Advanced Multidimensional Design

Knowledge objectives

1. Justify the existence of factless facts
2. Justify the use of degenerate dimensions
3. Justify the use of junk dimensions
4. Justify the implementation of any of the three kinds of slowly changing dimensions
5. Exemplify the three necessary conditions for summarizability

Understanding Objectives

1. Translate a sequence of multidimensional algebraic operations into SQL, and simplify it as much as possible
2. Solve the problems generated by general hierarchies

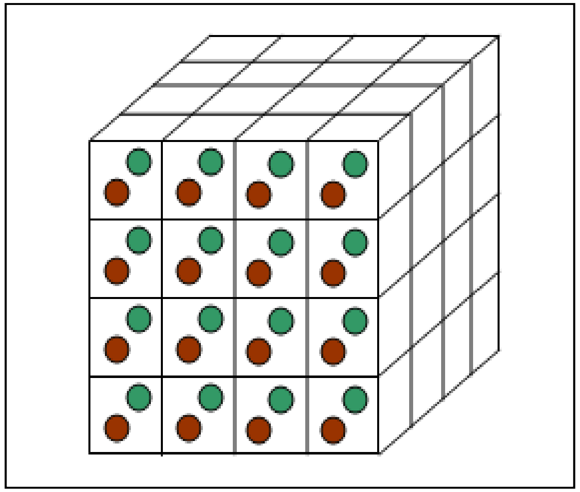
Application Objectives

1. Propose a sequence of algebraic operation to solve an informational need

ALGEBRAIC OPERATIONS

Operations set

- Selection (<cube>, <predicate on dimensions>)
- Projection (<cube>, <measures left>)
- Roll-Up (<cube>, <destination level>[, <aggregation function>])
- Drill-Down (<cube>, <destination level>)
- Drill-Across (<cube>, <new fact>[, <aggregation function>])
- ChangeBase (<cube>, <new base>)



Example or sequence of operations

A:=Selection(Purchases,Time.Day="1234")

B:=Roll-Up(A,Place.Region)

C:=Roll-up(B,Place.AllPlaces)

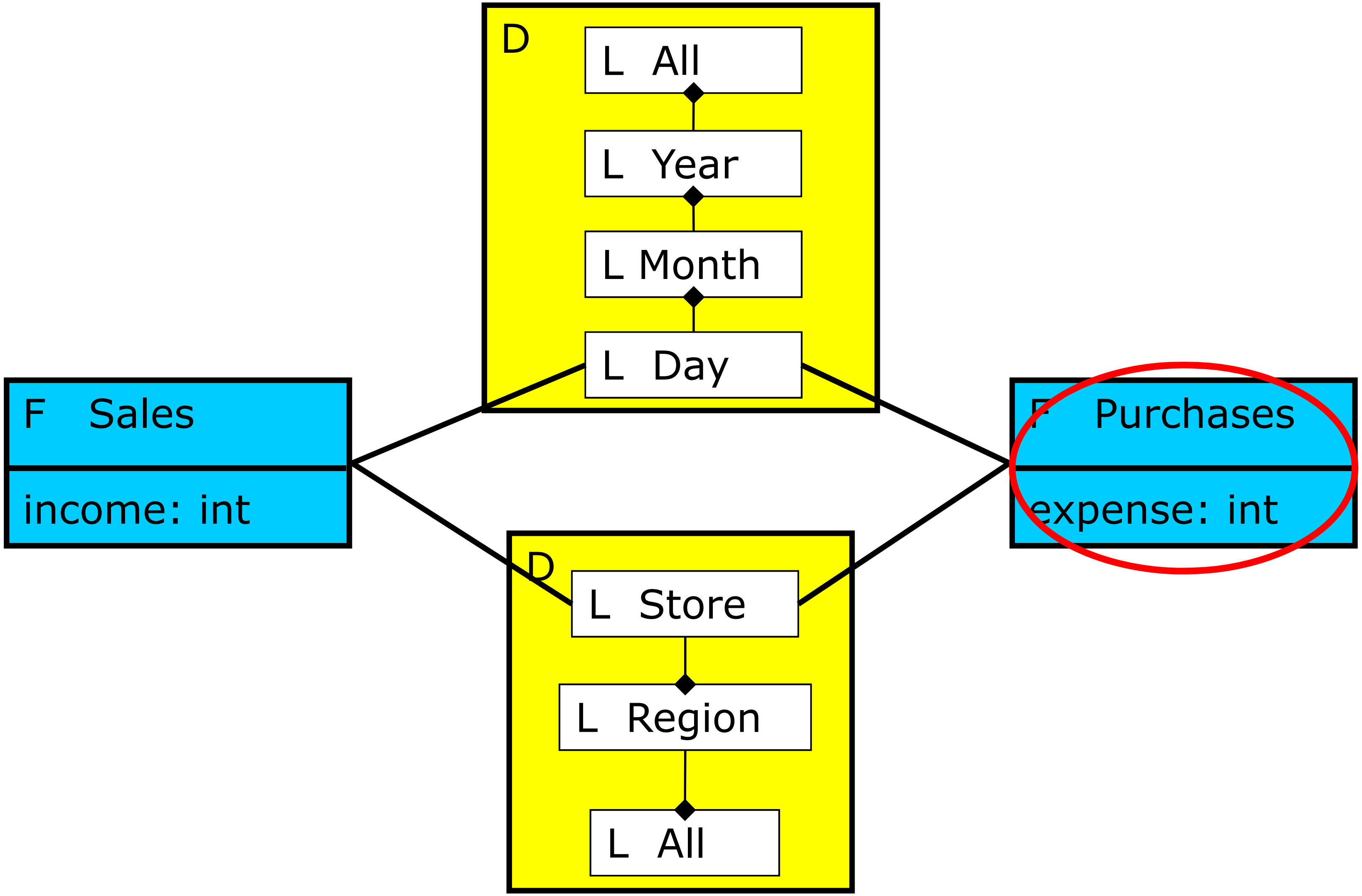
D:=ChangeBase(C,{Place,Time})

E:=ChangeBase(D,{Time})

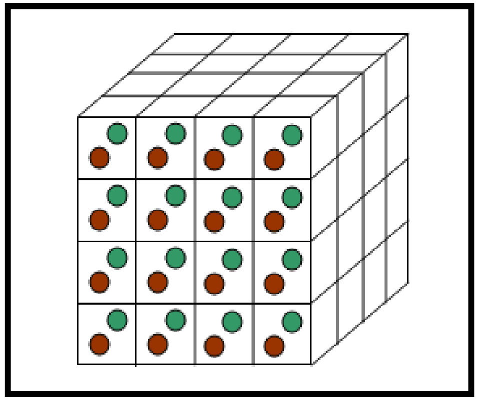
F:=Drill-Across(E,Sales)

G:=Projection(F, income)

Example of initial Cube-Query



Translation of the initial Cube-Query

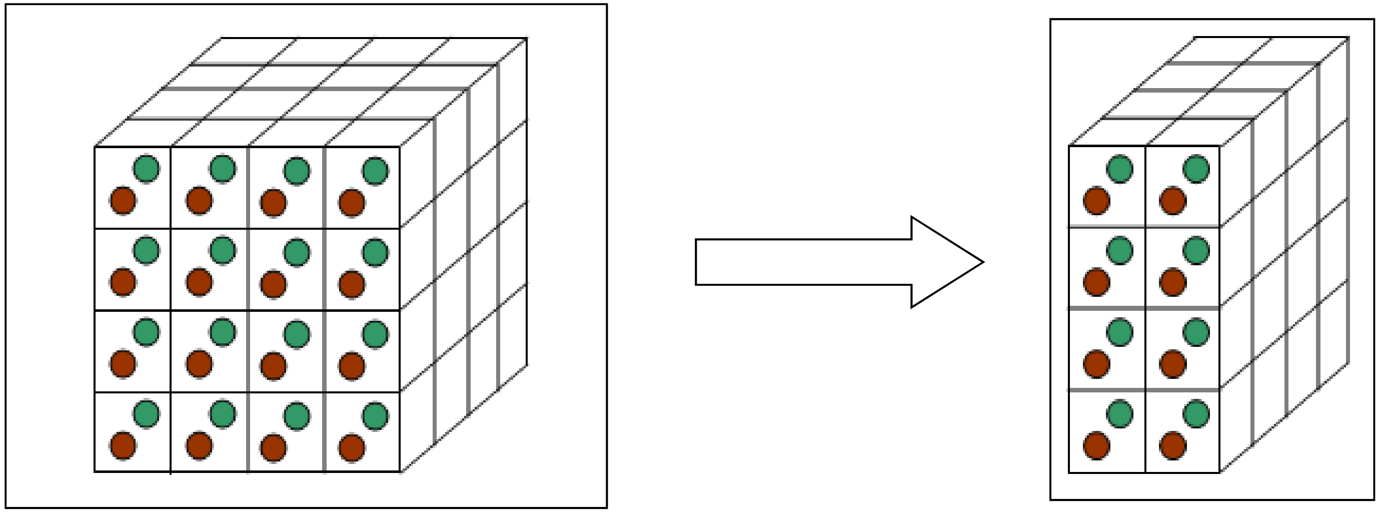


- *Cube-Query* to recover daily expense per store (Purchases):

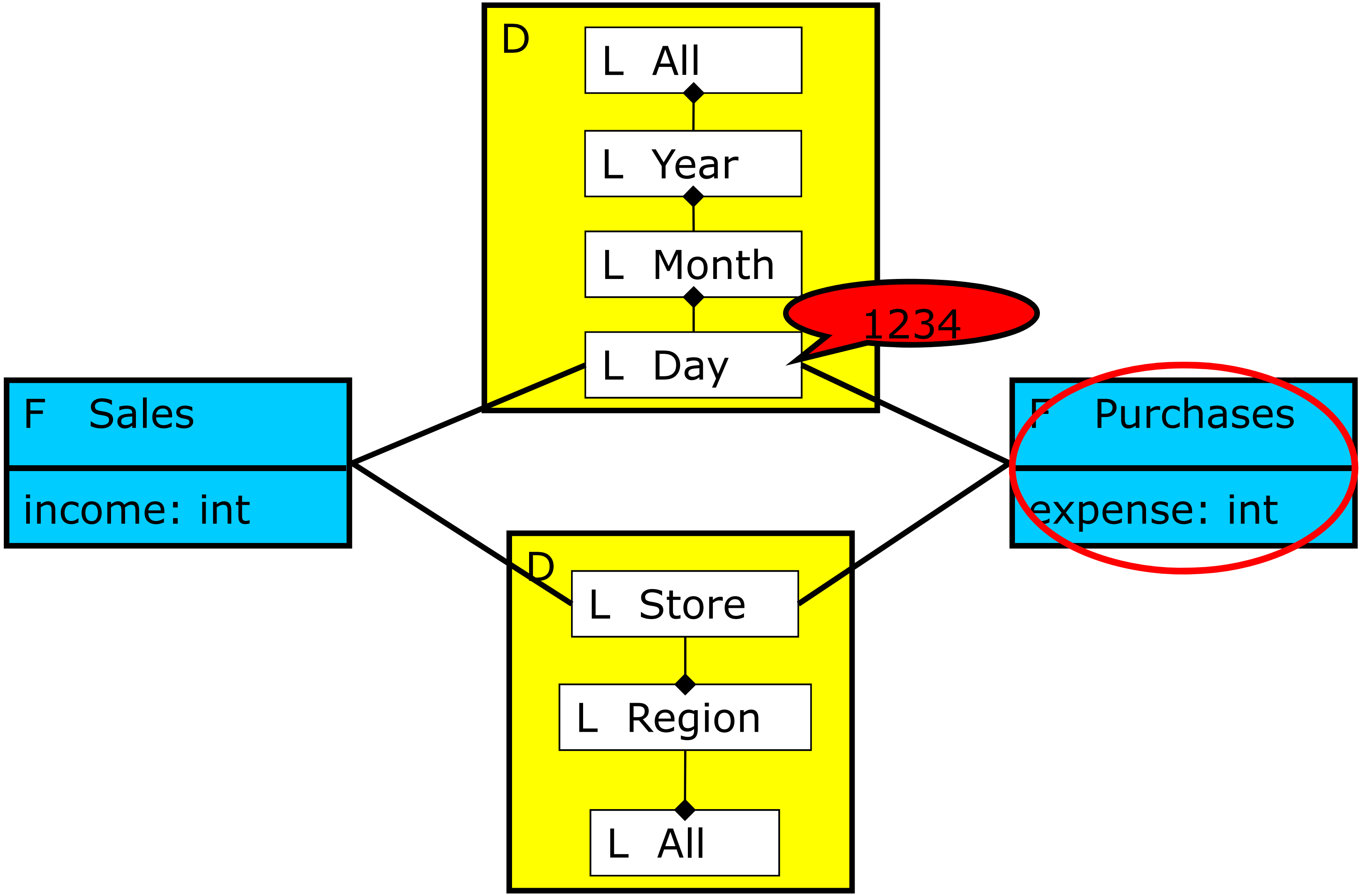
```
SELECT d.id, s.id, SUM(p.expense)
FROM purchases p, day d, store s
WHERE p.day=d.id AND p.store=s.id
GROUP BY d.id, s.id
ORDER BY d.id, s.id
```

Selection

- Allows the selection of a subset of cells from those available in the original cube



Example of selection



Translation of selection

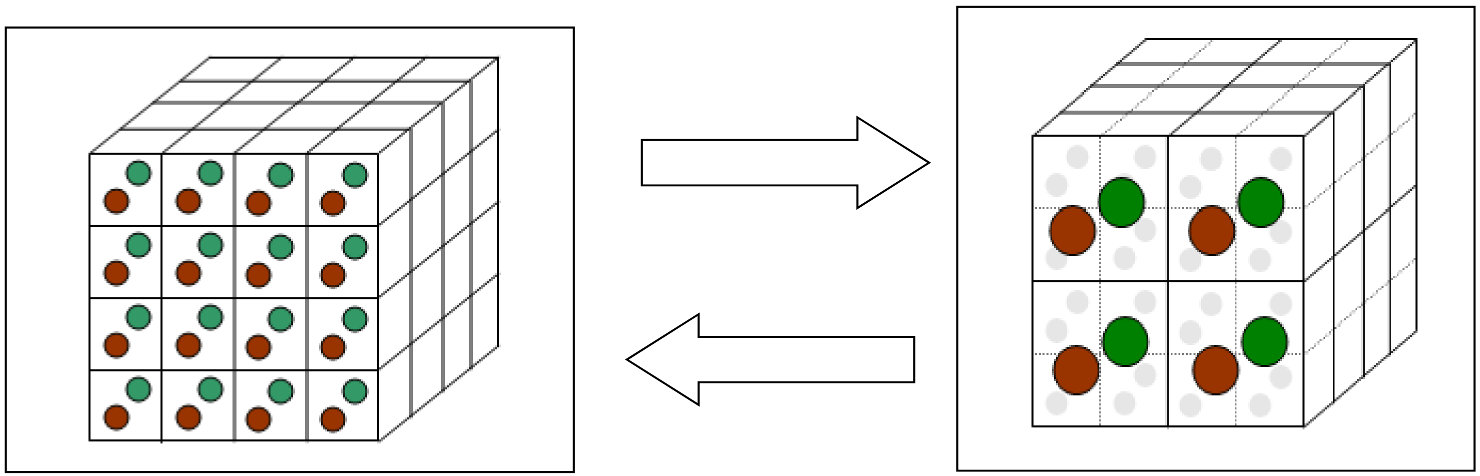
```
SELECT d.id, s.id, SUM(p.expense)
FROM purchases p, day d, store s
WHERE p.day=d.id AND p.store=s.id
GROUP BY d.id, s.id
ORDER BY d.id, s.id
```

- A:=Selection(Purchases,Time.Day="1234"): Adds conditions to the WHERE clause

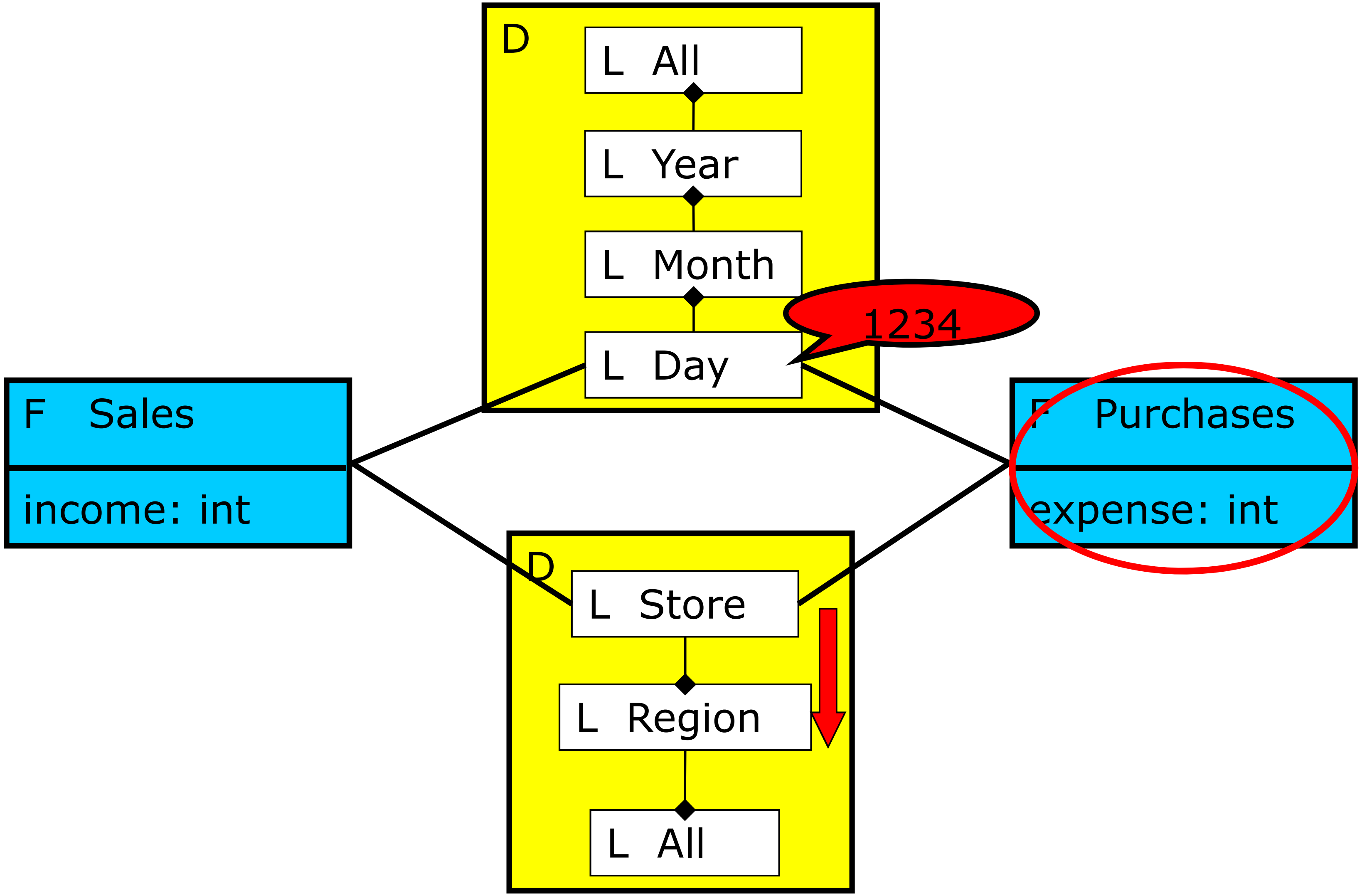
```
SELECT d.id, s.id, SUM(p.expense)
FROM purchases p, day d, store s
WHERE p.day=d.id AND p.store=s.id
          AND d.id="1234"
GROUP BY d.id, s.id
ORDER BY d.id, s.id
```

Roll-Up/Drill-Down

- Aggregates/Deaggregates the cells based on an aggregation hierarchy
- These operations increase/decrease the granularity of data



Example of Roll-up to an intermediate level



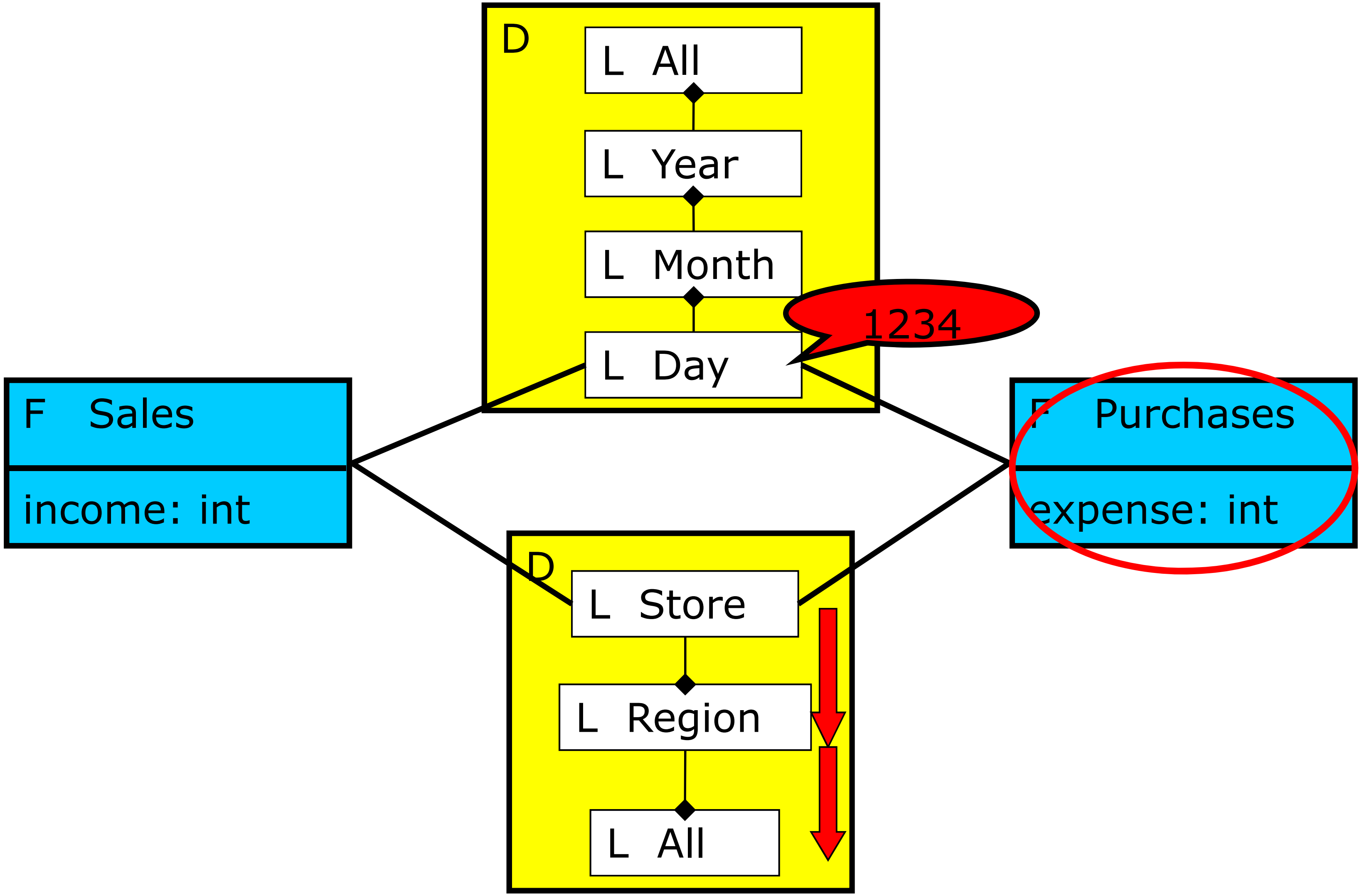
Translation of Roll-Up to an intermediate level

```
SELECT d.id, s.id, SUM(p.expense)
FROM purchases p, day d, store s
WHERE p.day=d.id AND p.store=s.id
          AND d.id="1234"
GROUP BY d.id, s.id
ORDER BY d.id, s.id
```

- B:=Roll-Up(A,Place.Region): Modifies SELECT, GROUP BY and ORDER BY

```
SELECT d.id, s.region, SUM(p.expense)
FROM purchases p, day d, store s
WHERE p.day=d.id AND p.store=s.id
          AND d.id="1234"
GROUP BY d.id, s.region
ORDER BY d.id, s.region
```

Example of Roll-up to *All* level



Translation of Roll-Up to All level

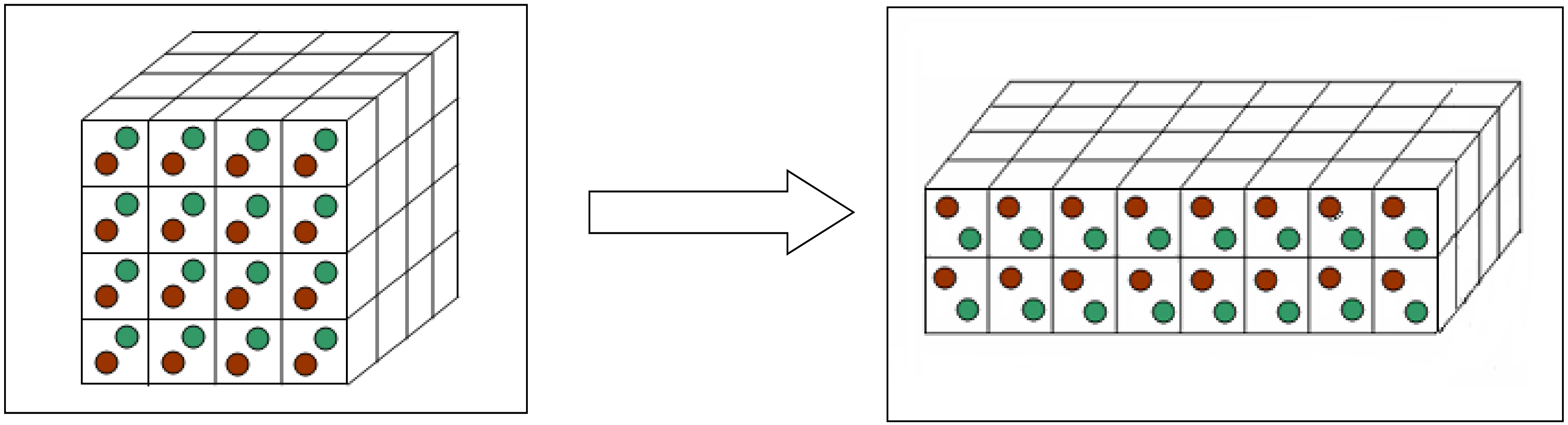
```
SELECT d.id, s.region, SUM(p.expense)
FROM purchases p, day d, store s
WHERE p.day=d.id AND p.store=s.id
          AND d.id="1234"
GROUP BY d.id, s.region
ORDER BY d.id, s.region
```

- C:=Roll-up(B,Place.AllPlaces): Modifies SELECT, GROUP BY and ORDER BY

```
SELECT d.id, "AllPlaces", SUM(p.expense)
FROM purchases p, day d, store s
WHERE p.day=d.id AND p.store=s.id
          AND d.id="1234"
GROUP BY d.id , region
ORDER BY d.id , region
```

ChangeBase

- Places exactly the same cells (instances) of a cube in a new n-dimensional space with the same number of points



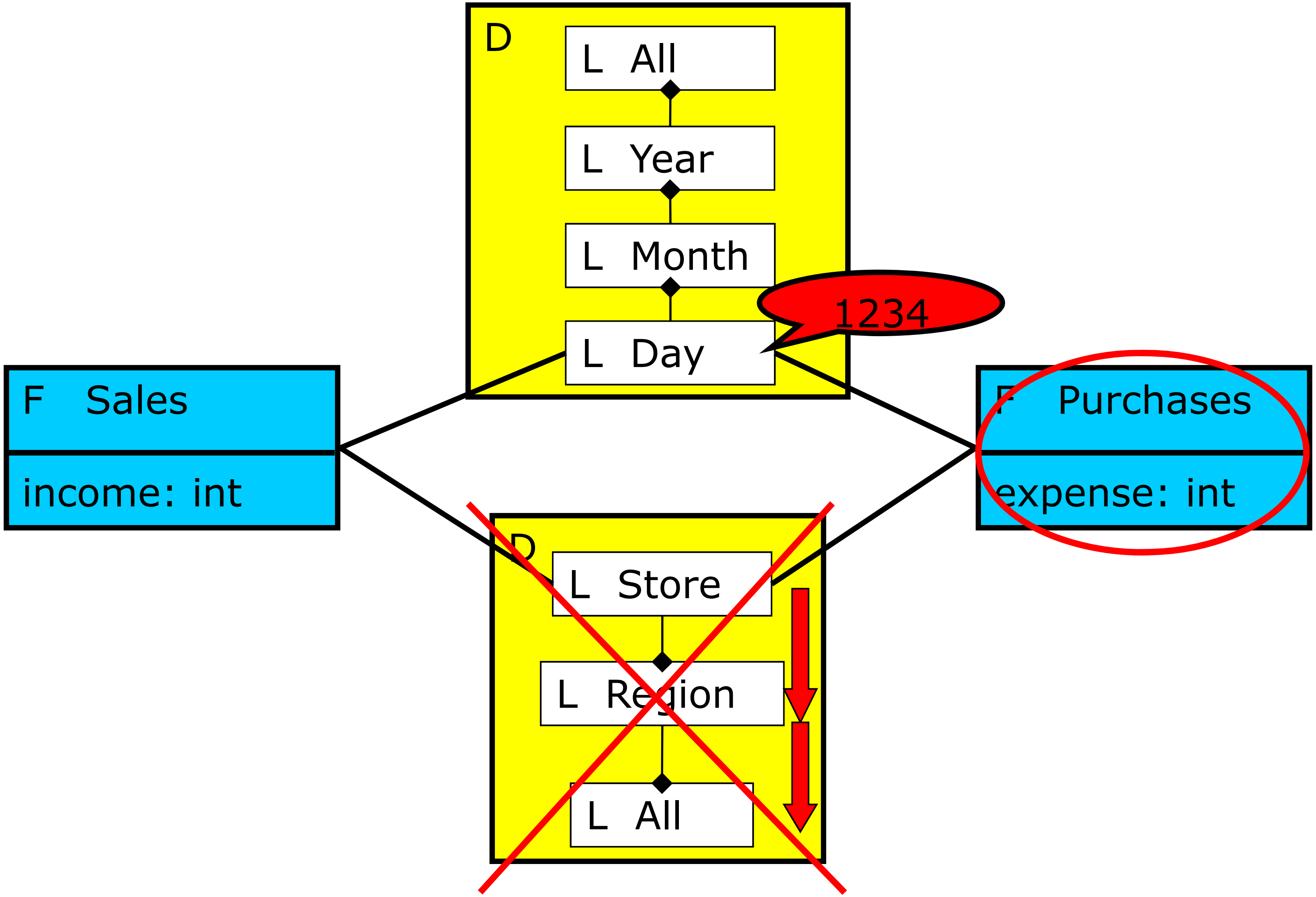
Translation of ChangeBase with reordering

```
SELECT d.id, "AllPlaces", SUM(p.expense)
FROM purchases p, day d, store s
WHERE p.day=d.id AND p.store=s.id
          AND d.id="1234"
GROUP BY d.id
ORDER BY d.id
```

- D:=ChangeBase (C,{Place,Time}): Reorders dimension attributes in SELECT and ORDER BY clauses

```
SELECT "AllPlaces", d.id, SUM(p.expense)
FROM purchases p, day d, store s
WHERE p.day=d.id AND p.store=s.id
          AND d.id="1234"
GROUP BY d.id
ORDER BY d.id
```

Example of ChangeBase with removal



Translation of ChangeBase with removal

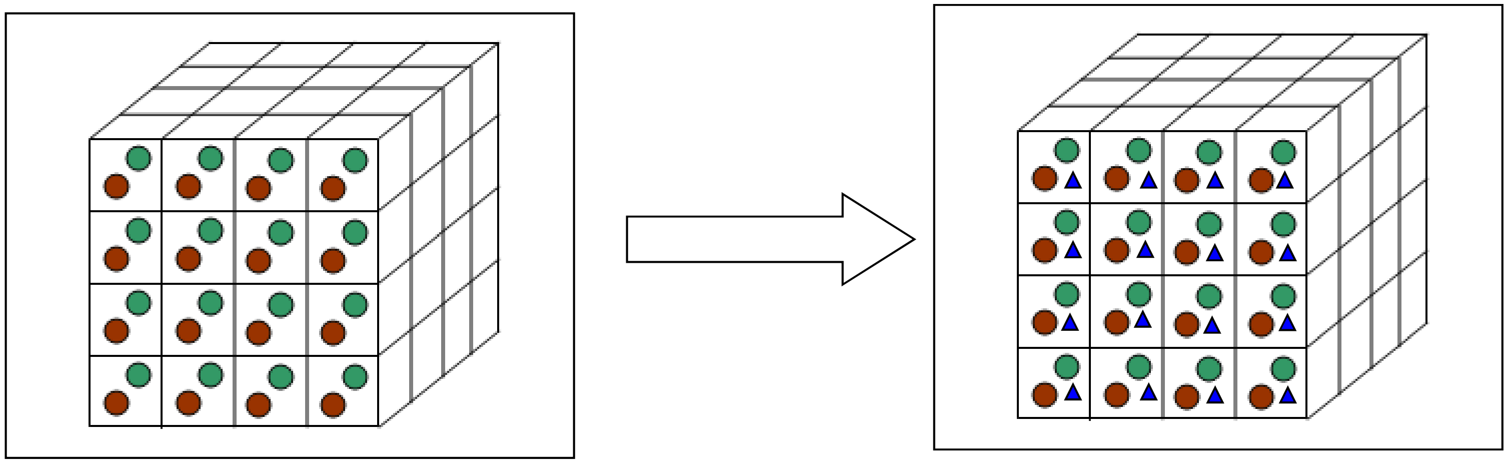
```
SELECT “AllPlaces”, d.id, SUM(p.expense)
FROM purchases p, day d, store s
WHERE c.day=d.id AND c.store=s.id
          AND d.id=“1234”
GROUP BY d.id
ORDER BY d.id
```

- E:=ChangeBase (D,{Time}): Removes the attributes of the dimensions from SELECT, GROUP BY and ORDER BY clauses

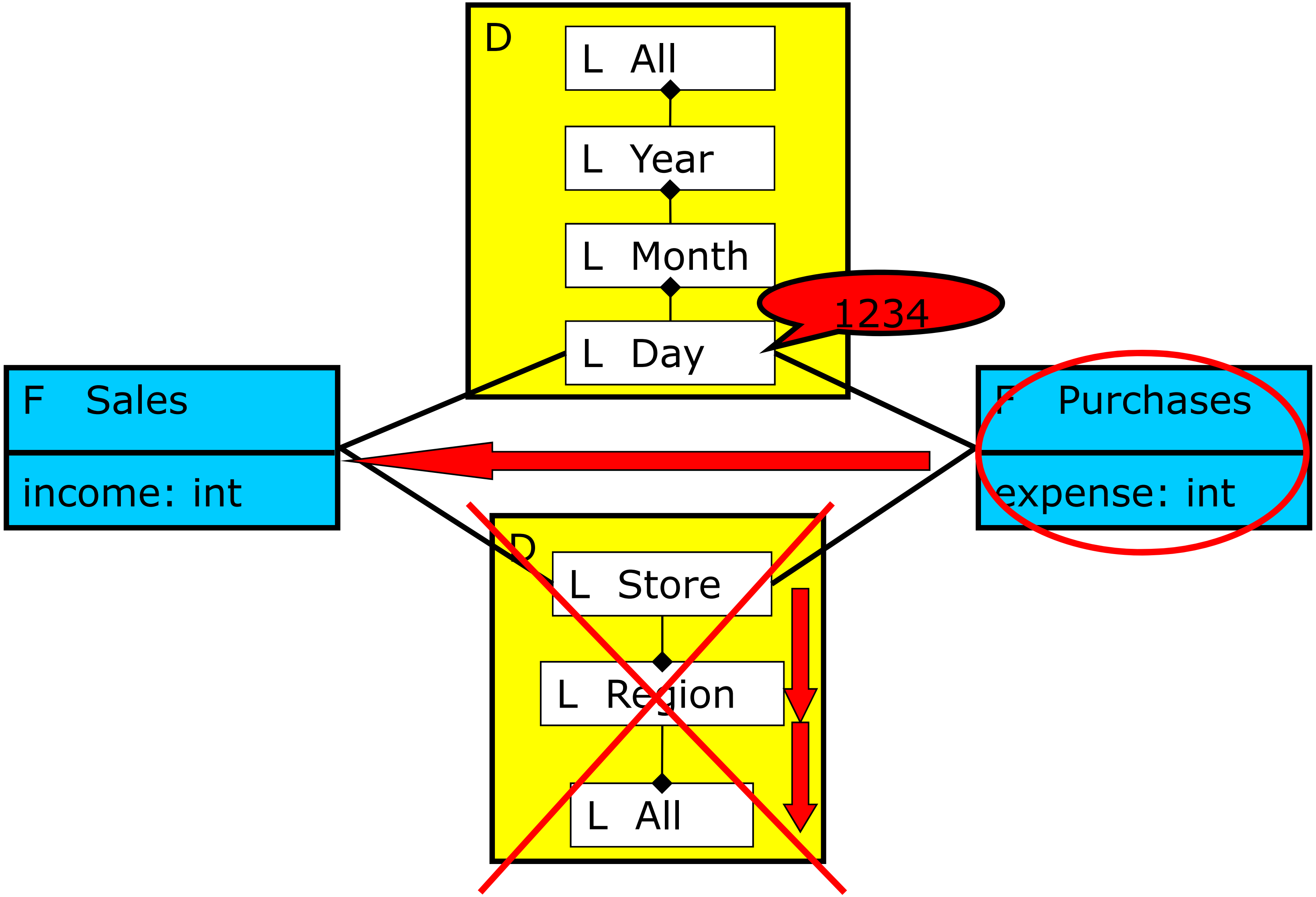
```
SELECT ”AllPlaces”, d.id, SUM(p.expense)
FROM purchases p, day d, store s
WHERE p.day=d.id AND p.store=s.id
          AND d.id=“1234”
GROUP BY d.id
ORDER BY d.id
```

Drill-Across

- Adds a new subject of analysis to those already in the cube



Example of Drill-across



Translation of Drill-Across

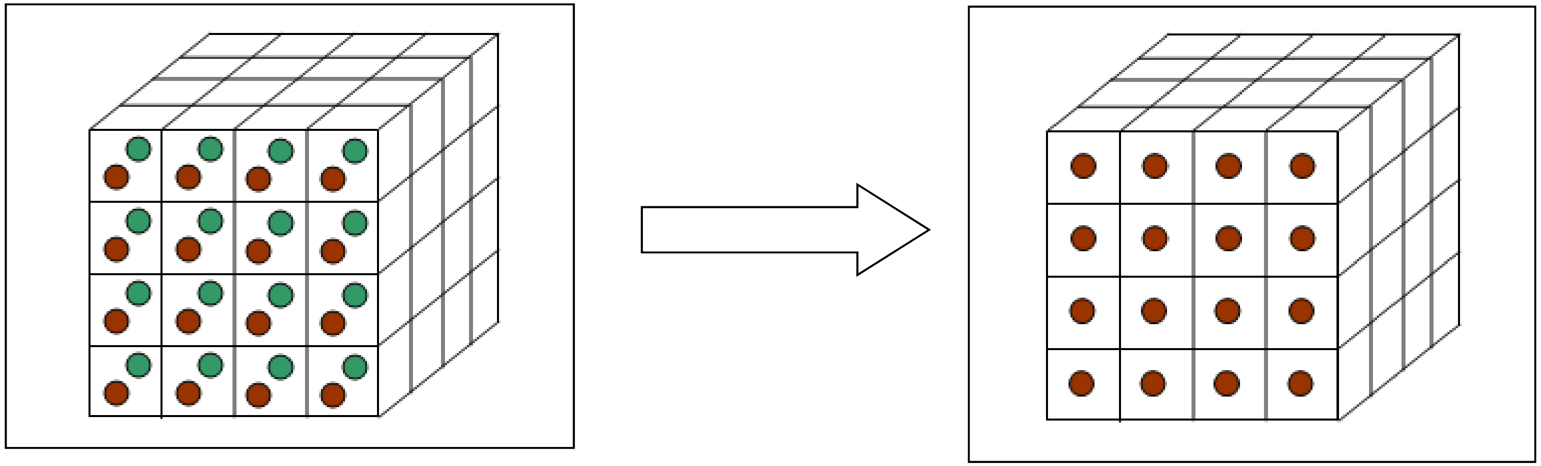
```
SELECT d.id, SUM(p.expense)
FROM purchases p, day d, store s
WHERE p.day=d.id AND p.store=s.id
          AND d.id="1234"
GROUP BY d.id
ORDER BY d.id
```

- F:=Drill-Across(E,Sales): Adds a new fact table to the FROM, its measures to the SELECT and the proper links to the WHERE

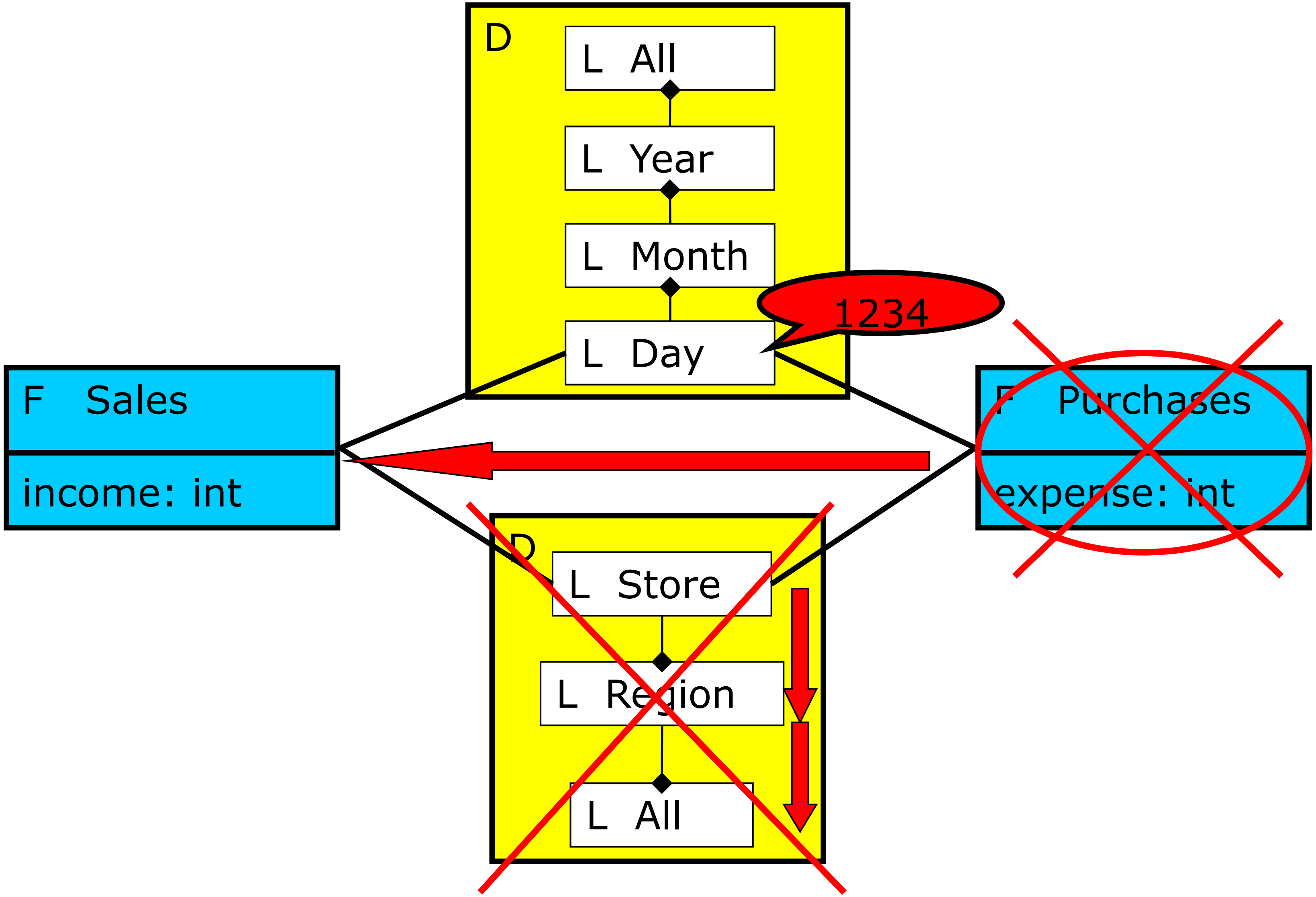
```
SELECT d.id, SUM(p.expense), SUM(sa.income)
FROM purchases p, sales sa, day d, store s
WHERE p.day=d.id AND p.store=s.id
          AND sa.day=d.id AND sa.store=s.id
          AND d.id="1234"
GROUP BY d.id
ORDER BY d.id
```

Projection

- Selects a subset of measures from those available in the cube



Example of Projection



Translation of Projection

```
SELECT d.id, SUM(p.expense), SUM(sa.income)
FROM purchases p, sales sa, day d, store s
WHERE p.day=d.id AND p.store=s.id
          AND sa.day=d.id AND sa.store=s.id
          AND d.id="1234"
GROUP BY d.id
ORDER BY d.id
```

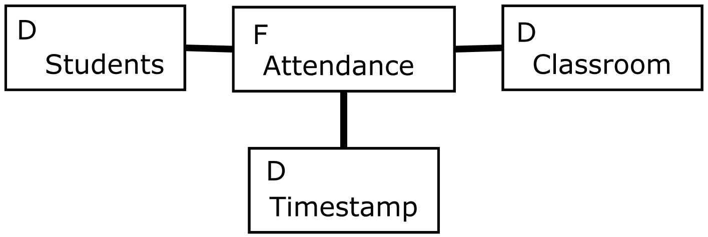
- G:=Projection(F, income): Removes measures from the SELECT clause

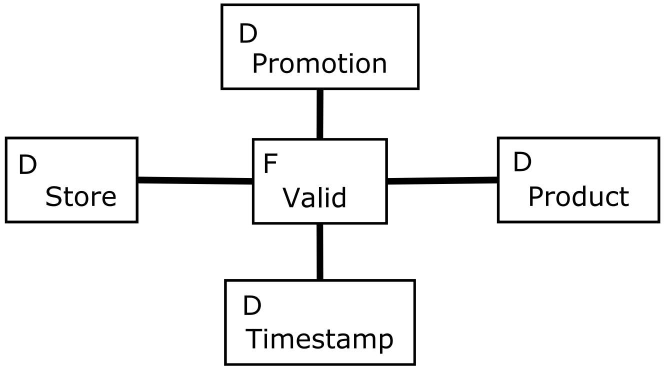
SELECT d.id, ~~SUM(p.expense),~~ SUM(sa.income)
FROM purchases p, sales sa, day d, store s
WHERE p.day=d.id AND p.store=s.id
          AND sa.day=d.id AND sa.store=s.id
          AND d.id="1234"
GROUP BY d.id
ORDER BY d.id

ADVANCED DESIGN

Factless Facts

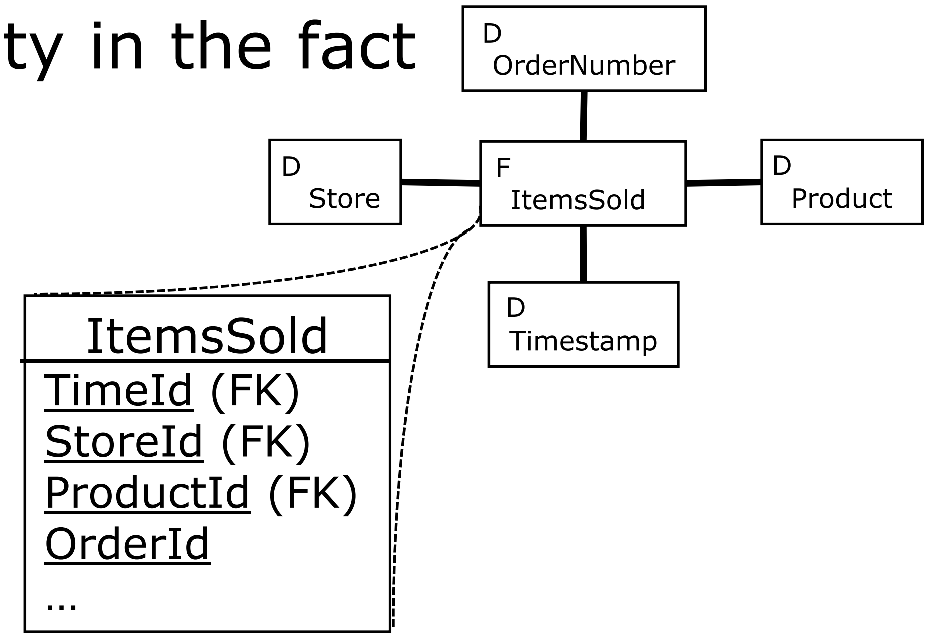
- Don't show measures
- Appear on recording data at the lowest granularity
- Two kinds of situations
  - Coverage
    - E.g., Products in promotion even if not sold
  - Events
    - E.g., Student's attendance





Degenerated Dimensions

- Do not contain other attributes than the key
  - No dimension table is necessary
  - We can save space by using just an integer
- Usefulness:
  a) Show finer granularity in the fact
  b) Group related facts
- Typical examples:
  - Order number
  - Invoice number
  - Ticket number



Junk Dimensions (I)

- Contain miscellaneous flags and text attributes
  a) Make one dimension each
  b) Leave them in the fact table
  c) Strip all them out
- It should be translated into one table with one combination per row
  - Complicates queries a little bit, but saves space in the fact table
  - If some are correlated, they should be packed together, to avoid the cartesian product
    - Data should be mined for relationships

Junk Dimensions (II)

a)

Sales
- TimeId (FK)
- StoreId (FK)
- ProductId (FK)
- GenreId (FK)
- AgeRangeID (FK)
- SalaryRangeID (FK)
- …

b)

Sales
- TimeId (FK)
- StoreId (FK)
- ProductId (FK)
- Genre
- AgeRange
- SalaryRange
- …

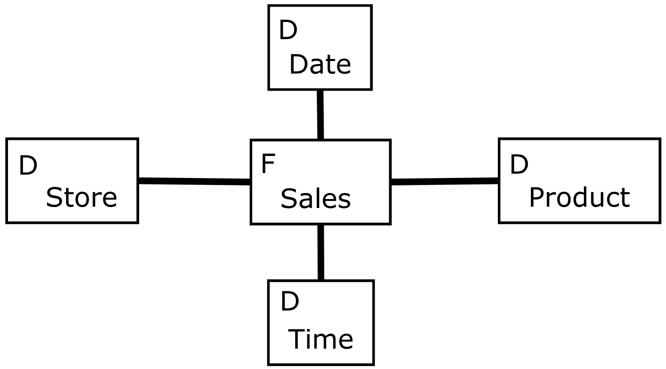
c)

Sales
- TimeId (FK)
- StoreId (FK)
- ProductId (FK)
- DemographicsID(FK)
- …

Demographics
- ID
- GenreId
- AgeRange
- SalaryRange
- …

Time dimensions

- One time dimension may show too many values:
  - 60*60*24*365*10 = 315,360,000
- Better to split it:
  - Time x Date
  - Time x Day x Year



Slowly Changing Dimensions (I)

"There is nothing permanent except change."

Heraclitus of Ephesus

- Dimensions also need to evolve
  - Reality changes
  - Correct errors

StoreId	Municipality	Region	Country
1	Pristina	Kosovo	Serbia
2	Barcelona	Arkansas	USA
3	…	…	…

Slowly Changing Dimensions (II)

Municipality	Region
M1	R1
M2	R2

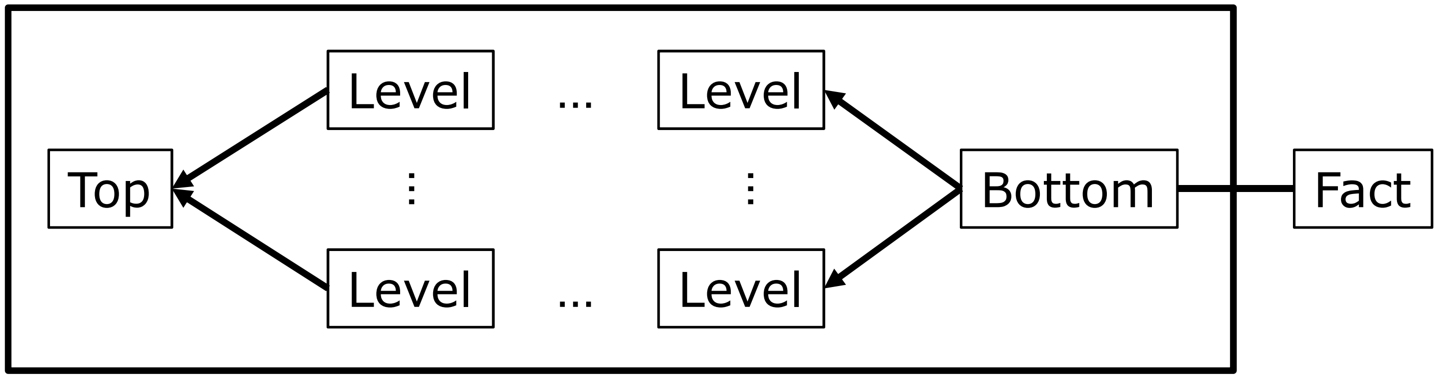
Municipality	Region
M1	R1
M2	R3

Municipality	Region	Reg_old
M1	R1	NULL
M2	R3	R2

Municipality	Region	Time
M1	R1	Now
M2	R2	T1
M2	R3	Now

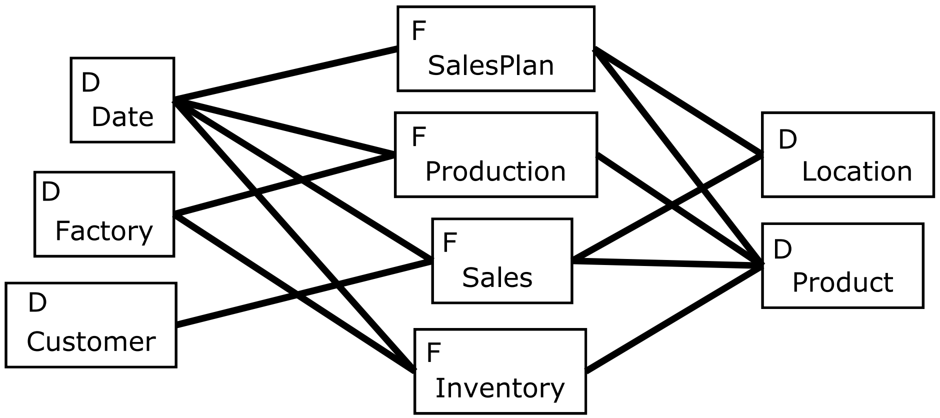
Aggregation hierarchies

- Can be explicit (snowflake) or implicit (star)
- Must be a lattice
  - One top and one bottom levels
  - Can be linear or not (i.e., multiple)
    - Parallel paths can be more or less independent
      - E.g., Fiscal year vs Calendar year
      - E.g., Book's author vs Book's genre



Conformed hierarchies/dimensions

	Customer	Factory	Date	Location	Product
Sales	X		X	X	X
Sales plan			X	X	X
Production		X	X		X
Inventory		X	X		X



NECESSARY SUMMARIZABILITY CONDITIONS

Aggregation problems (I)

- Number of students per department and year, assuming the students follow a two-year program

	1994	1995	1996	All
Informatics	15	17	13	28
Statistics	10	15	11	21
All	25	32	24	49

- Number of students per department and year, assuming the students follow a two-year program where there are inter-department courses

	1994	1995	1996	All
Informatics	15	17	13	28
Statistics	10	15	11	21
All	23	30	24	47

Necessary condition (I)

- Number of students per department and year, assuming the students follow a two-year program

	1994	1995	1996	[illegible]
Informatics	15	17	[illegible]	[illegible]
Statistics	10	15	[illegible]	21
All	25	[illegible]	24	49

Disjoint

- Number of students per department and year, assuming the students follow a two-year program where there are inter-department courses

	1994	1995	1996	All
Informatics	15	17	13	28
Statistics	10	15	11	21
All	23	30	24	47

Aggregation problems (II)

- Number of car accidents per province chief town and year

	1994	1995	1996	All
Barcelona	5	6	3	14
Tarragona	1	0	1	2
Lleida	0	2	1	3
Girona	3	5	6	14
Catalunya	20	23	22	65

Necessary condition (II)

- Number of car accidents per province chief town and year

	1994	1995	1996	All
Barcelona	5	6	3	14
Tarragona	1	0	1	2
Lleida	0	2	1	3
Girona	3	5	6	14
Catalunya	20	23	22	65

Complete

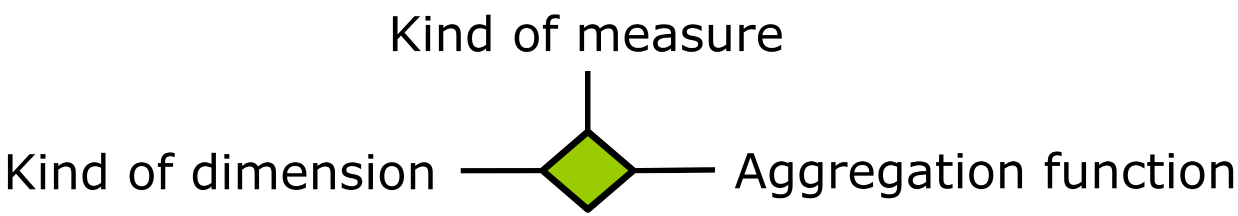
Aggregation problems (III)

	Cumulative	State	Value per unit
min	No problem	No problem	No problem
max	No problem	No problem	No problem
sum	No problem	Non-temporal	Never
avg	No problem	No problem	No problem

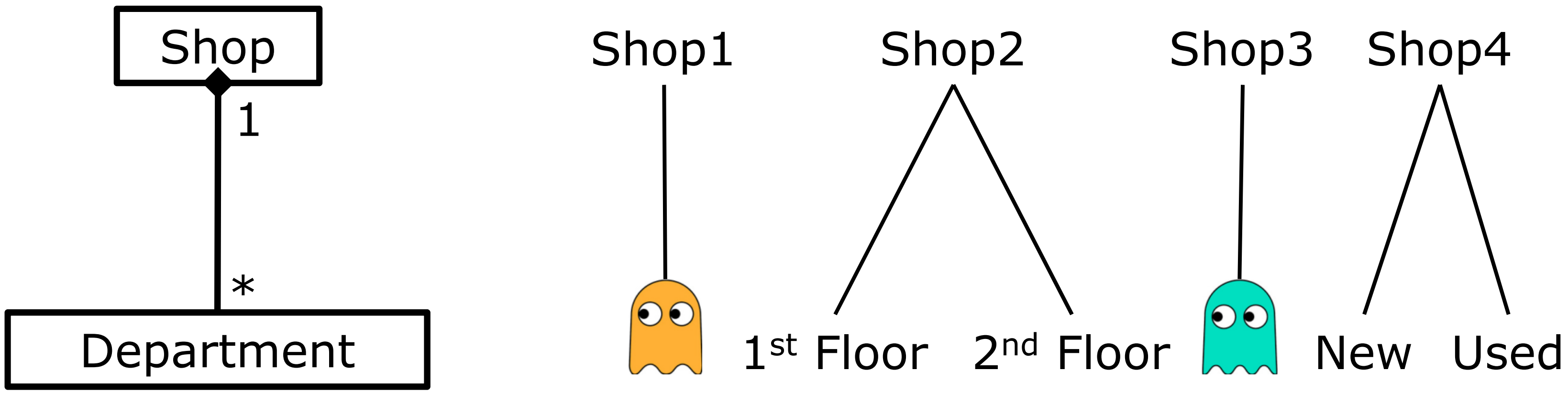
Necessary condition (III)

	Cumulative	State	Value per unit
min	No problem	No problem	No problem
max	No problem	No problem	No problem
sum	No problem	Non-temporal	Never
avg	No problem	No problem	No problem

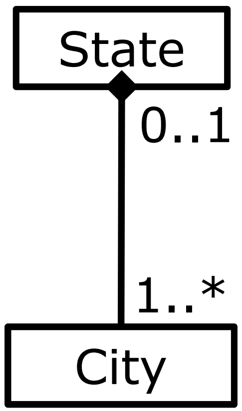
Compatible

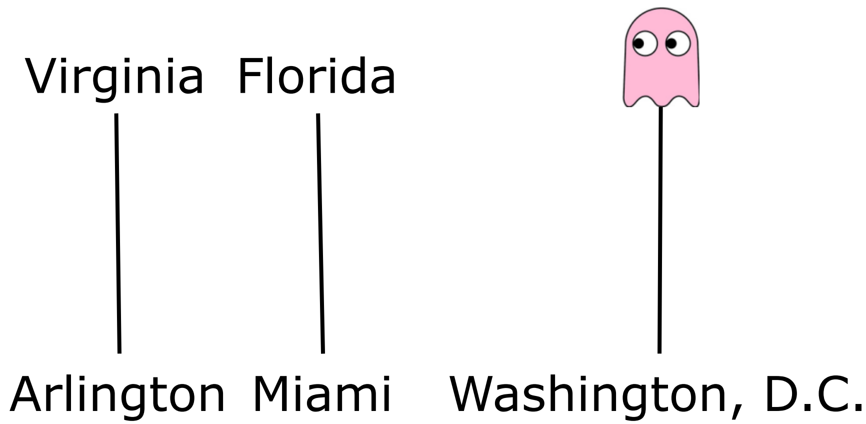


Unbalanced hierarchies

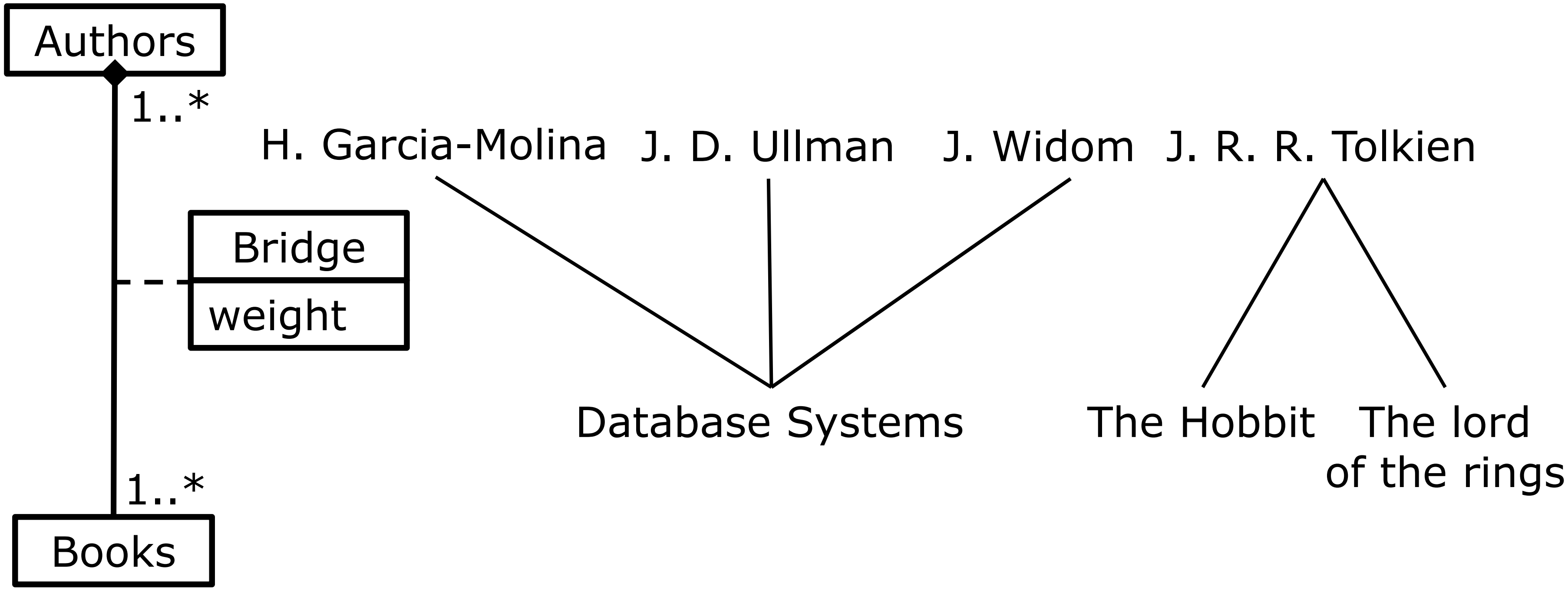


Non-covering hierarchies





Non-strict hierarchies



CLOSING

Summary

- Multidimensional operations
- Factless facts
- Degenerated dimensions
- Junk dimensions
- Slowly Changing Dimensions
- Aggregation problems
  - Summarizability necessary conditions
- Generalization of hierarchies
  - Unbalanced
  - Non-covering
  - Non-strict

Bibliography

- H. J. Lenz and A. Shoshani. *Summarizability in OLAP and statistical databases*. In *Proceedings of SSDBM'1997*. IEEE, 1997
- R. Kimball, L. Reeves, M. Ross and W. Thornthwaite. *The Data Warehouse lifecycle toolkit*. John Wiley & Sons, 1998
- C. S. Jensen, T. B. Pedersen, C. Thomsen. *Multidimensional Databases and Data Warehousing*. Morgan&Claypool Pulbishers, 2010
- M. Golfarelli and S. Rizzi. *Data Warehouse Design*. McGrau-Hill, 2009