Pine	Pine	12
1/2in X 12in X 10ft Oak	Oak	12
5/8in X 12in X 16ft Pine	Pine	12
2in X 12in X 12ft Pine	Pine	12
12in X 12in X 16ft Birch	Birch	12
4in X 12in X 12ft Cherry	Cherry	12
3/4in X 12in X 16ft Oak	Oak	12
3/4in X 12in X 4ft Pine	Pine	12

Also the query results show the difference

Source: http://mediaplayer.pearsoncmg.com/_ph_bp2_cc_set.title.Chapter_6%26%2358%3B_Writing_SQL_Queries%26%2358%3B_Part_1_%26%23150%3B_Single_Table_Queries__/bp_hoffer_mdm_10/hoffer10e_ch6_video.m4v

Set & Multiset operations (UNION, INTERSECT, EXCEPT)

Question

Student

sid	name	year
861	Alice	2
753	Bob	1
955	Charly	2

Employee

id	name
233	Alice
651	Dora
122	Emil

With what we have so far, could you find the ids of all persons (students and employees)?

Set Operations

- We can apply union, intersection and difference to two (or more) queries
  - (Q_1) UNION (Q_2)
  - (Q_1) INTERSECT (Q_2)
  - (Q_1) EXCEPT (Q_2)
- Subqueries must be union compatible in a *weak sense*
  - Same #attributes
  - Types of corresponding attributes must be *convertible to each other* (e.g., int→float)
  - The output adopts names of 1st subquery

Bag or Set Semantics?

- Default is *set semantics*:
  1. Eliminate duplicates
  2. Apply operator
  3. Eliminate duplicates

- For bag semantics, use the keyword ALL
  - (Q_1) UNION ALL (Q_2)
  - (Q_1) INTERSECT ALL (Q_2)
  - (Q_1) EXCEPT ALL (Q_2)

Question Revisited

Student

sid	name	year
861	Alice	2
753	Bob	1
955	Charlie	2

Employee

id	name
233	Alice
651	Dora
122	Emil

```
(SELECT sid FROM Student)
UNION
(SELECT id FROM Employee)
```

sid
861
753
955
233
651
122

What are the Results?

Student

sid	name	year
861	Alice	2
753	Bob	1
955	Charlie	2

Employee

id	name
233	Alice
651	Dora
122	Emil

```
(SELECT name FROM Student)
UNION
(SELECT name FROM Employee)
```

```
(SELECT name FROM Student)
UNION ALL
(SELECT name FROM Employee)
```

```
(
 (SELECT name FROM Student)
 UNION ALL
 (SELECT name FROM Employee)
)
EXCEPT ALL
(SELECT name FROM Employee)
```

Do we need a "union"?

R

a
1
2

U

a
2
3
4

305

```
SELECT a
FROM   R
UNION
SELECT a
FROM   U
```

Do we need "union"?

R

a
1
2

U

a
2
3
4

```
SELECT a
FROM   R
UNION
SELECT a
FROM   U
```

a
1
2
3
4

Do we need "union"?

R

a
1
2

U

a
2
3
4

```
SELECT a
FROM   R
UNION
SELECT a
FROM   U
```

⇨

a
1
2
3
4

⇦

```
SELECT COALESCE(R.a,U.a) as a
FROM   R FULL JOIN U
on     R.a = U.a
```

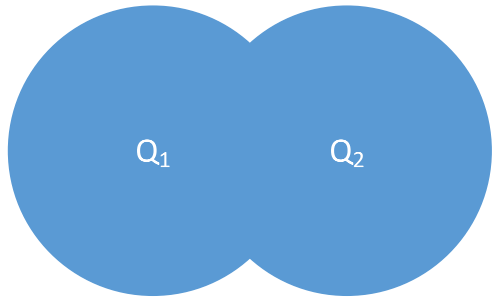
Do we need "union"? Whiteboard

-> HW 4

UNION

```
SELECT  R.A
FROM    R, S
WHERE   R.A=S.A
UNION
SELECT R.A
FROM    R, T
WHERE   R.A=T.A
```

$$\{r.A \mid r.A = s.A\} \cup \{r.A \mid r.A = t.A\}$$



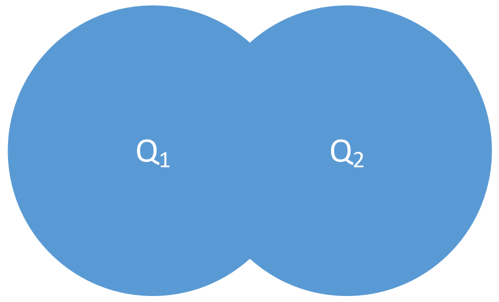
Why aren’t there duplicates?

By default: SQL uses set semantics for INTERSECT and UNION!

What if we want duplicates?

UNION ALL

$$\{r.A \mid r.A = s.A\} \cup \{r.A \mid r.A = t.A\}$$



```
SELECT  R.A
FROM    R, S
WHERE   R.A=S.A
UNION ALL
SELECT R.A
FROM    R, T
WHERE   R.A=T.A
```

ALL indicates Multiset operations

Recall Multisets (Bags)

Multiset X

Tuple
(1, a)
(1, a)
(1, b)
(2, c)
(2, c)
(2, c)
(1, d)
(1, d)

Equivalent Representations of a <u>**Multiset**</u>

$\boldsymbol{\lambda(X)}$= "Count of tuple in X"
(Items not listed have implicit count 0)

Multiset X

Tuple	$\lambda(X)$
(1, a)	2
(1, b)	1
(2, c)	3
(1, d)	2

Note: In a set all counts are {0,1}.

Generalizing Set Operations to Multiset Operations

Multiset X

Tuple	$\lambda(X)$
(1, a)	2
(1, b)	0
(2, c)	3
(1, d)	0

$\cap$

Multiset Y

Tuple	$\lambda(Y)$
(1, a)	5
(1, b)	1
(2, c)	2
(1, d)	2

$=$

Multiset Z

Tuple	$\lambda(Z)$
(1, a)	2
(1, b)	0
(2, c)	2
(1, d)	0

$$\boldsymbol{\lambda(Z) = min(\lambda(X), \lambda(Y))}$$

For sets, this is **intersection**

Generalizing Set Operations to Multiset Operations

Multiset X

Tuple	$\lambda(X)$
(1, a)	2
(1, b)	0
(2, c)	3
(1, d)	0

$\cup$

Multiset Y

Tuple	$\lambda(Y)$
(1, a)	5
(1, b)	1
(2, c)	2
(1, d)	2

$=$

Multiset Z

Tuple	$\lambda(Z)$
(1, a)	7
(1, b)	1
(2, c)	5
(1, d)	2

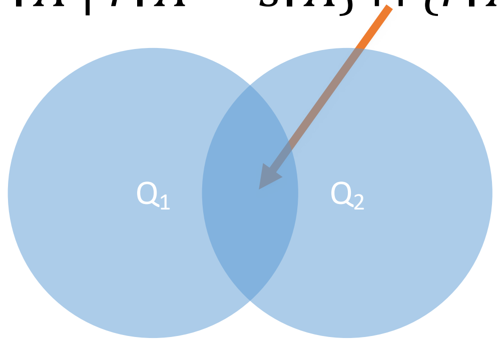
$$\lambda(Z) = \lambda(X) + \lambda(Y)$$

For sets,
this is **union**

Explicit Set Operators: INTERSECT

```
SELECT R.A
FROM   R, S
WHERE  R.A=S.A
INTERSECT
SELECT R.A
FROM   R, T
WHERE  R.A=T.A
```

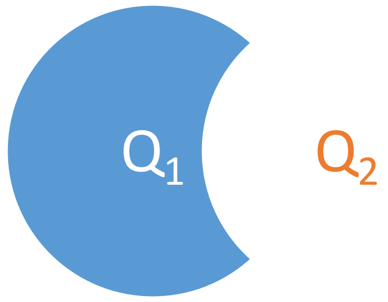
$$\{r.A \mid r.A = s.A\} \cap \{r.A \mid r.A = t.A\}$$



EXCEPT

```
SELECT R.A
FROM   R, S
WHERE  R.A=S.A
EXCEPT
SELECT R.A
FROM   R, T
WHERE  R.A=T.A
```

$$\{r.A \mid r.A = s.A\} \setminus \{r.A \mid r.A = t.A\}$$



What is the multiset version?

INTERSECT and EXCEPT*

R(a,b)
S(a,b)

```
(SELECT R.a, R.b
 FROM    R)

INTERSECT

(SELECT S.a, S.b
 FROM    S)
```

```
(SELECT R.A, R.B
 FROM    R)

EXCEPT

(SELECT S.A, S.B
 FROM    S)
```

*Not in all DBMSs. (SQLlite does not like the parentheses, Oracle uses "MINUS" instead of "EXCEPT")

INTERSECT and EXCEPT*

```
R(a,b)
S(a,b)
```

```
(SELECT R.a, R.b
 FROM    R)

INTERSECT

(SELECT S.a, S.b
 FROM    S)
```

```
SELECT  R.a, R.b
FROM    R
WHERE
EXISTS  (SELECT  *
         FROM    S
         WHERE   R.a=S.a
         and     R.b=S.b)
```

```
(SELECT R.A, R.B
 FROM    R)

EXCEPT

(SELECT S.A, S.B
 FROM    S)
```

```
SELECT R.A, R.B
FROM    R
WHERE
NOT  EXISTS  (SELECT  *
              FROM    S
              WHERE   R.A=S.A
              and     R.B=S.B)
```

*Not in all DBMSs. (SQLlite does not like the parentheses, Oracle uses "MINUS" instead of "EXCEPT")

INTERSECT and EXCEPT*

```
R(a,b)
S(a,b)
```

```
(SELECT R.a, R.b
 FROM     R)

INTERSECT

(SELECT S.a, S.b
 FROM    S)
```

```
SELECT  R.a, R.b
FROM    R
WHERE
EXISTS  (SELECT  *
         FROM    S
         WHERE   R.a=S.a
         and     R.b=S.b)
```

If R, S have no duplicates, then can write without sub-queries (HOW?)

```
(SELECT R.A, R.B
 FROM     R)

EXCEPT

(SELECT S.A, S.B
 FROM    S)
```

```
SELECT R.A, R.B
FROM    R
WHERE
NOT  EXISTS  (SELECT  *
              FROM    S
              WHERE   R.A=S.A
              and     R.B=S.B)
```

*Not in all DBMSs. (SQLlite does not like the parentheses, Oracle uses "MINUS" instead of "EXCEPT")

Top k

Top-k Tuples

SQL allows to limit the result to only the first k answers, for some number k of choice

```
  SELECT A₁,...,Aₖ
    FROM R₁,...,Rₙ
   WHERE Condition(B₁,...,Bₘ)
   ORDER BY C₁,...,Cₖ
   LIMIT k
```

Example:

```
  SELECT *
    FROM Student, Enroll
   WHERE Student.sid = Enroll.sid
ORDER BY name
   LIMIT 8
```

Example

Student

sid	name	year
861	Alice	2
753	Bob	1
955	Charlie	2

Enroll

sid	course
861	DB
861	PL
753	PL
753	AI
753	DC

```
  SELECT *
    FROM Student, Enroll
   WHERE Student.sid = Enroll.sid
ORDER BY name, course
   LIMIT 3
```

Example

Student

sid	name	year
861	Alice	2
753	Bob	1
955	Charlie	2

Enroll

sid	course
861	DB
861	PL
753	PL
753	AI
753	DC

```
  SELECT *
    FROM Student, Enroll
   WHERE Student.sid = Enroll.sid
ORDER BY name, course
   LIMIT 3
```

Student.sid	name	year	Enroll.sid	course
861	Alice	2	861	DB
861	Alice	2	861	PL
753	Bob	1	753	AI

Can you solve witnesses with top-1?

Product (pname, price, cid)

Q: Find the most expensive product + its price:

```
SELECT  P2.pname, P2.price
FROM    Product P2
WHERE   P2.price =
        (SELECT  max(P1.price)
         FROM    Product P1)
```

Can you solve witnesses with top-1?

Product (pname, price, cid)

Q: Find the most expensive product + its price:

```
SELECT  P2.pname, P2.price
FROM    Product P2
WHERE   P2.price =
        (SELECT  max(P1.price)
         FROM    Product P1)
```

```
SELECT  pname, price
FROM    Product
ORDER BY price
LIMIT 1
```

Can you solve witnesses with top-1?

Product (pname, price, cid)

Q: Find the most expensive product + its price:

```
SELECT  P2.pname, P2.price
FROM    Product P2
WHERE   P2.price =
        (SELECT  max(P1.price)
         FROM    Product P1)
```

```
SELECT  pname, price
FROM    Product
ORDER BY price
LIMIT 1
```

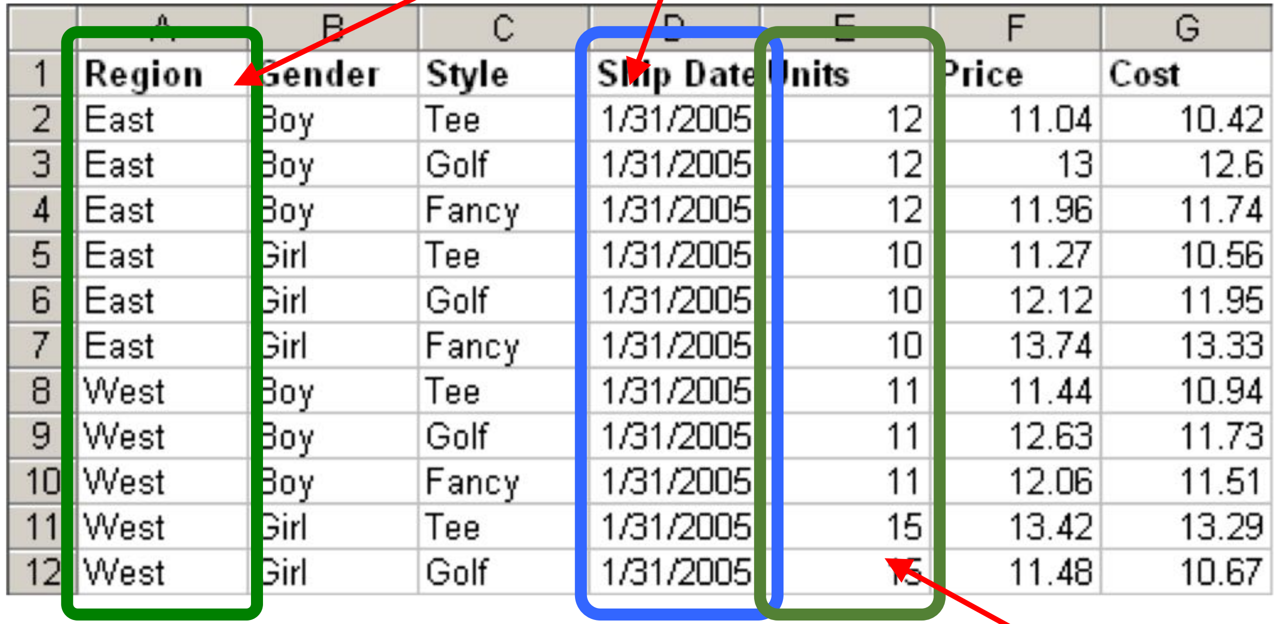
Product

PName	Price	cid
Gizmo	15	1
SuperGizmo	20	1
iTouch1	300	2
iTouch2	300	2

Excel Pivot and SQL OLAP "Online Analytical Processing"

Pivot tables

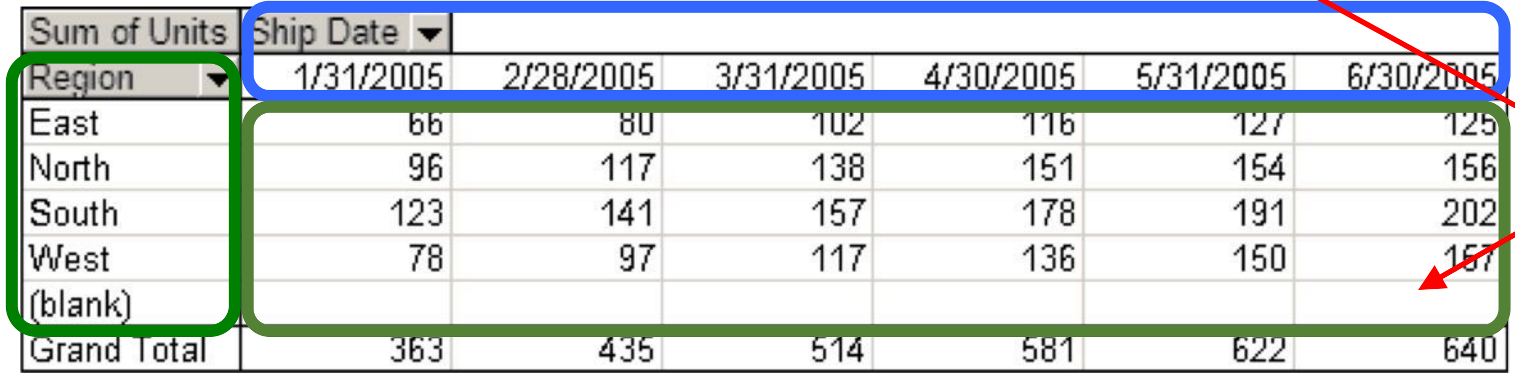
dimensions ("members, category")



	A	B	C	D	E	F	G
1	**Region**	**Gender**	**Style**	**Ship Date**	**Units**	**Price**	**Cost**
2	East	Boy	Tee	1/31/2005	12	11.04	10.42
3	East	Boy	Golf	1/31/2005	12	13	12.6
4	East	Boy	Fancy	1/31/2005	12	11.96	11.74
5	East	Girl	Tee	1/31/2005	10	11.27	10.56
6	East	Girl	Golf	1/31/2005	10	12.12	11.95
7	East	Girl	Fancy	1/31/2005	10	13.74	13.33
8	West	Boy	Tee	1/31/2005	11	11.44	10.94
9	West	Boy	Golf	1/31/2005	11	12.63	11.73
10	West	Boy	Fancy	1/31/2005	11	12.06	11.51
11	West	Girl	Tee	1/31/2005	15	13.42	13.29
12	West	Girl	Golf	1/31/2005	15	11.48	10.67

Flat table

"How many *Units* did we sell in each *Region* for every *Ship Date*?"

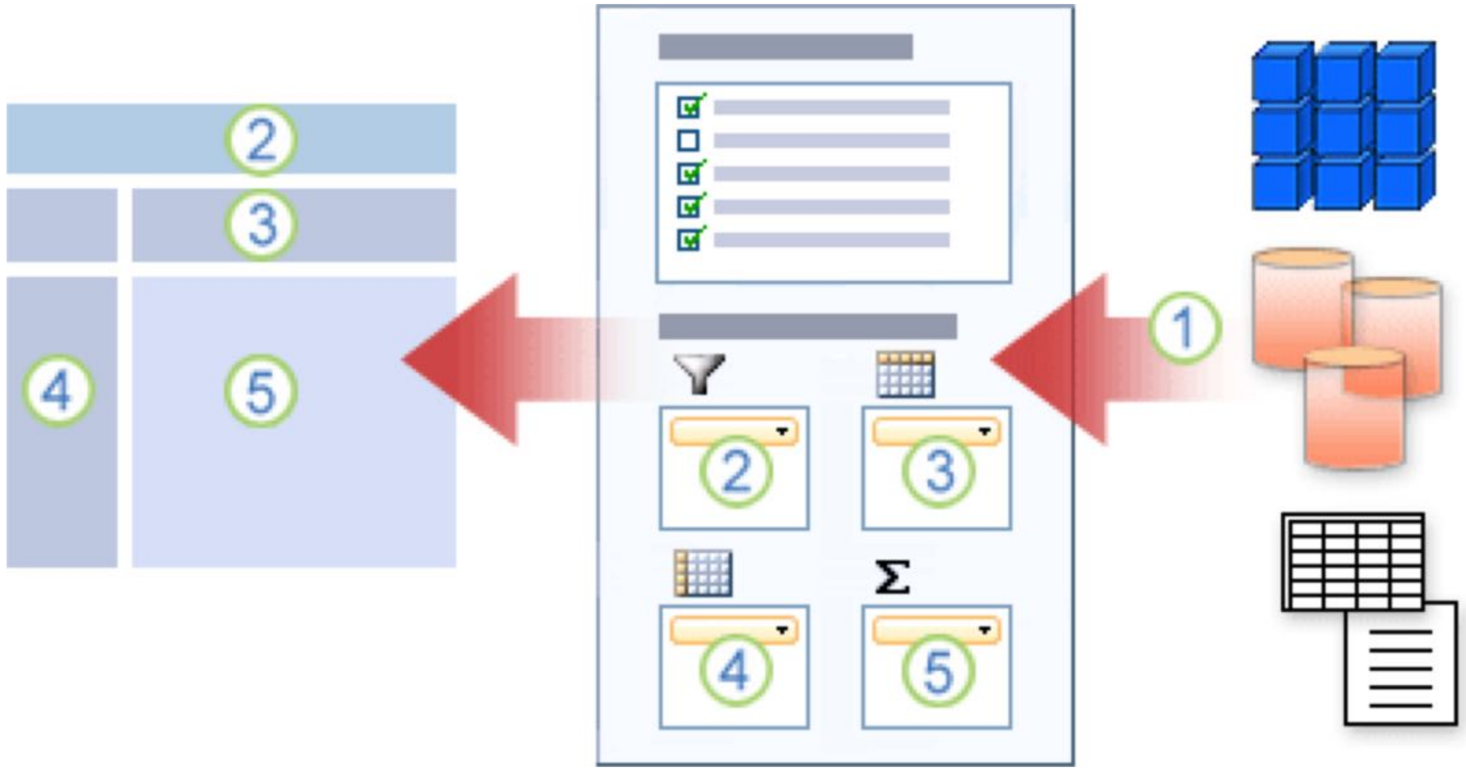


Sum of Units	Ship Date ▼					
Region ▼	1/31/2005	2/28/2005	3/31/2005	4/30/2005	5/31/2005	6/30/2005
East	66	80	102	116	127	125
North	96	117	138	151	154	156
South	123	141	157	178	191	202
West	78	97	117	136	150	157
(blank)						
Grand Total	363	435	514	581	622	640

facts
("measures")

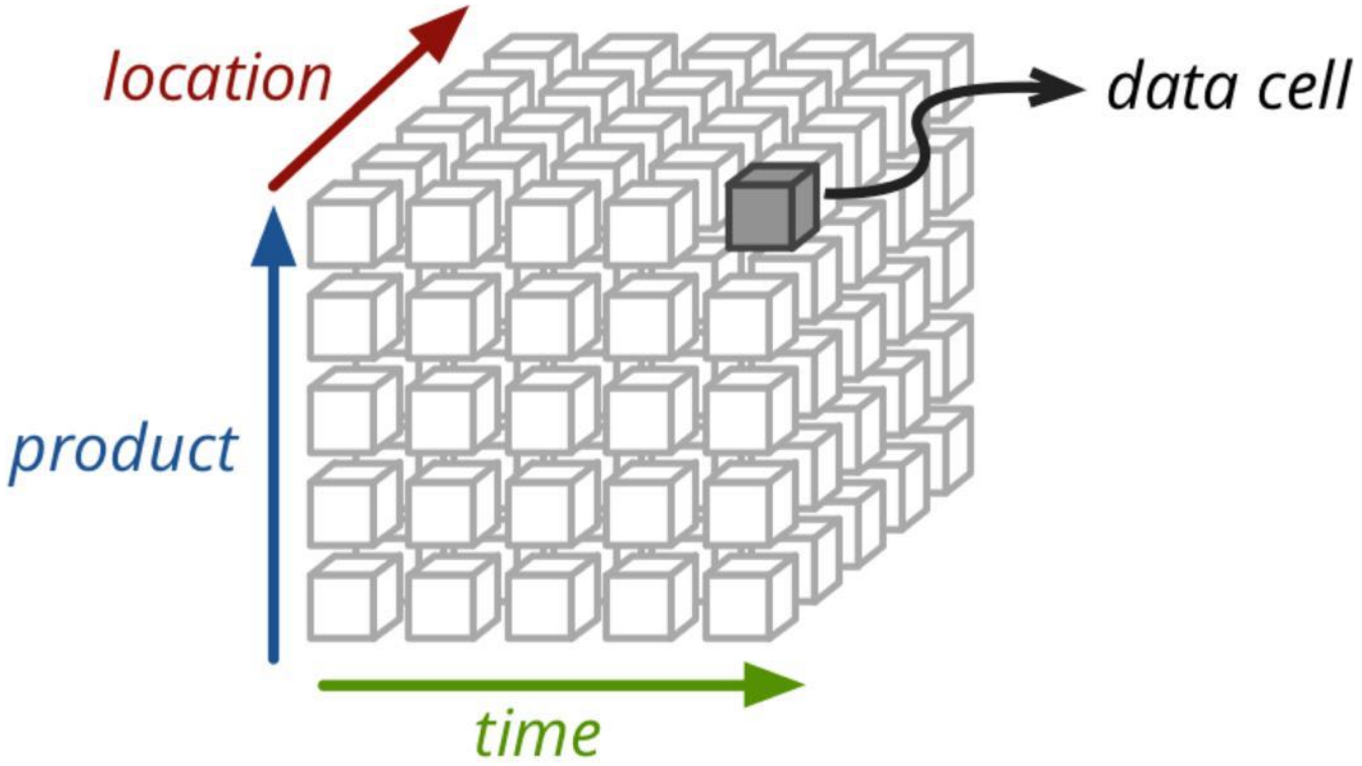
Source: http://en.wikipedia.org/wiki/Pivot_table

Pivot tables and OLAP: Online Analytical Processing



Source: https://support.office.com/en-au/article/Create-and-change-the-field-layout-in-a-PivotTable-or-PivotChart-report-122f718b-0d61-44ff-ab34-ce49373cf2ce

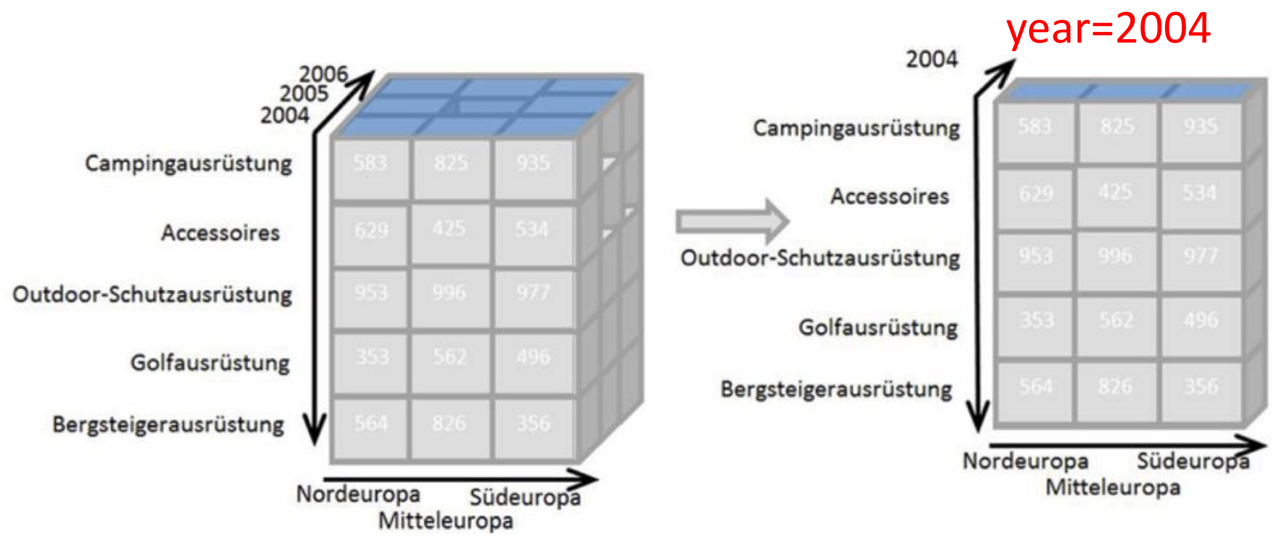
OLAP cube: dimensions vs. measures

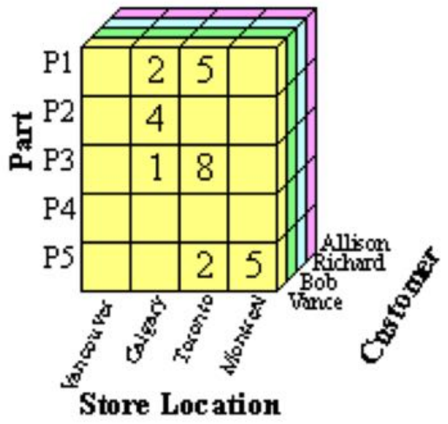


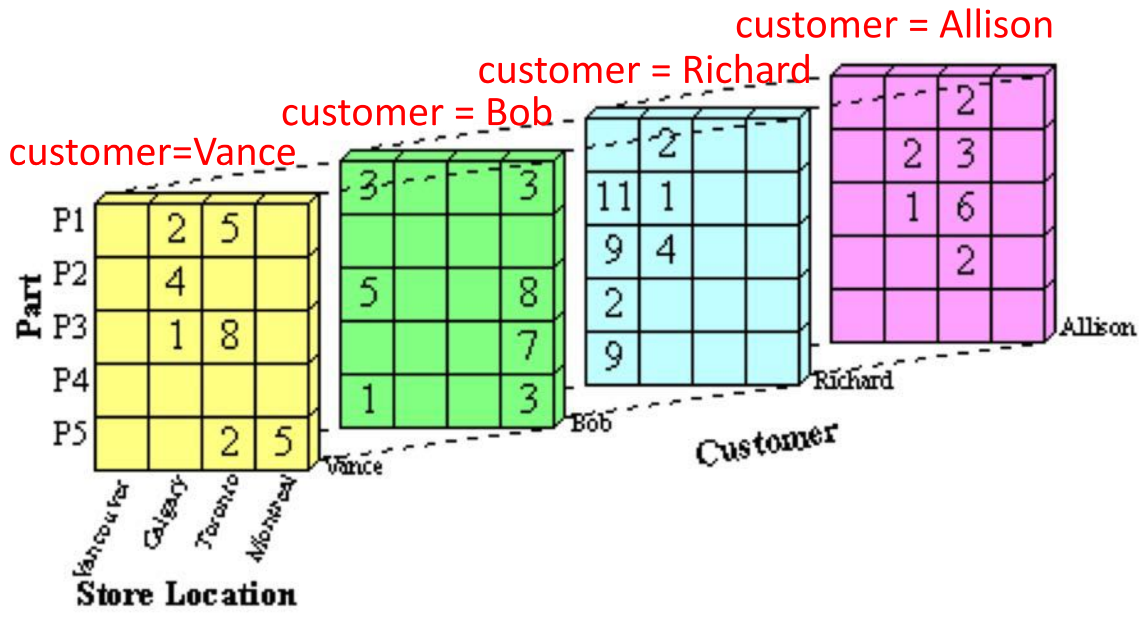
Source: https://pythonhosted.org/cubes/introduction.html

OLAP: slice

Slice: choosing a single value for one of its dimensions (thus reducing the dimensions)
= equality selection





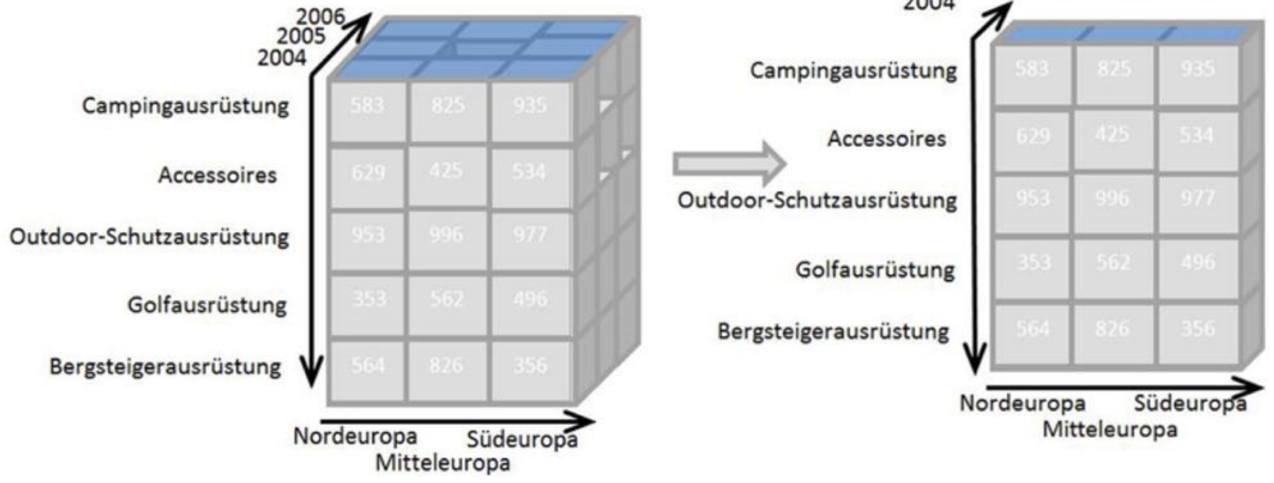


Source: http://en.wikipedia.org/wiki/OLAP_cube , http://www2.cs.uregina.ca/~dbd/cs831/notes/dcubes/dcubes.html

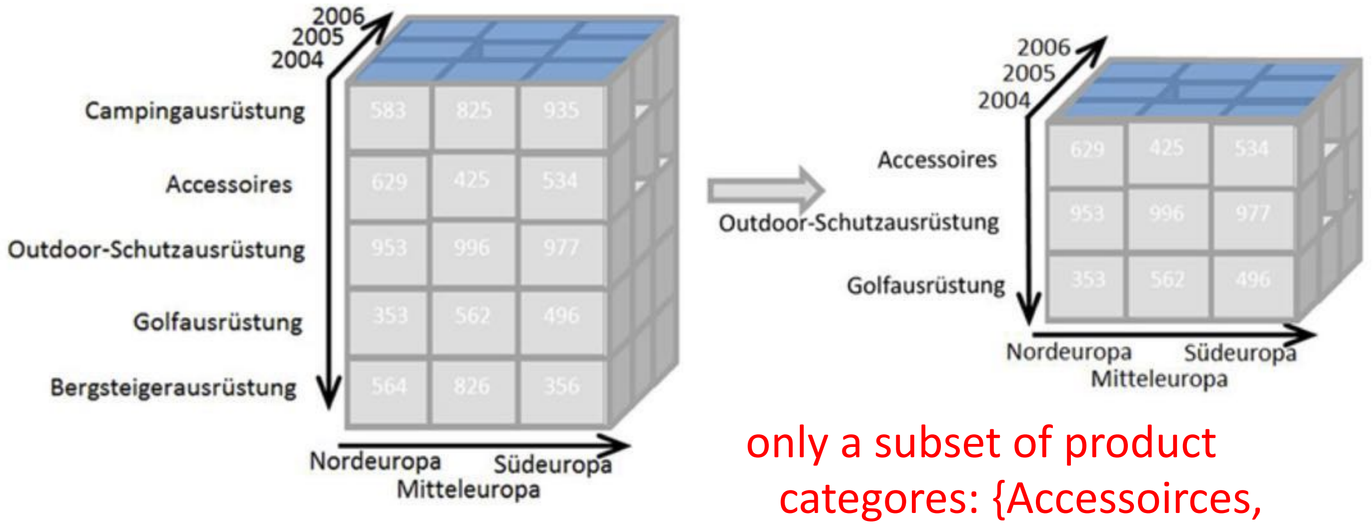
OLAP: slice & dice

Slice: choosing a single value for one of its dimensions (thus reducing the dimensions)
= equality selection

year=2004



Dice: pick specific values of multiple dimensions
= range selection

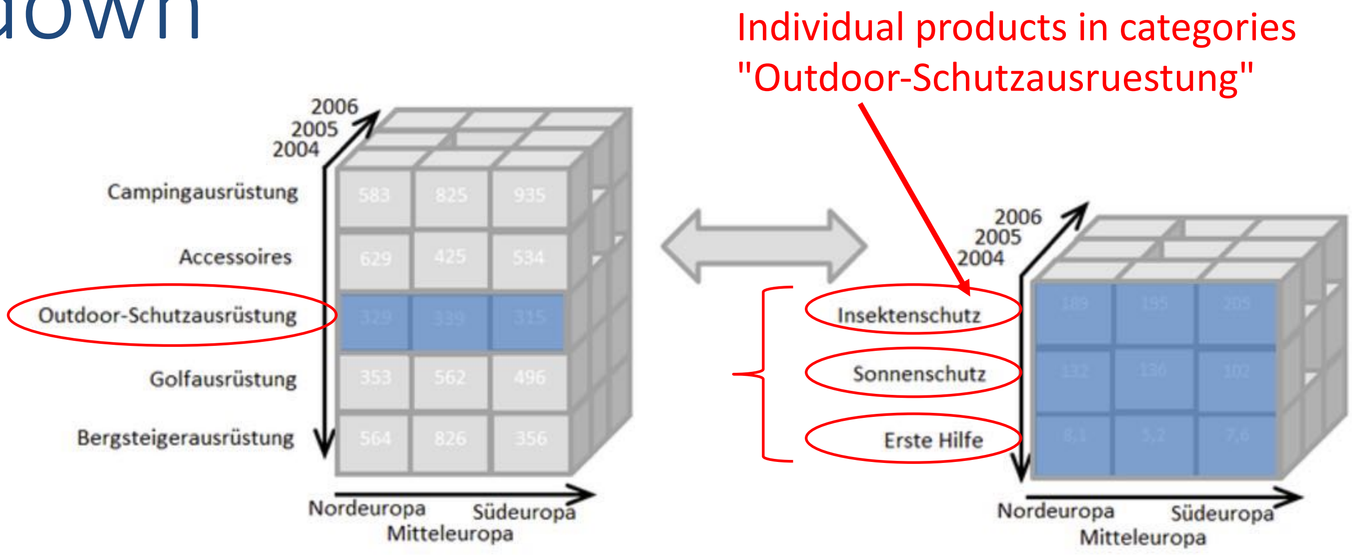


only a subset of product categores: {Accessoirces, Outdoor-Schutzausrüstung, Golfausrüstung}

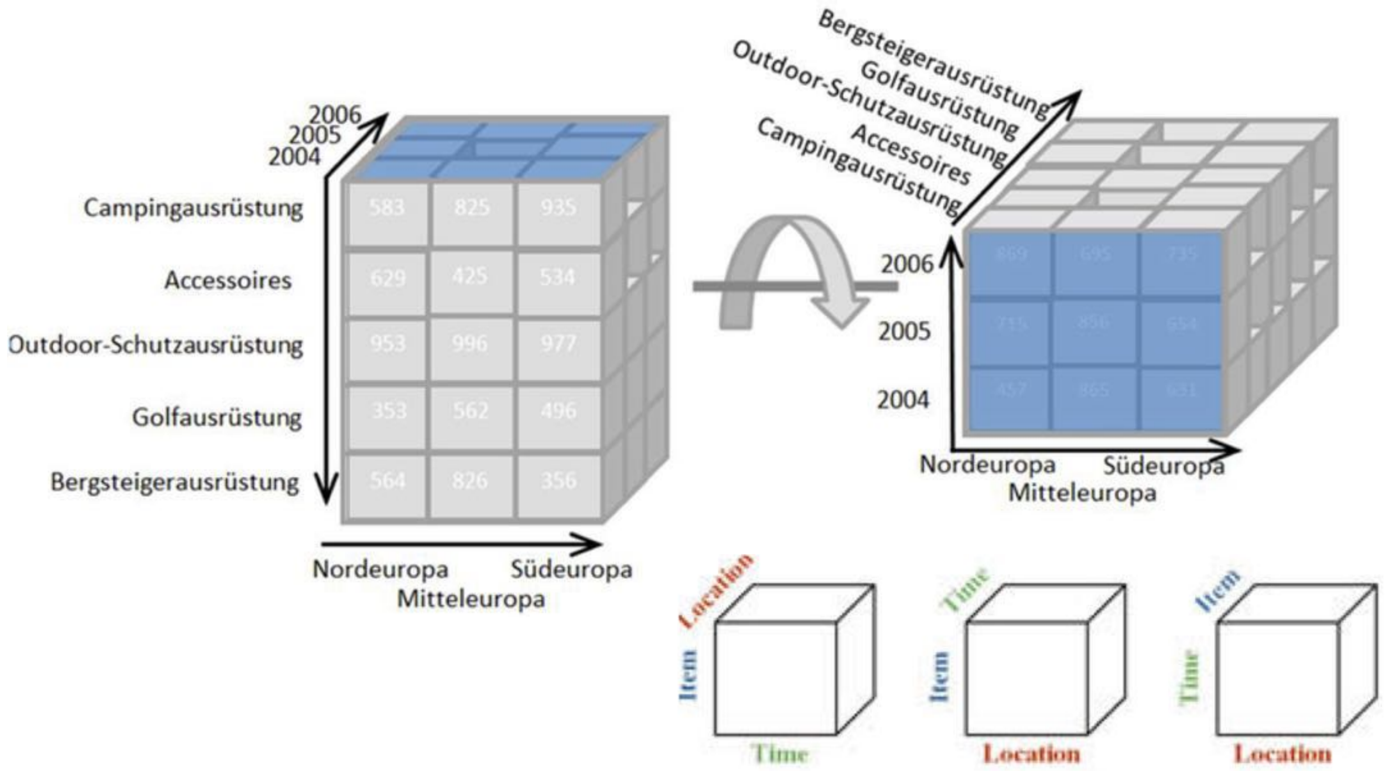
Source: http://en.wikipedia.org/wiki/OLAP_cube

OLAP: roll-up/drill-down

Roll-up/drill-down: navigate among levels of data ranging from the most summarized (up) to the most detailed (down).

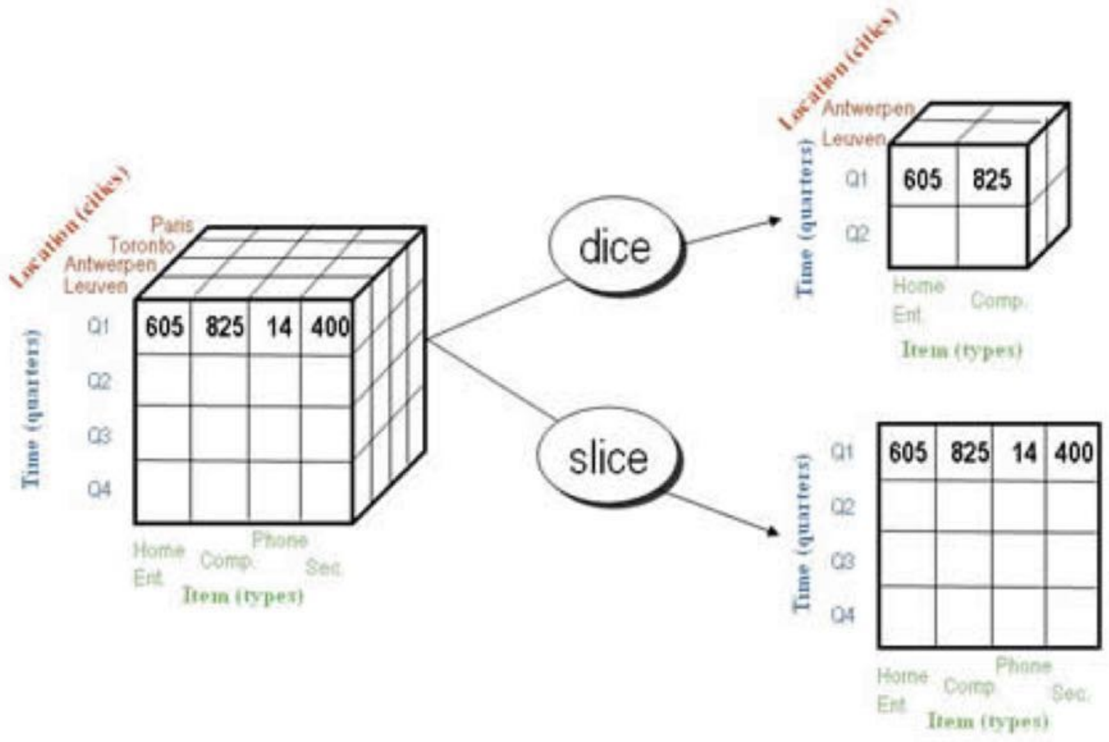


Pivot: rotate the cube in space to see its various faces.



Source: http://en.wikipedia.org/wiki/OLAP_cube

OLAP: slice & dice



Slice:

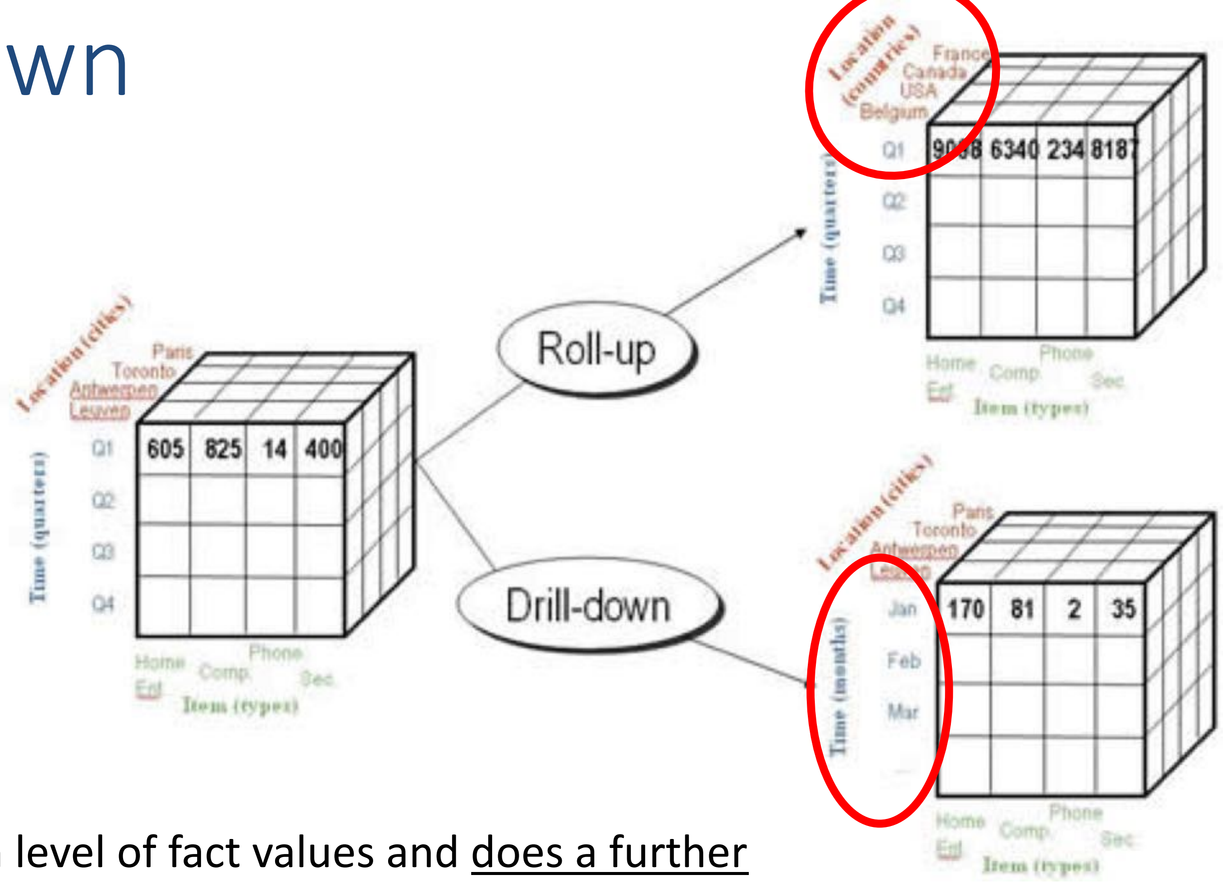
- Performs a selection on one dimension of the given cube, resulting in a sub-cube.
- Reduces the dimensionality of the cubes.
- Sets one or more dimensions to specific values and keeps a subset of dimensions for selected values.

Dice:

- Define a sub-cube by performing a selection of one or more dimensions.
- Refers to range select condition on one dimension, or to select condition on more than one dimension.
- Reduces the number of member values of one or more dimensions.

Source: http://cci.drexel.edu/faculty/song/courses/info%20607/tutorial_OLAP/operations.htm (recommended)

OLAP: roll-up / drill-down



Roll-up:

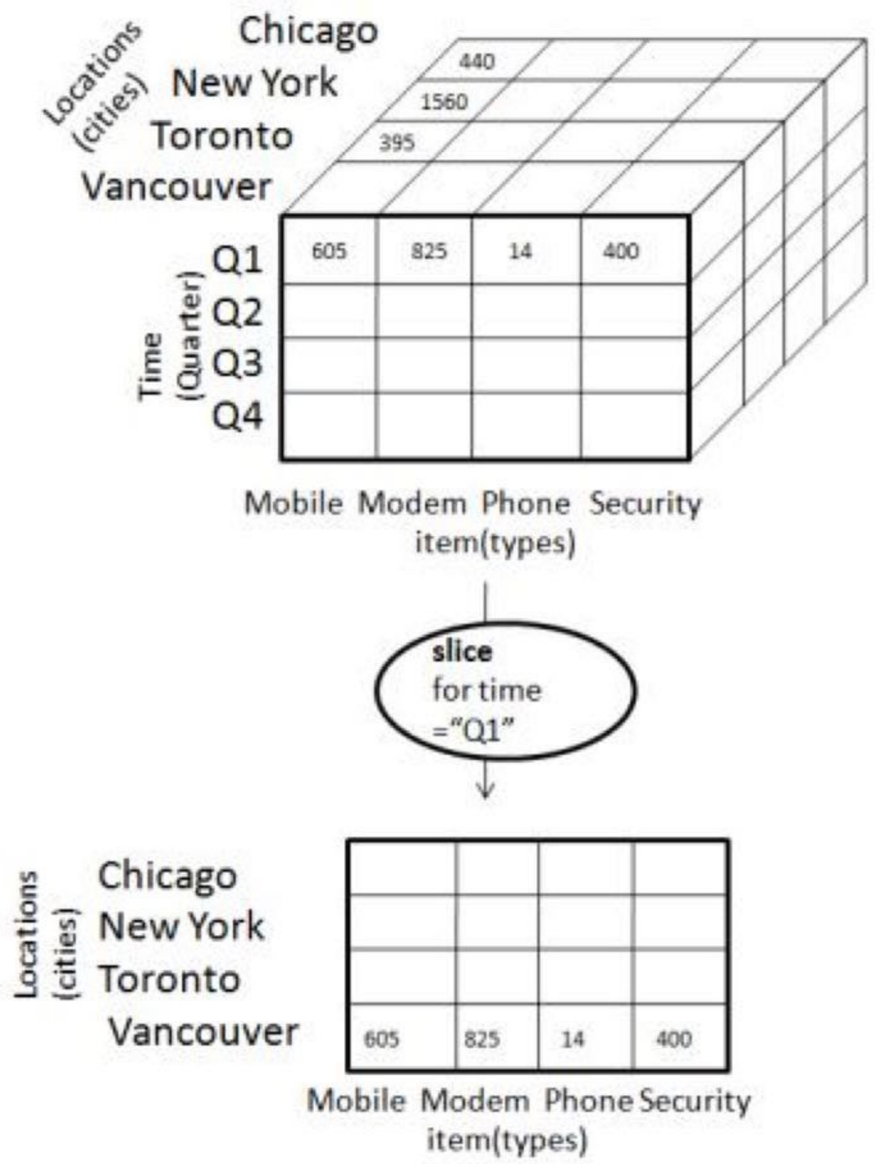
- Takes the current aggregation level of fact values and <u>does a further aggregation</u> on one or more of the dimensions.
- Equivalent to doing <u>GROUP BY</u> to this dimension by using attribute hierarchy.
- Decreases a number of dimensions - removes row headers.

Drill-down:

- Opposite of roll-up.
- Summarizes data at a lower level of a dimension hierarchy, thereby viewing data in a more specialized level within a dimension.
- <u>Increases number of dimensions</u> - adds new headers

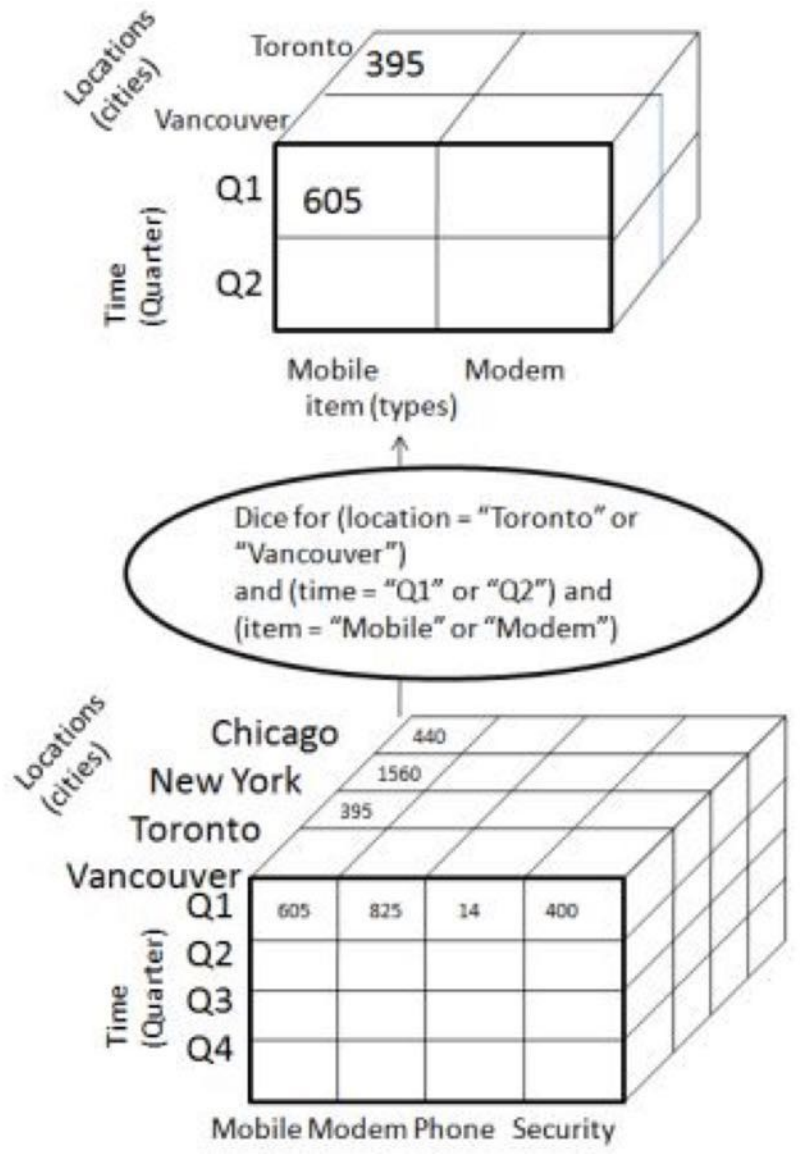
Source: http://cci.drexel.edu/faculty/song/courses/info%20607/tutorial_OLAP/operations.htm (recommended)

OLAP: slice



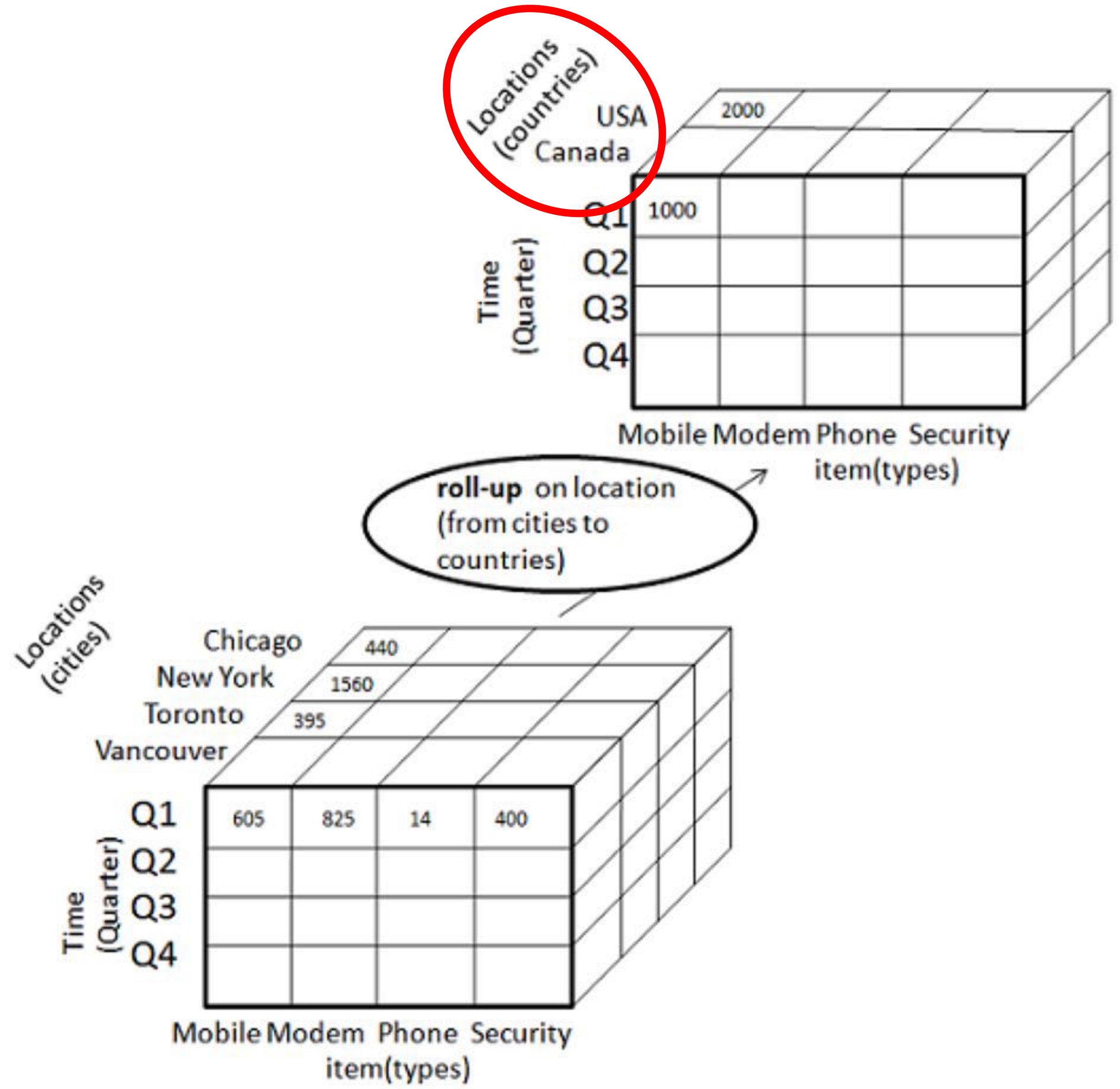
Source: http://www.tutorialspoint.com/dwh/dwh_olap.htm

OLAP: dice



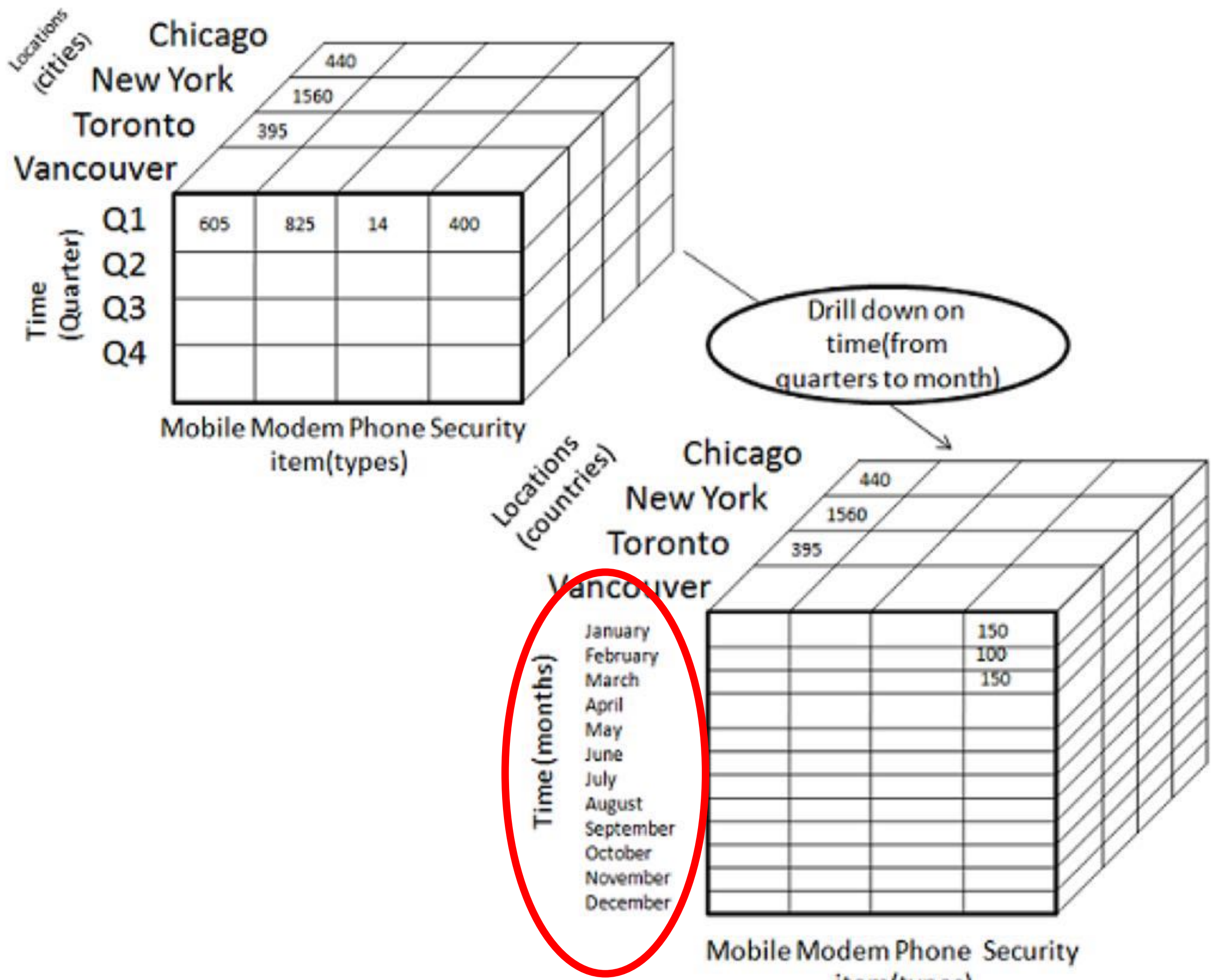
Source: http://www.tutorialspoint.com/dwh/dwh_olap.htm

OLAP: roll-up



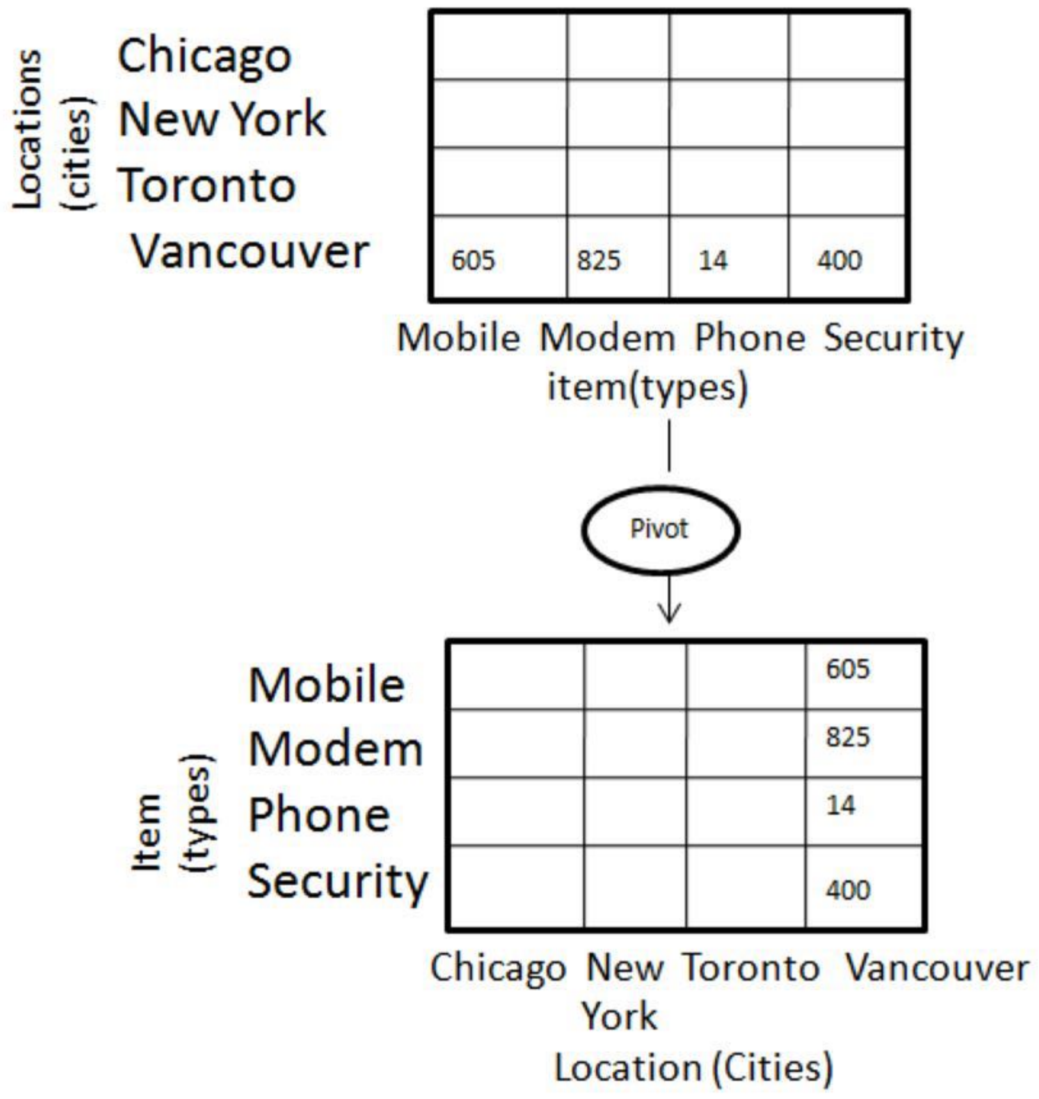
Source: http://www.tutorialspoint.com/dwh/dwh_olap.htm

OLAP: drill-down



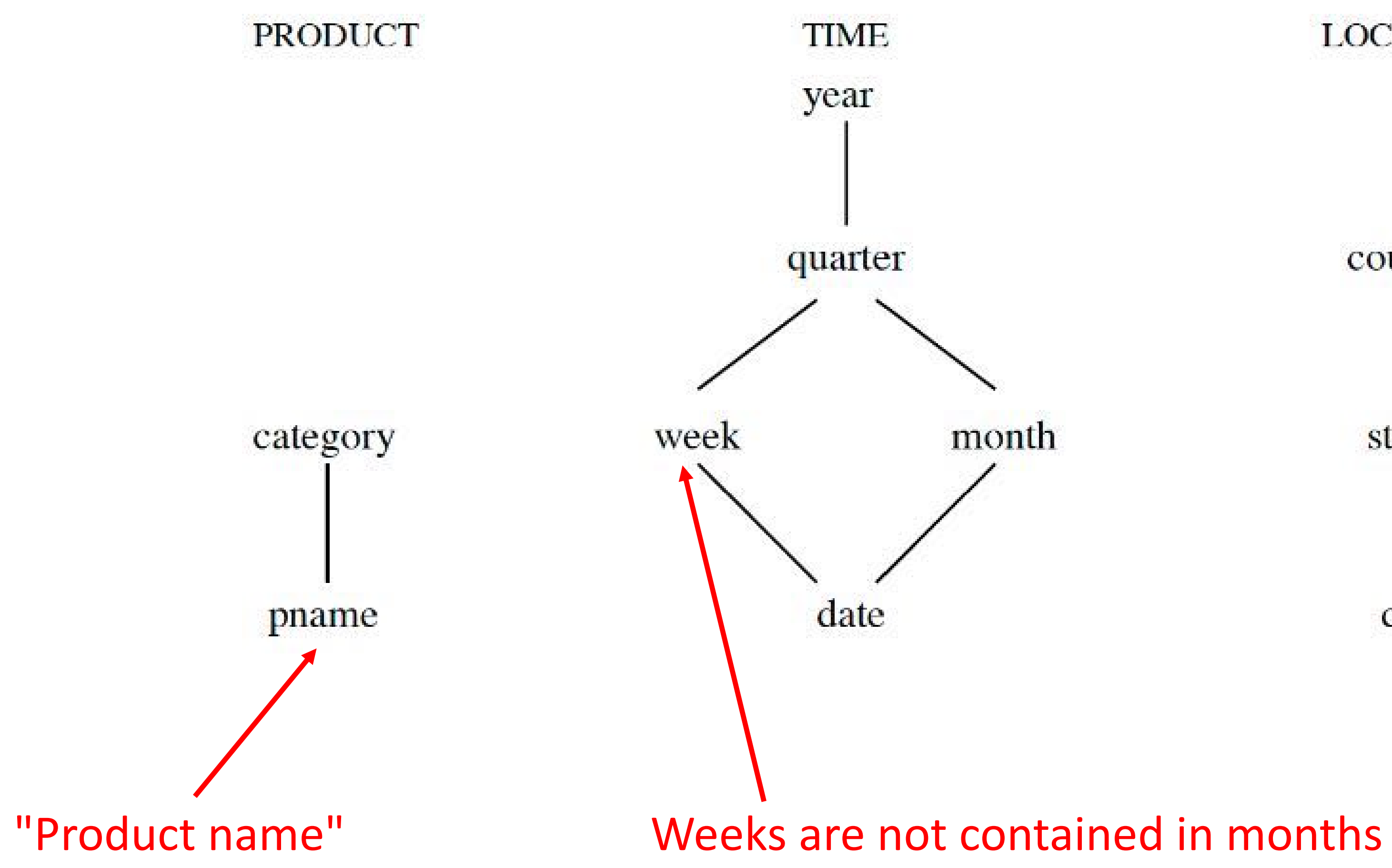
Source: http://www.tutorialspoint.com/dwh/dwh_olap.htm

OLAP: pivot



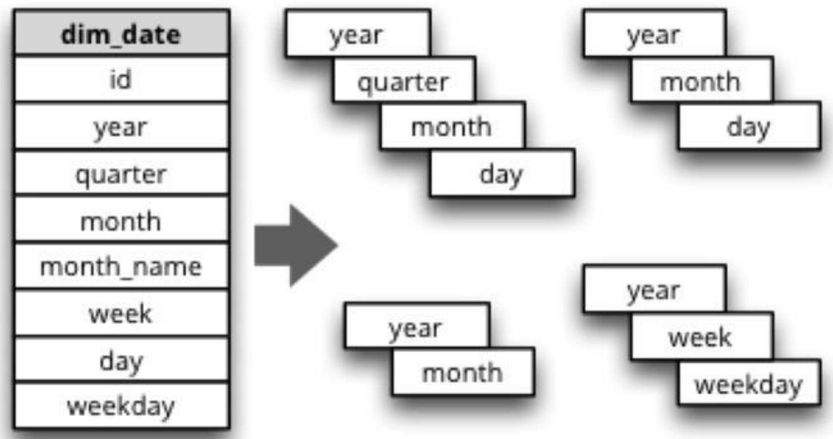
Source: http://www.tutorialspoint.com/dwh/dwh_olap.htm

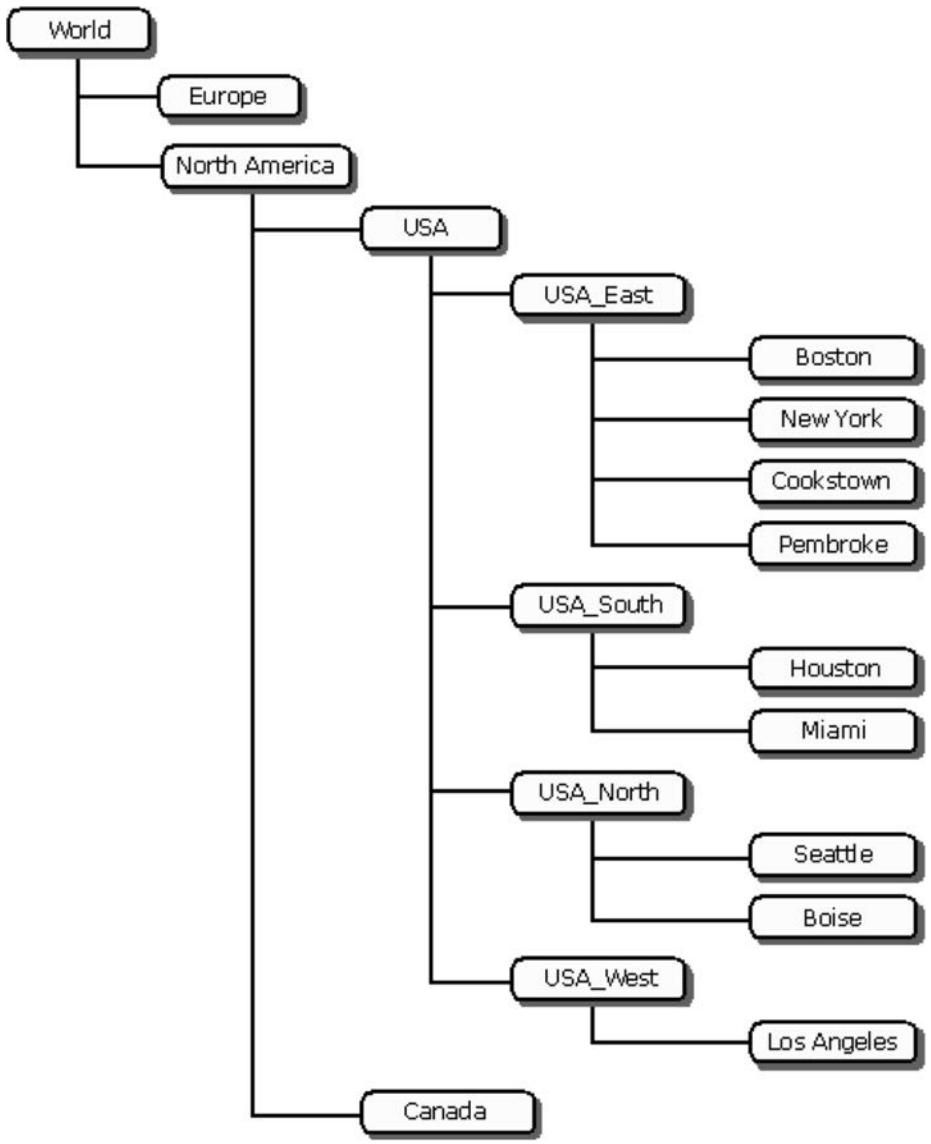
OLAP: Dimension hierarchies



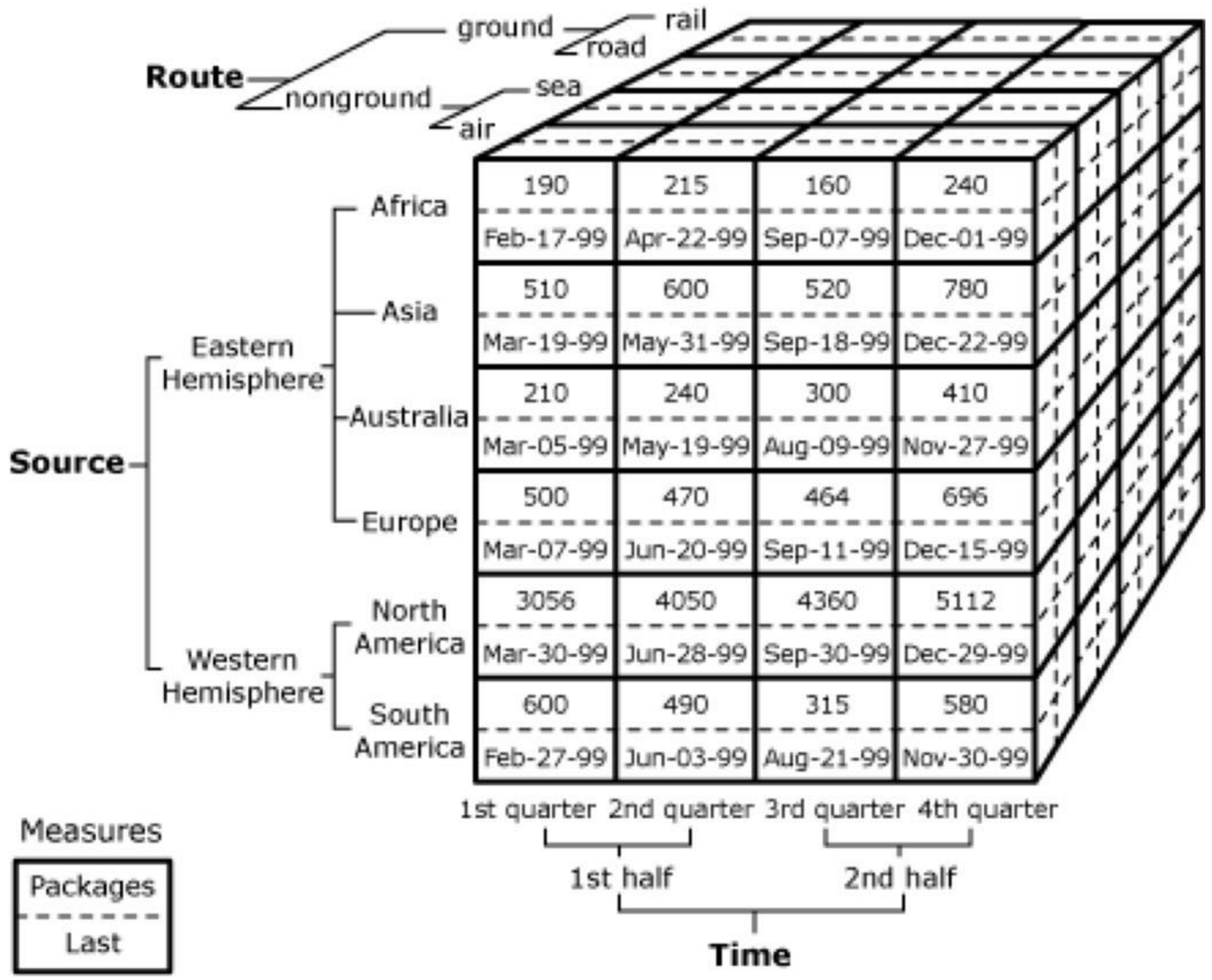
Source: Fig 23.4 from Ramakrishnan, Gehrke: Database management systems, 2nd ed, 2000

OLAP: Dimension hierarchies



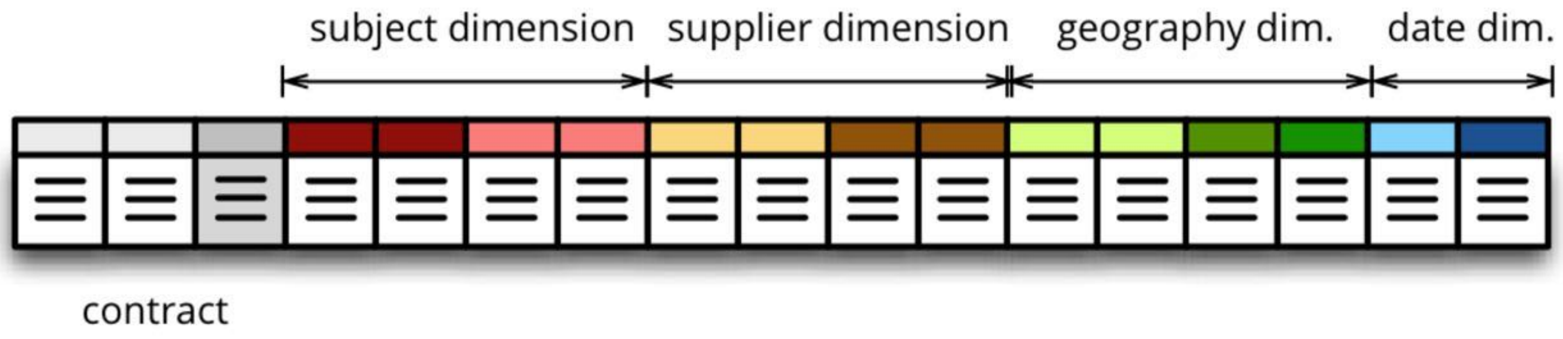


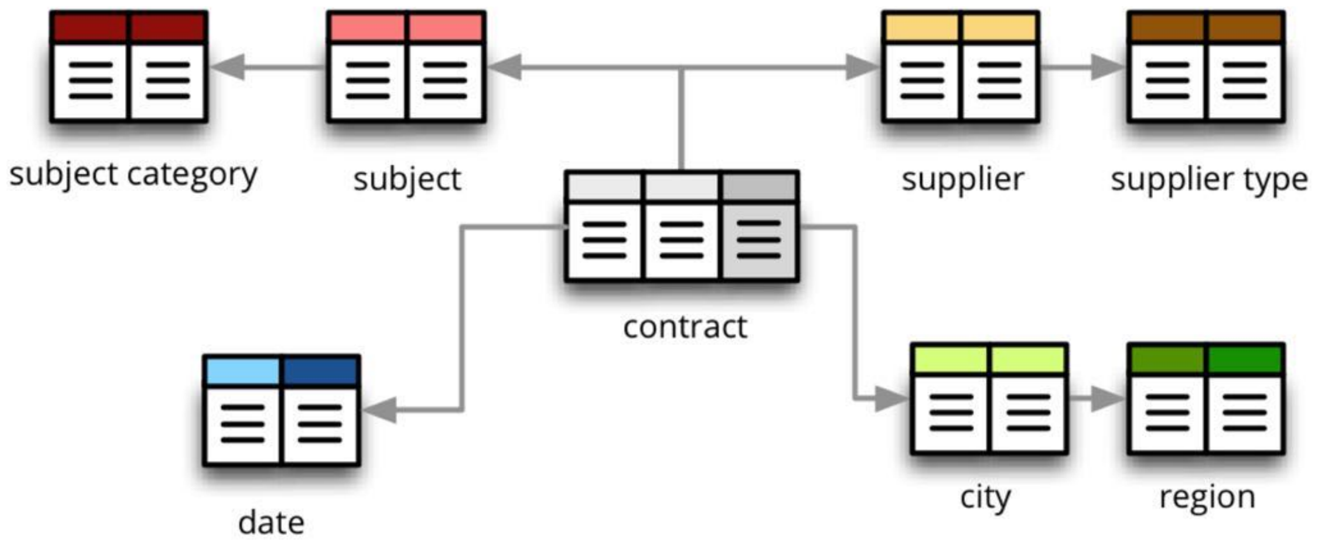
Source: https://pythonhosted.org/cubes/schemas.html, http://msdn.microsoft.com/en-us/library/windows/desktop/ms711518%28v=vs.85%29.aspx



		1st quarter	2nd quarter	3rd quarter	4th quarter
Eastern Hemisphere	Africa	190 / Feb-17-99	215 / Apr-22-99	160 / Sep-07-99	240 / Dec-01-99
	Asia	510 / Mar-19-99	600 / May-31-99	520 / Sep-18-99	780 / Dec-22-99
	Australia	210 / Mar-05-99	240 / May-19-99	300 / Aug-09-99	410 / Nov-27-99
	Europe	500 / Mar-07-99	470 / Jun-20-99	464 / Sep-11-99	696 / Dec-15-99
Western Hemisphere	North America	3056 / Mar-30-99	4050 / Jun-28-99	4360 / Sep-30-99	5112 / Dec-29-99
	South America	600 / Feb-27-99	490 / Jun-03-99	315 / Aug-21-99	580 / Nov-30-99

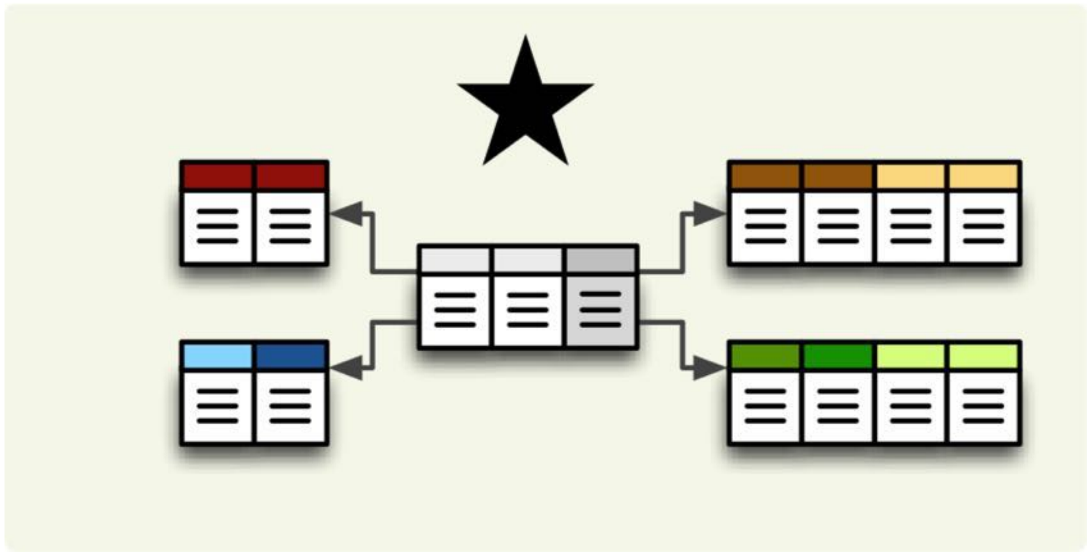
Source: http://technet.microsoft.com/en-us/library/ms174587%28v=sql.110%29.aspx



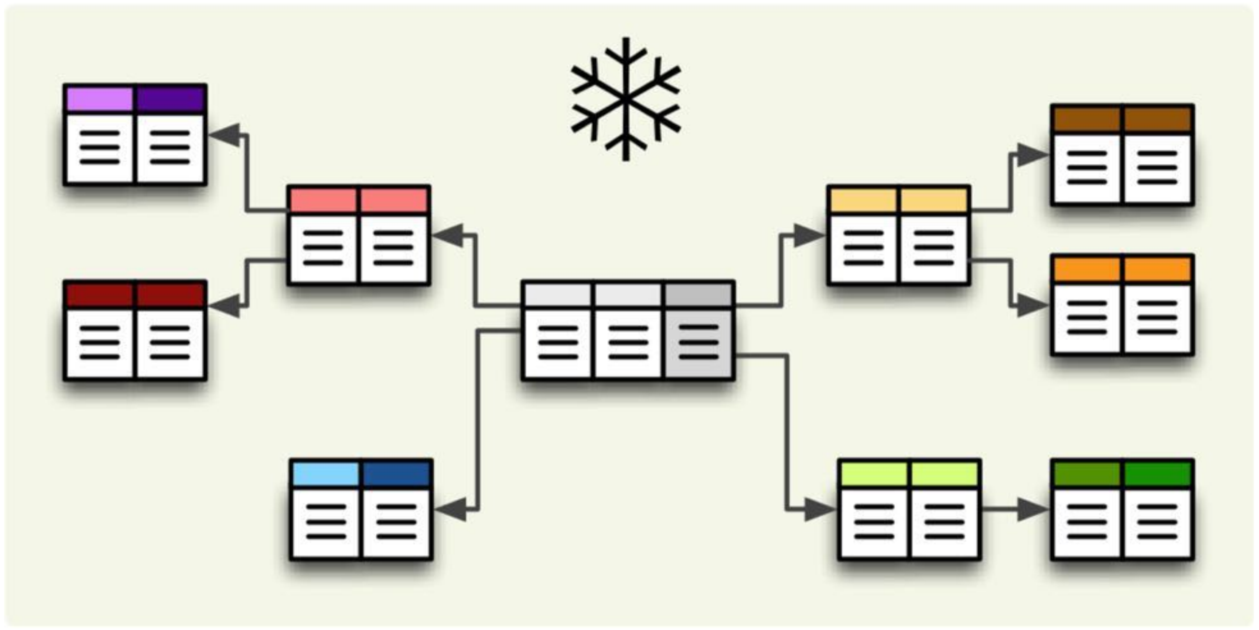


Source: http://cubes.readthedocs.org/en/latest/backends/sql.html

Star schema



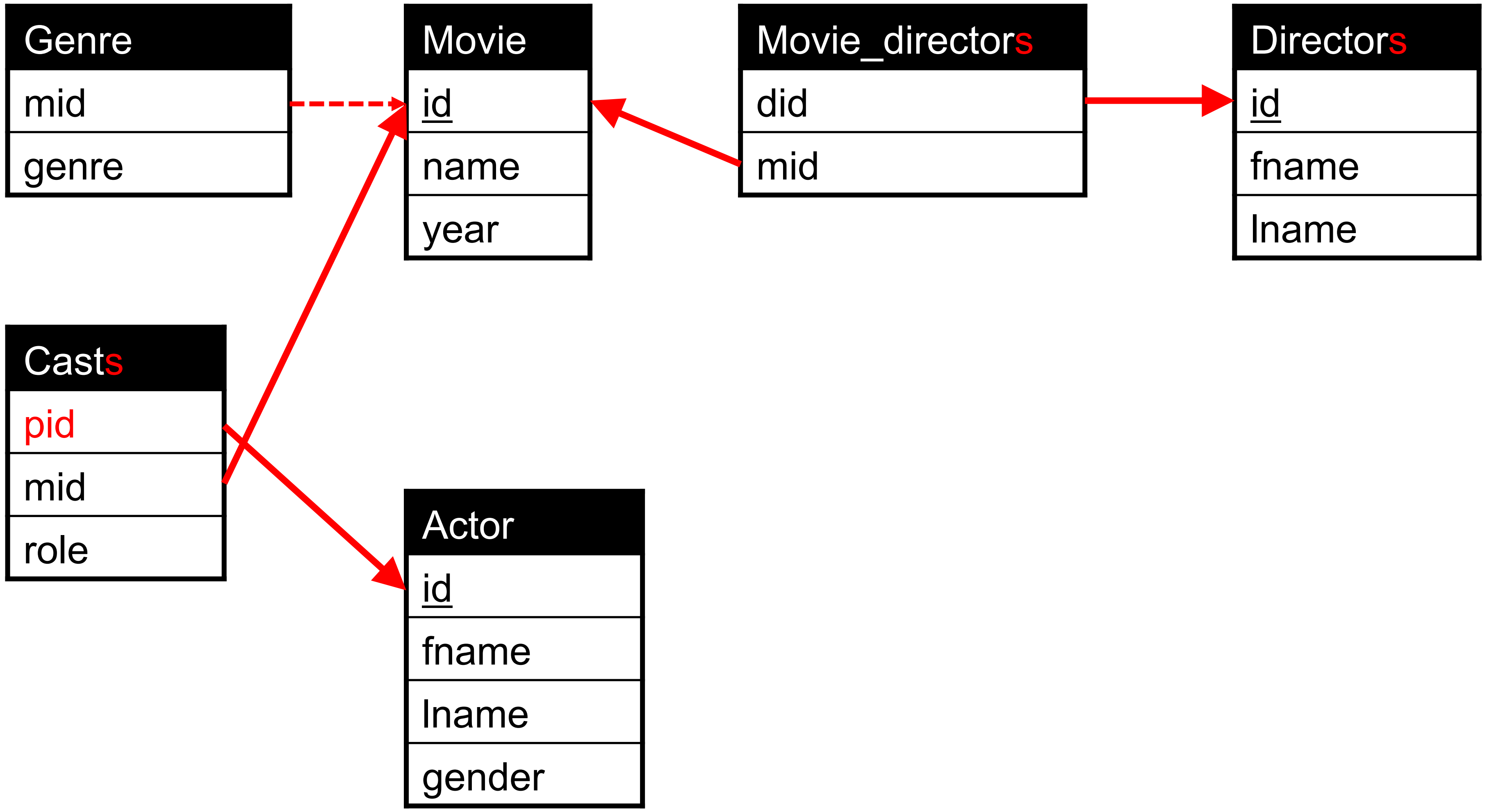
Snowflake schema



Source: http://cubes.readthedocs.org/en/latest/backends/sql.html

Practice

Big IMDB schema (Postgres)



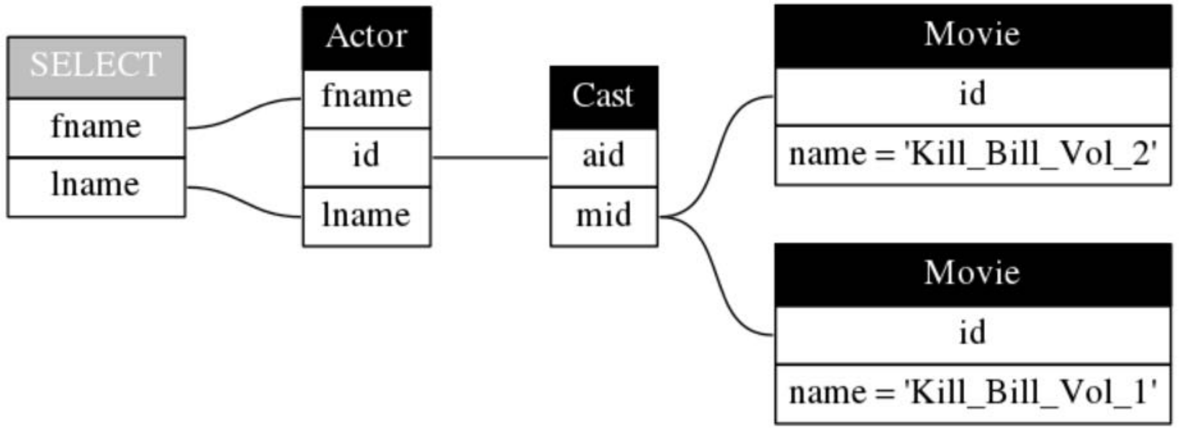
Quiz

IMDB (postgres)

Find the first/last names of all actors who appeared in both of the following movies: Kill Bill: Vol. 1 and Kill Bill: Vol. 2.

```
SELECT DISTINCT A.fname, A.lname
FROM     Actor A, Casts C, Movie M1, Movie M2
WHERE    M1.name = 'Kill Bill: Vol. 1'
    and  M2.name = 'Kill Bill: Vol. 2'
    and  M1.id = C.mid
    and  M2.id = C.mid
    and  C.pid = A.id
```

☹



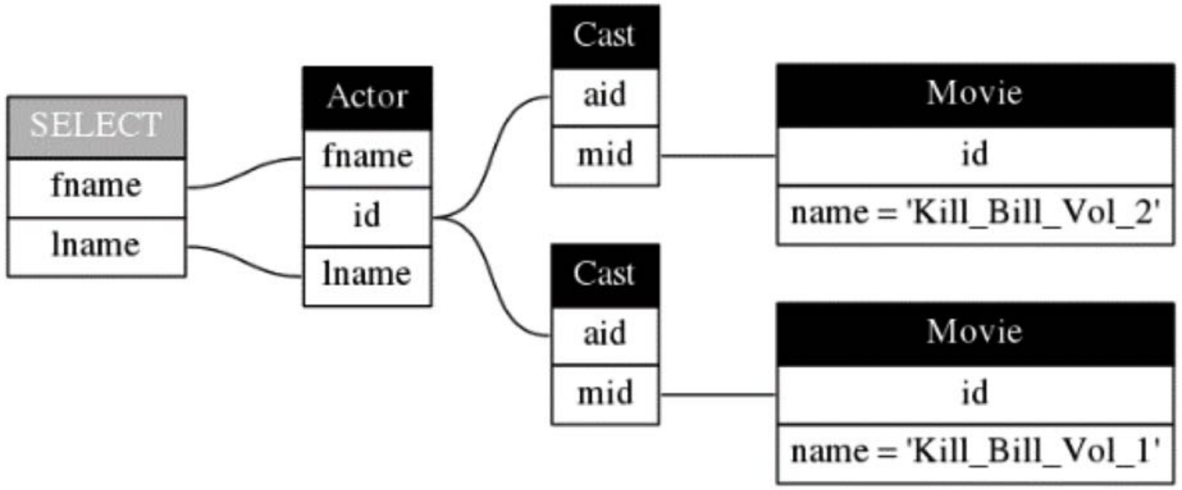
Picture Source: http://queryviz.com/online

Quiz

IMDB (postgres)

Find the first/last names of all actors who appeared in both of the following movies: Kill Bill: Vol. 1 and Kill Bill: Vol. 2.

```
SELECT DISTINCT A.fname, A.lname
FROM    Actor A, Casts C, Movie M1, Movie M2, Casts C2
WHERE   M1.name = 'Kill Bill: Vol. 1'
   and  M2.name = 'Kill Bill: Vol. 2'
   and  M1.id = C.mid
   and  M2.id = C2.mid
   and  C.pid = A.id
   and  C2.pid = A.id
```



Picture Source: http://queryviz.com/online

Quiz

IMDB (postgres)

Find the first/last names of all actors who appeared in both of the following movies: Kill Bill: Vol. 1 and Kill Bill: Vol. 2.

```
SELECT  A.id, A.lname, A.fname,
FROM    actor A, cast C, movie M
WHERE   M.id = C.mid
   AND  A.id = C.pid
   AND  (M.name = 'Kill Bill: Vol. 1'
        OR M.name = 'Kill Bill: Vol. 2')
GROUP BY A.id, A.lname, A.fname
HAVING  count(M.id) > 1
```

Quiz

IMDB (postgres)

Find the first/last names of all actors who appeared in both of the following movies: Kill Bill: Vol. 1 and Kill Bill: Vol. 2.

```
SELECT A.id, A.lname, A.fname,
FROM   actor A, cast C, movie M
WHERE  M.id = C.mid
   AND    A.id = C.pid
   AND    (M.name = 'Kill Bill: Vol. 1'
          OR M.name = 'Kill Bill: Vol. 2')
GROUP BY A.id, A.lname, A.fname
HAVING  count(M.id) > 1
```

What if an actor played two roles in Kill Bill 1?

More on WITH

Recall: Witnesses: with aggregates per group (8/8)

Second: How to get the product that is sold with max sales?

Purchase

Product	Price	Quantity
Bagel	3	20
Bagel	2	20
Banana	1	50
Banana	2	10
Banana	4	10

Product	sales
Banana	70

```
SELECT     product, sum(quantity) as sales
FROM       Purchase
GROUP BY product
HAVING     sum(quantity) =
           (SELECT max (Q)
           FROM  (SELECT      sum(quantity) Q
                 FROM         Purchase
                 GROUP BY product) X        )
```

WITH clause

```
WITH X AS
            (SELECT   product, SUM(quantity) sales
            FROM      Purchase
            GROUP BY product)
```

```
SELECT      product, sum(quantity) as sales
FROM        Purchase
GROUP BY product
HAVING      sum(quantity) =
            (SELECT max (Q)
            FROM   (SELECT     sum(quantity) Q
                    FROM       Purchase
                    GROUP BY product) X       )
```

WITH clause

```
WITH X AS
                (SELECT   product, SUM(quantity) sales
                 FROM      Purchase
                 GROUP BY product)
SELECT          *
FROM            X
WHERE
```

```
SELECT          product, sum(quantity) as sales
FROM            Purchase
GROUP BY product
HAVING          sum(quantity) =
                (SELECT max (Q)
                 FROM    (SELECT    sum(quantity) Q
                          FROM       Purchase
                          GROUP BY product) X       )
```

WITH clause

```
WITH X AS
                (SELECT    product, SUM(quantity) sales
                FROM       Purchase
                GROUP BY product)
SELECT          *
FROM            X
WHERE           sales =
                (SELECT    MAX (sales)
                FROM       X)
```

```
SELECT      product, sum(quantity) as sales
FROM        Purchase
GROUP BY product
HAVING      sum(quantity) =
            (SELECT max (Q)
            FROM    (SELECT      sum(quantity) Q
                    FROM         Purchase
                    GROUP BY product) X          )
```

WITH clause

```
WITH X AS
            (SELECT   product, SUM(quantity) sales
            FROM      Purchase
            GROUP BY product),
Y AS
            (SELECT   MAX (sales) maxs
            FROM      X)
SELECT      *
FROM        X
WHERE       sales = (SELECT maxs FROM Y))
```

```
SELECT      product, sum(quantity) as sales
FROM        Purchase
GROUP BY product
HAVING      sum(quantity) =
            (SELECT max (Q)
            FROM (SELECT    sum(quantity) Q
                  FROM      Purchase
                  GROUP BY product) X     )
```

More Practice

1. What does this query return?

Actor

id	name	gender
1	Alice	f
2	Bob	m

Casts

aid	mid	role
1	1	role 1
2	1	role 2
2	1	role 3

Movie

id	name
1	Kill Bill
2	Kill Bill

```
SELECT movie.name
FROM   movie, casts, actor
WHERE  casts.aid = actor.id
  AND  movie.id = casts.mid
  AND  actor.id NOT IN
       (SELECT a2.id
        FROM   actor a2
        WHERE  a2.gender = 'm')
```

⇨ ?

1. What does this query return?

Actor

id	name	gender
1	Alice	f
2	Bob	m

Casts

aid	mid	role
1	1	role 1
2	1	role 2
2	1	role 3

Movie

id	name
1	Kill Bill
2	Kill Bill

```
SELECT  movie.name
FROM    movie, casts, actor
WHERE   casts.aid = actor.id
  AND   movie.id = casts.mid
  AND   actor.id NOT IN
        (SELECT  a2.id
         FROM    actor a2
         WHERE   a2.gender = 'm')
```

name
Kill Bill

1. The Full join (aka "Universal Relation")

Select *

id	name	gender	aid	mid	role	id	name
1	Alice	f	1	1	role 1	1	Kill Bill
2	Bob	m	2	1	role 2	1	Kill Bill
2	Bob	m	2	1	role 3	1	Kill Bill

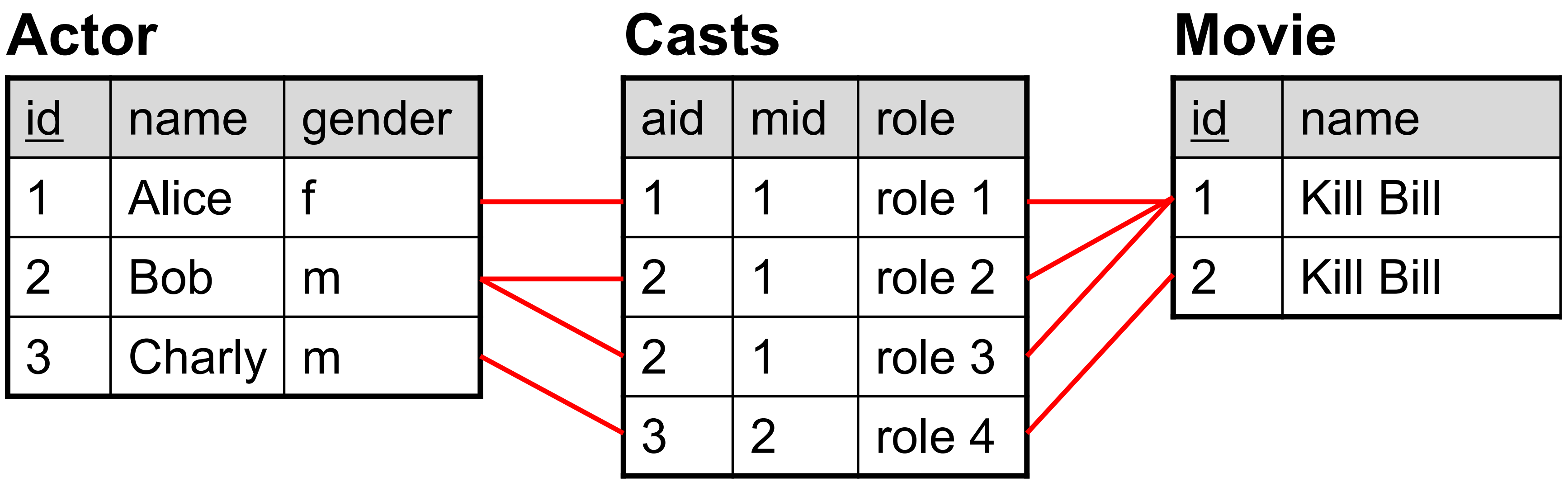
```
SELECT  *
FROM    movie, casts, actor
WHERE   casts.aid = actor.id
 AND    movie.id = casts.mid
```

```
SELECT  a2.id
FROM    actor a2
WHERE   a2.gender = 'm'
```

id
2

More information about the "universal relation": https://en.wikipedia.org/wiki/Universal_relation_assumption

2. How to get the number of casts for each movie?



Actor

id	name	gender
1	Alice	f
2	Bob	m
3	Charly	m

Casts

aid	mid	role
1	1	role 1
2	1	role 2
2	1	role 3
3	2	role 4

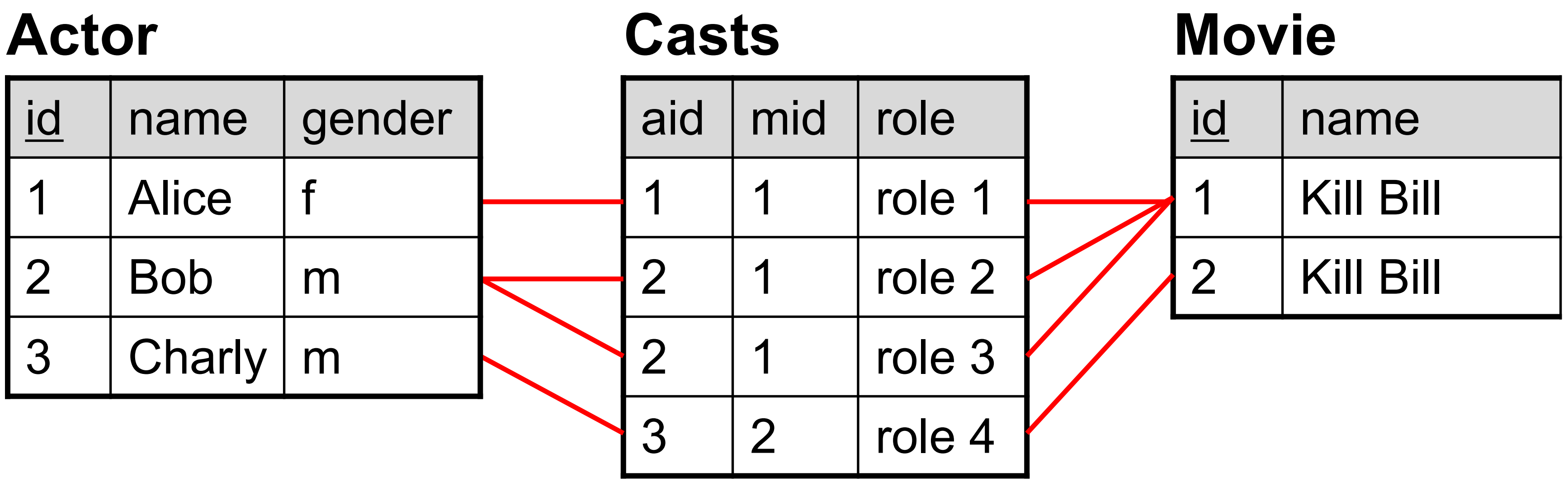
Movie

id	name
1	Kill Bill
2	Kill Bill

```
SELECT  m.name
FROM
WHERE
GROUP BY
```

?

2. How to get the number of casts for each movie?



Actor

<u>id</u>	name	gender
1	Alice	f
2	Bob	m
3	Charly	m

Casts

aid	mid	role
1	1	role 1
2	1	role 2
2	1	role 3
3	2	role 4

Movie

<u>id</u>	name
1	Kill Bill
2	Kill Bill

```
SELECT  m.name, count(c.aid)
FROM    movie m, casts c
WHERE   m.id = c.mid
GROUP BY m.name
```

name	(no name)
Kill Bill	4

2. How to get the number of casts for each movie?

Actor

id	name	gender
1	Alice	f
2	Bob	m
3	Charly	m

Casts

aid	mid	role
1	1	role 1
2	1	role 2
2	1	role 3
3	2	role 4

Movie

id	name
1	Kill Bill
2	Kill Bill

```
SELECT  m.name, count(c.aid)
FROM    movie m, casts c
WHERE   m.id = c.mid
GROUP BY m.id
```

$\Rightarrow$?

2. How to get the number of casts for each movie?

Actor

id	name	gender
1	Alice	f
2	Bob	m
3	Charly	m

Casts

aid	mid	role
1	1	role 1
2	1	role 2
2	1	role 3
3	2	role 4

Movie

id	name
1	Kill Bill
2	Kill Bill

```
SELECT  m.name, count(c.aid)
FROM    movie m, casts c
WHERE   m.id = c.mid
GROUP BY m.id
```

name	(no name)
Kill Bill	3
Kill Bill	1

Notice that this query gives an error on SQL server and used to give errors on some other databases. Now PostgreSQL can interpret the PK m.id -> m.name

2. How to get the number of casts for each movie?

Actor

id	name	gender
1	Alice	f
2	Bob	m
3	Charly	m

Casts

aid	mid	role
1	1	role 1
2	1	role 2
2	1	role 3
3	2	role 4

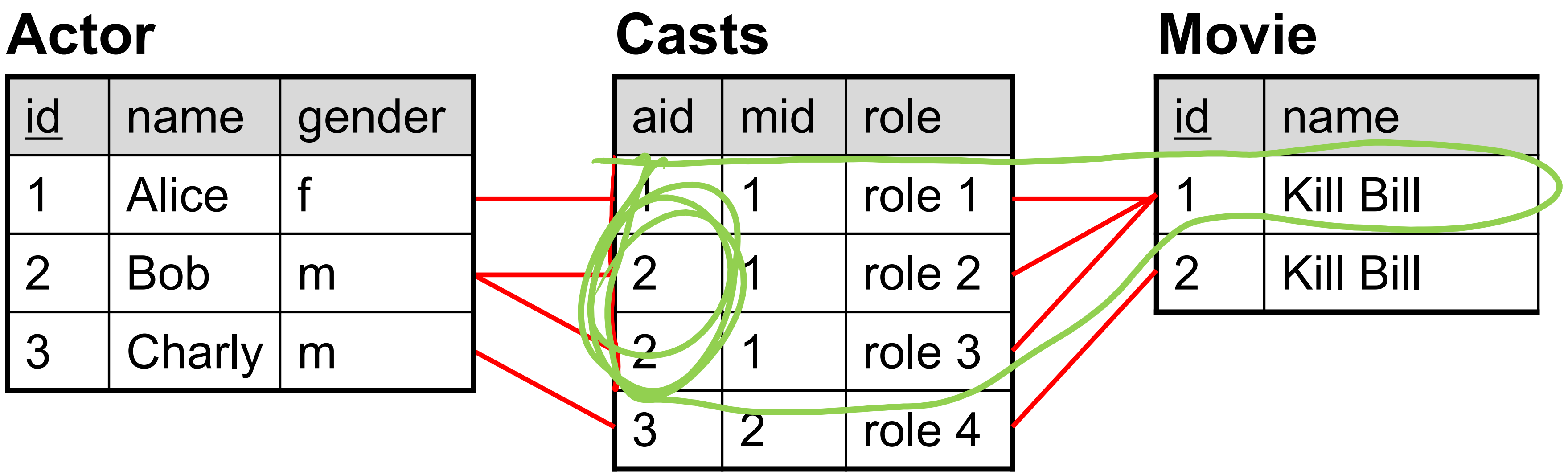
Movie

id	name
1	Kill Bill
2	Kill Bill

```
SELECT  m.name, count(c.aid)
FROM    movie m, casts c
WHERE   m.id = c.mid
GROUP BY m.id, m.name
```

name	(no name)
Kill Bill	3
Kill Bill	1

2. How to get the number of casts for each movie?



Actor

id	name	gender
1	Alice	f
2	Bob	m
3	Charly	m

Casts

aid	mid	role
1	1	role 1
2	1	role 2
2	1	role 3
3	2	role 4

Movie

id	name
1	Kill Bill
2	Kill Bill

```
SELECT  m.name, count(distinct c.aid)
FROM    movie m, casts c
WHERE   m.id = c.mid
GROUP BY m.id, m.name
```

name	(no name)
Kill Bill	2
Kill Bill	1

3. How to get the number of casts for '%Bill%'?

Actor

id	name	gender
1	Alice	f
2	Bob	m
3	Charly	m

Casts

aid	mid	role
1	1	role 1
2	1	role 2
2	1	role 3
3	2	role 4

Movie

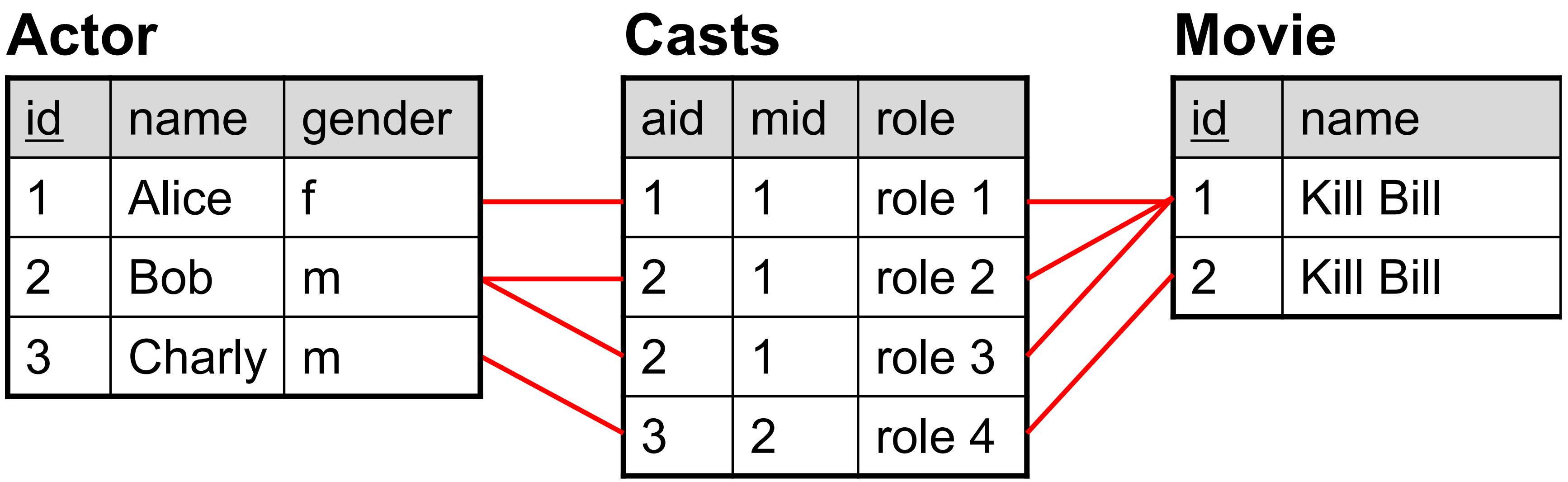
id	name
1	Kill Bill
2	Kill Bill

```
SELECT  m.name, count(distinct c.aid)
FROM    movie m, casts c
WHERE   m.id = c.mid

GROUP BY m.id, m.name
```

?

3. How to get the number of casts for '%Bill%'?



Actor

id	name	gender
1	Alice	f
2	Bob	m
3	Charly	m

Casts

aid	mid	role
1	1	role 1
2	1	role 2
2	1	role 3
3	2	role 4

Movie

id	name
1	Kill Bill
2	Kill Bill

```
SELECT  m.name, count(distinct c.aid)
FROM    movie m, casts c
WHERE   m.id = c.mid
   AND  m.name like '%Bill%'
GROUP BY m.id, m.name
```

Also possible but *not* recommended: selection in HAVING clause

4. How to get the number of casts for each actor?

Actor

<u>id</u>	name	gender
1	Alice	f
2	Bob	m
3	Charly	m

Casts

aid	mid	role
1	1	role 1
2	1	role 2
2	1	role 3
3	2	role 4

Movie

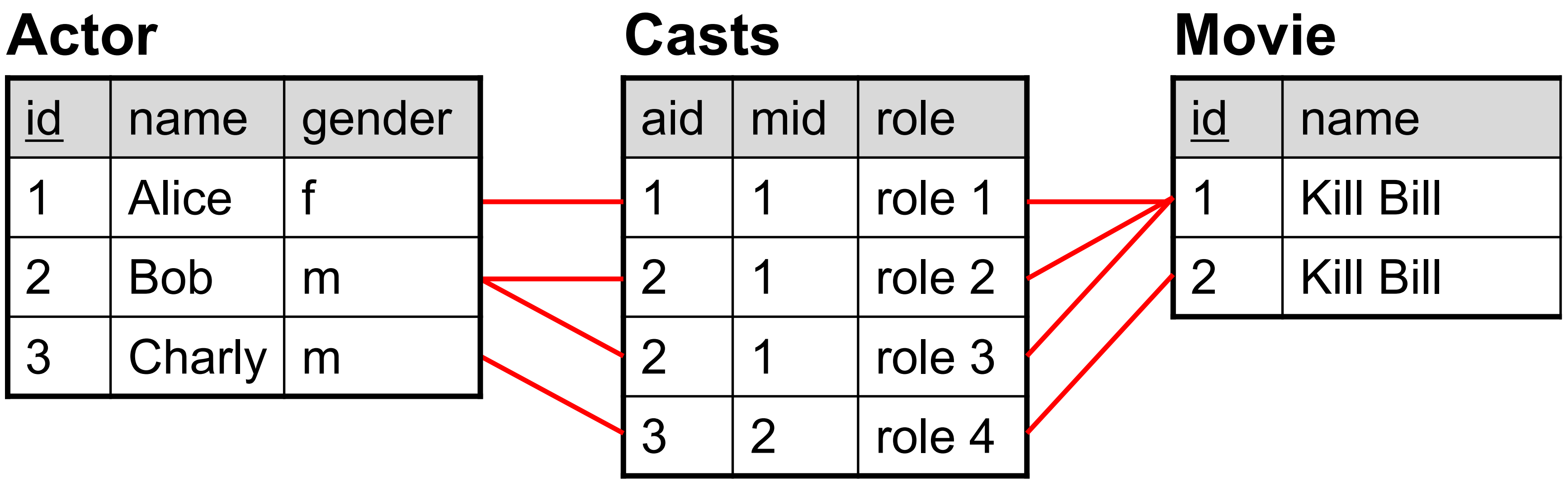
<u>id</u>	name
1	Kill Bill
2	Kill Bill

```
SELECT
FROM
WHERE
GROUP BY
```

?

name	(no name)
Alice	1
Bob	2
Charly	1

4. How to get the number of casts for each actor?



Actor

id	name	gender
1	Alice	f
2	Bob	m
3	Charly	m

Casts

aid	mid	role
1	1	role 1
2	1	role 2
2	1	role 3
3	2	role 4

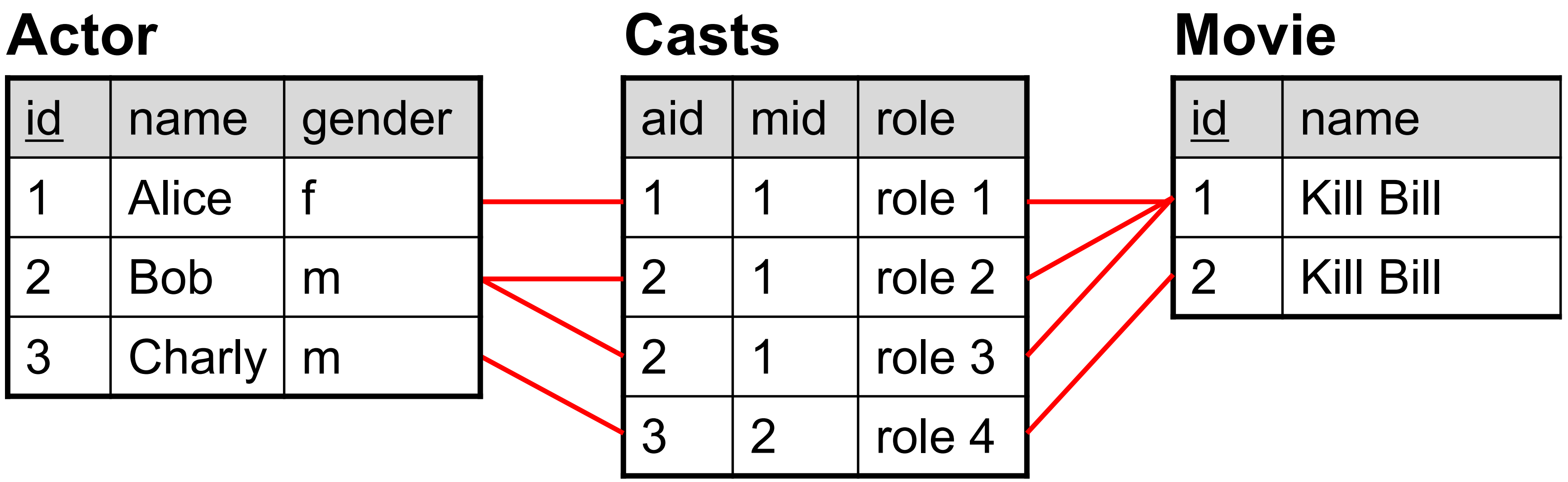
Movie

id	name
1	Kill Bill
2	Kill Bill

```
SELECT  a.name, count(*)
FROM    actor a, casts c
WHERE   a.id = c.aid
GROUP BY a.id, a.name
```

name	(no name)
Alice	1
Bob	2
Charly	1

4. How to get the number of casts for each actor?



Actor

id	name	gender
1	Alice	f
2	Bob	m
3	Charly	m

Casts

aid	mid	role
1	1	role 1
2	1	role 2
2	1	role 3
3	2	role 4

Movie

id	name
1	Kill Bill
2	Kill Bill

```
SELECT  a.name, count(distinct c.aid)
FROM    actor a, casts c
WHERE   a.id = c.aid
GROUP BY a.id, a.name
```

⇨ ?

4. How to get the number of casts for each actor?

Actor

id	name	gender
1	Alice	f
2	Bob	m
3	Charly	m

Casts

aid	mid	role
1	1	role 1
2	1	role 2
2	1	role 3
3	2	role 4

Movie

id	name
1	Kill Bill
2	Kill Bill

```sql
SELECT  a.name, count(distinct c.aid)
FROM    actor a, casts c
WHERE   a.id = c.aid
GROUP BY a.id, a.name
```

name	(no name)
Alice	1
Bob	1
Charly	1

Will always show 1 (since there is only one distinct aid per group grouped by aid *by def.*)

4. How to get the number of movies for each actor?

Actor

<u>id</u>	name	gender
1	Alice	f
2	Bob	m
3	Charly	m

Casts

aid	mid	role
1	1	role 1
2	1	role 2
2	1	role 3
3	2	role 4

Movie

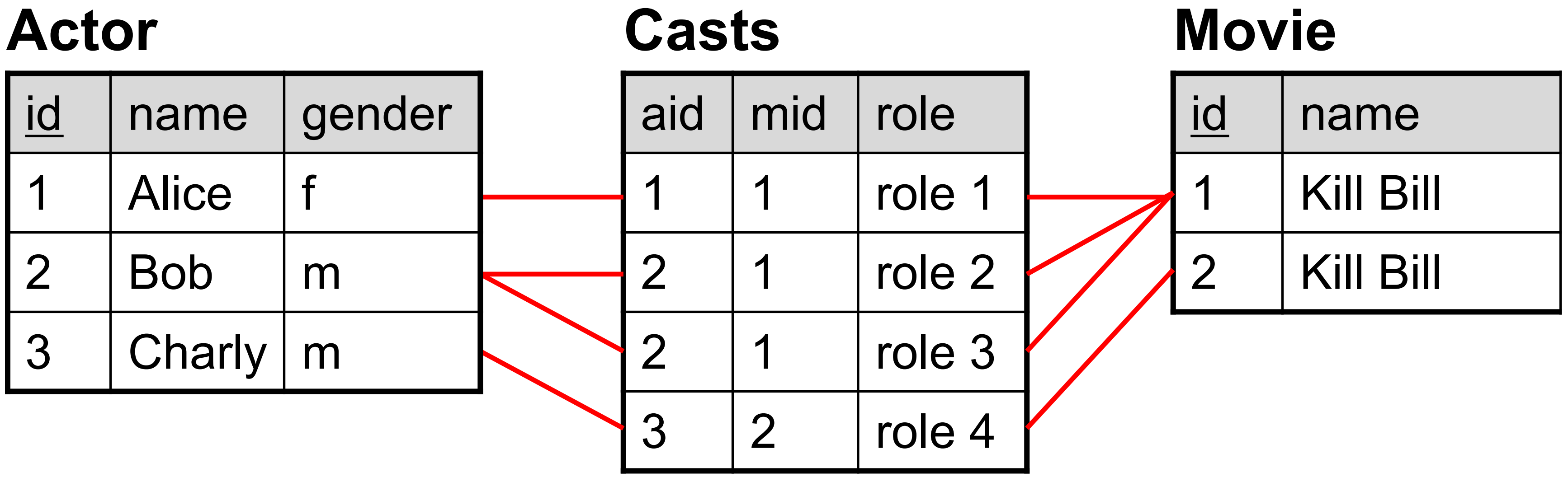
<u>id</u>	name
1	Kill Bill
2	Kill Bill

```
SELECT
FROM
WHERE
GROUP BY
```

?

name	(no name)
Alice	1
Bob	1
Charly	1

4. How to get the number of movies for each actor?



Actor

id	name	gender
1	Alice	f
2	Bob	m
3	Charly	m

Casts

aid	mid	role
1	1	role 1
2	1	role 2
2	1	role 3
3	2	role 4

Movie

id	name
1	Kill Bill
2	Kill Bill

```
SELECT  a.name, count(distinct c.mid)
FROM    actor a, casts c
WHERE   a.id = c.aid
GROUP BY a.id, a.name
```

name	(no name)
Alice	1
Bob	1
Charly	1

4. How to get the number of casts for each male actor?

Actor

id	name	gender
1	Alice	f
2	Bob	m
3	Charly	m

Casts

aid	mid	role
1	1	role 1
2	1	role 2
2	1	role 3
3	2	role 4

Movie

id	name
1	Kill Bill
2	Kill Bill

```
SELECT
FROM
WHERE
      AND
GROUP BY
```

?

name	(no name)
Bob	2
Charly	1

4. How to get the number of casts for each male actor?

Actor

id	name	gender
1	Alice	f
2	Bob	m
3	Charly	m

Casts

aid	mid	role
1	1	role 1
2	1	role 2
2	1	role 3
3	2	role 4

Movie

id	name
1	Kill Bill
2	Kill Bill

```
SELECT  a.name, count(distinct a.id)
FROM    actor a, casts c
WHERE   a.id = c.aid
     AND a.gender = 'm'
GROUP BY a.id, a.name
```

name	(no name)
Bob	2
Charly	1

5. How to get male actors with number of casts> 1

Actor

id	name	gender
1	Alice	f
2	Bob	m
3	Charly	m

Casts

aid	mid	role
1	1	role 1
2	1	role 2
2	1	role 3
3	2	role 4

Movie

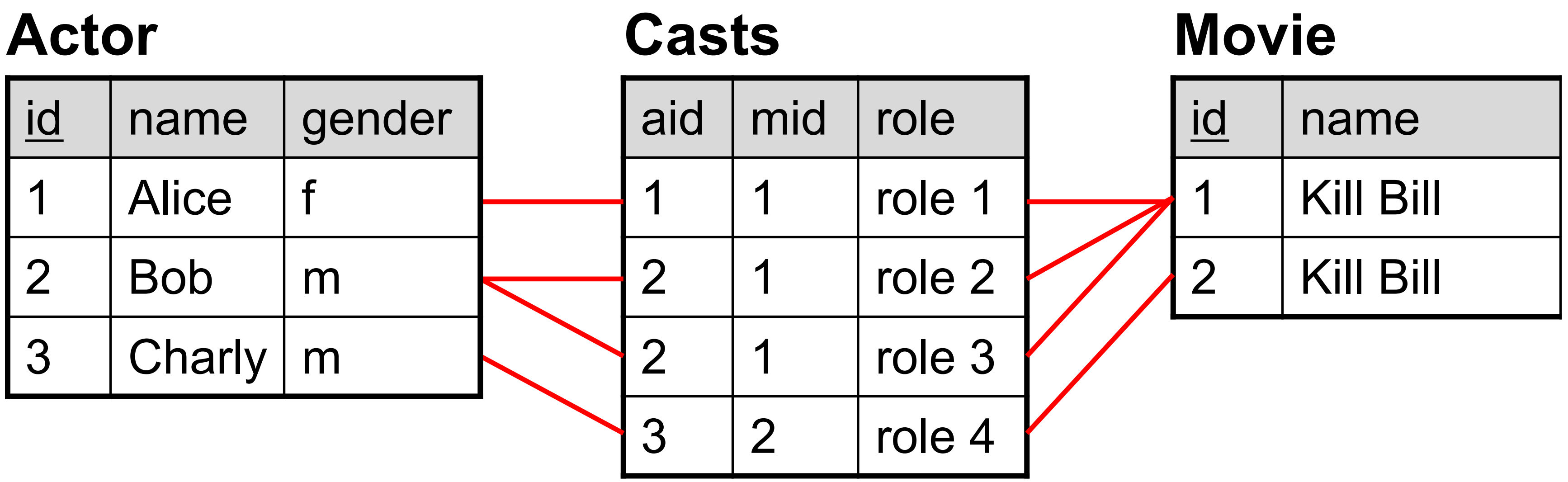
id	name
1	Kill Bill
2	Kill Bill

```
SELECT
FROM
WHERE
      AND
GROUP BY
```

?

name	(no name)
Bob	2

5. How to get male actors with number of casts> 1



Actor

id	name	gender
1	Alice	f
2	Bob	m
3	Charly	m

Casts

aid	mid	role
1	1	role 1
2	1	role 2
2	1	role 3
3	2	role 4

Movie

id	name
1	Kill Bill
2	Kill Bill

```
SELECT  a.name, count(*)
FROM    actor a, casts c
WHERE   a.id = c.aid
    AND a.gender = 'm'
GROUP BY a.id, a.name
HAVING count(*) > 1
```

name	(no name)
Bob	2

Small IMDB (SQLite) Practice examples

IMDB practice

300

Actor(id, fname, lname, gender)
"Cast"(aid, mid, role)
Movie(id, name, year)
Movie_director(did, mid)
Director(id, fname, lname)
Movie_genre(mid, genre)

Which genres have more than 10 movies associated with it?
Return: (genre, #movies)

[4 results on SQLite, 27 on Azure]

⇨ ?

IMDB practice

300

```
Actor(id, fname, lname, gender)
"Cast"(aid, mid, role)
Movie(id, name, year)
Movie_director(did, mid)
Director(id, fname, lname)
Movie_genre(mid, genre)
```

Which genres have more than 10 movies associated with it?
Return: (genre, #movies)

[4 results on SQLite, 27 on Azure]

```
SELECT      movie_genre.genre,
            count(movie.name)
FROM        movie_genre
            INNER JOIN movie
            ON movie_genre.mid = movie.id
GROUP BY    movie_genre.genre
HAVING      count(movie.name) > 10
```

Can you simplify this query?

IMDB practice

Which genres have more than 10 movies associated with it?
Return: (genre, #movies)

[4 results on SQLite, 27 on Azure]

Actor(id, fname, lname, gender)
"Cast"(aid, mid, role)
Movie(id, name, year)
Movie_director(did, mid)
Director(id, fname, lname)
Movie_genre(mid, genre)

```
SELECT      movie_genre.genre,
            count(movie_genre.mid)
FROM        movie_genre


GROUP BY    movie_genre.genre
HAVING      count(movie_genre.mid) > 10
```

We don't need the "movie" table

IMDB practice

Which movies have roles (e.g. "Stormtrooper" and not empty entries "") with more than 10 actors for that role?
Return: (name, role,#actors)

[3 results on SQLite, >36k on Azure]

Actor(<u>id</u>, fname, lname, gender)
"Cast"(aid, mid, role)
Movie(<u>id</u>, name, year)
Movie_director(did, mid)
Director(<u>id</u>, fname, lname)
Movie_genre(mid, genre)

300

⇨ ?

IMDB practice

300

```
Actor(id, fname, lname, gender)
"Cast"(aid, mid, role)
Movie(id, name, year)
Movie_director(did, mid)
Director(id, fname, lname)
Movie_genre(mid, genre)
```

Which movies have roles (e.g. "Stormtrooper" and not empty entries "") with more than 10 actors for that role?
Return: (name, role,#actors)

[3 results on SQLite, >36k on Azure]

```
SELECT      movie.name, "cast".role,
            count(actor.id)
FROM        actor
            INNER JOIN "cast"
            ON "cast".aid = actor.id
            INNER JOIN movie
            ON movie.id = "cast".mid
WHERE       "cast".role <> ''
GROUP BY    movie.name, "cast".role
HAVING      count(actor.id) > 10
```

Again: Can you simplify this query?

IMDB practice

300

Actor(id, fname, lname, gender)
"Cast"(aid, mid, role)
Movie(id, name, year)
Movie_director(did, mid)
Director(id, fname, lname)
Movie_genre(mid, genre)

Which movies have roles (e.g. "Stormtrooper") with more than 10 actors for that role?
Return: (name, role,#actors)

[3 results on SQLite, >36k on Azure]

```
SELECT      movie.name, "cast".role,
            count("cast".aid)
FROM
            "cast"

            INNER JOIN movie
            ON movie.id = "cast".mid
WHERE       "cast".role <> ''
GROUP BY    movie.name, "cast".role
HAVING      count("cast".aid) > 10
```

We don't need the "Actor" table!

But we should probably include the movie.id in the grouping

IMDB practice

300

Actor(id, fname, lname, gender)
"Cast"(aid, mid, role)
Movie(id, name, year)
Movie_director(did, mid)
Director(id, fname, lname)
Movie_genre(mid, genre)

Which movies had co-directors and how many?
Return: (name, #directors)

[5 results on SQLite, >83k on Azure]

$\Rightarrow$?

IMDB practice

300

Actor(id, fname, lname, gender)
"Cast"(aid, mid, role)
Movie(id, name, year)
Movie_director(did, mid)
Director(id, fname, lname)
Movie_genre(mid, genre)

Which movies had co-directors and how many?
Return: (name, #directors)

[5 results on SQLite, >83k on Azure]

```
SELECT      movie.name,
            count(movie_director.did)
FROM        movie
            INNER JOIN movie_director
            ON movie_director.mid = movie.id
GROUP BY    movie.id
HAVING      count(movie_director.did) > 1
```

Incorrect for major databases, only
SQLite misbehaves and accepts it ☹

IMDB practice

300

Actor(id, fname, lname, gender)
"Cast"(aid, mid, role)
Movie(id, name, year)
Movie_director(did, mid)
Director(id, fname, lname)
Movie_genre(mid, genre)

Which movies had co-directors and how many?
Return: (name, #directors)

[5 results on SQLite, >83k on Azure]

```
SELECT     movie.name,
           count(movie_director.did)
FROM       movie
           INNER JOIN movie_director
           ON movie_director.mid = movie.id
GROUP BY   movie.id, movie.name
HAVING     count(movie_director.did) > 1
```

Rule: queries with "GROUP BY" can only contain attributes in the SELECT clause that also appear in the GROUP BY clause!

IMDB practice

```
Actor(id, fname, lname, gender)
"Cast"(aid, mid, role)
Movie(id, name, year)
Movie_director(did, mid)
Director(id, fname, lname)
Movie_genre(mid, genre)
```

How many roles did each actor gender (male and female) play?

Return: (gender, #roles)

[2 for F and M on both]

⇨ ?

IMDB practice

300

Actor(id, fname, lname, gender)
"Cast"(aid, mid, role)
Movie(id, name, year)
Movie_director(did, mid)
Director(id, fname, lname)
Movie_genre(mid, genre)

How many roles did each actor gender (male and female) play?

Return: (gender, #roles)

[2 for F and M on both]

```
SELECT      actor.gender, count("cast".mid)
FROM        actor
            INNER JOIN "cast"
            ON "cast".aid = actor.id
GROUP BY    actor.gender
```

IMDB practice

300

Actor(<u>id</u>, fname, lname, gender)
"Cast"(aid, mid, role)
Movie(<u>id</u>, name, year)
Movie_director(did, mid)
Director(<u>id</u>, fname, lname)
Movie_genre(mid, genre)

Which actors haven't acted in any movies yet?

Return: (fname, lname)

[0 on SQLite, >325k on Azure]

⇨ ?

Can you write this query first with an Outer join?

⇨ ?

Can you write this query then without an Outer join?

IMDB practice

300

Actor(id, fname, lname, gender)
"Cast"(aid, mid, role)
Movie(id, name, year)
Movie_director(did, mid)
Director(id, fname, lname)
Movie_genre(mid, genre)

Which actors haven't acted in any movies yet?

Return: (fname, lname)

[0 on SQLite, >325k on Azure]

```
SELECT    fname, lname
FROM      actor
          LEFT JOIN "cast"
          ON actor.id = "cast".aid
WHERE     aid IS NULL
```

?

Can you write this query then without an Outer join?

IMDB practice

300

```
Actor(id, fname, lname, gender)
"Cast"(aid, mid, role)
Movie(id, name, year)
Movie_director(did, mid)
Director(id, fname, lname)
Movie_genre(mid, genre)
```

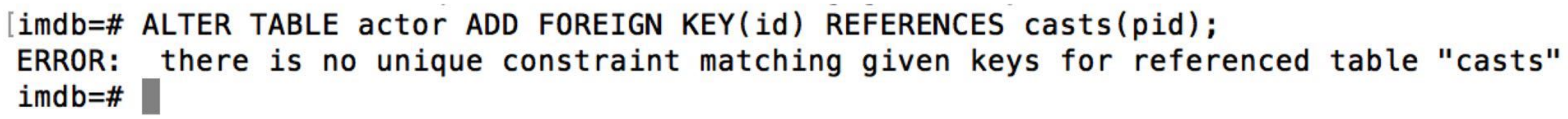
Which actors haven't acted in any movies yet?

Return: (fname, lname)

[0 on SQLite, >325k on Azure]

```
SELECT    fname, lname
FROM      actor
          LEFT JOIN "cast"
          ON actor.id = "cast".aid
WHERE     aid IS NULL
```

```
SELECT    fname, lname
FROM      actor
WHERE     id not in
          (SELECT   aid
          FROM      "cast")
```



```
imdb=# ALTER TABLE actor ADD FOREIGN KEY(id) REFERENCES casts(pid);
ERROR:  there is no unique constraint matching given keys for referenced table "casts"
imdb=# 
```

SQL injection

Simple SQL Query

Product

PName	Price	Category	Manufacturer
Gizmo	$19.99	Gadgets	GizmoWorks
Powergizmo	$29.99	Gadgets	GizmoWorks
SingleTouch	$149.99	Photography	Canon
MultiTouch	$203.99	Household	Hitachi

```
SELECT  PName
FROM    Product
WHERE   category='Gadgets'
```

$\Rightarrow$

PName
Gizmo
Powergizmo

Parameterized SQL Query

Varies between DBMSs. The following is the semantics for SQL server. You do not need to know that for exams

Product

PName	Price	Category	Manufacturer
Gizmo	$19.99	Gadgets	GizmoWorks
Powergizmo	$29.99	Gadgets	GizmoWorks
SingleTouch	$149.99	Photography	Canon
MultiTouch	$203.99	Household	Hitachi

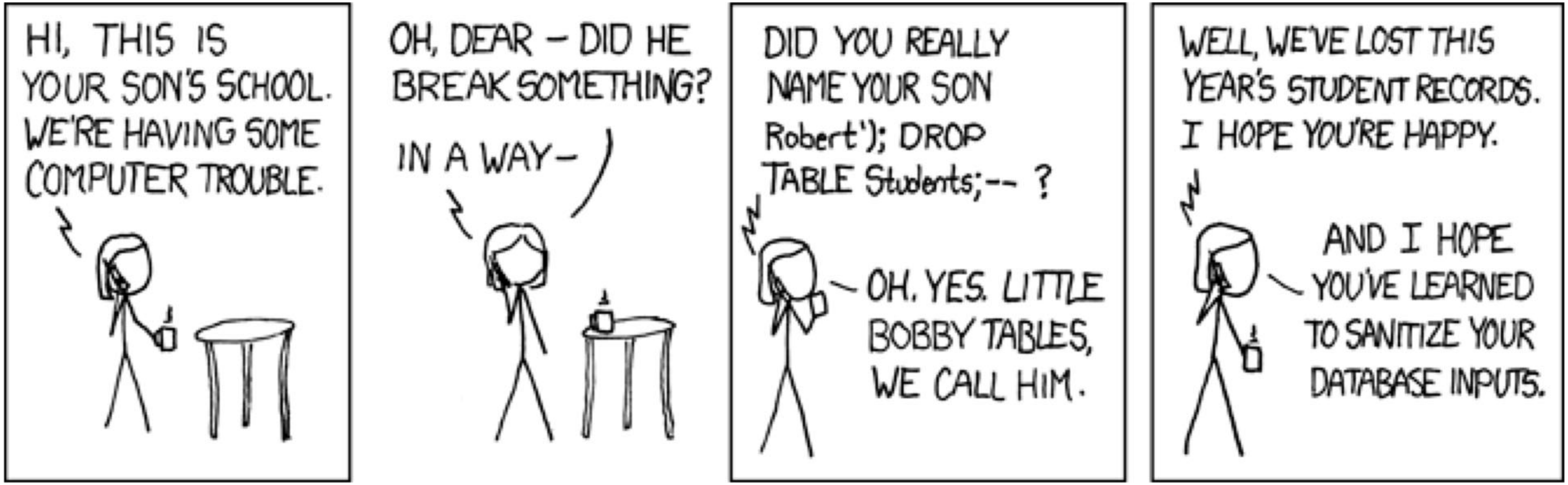
```
DECLARE @num VARCHAR(50)
SET @num = 'Gadgets'

SELECT  PName
FROM    Product
WHERE   category=@num
```

$\Rightarrow$

PName
Gizmo
Powergizmo

What happened here?

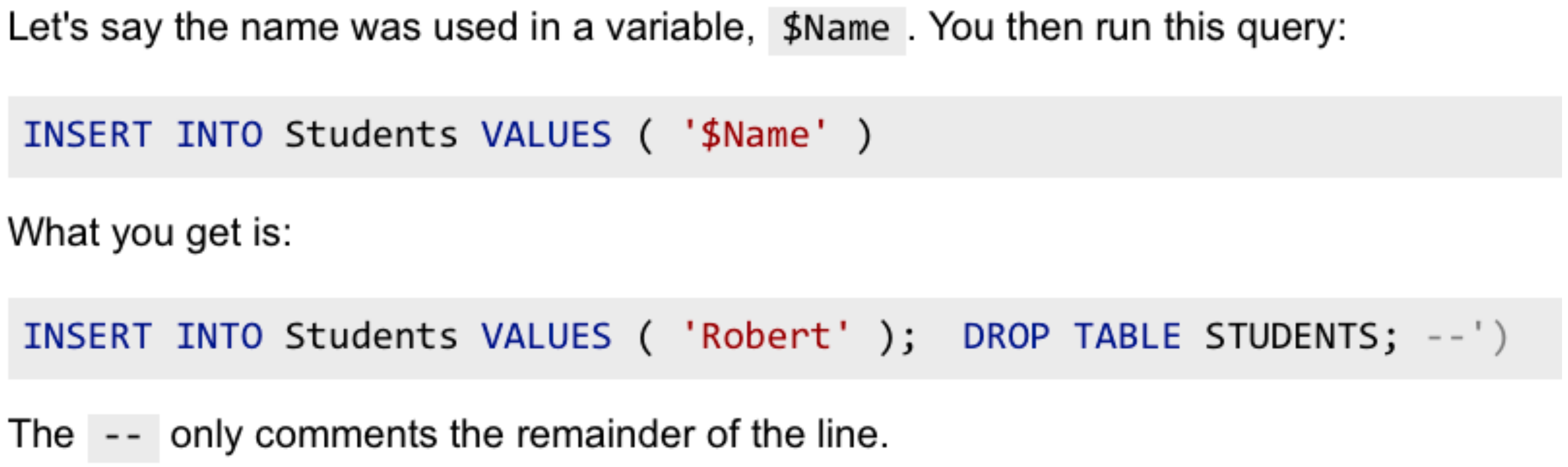

HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR – DID HE BREAK SOMETHING?
IN A WAY–
DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?
OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.
WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.

What does this SQL do:

```
Robert'); DROP
TABLE STUDENTS; --
```

Source: http://xkcd.com/327/, http://stackoverflow.com/questions/332365/xkcd-sql-injection-please-explain

It's called SQL injection: Version 1



Let's say the name was used in a variable, `$Name`. You then run this query:

```
INSERT INTO Students VALUES ( '$Name' )
```

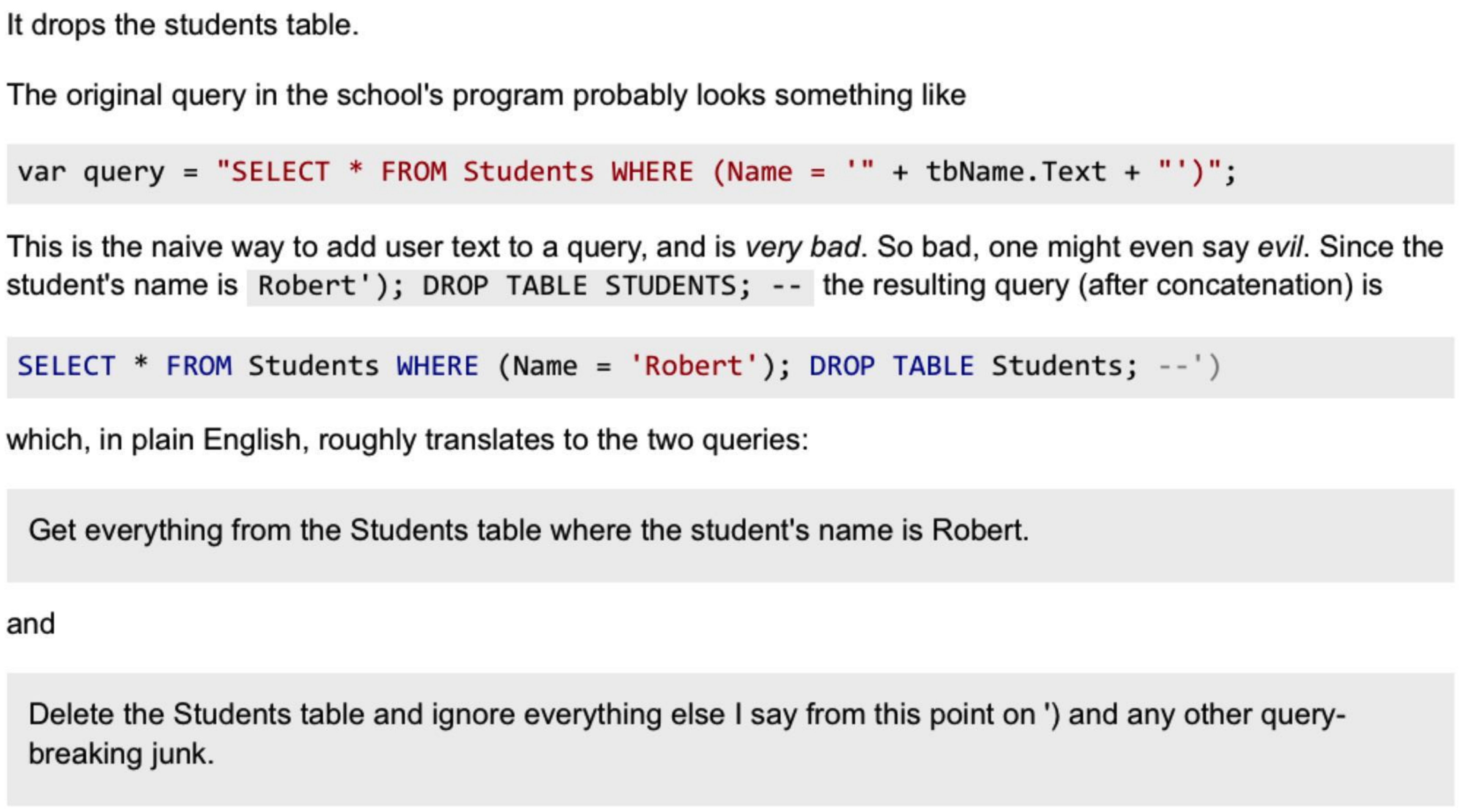
What you get is:

```
INSERT INTO Students VALUES ( 'Robert' );  DROP TABLE STUDENTS; --')
```

The `--` only comments the remainder of the line.

Source: http://xkcd.com/327/, http://stackoverflow.com/questions/332365/xkcd-sql-injection-please-explain

It's called SQL injection: Version 2



It drops the students table.

The original query in the school's program probably looks something like

```
var query = "SELECT * FROM Students WHERE (Name = '" + tbName.Text + "')";
```

This is the naive way to add user text to a query, and is *very bad*. So bad, one might even say *evil*. Since the student's name is `Robert'); DROP TABLE STUDENTS; --` the resulting query (after concatenation) is

```
SELECT * FROM Students WHERE (Name = 'Robert'); DROP TABLE Students; --')
```

which, in plain English, roughly translates to the two queries:

> Get everything from the Students table where the student's name is Robert.

and

> Delete the Students table and ignore everything else I say from this point on ') and any other query-breaking junk.

Source: http://xkcd.com/327/, http://stackoverflow.com/questions/332365/xkcd-sql-injection-please-explain

SQL injection

```
statement = "SELECT * FROM users WHERE name = '" + userName + "';"
```

```
' or '1'='1
' or '1'='1' -- '
' or '1'='1' ({ '
' or '1'='1' /* '
```

```
SELECT * FROM users WHERE name = '' OR '1'='1';
SELECT * FROM users WHERE name = '' OR '1'='1' -- ';
```

Source: http://en.wikipedia.org/wiki/SQL_injection

TJX Credit Card Numbers Theft

- In 2006, a $17.5 billion Fortune 500 firm
- Unauthorized intrusion resulting in lost of credit/debit card information
  - From May 2006 to January 2007?
  - From July 2005 to December 2006?
- TJX's overall losses: $1.35 billion – $4.5 billion

SQL injection: an interesting topic by itself

Pointers in case you are interested to learn more:

- https://www.owasp.org/index.php/SQL_Injection_Prevention_Cheat_Sheet
- https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/
- https://en.wikipedia.org/wiki/SQL_injection

Outlook

Database design
= Relational Data Modeling

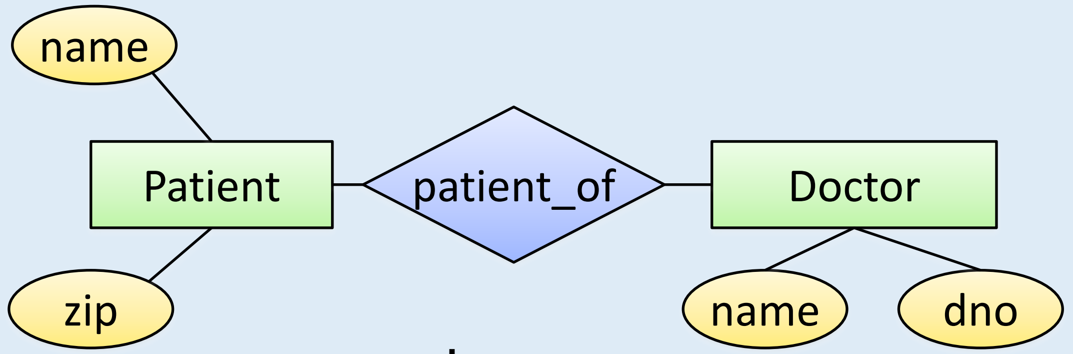
Data modeling and Database Design Process

1. ER Diagram

Conceptual Model:

("<u>technology independent</u>")

describe main data items



2. Relational Database Design

Logical Model

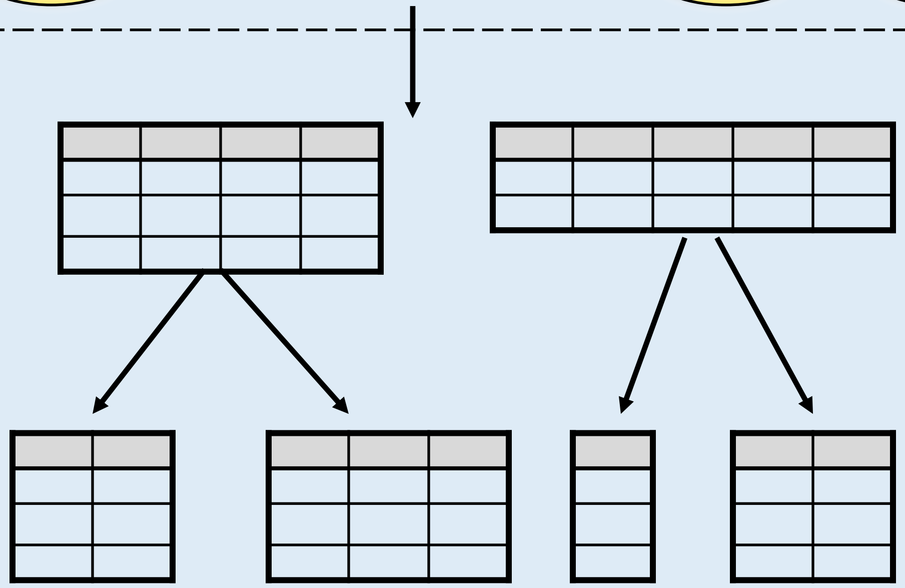
("<u>for relational databases</u>"):

Tables, Constraints

Functional Dependencies

Normalization:

Eliminates anomalies

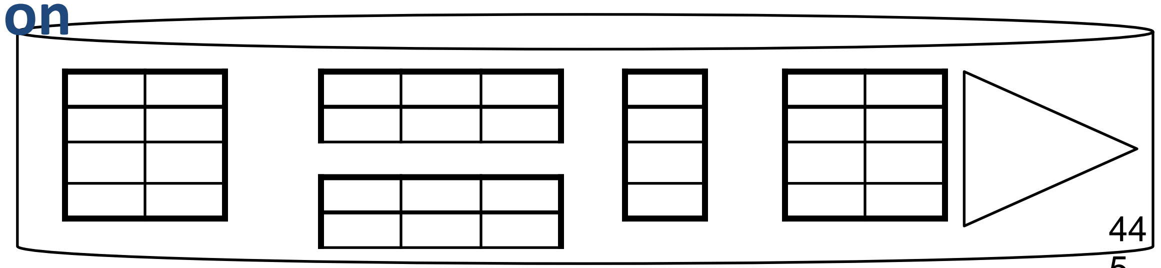


3. Database Implementation

Physical Model

Physical storage details

Result: Physical Schema



Graphicacy

"Graphicacy is concerned with the capacities people require in order to interpret and <u>generate information in the form of graphics</u>."

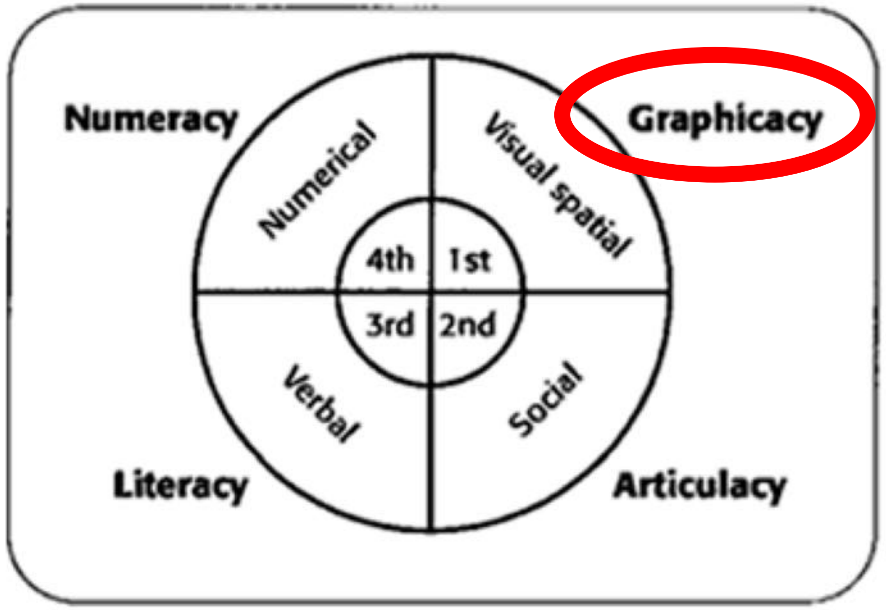
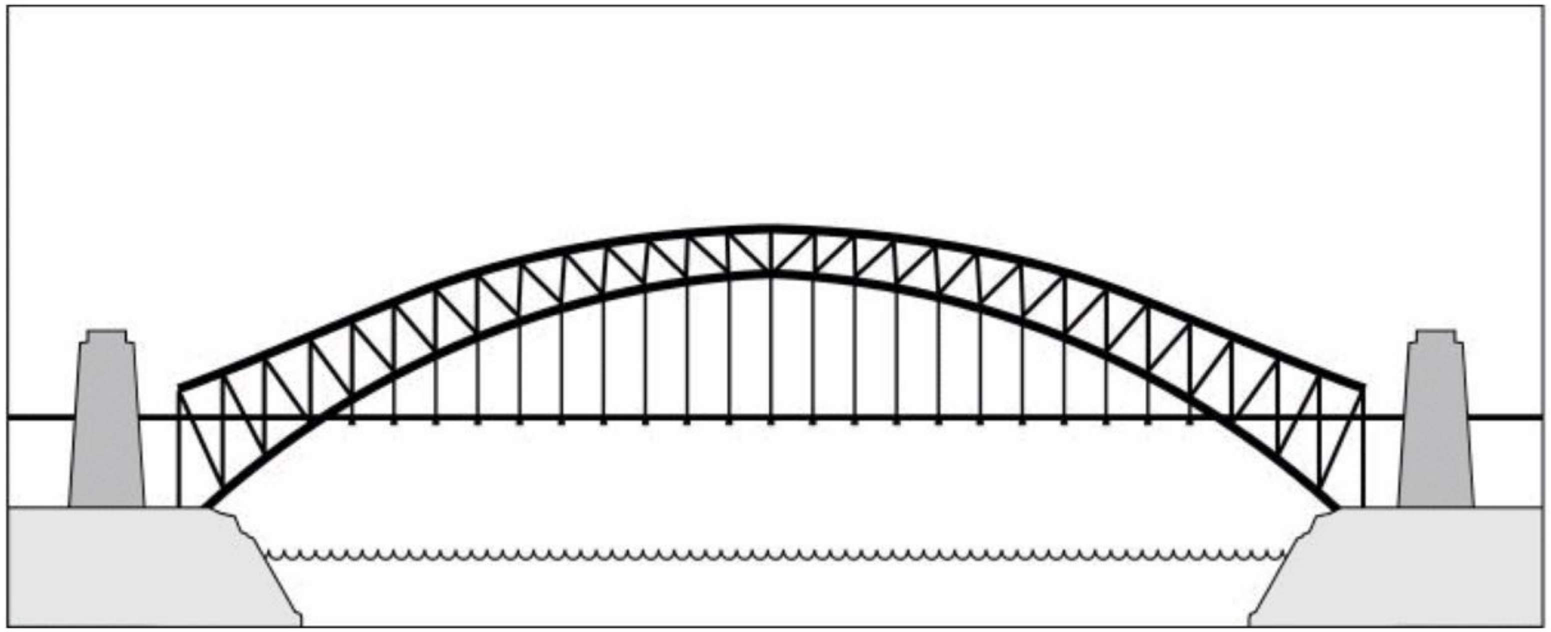
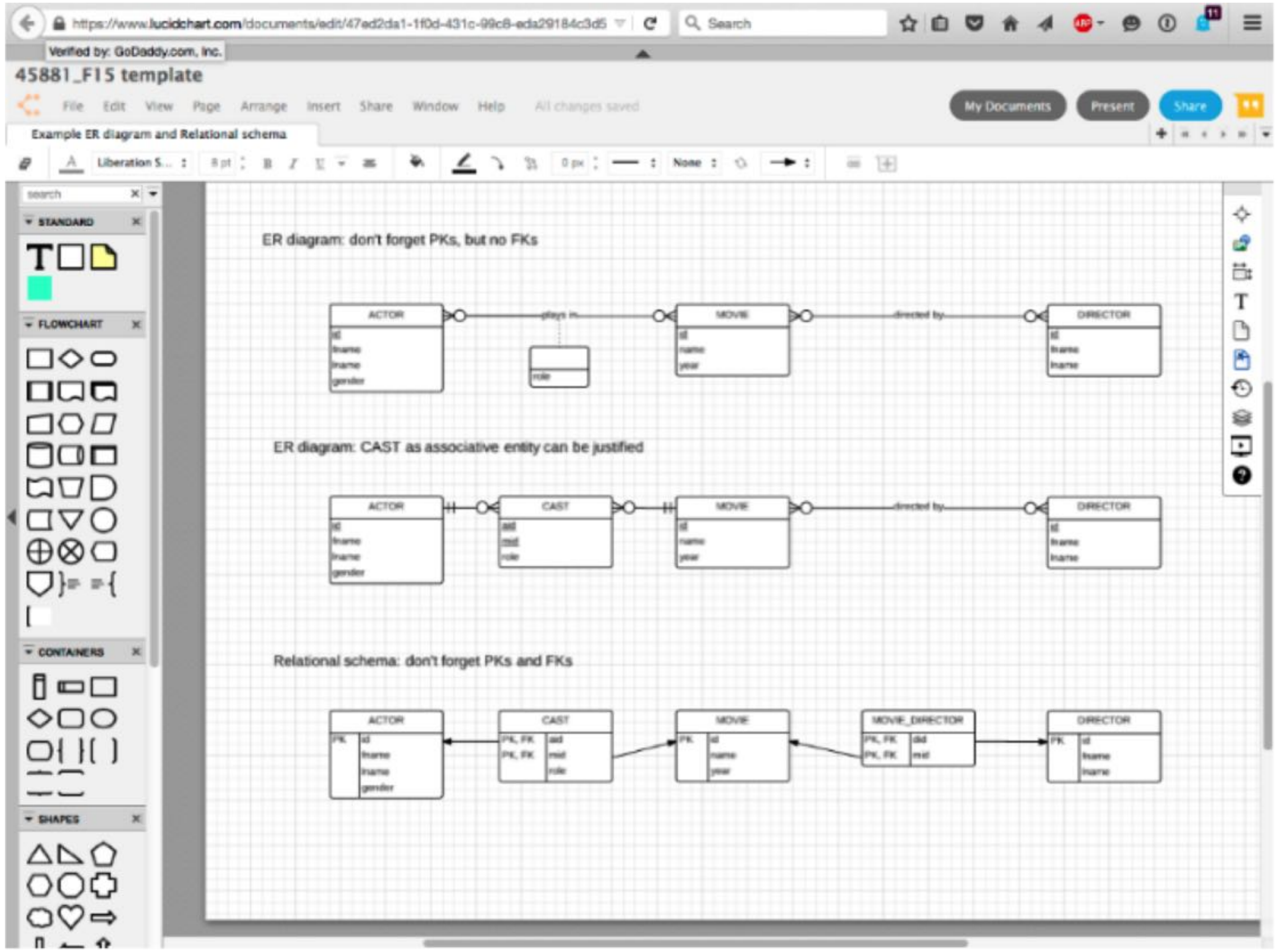


Figure 2. Balchin's "four types of ability."



Source: W.G.V Balchin, Graphicacy 1976, http://en.wikipedia.org/wiki/Graphicacy

Lucidchart.com


https://www.lucidchart.com/documents/edit/47ed2da1-1f0d-431c-99c8-eda29184c3d5

45881_F15 template

Example ER diagram and Relational schema

ER diagram: don't forget PKs, but no FKs

ER diagram: CAST as associative entity can be justified

Relational schema: don't forget PKs and FKs

I recommend to always first draw "free-form" by hand.
Only then (once you have a sketch) use some drawing tool.