



# Εξόρυξη Γνώσης από Δεδομένα

2ο μάθημα

Ηρακλής Βαρλάμης



# Αποθήκες δεδομένων και αναλυτική επεξεργασία δεδομένων



- Αποθήκη δεδομένων (Data warehouse):
- Είναι μια βάση δεδομένων για την υποστήριξη αποφάσεων
  - Διατηρείται ξεχωριστά από τη λειτουργική ΒΔ του οργανισμού
  - Συγκεντρώνει και οργανώνει ιστορικά δεδομένα του οργανισμού και τα προσφέρει για ανάλυση
- Η ανάλυση στις αποθήκες δεδομένων δε γίνεται σε επίπεδο συναλλαγής, αλλά σε μεγαλύτερες κλίμακες (χώρου, χρόνου κ.ά. Διαστάσεων)

# Χαρακτηριστικά ενός DW

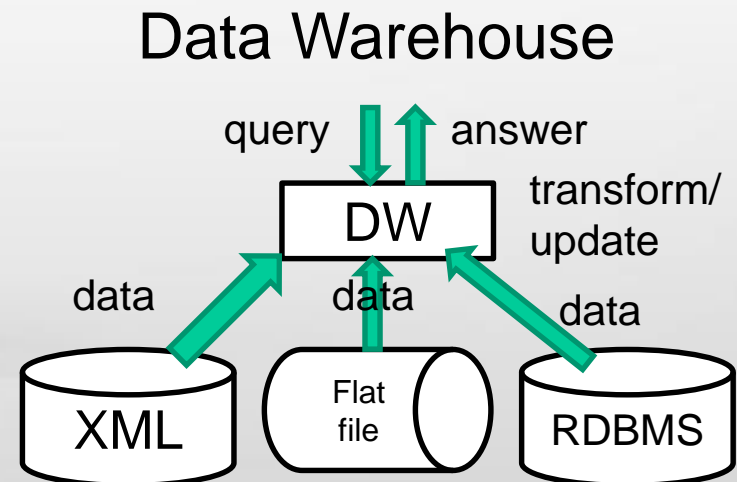
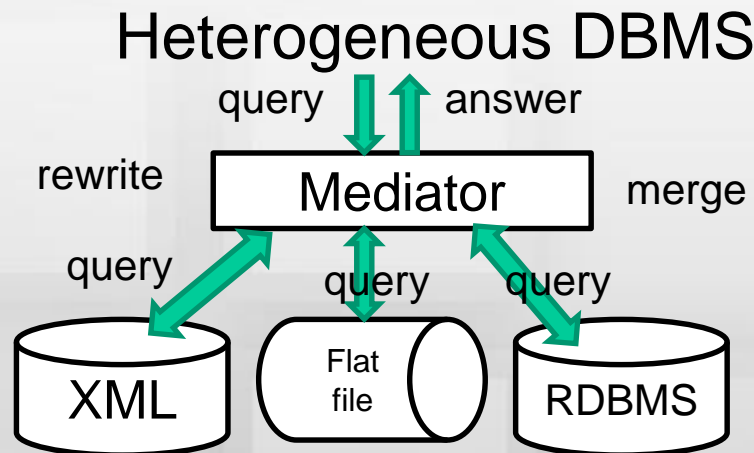
1. Εστιάζει σε συγκεκριμένο πρόβλημα
  - Πελάτες, προϊόντα, πωλήσεις
2. Ολοκληρώνει πληροφορία από διαφορετικές πηγές
  - Σχεσιακές ΒΔ, αρχεία, δεδομένα συναλλαγών
  - Αφήνει απέξω δεδομένα που δε συμβάλλουν στη λήψη αποφάσεων
  - Τα δεδομένα που μετακινούνται στην αποθήκη καθαρίζονται και τροποποιούνται
3. Εξελίσσεται με το χρόνο
  - Σε αντίθεση με τη λειτουργική ΒΔ, κρατά ιστορικό των δεδομένων για αρκετά χρόνια πριν
4. Είναι μόνιμα αποθηκευμένη (non volatile)
  - Δύο βασικές λειτουργίες: φόρτωση δεδομένων και πρόσβαση

*(William Inmon στα 80s)*



# Αποθήκες δεδομένων & ετερογενείς ΒΔ

- Και οι δύο ολοκληρώνουν πληροφορία από πολλές ΒΔ
- Οι ετερογενείς χρησιμοποιούν wrappers/mediators πάνω από κάθε ΒΔ
- Κάθε ερώτημα μετασχηματίζεται σε ερώτημα στις επιμέρους ΒΔ
- Οι αποθήκες δεδομένων συλλέγουν εκ των προτέρων τα δεδομένα από τις πηγές και τα ερωτήματα γίνονται απευθείας πάνω στα μετασχηματισμένα δεδομένα τους





# Αποθήκες δεδομένων και Λειτουργικές ΒΔ



- OLTP (on-line transaction processing)
  - Βασική εργασία στα σχεσιακά ΣΔΒΔ
  - Καθημερινές λειτουργίες: αγορές, τραπεζικές συναλλαγές, μισθοδοσία, παραγωγή κλπ.
- OLAP (on-line analytical processing)
  - Βασική εργασία στις αποθήκες δεδομένων
  - Ανάλυση δεδομένων και στήριξη αποφάσεων
- Διαφορές (OLTP vs. OLAP):
  - Προσανατολισμός στο χρήστη vs στο σύστημα: customer vs. market
  - Δεδομένα: τρέχοντα, λεπτομερή vs. Ιστορικά, συγκεντρωτικά
  - Σχεδίαση της ΒΔ: ER διάγραμμα vs. star ή snowflake
  - Πρόσβαση: ενημερώσεις και τετριμμένα queries vs. read-only σύνθετα ερωτήματα



# OLTP vs. OLAP

	OLTP	OLAP
<b>users</b>	clerk, IT professional	knowledge worker
<b>function</b>	day to day operations	decision support
<b>DB design</b>	application-oriented	subject-oriented
<b>data</b>	current, up-to-date detailed, flat relational isolated	historical, summarized, multidimensional integrated, consolidated
<b>usage</b>	repetitive	ad-hoc
<b>access</b>	read/write index/hash on prim. key	lots of scans
<b>unit of work</b>	short, simple transaction	complex query
<b># records accessed</b>	tens	millions
<b>#users</b>	thousands	hundreds
<b>DB size</b>	100MB-GB	100GB-TB
<b>metric</b>	transaction throughput	query throughput, response



# Γιατί χρειαζόμαστε ξεχωριστό DW

- Υψηλές επιδόσεις στα δύο συστήματα
  - Το ΣΔΒΔ έχει ρυθμιστεί για OLTP: ευρετήρια, έλεγχος συνδρομικότητας, ανάνηψη
  - Η αποθήκη δεδομένων έχει ρυθμιστεί για OLAP: σύνθετα ερωτήματα, πολυδιάστατες όψεις, συγκεντρωτικά δεδομένα
- Διαφορετικές λειτουργίες, διαφορετικά δεδομένα
  - Missing data: Η λήψη αποφάσεων απαιτεί δεδομένα σε βάθος χρόνου που δεν διατηρούνται στη λειτουργική ΒΔ.
  - Data consolidation: Για τη λήψη αποφάσεων απαιτείται ολοκλήρωση, περίληψη κλπ δεδομένων
  - Data quality: διαφορετικές πηγές μπορεί να έχουν διαφορετικές και ασύμβατες αναπαραστάσεις των δεδομένων

# Πολυδιάστατο μοντέλο δεδομένων

- Στηρίζεται σε πίνακες αλλά έχει πολλές διαφορές από το σχεσιακό μοντέλο
- Βασικό μοντέλο δεδομένων για την αποθήκη δεδομένων είναι ο πολυδιάστατος κύβος δεδομένων (data cube)
- Ένας κύβος για τις πωλήσεις (sales) έχει πολλαπλές διαστάσεις
  - Ένα πίνακα για κάθε διάσταση (dimension table): προϊόν (item), χρονική περίοδος (time)
  - Ένα πίνακα μετρικών (fact table) με πεδία που αφορούν τον τζίρο ή το πλήθος αντικειμένων που πουλήθηκαν και κλειδιά για κάθε πίνακα διάστασης







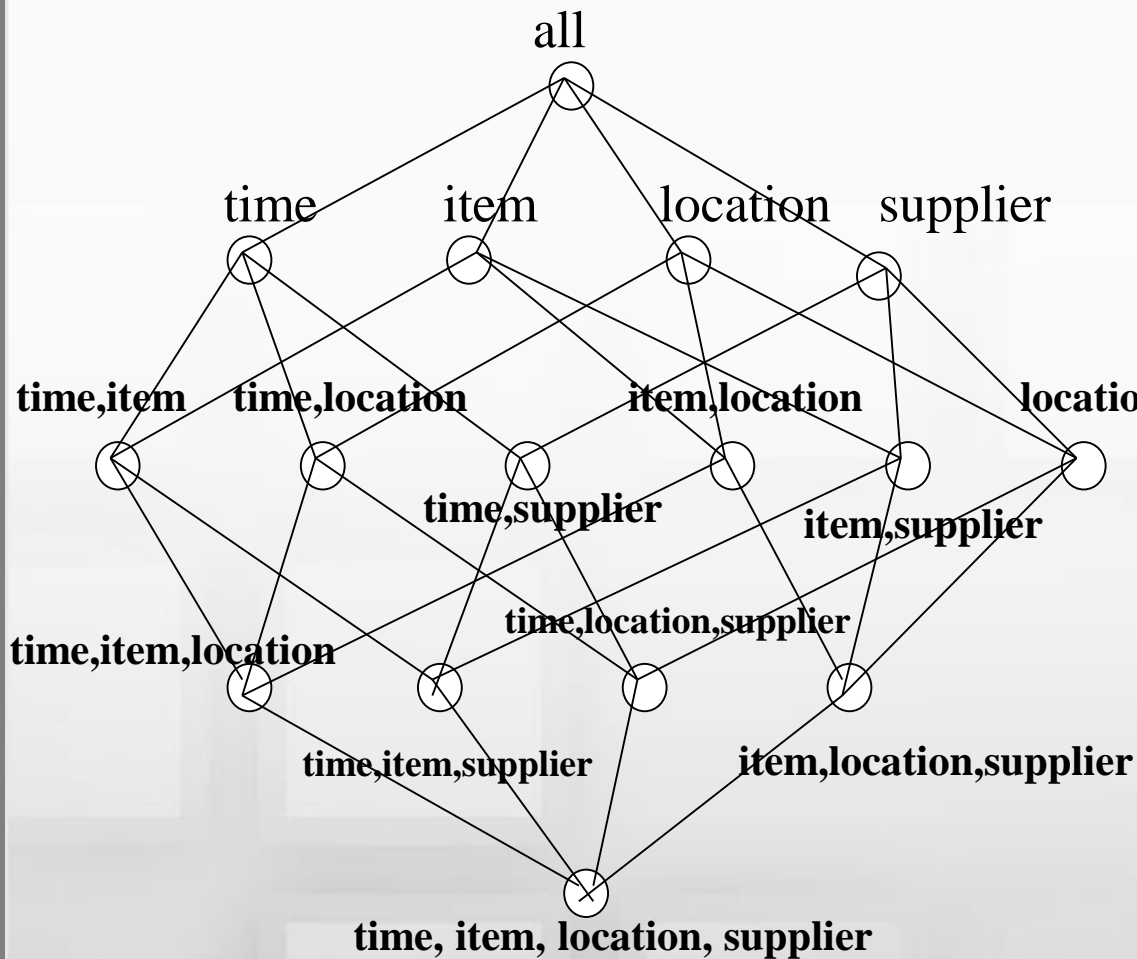
# Κύβος δεδομένων



- Σε κάθε διάσταση μπορούμε να εξειδικεύουμε (drill down) ή να γενικεύουμε (roll up) εστιάζοντας σε διαφορετικό βαθμό συνάθροισης (π.χ. μήνα, τρίμηνο, εξάμηνο, έτος)
- Ο n-D κύβος με τον χαμηλότερο βαθμό γενίκευσης (για την εκάστοτε διάσταση) ονομάζεται base cuboid (πλησιάζει στο επίπεδο συναλλαγής)
- Ο κορυφαίος 0-D cuboid έχει το υψηλότερο βαθμό περίληψης (γενίκευσης) και ονομάζεται apex cuboid
- Το σύνολο όλων των cuboids αποτελεί τον κύβο δεδομένων (data cube)



# Ένα πλέγμα κύβων



0-D(apex) cuboid

1-D cuboids

2-D cuboids

3-D cuboids

4-D(base) cuboid

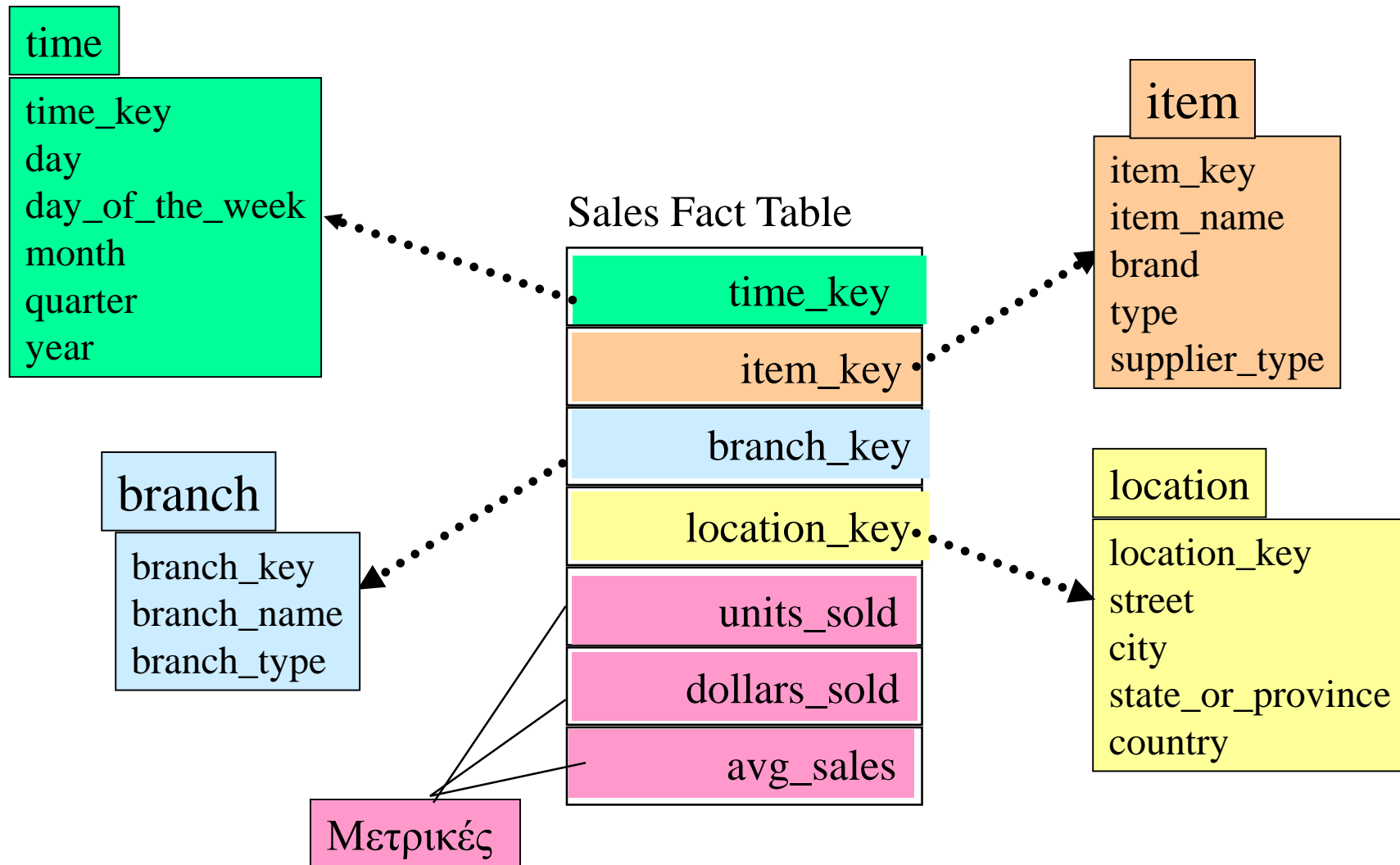
# Απεικόνιση του μοντέλου



- Star schema: Ο πίνακας μετρικών (fact table), στο κέντρο, συνδέεται με καθένα από τους πίνακες διαστάσεων με ξένα κλειδιά
- Snowflake schema: Όπως το star schema μόνο που ορισμένοι πίνακες διαστάσεων κανονικοποιούνται (σε περισσότερους πίνακες)
- Fact constellations: Πολλοί πίνακες μετρικών μοιράζονται από κοινού πίνακες διαστάσεων



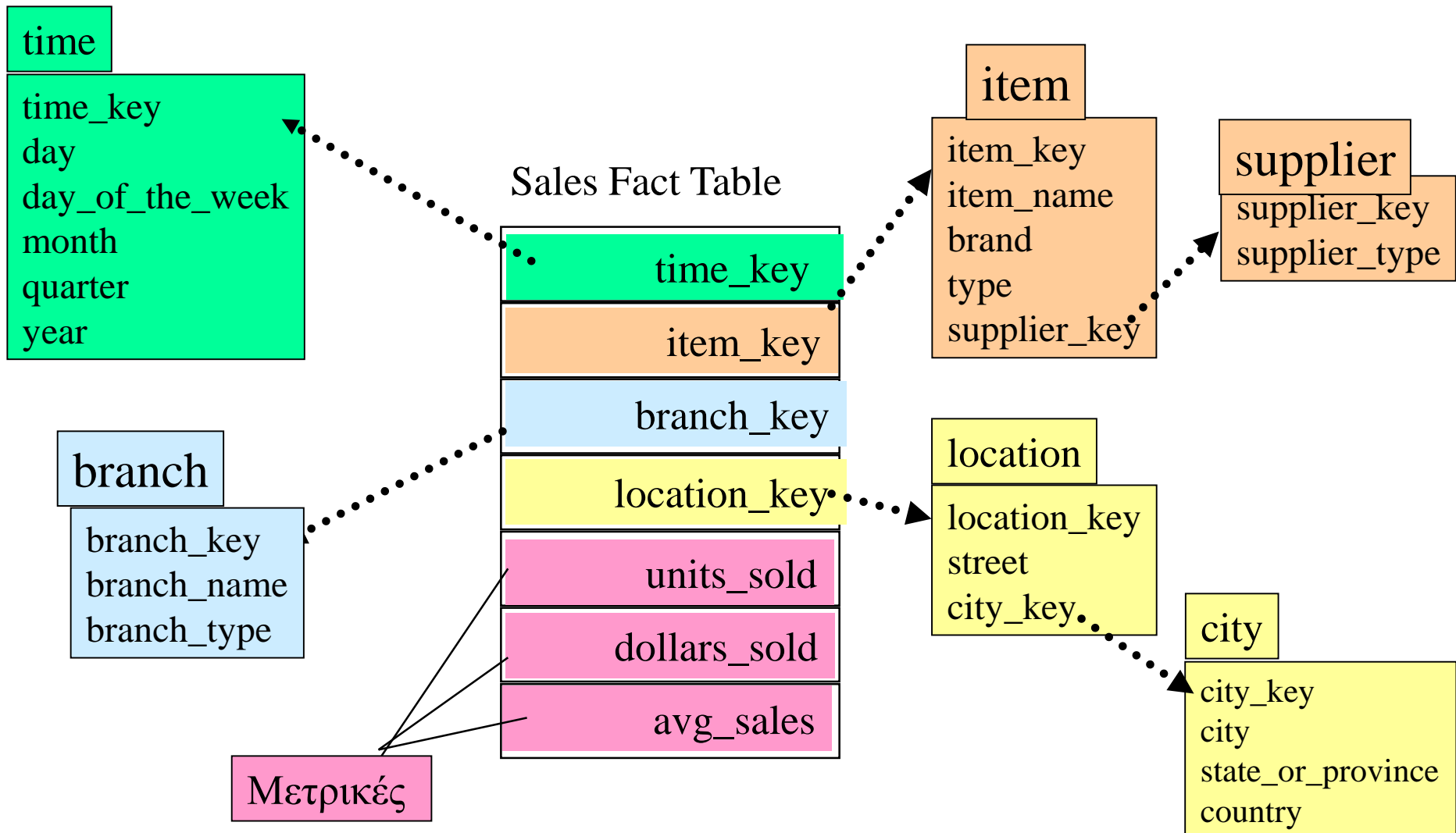
# Παράδειγμα Star Schema





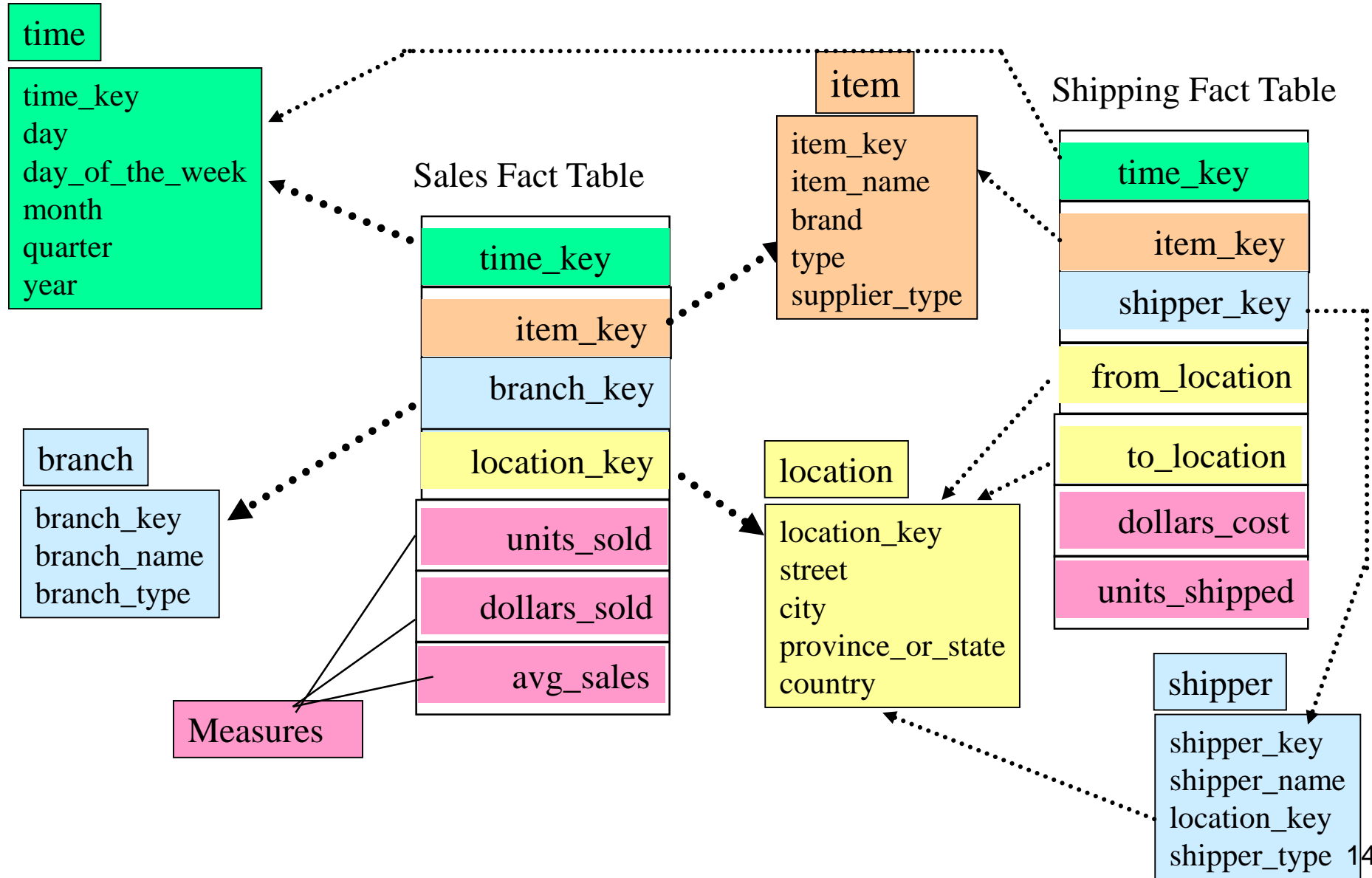


# Παράδειγμα Snowflake Schema





# Παράδειγμα Fact Constellation





# DML: Η γλώσσα ορισμού σχήματος

Sales Fact Table

time_key
item_key
branch_key
location_key
units_sold
dollars_sold
avg_sales

**define cube** sales\_star [time, item, branch, location]:

dollars\_sold = sum(sales\_in\_dollars), avg\_sales = avg(sales\_in\_dollars), units\_sold = count(\*)

**define dimension** time **as** (time\_key, day, day\_of\_week, month, quarter, year)

**define dimension** item **as** (item\_key, item\_name, brand, type, supplier\_type)

**define dimension** branch **as** (branch\_key, branch\_name, branch\_type)

**define dimension** location **as** (location\_key, street, city, province\_or\_state, country)





# Μετρικές

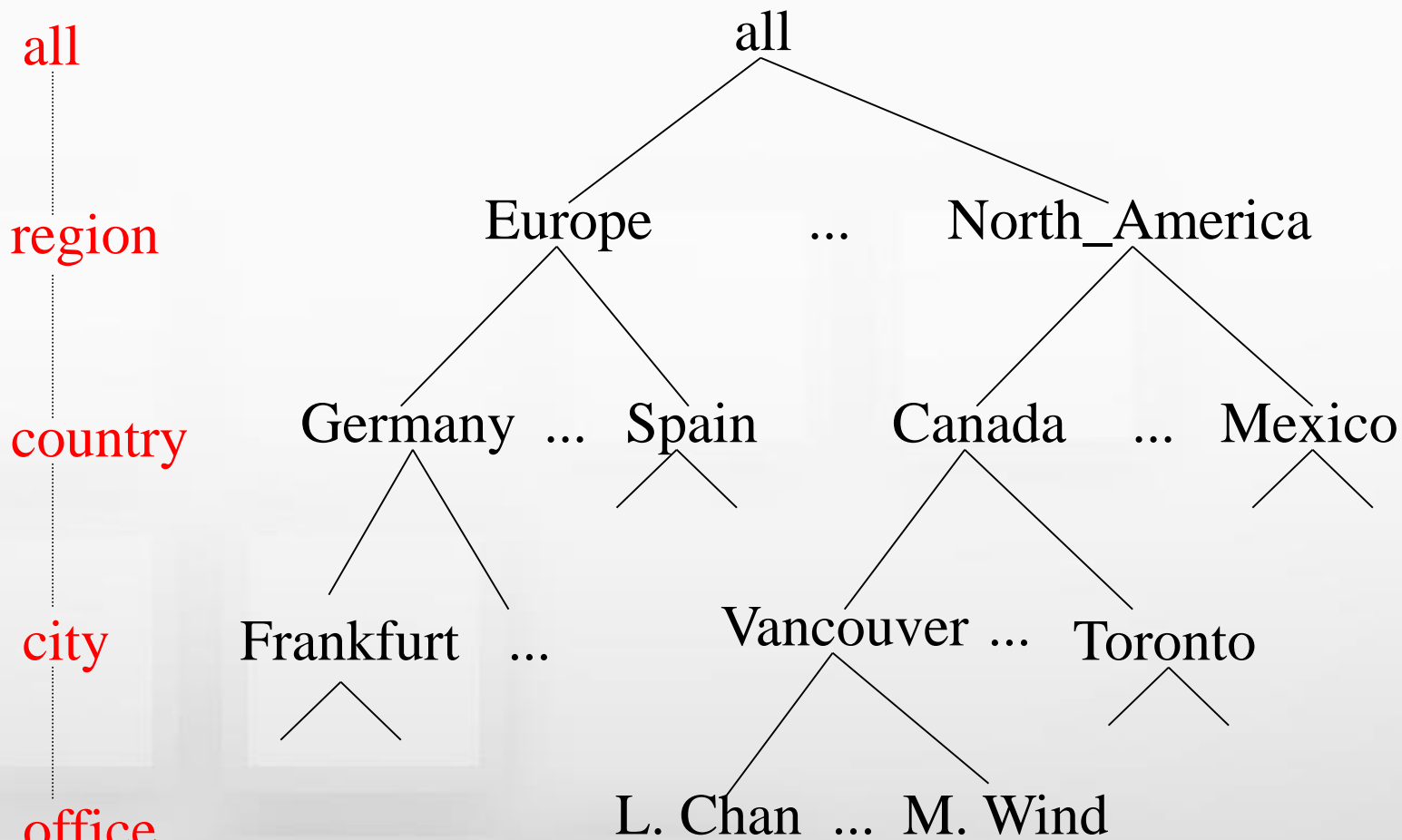


- **distributive:** Αν το αποτέλεσμα που προκύπτει αν εφαρμόσουμε τη συνάρτηση σε  $N$  αθροιστικές τιμές είναι το ίδιο με το να εφαρμόζαμε απευθείας τη συνάρτηση σε όλες τις τιμές χωρίς ομαδοποίηση
  - Π.χ. `count()`, `sum()`, `min()`, `max()`
- **algebraic:** Αν το αποτέλεσμα μπορεί να υπολογιστεί από μια αλγεβρική συνάρτηση με  $M$  ορίσματα ( $M$  φραγμένος ακέραιος), καθένα από τα οποία υπολογίζεται εφαρμόζοντας μια distributive function
  - Π.χ. `avg()`, `min_N()`, `standard_deviation()`
- **holistic:** αν δεν υπάρχει σταθερά που να εξαρτάται από το μέγεθος του αποτελέσματος της συνάθροισης
  - Π.χ. `median()`, `mode()`, `rank()`





# Παράδειγμα διάστασης: Location



# Ιεραρχίες διαστάσεων

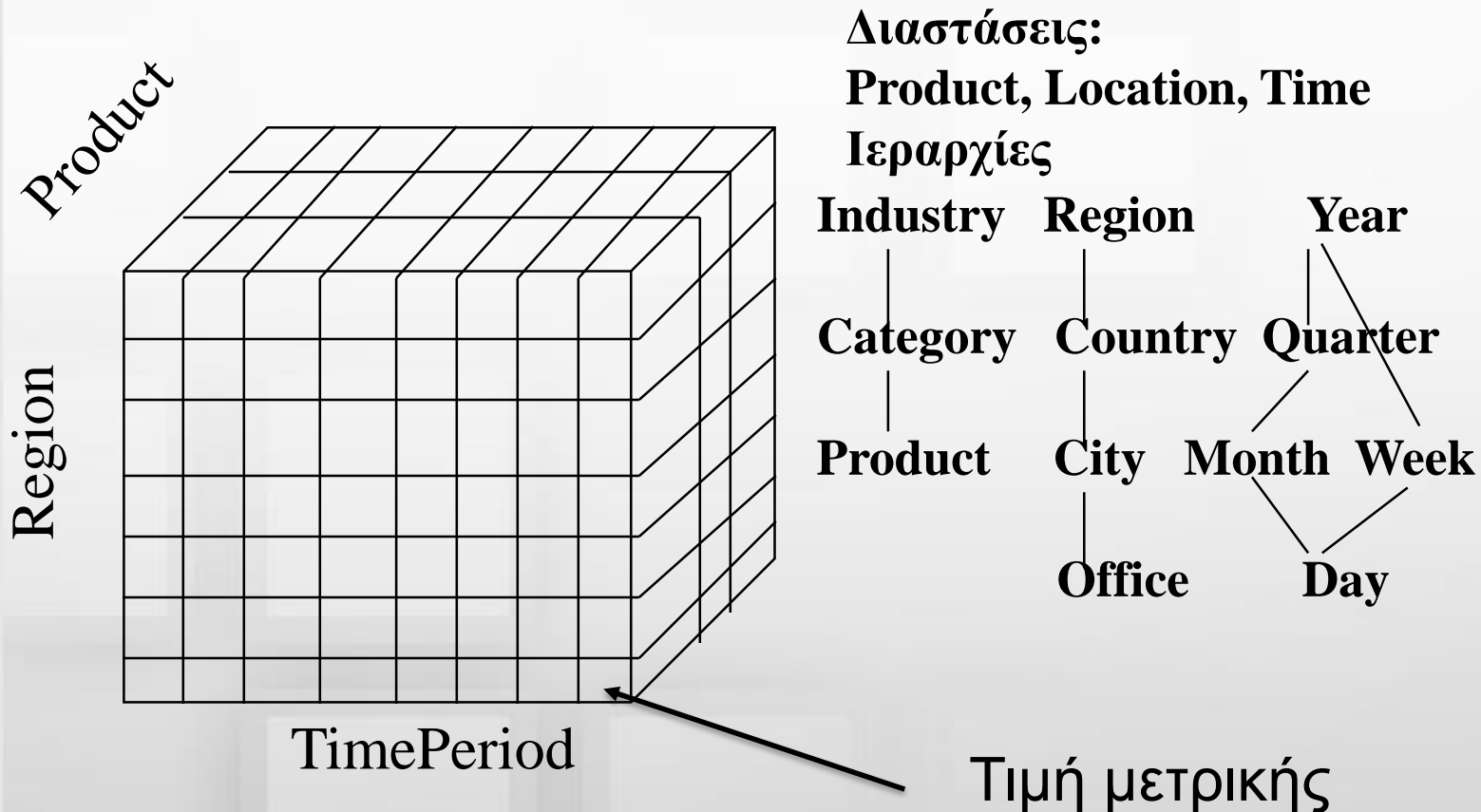
- Ενοιολογική ιεραρχία (concept hierarchy)
  - π.χ. all < region < country < city < office
- Ιεραρχία σχήματος (schema hierarchy)
  - π.χ. day < {week | month} < quarter < year
- Ιεραρχία ομαδοποίησης (set\_grouping)
  - π.χ. {0..30} young < {31-50} midaged < {50-..} old





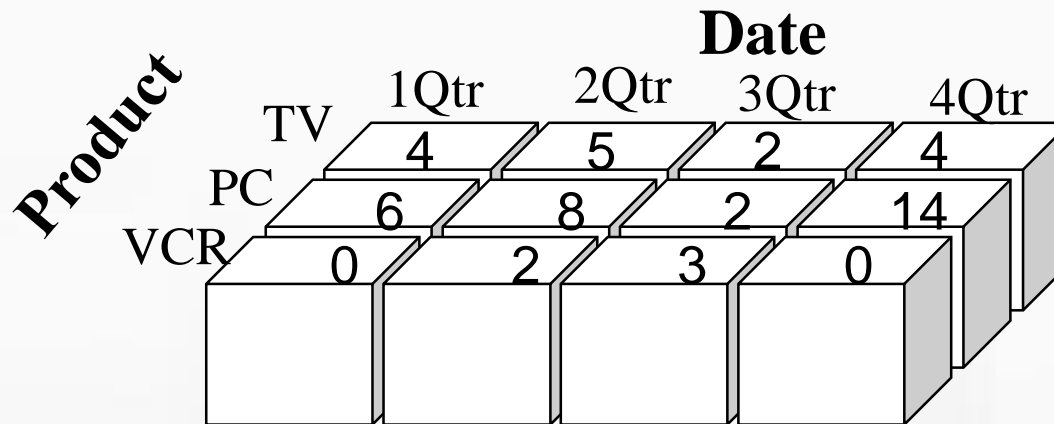
# Πολυδιάστατος κύβος

- Όγκος πωλήσεων σε συνάρτηση με το προϊόν την περίοδο και την περιοχή





# Παράδειγμα: Πωλήσεις ηλεκτρονικών συσκευών



U.S.A

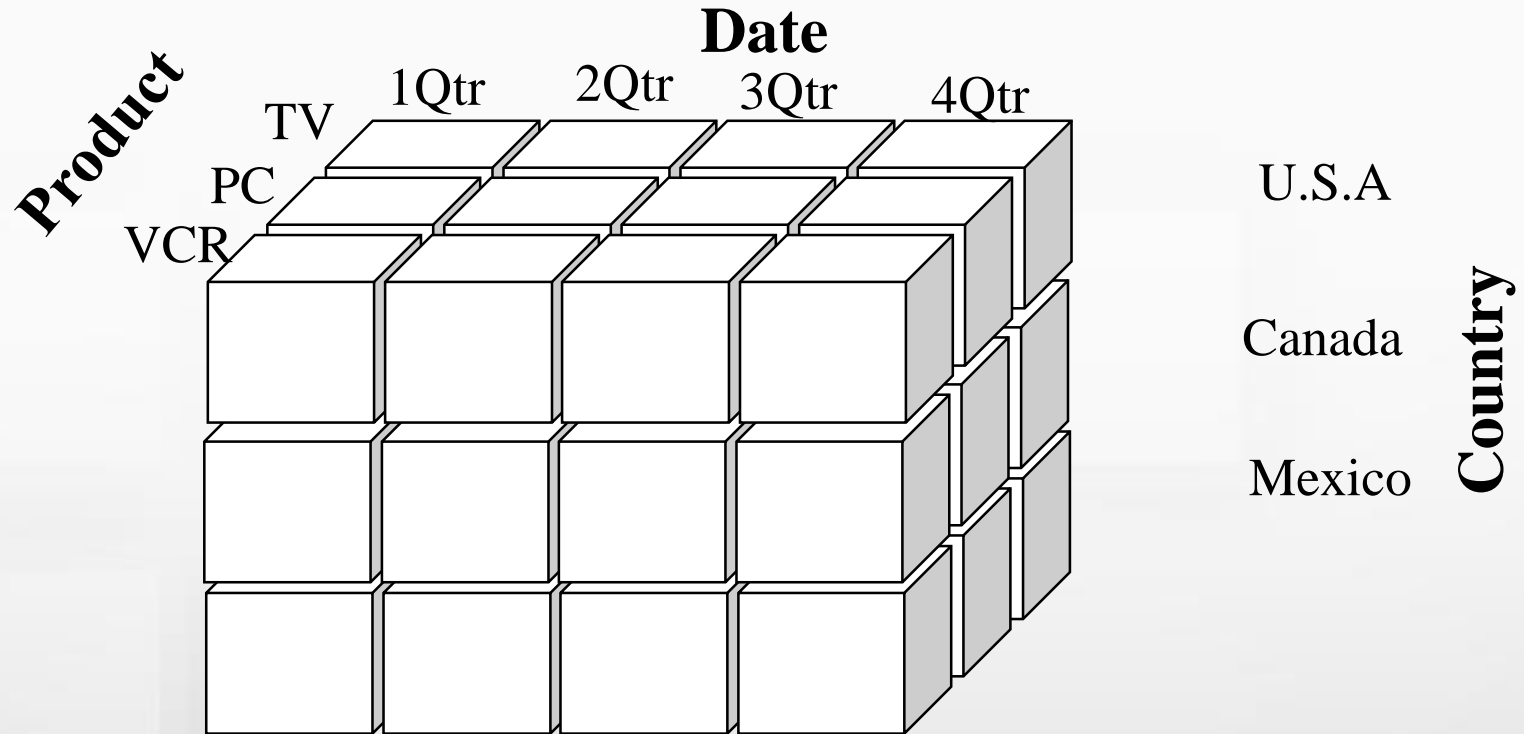
Canada

Mexico

**Country**

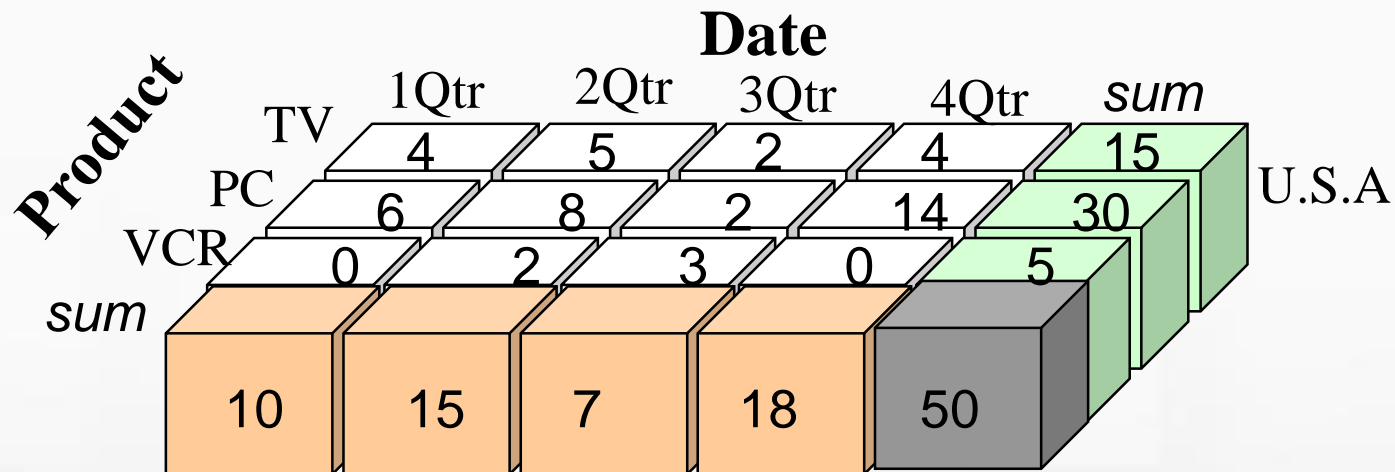


# Παράδειγμα: Πωλήσεις ηλεκτρονικών συσκευών



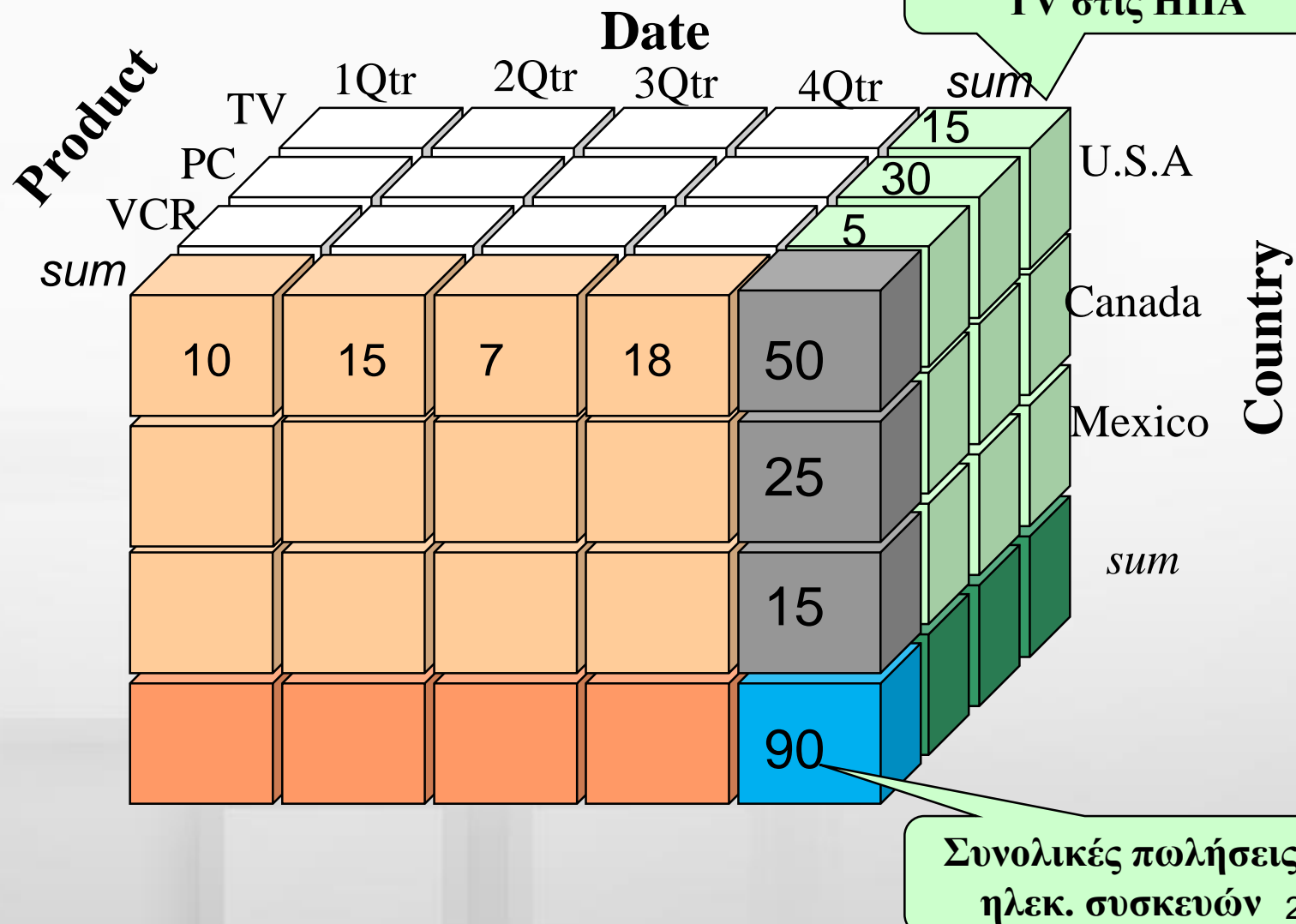


# Παράδειγμα: Πωλήσεις ηλεκτρονικών συσκευών



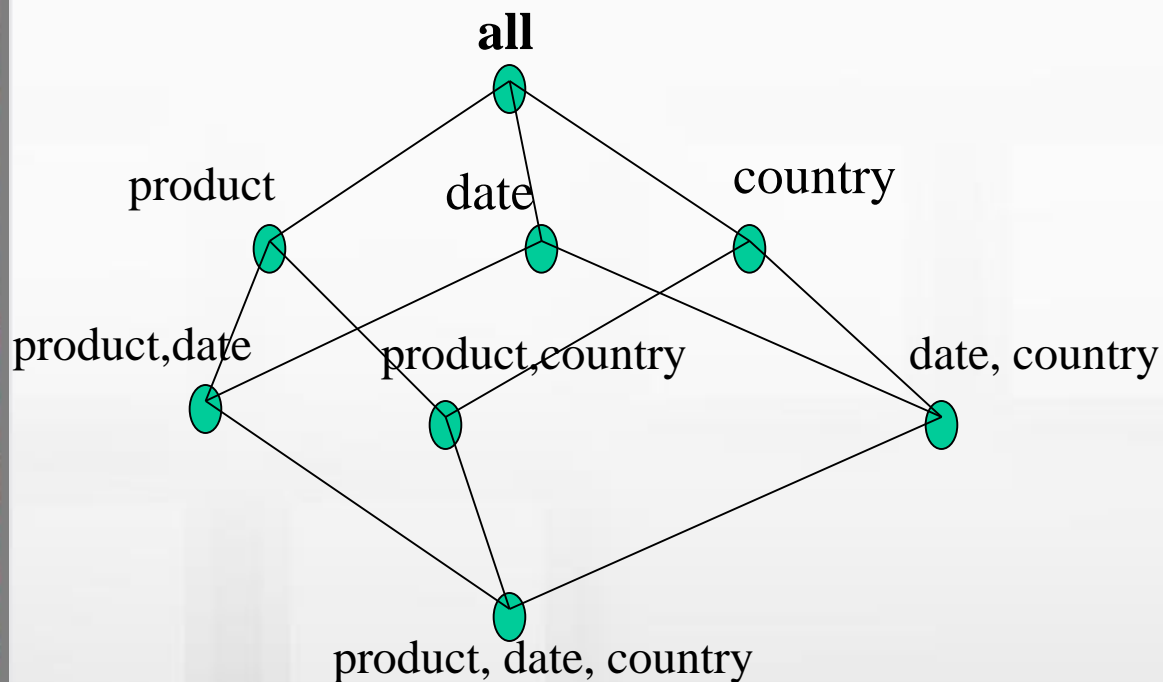


# Παράδειγμα





# Cuboids που υλοποιούνται



0-D(apex) cuboid

1-D cuboids

2-D cuboids

3-D(base) cuboid





# Τυπικές ενέργειες OLAP

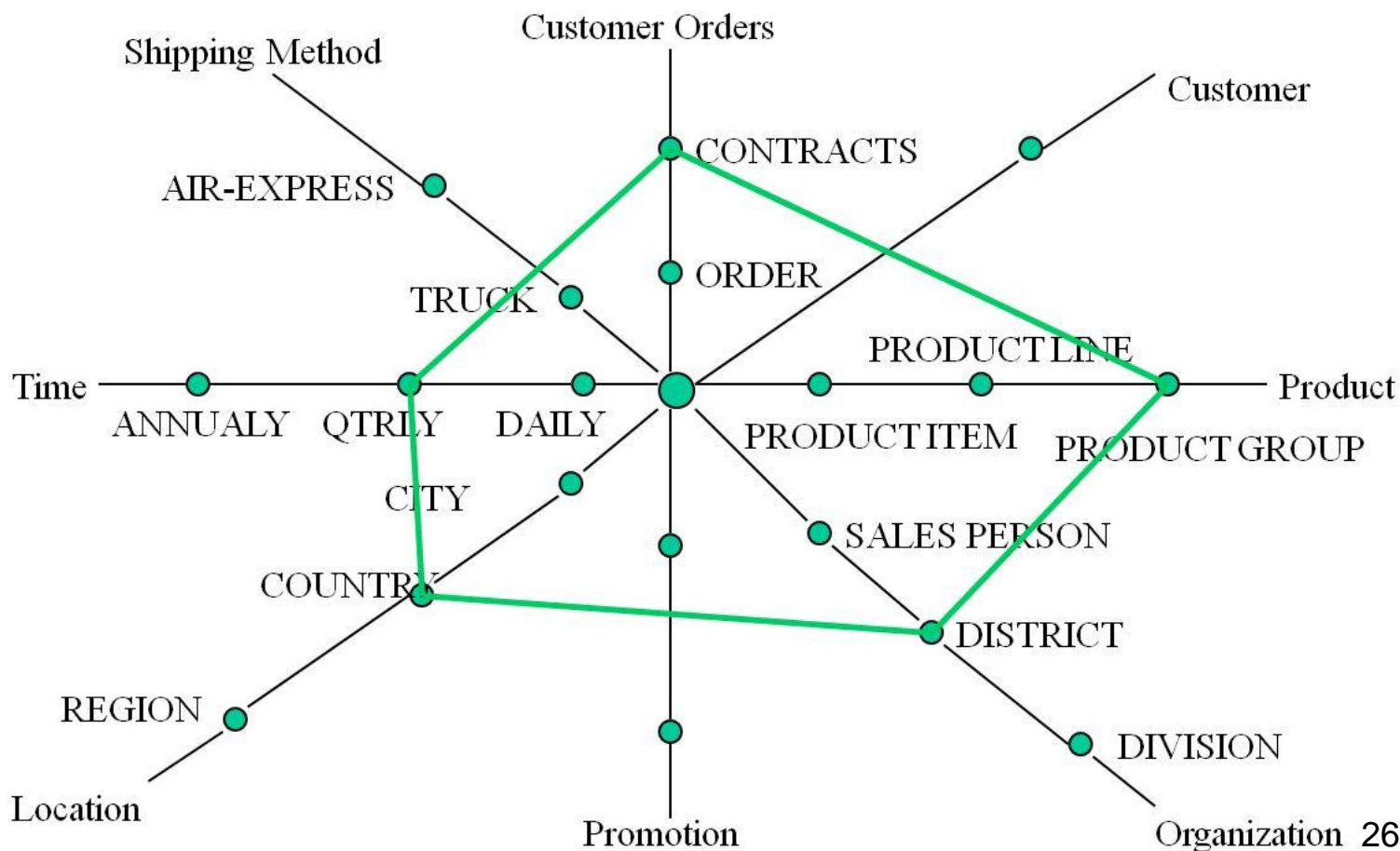


- Roll up (drill-up): Περίληψη (σύνοψη) δεδομένων
  - Ανεβαίνοντας την ιεραρχία ή μειώνοντας τις διαστάσεις
- Drill down (roll down): Ανάλυση δεδομένων
  - Κατεβαίνοντας προς πιο αναλυτικά δεδομένα ή βάζοντας νέες διαστάσεις
- Slice and dice: προβολή και επιλογή δεδομένων
- Pivot (rotate): αλλαγή προσανατολισμού του κύβου, οπτικοποίηση προβολή σε 3Δ, 2Δ κλπ
- Drill through: ανάλυση που φτάνει μέχρι τα δεδομένα στο σχεσιακό μοντέλο
- Drill across: ταυτόχρονη ανάλυση περισσότερων fact tables (και συνεπώς κύβων)



# Σύνδεση με SQL

- Κάθε κύκλος είναι ένα ερώτημα συνάθροισης στο σχεσιακό μοντέλο
- Η ακμή ορίζει ένα κύβο





# APXITEKTONIKH TOY DW



# Πλαίσιο ανάλυσης δεδομένων



- Top down view
  - Επιλογή των πληροφοριών που θα εισαχθούν στην αποθήκη
- Data source view
  - Καθορισμός των δεδομένων που θα συλλέγονται και θα αποθηκεύονται στη λειτουργική ΒΔ του οργανισμού
- Data warehouse view
  - Καθορισμών των fact και dimension πινάκων
- Business query view
  - Ποια είναι η γνώση που θέλουμε να εξάγουμε προς τον τελικό χρήστη





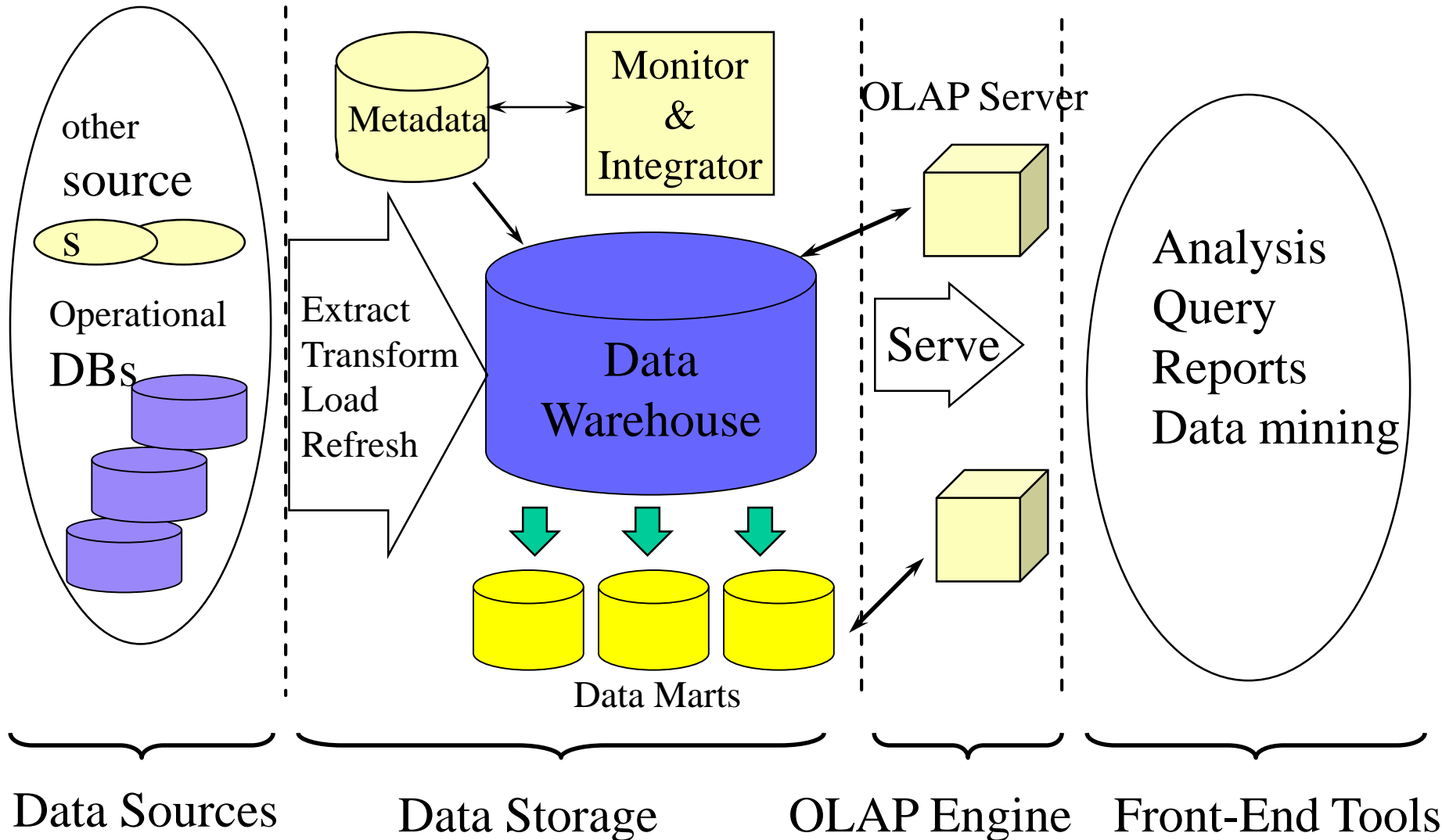
# Σχεδίαση της αποθήκης



- Επιλογή της επιχειρηματικής διαδικασίας που θα αναλυθεί (π.χ. Παραγγελίες, πληρωμές κλπ)
- Επιλογή του επιπέδου λεπτομέρειας (granularity) των δεδομένων που θα χρησιμοποιηθούν
- Επιλογή των διαστάσεων που θα περιέχει κάθε fact table
- Επιλογή της μετρικής που θα γεμίσει κάθε εγγραφή του fact table



# Πολυεπίπεδη αρχιτεκτονική





# Εκδοχές του Data Warehouse



- Enterprise warehouse
  - Συλλέγει όλα τα δεδομένα για όλους τους τομείς δραστηριότητας του οργανισμού
- Data Mart
  - Περιέχει ένα υποσύνολο των δεδομένων του οργανισμού που αφορούν συγκεκριμένες ομάδες χρηστών, π.χ. Marketing
- Virtual warehouse
  - Δεν έχει δικά του δεδομένα
  - Υλοποιείται με όψεις στη λειτουργική ΒΔ



# Εναλλακτικές αρχιτεκτονικές



- Relational OLAP (ROLAP)
  - Χρησιμοποιεί ένα σχεσιακό ΣΔΒΔ για να αποθηκεύει και να διαχειρίζεται τα δεδομένα και ένα OLAP middleware για τις υπόλοιπες λειτουργίες (Microsoft Analysis Services, SAP Business Objects, Oracle Business Intelligence Suite Enterprise Edition (former Siebel Analytics))
  - Βελτιστοποιεί το DBMS backend, υλοποιώντας τη λογική του roll-up/drill-down στις συναθροίσεις
  - Προσφέρει καλύτερη κλιμάκωση
- Multidimensional OLAP (MOLAP)
  - Αποθηκεύει σε πολυδιάστατους πίνακες (sparse matrix techniques) (array-based)
  - Γρήγορη ευρετηρίαση σε προϋπολογισμένες περιλήψεις των δεδομένων
- Hybrid OLAP (HOLAP)
  - Συνδυάζουν σχεσιακό μοντέλο σε χαμηλό επίπεδο και array storage στα πιο ψηλά επίπεδα

# Αποδοτική υλοποίηση του data cube

- Πλέγμα cuboids

- Στο κατώτερο επίπεδο έχουμε τα base cuboids
- Στο ανώτερο επίπεδο συνάθροισης έχουμε το apex cuboid με ένα μόνο κελί
- Σε ένα n-διάστατο κύβο με  $L_i$  επίπεδα σε κάθε διάσταση έχω: 
$$T = \prod_{i=1}^n (L_i + 1) \text{ cuboids}$$

- Υλοποίηση (materialization)

- Πλήρης (όλα τα cuboids), καμία, ή μερική
- Επιλογή των cuboids που θα υλοποιήσω με βάση: το μέγεθος, τον διαμοιρασμό, τη συχνότητα πρόσβασης κλπ.



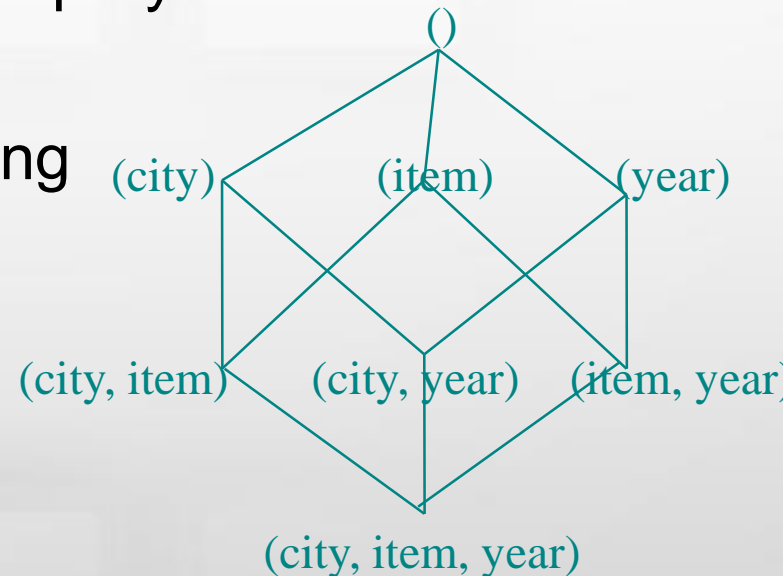


# Υλοποίηση κύβου

- Δημιουργία των πινάκων του star schema
- Ορισμός του κύβου (διαστάσεις και cuboid)
- Υλοποίηση

– Υπολογισμός των group-by  
σε κάθε διάσταση

Μπορεί να απαιτεί sorting  
hashing, grouping στα  
γνωρίσματα κάθε  
διάστασης.





# Δημιουργία cuboid (1/3)

- Ορισμός διαστάσεων
- CREATE DIMENSION products  
LEVEL item IS (products.item\_key)  
LEVEL subtype IS (products.subtype\_key)  
LEVEL type IS (products.type\_key)  
LEVEL category IS (products.category\_key)  
LEVEL department IS (products.department\_key)  
HIERARCHY departments (  
item CHILD OF  
subtype CHILD OF  
type CHILD OF  
category CHILD OF  
department)  
ATTRIBUTE item DETERMINES (item\_name)  
ATTRIBUTE subtype DETERMINES (subtype\_name)  
ATTRIBUTE type DETERMINES (type\_name)  
ATTRIBUTE category DETERMINES (category\_name)  
ATTRIBUTE department DETERMINES (department\_name);



# Δημιουργία cuboid (2/3)

- Ορισμός ενός cuboid ως materialized view
- CREATE materialized VIEW sales\_mon\_ite\_cit\_cha\_mv build deferred  
AS  
SELECT t.month\_name, p.item\_name, cu.city\_name,  
ch.channel\_name, SUM(f.sales) sales, SUM(f.quantity) quantity  
FROM times t, products p, customers cu, channels ch,  
sales\_fact f  
WHERE t.day\_key = f.day\_key  
AND p.item\_key = f.product  
AND cu.customer\_key = f.customer  
AND ch.channel\_key = f.channel  
GROUP BY t.month\_name,  
p.item\_name,  
cu.city\_name,  
ch.channel\_name;





# Δημιουργία cuboid (3/3)

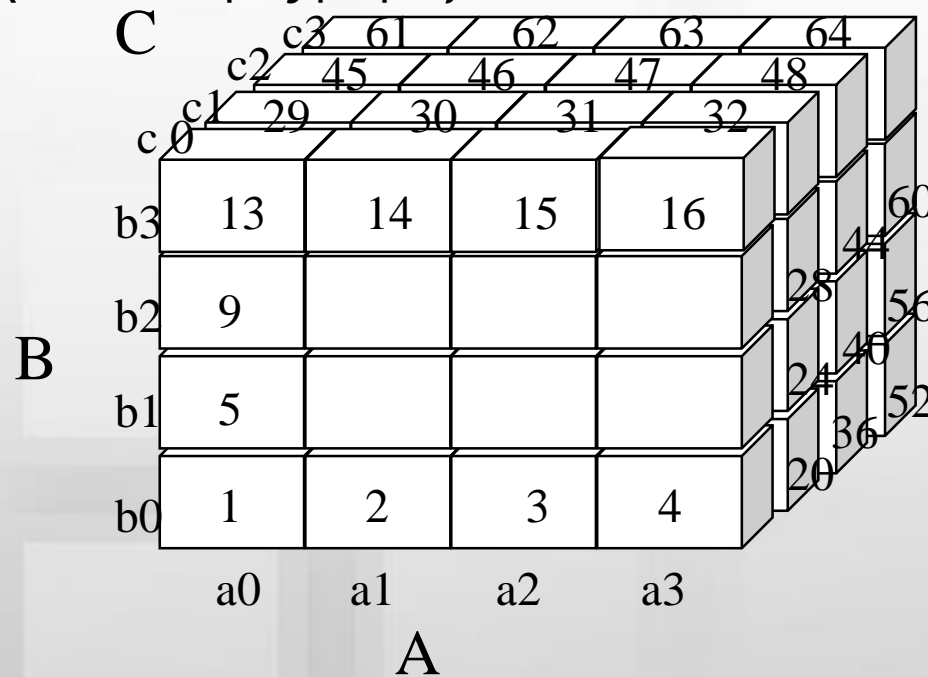
- Υπολογισμός του κύβου
- declare  
myCubeMv varchar2(32);  
begin  
myCubeMv := **dbms\_cube.create\_mview**( mvOwner =>user,  
mvName =>'sales\_mon\_ite\_cit\_cha\_mv',  
sam\_parameters=>'logDest=serverout,build=immediate' );  
end;



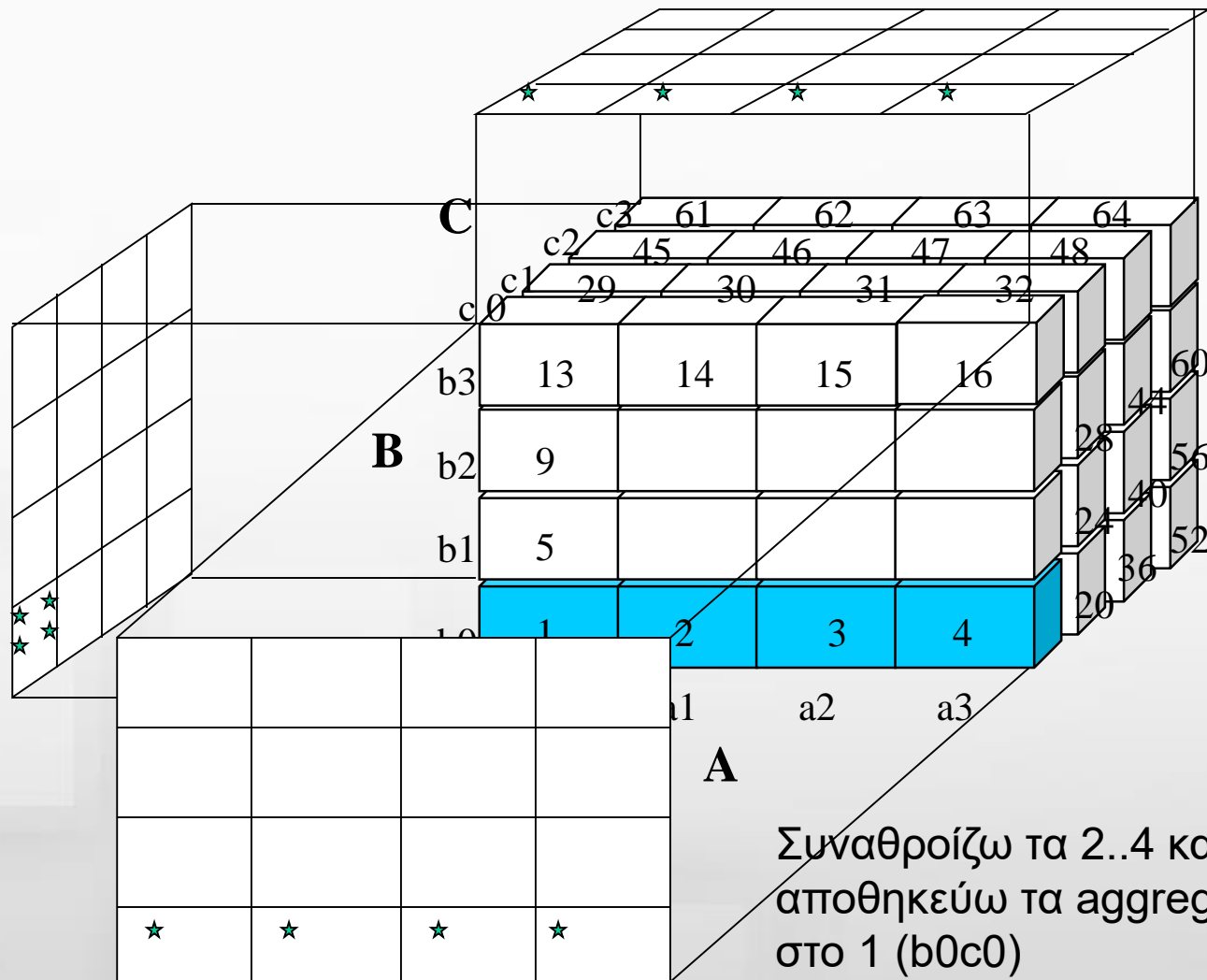


# Κύβος = Συναθροίσεις

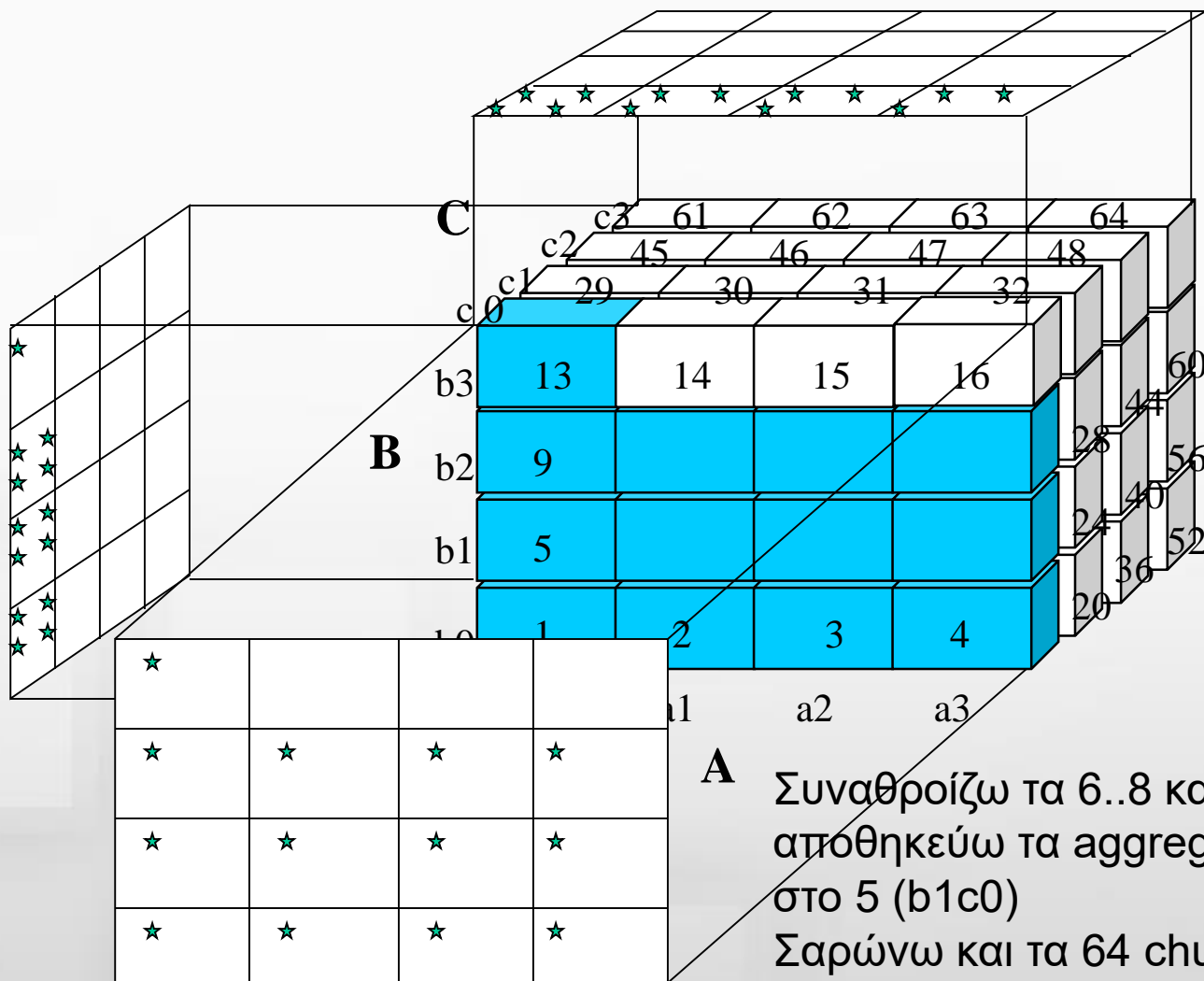
- Συναθροίσεις σε πολλούς άξονες
  - Τεμαχισμός του κύβου σε κομμάτια (chunks) που χωρούν στη μνήμη
  - Υπολογισμός των συναθροίσεων σε πολλούς άξονες ταυτόχρονα: επισκεπτόμαστε τα chunks με τρόπο που να ελαχιστοποιεί τις επισκέψεις κάθε chunk (τα I/Os)
  - π.χ. ο 3-D κύβος μοιράζεται σε 64 chunks











**A** Συναθροίζω τα 6..8 και αποθηκεύω τα aggregates στο 5 (b1c0)  
 Σαρώνω και τα 64 chunks  
 Υπολογίζω το BC cuboid



# Ταχύτερος υπολογισμός

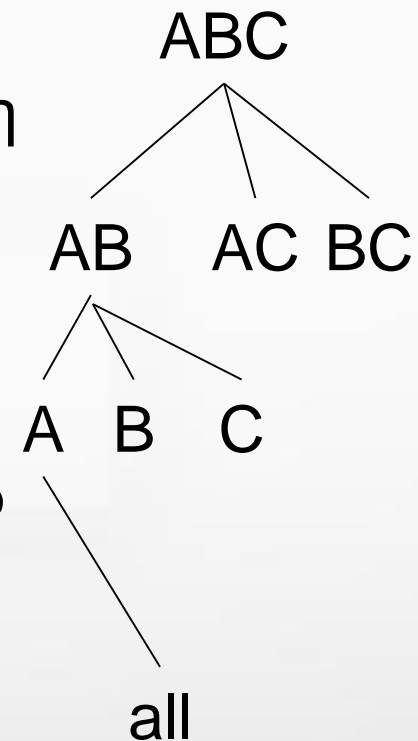


- Πώς μπορώ να αποφύγω να σαρώσω τα 64 chunks για τον επανυπολογισμό του AB cuboid;
- Multiway computation: υπολογίζω και αποθηκεύω με μία σάρωση τα aggregates στα οποία συμμετέχει το κάθε chunk (π.χ. το a0b0c0)
- Με ποια σειρά θα γίνει η σάρωση;
- Μέθοδος: οι επιφάνειες ταξινομούνται σε αύξουσα σειρά μεγέθους
- Η σάρωση ξεκινά από τη μικρότερη (σε μέγεθος διάσταση και ακολουθούν οι υπόλοιπες
  - Η μικρότερη (σε μέγεθος) επιφάνεια κρατιέται στην κύρια μνήμη, για τη μεγαλύτερη επιφάνεια καλείται στη μνήμη ένα κομμάτι κάθε φορά



# Περιορισμοί

- Αν είναι μεγάλος ο αριθμός διαστάσεων η top-down προσέγγιση δεν αποδίδει
- Εναλλακτικά
  - ο υπολογισμός “shell-cube” λίγων διαστάσεων
  - ο υπολογισμός του “iceberg cube” (μόνο των κελιών που υπερβαίνουν ένα κατώφλι τιμής)



---

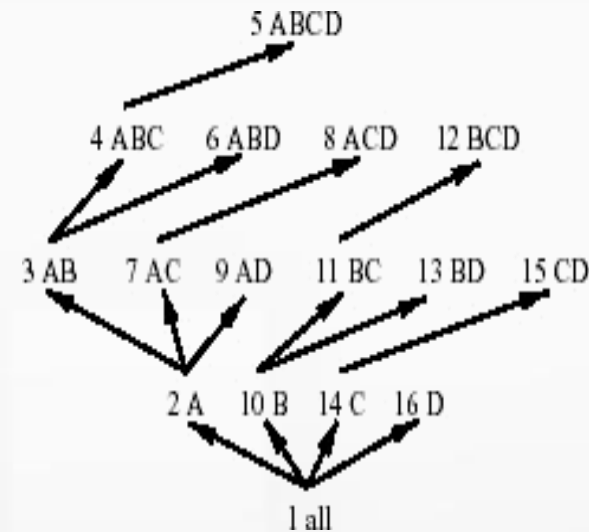
compute cube sales iceberg as  
select month, city, customer group, count(\*)  
from salesInfo  
cube by month, city, customer group  
having count(\*) >= min sup





# Bottom-Up Computation (BUC)


- Ξεκινώ τις συναθροίσεις από το apex cuboid
- Ταξινομώ τις διακριτές τιμές σε κάθε διάσταση
- Δημιουργώ μόνο τα iceberg cubes σε κάθε επίπεδο
- Αν ένα επίπεδο δεν ικανοποιεί το min\_sur δεν υπολογίζω τα επιπλέον cuboids
- Καλή λύση για αραιούς κύβους
- Οι iceberg κύβοι δεν υπολογίζονται για σύνθετες μετρικές



		b2	d1	d2
a1	b1	c1		
		c2		
	b3			
	b4			
a2				
a3				
a4				



# Σύνθετες συνθήκες



<i>month</i>	<i>day</i>	<i>city</i>	<i>cust_group</i>	<i>item</i>	<i>price</i>
Jan	10	Chicago	Education	HP Printer	485
Jan	15	Chicago	Household	Sony TV	1,200
Jan	20	New York	Education	Canon Camera	1,280
Feb	20	New York	Business	IBM Laptop	2,500
Mar	4	Vancouver	Education	Seagate HD	520
...	...	...	...	...	...

- Βρες πωλήσεις με πάνω από 50 τμχ και μέση τιμή πάνω από 800
- compute cube sales avg iceberg as  
select month, city, customer group, avg(price), count()  
from salesInfo  
cube by month, city, customer group  
having avg(price) >= 800 and count() >= 50
- Η μέση τιμή δεν είναι αντι-μονότονη. Δεν μπορώ να εφαρμόσω pruning στα iceberg cubes
- Μπορώ όμως να χρησιμοποιώ κάθε φορά το avg<sup>topk</sup>



# Ευρετήρια στο OLAP

- Ευρετήρια bitmap σε μία στήλη
- Κάθε διαφορετική τιμή στη στήλη έχει ένα bit vector μεγέθους όσο το πλήθος των εγγραφών στο βασικό πίνακα
- Δεν είναι αποδοτικά για πεδία με μεγάλο πλήθος διαφορετικών τιμών (cardinality)

**Base table**

Cust	Region	Type
C1	Asia	Retail
C2	Europe	Dealer
C3	Asia	Dealer
C4	America	Retail
C5	Europe	Dealer

**Index on Region**

RecID	Asia	Europe	America
1	1	0	0
2	0	1	0
3	1	0	0
4	0	0	1
5	0	1	0

**Index on Type**

RecID	Retail	Dealer
1	1	0
2	0	1
3	0	1
4	1	0
5	0	1

# Ευρετήρια συνένωσης: join indices

- Στο σχεσιακό μοντέλο
  - ένα join index για το  $S \bowtie R$  περιλαμβάνει για κάθε διαφορετική τιμή του  $S.id$  μια λίστα με δείκτες σε εγγραφές του  $R$  που έχουν το ίδιο  $R.sid$
  - επιταχύνει μια σχεσιακή συνένωση που έχει μεγάλο κόστος χωρίς το ευρετήριο
- Στις αποθήκες δεδομένων
  - ένα join index σχετίζει τις διαφορετικές τιμές στις διαστάσεις ενός star schema με εγγραφές του fact table
  - το ίδιο join index καλύπτει περισσότερες από μία διαστάσεις





# Παράδειγμα



Join index table for  
*location/sales*

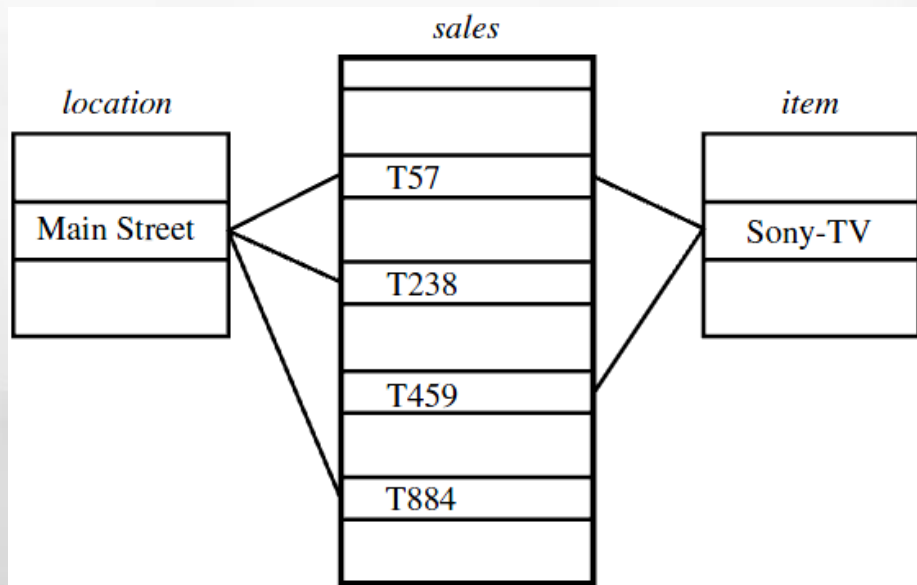
<i>location</i>	<i>sales_key</i>
...	...
Main Street	T57
Main Street	T238
Main Street	T884
...	...

Join index table for  
*item/sales*

<i>item</i>	<i>sales_key</i>
...	...
Sony-TV	T57
Sony-TV	T459
...	...

Join index table linking two dimensions  
*location/item/sales*

<i>location</i>	<i>item</i>	<i>sales_key</i>
...	...	...
Main Street	Sony-TV	T57
...	...	...





# Αποτελεσματική διαχείριση των OLAP queries



- Καθορισμός των λειτουργιών που θα πραγματοποιηθούν στα υπάρχοντα cuboids
  - Μετασχηματισμός των drill, roll, κλπ σε SQL και OLAP λειτουργίες, π.χ. dice = selection + projection
- Καθορισμός των cuboids που επηρεάζει κάθε λειτουργία
- Χρήση των index και των αραιών ή πυκνών array structures για τις συναθροίσεις





# Αποθετήριο μεταδεδομένων

## Περιέχει

- περιγραφές των δομών της αποθήκης δεδομένων: schema, view, dimensions, hierarchies, derived data definitions, data mart locations and contents
- λειτουργικά μεταδεδομένα: ιστορικό μετάπτωσης και μετασχηματισμού δεδομένων, κατάσταση δεδομένων, δεδομένα χρήσης, σφαλμάτων κλπ
- τα scripts μετάπτωσης από το λειτουργικό περιβάλλον στην αποθήκη
- τους αλγορίθμους για τη σύνοψη
- business data: ιδιοκτησία δεδομένων, πολιτικές χρέωσης κλπ.







# Υποστηρικτικές λειτουργίες



- Εξαγωγή δεδομένων
  - Από πολλαπλές, ετερογενείς και εξωτερικές πηγές
- Καθαρισμός δεδομένων
  - Ανίχνευση σφαλμάτων στα δεδομένα και επιδιόρθωση
- Μετασχηματισμός δεδομένων
  - Από το πηγαίο format σε αυτό της αποθήκης
- Φόρτωση
  - Ταξινόμηση, σύνοψη, υπολογισμός όψεων, έλεγχος συνέπειας, δημιουργία ευρετηρίων
- Ανανέωση
  - Κάθε φορά που αλλάζουν τα δεδομένα στην πηγή, οι αλλαγές θα πρέπει να απεικονίζονται στην αποθήκη



# Οπτικοποίηση

Συνάθροιση πωλήσεις για ένα έτος, ανά τοποθεσία και είδος

<i>location</i>	<i>item</i>	<i>sales (in million dollars)</i>	<i>count (in thousands)</i>
Asia	TV	15	300
Europe	TV	12	250
North_America	TV	28	450
Asia	computer	120	1000
Europe	computer	150	1200
North_America	computer	200	1800

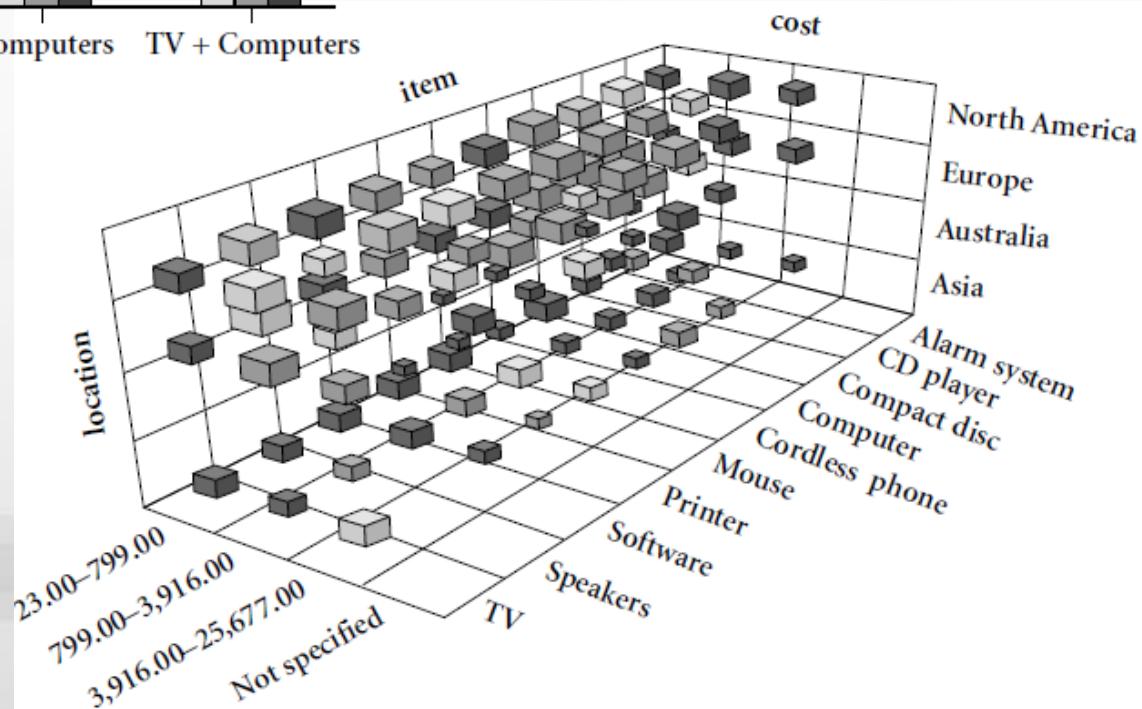
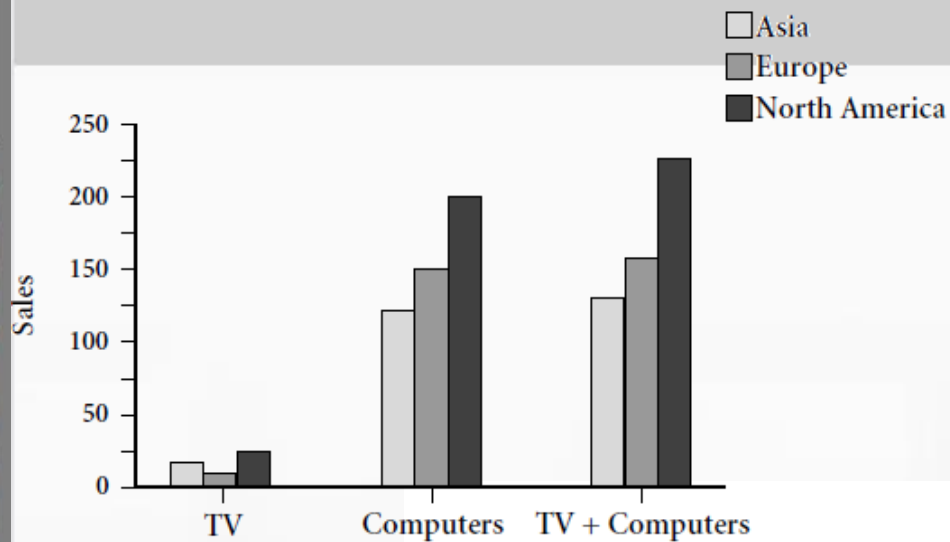
Προβολή crosstab

	<i>item</i>					
	<i>TV</i>		<i>computer</i>		<i>both_items</i>	
<i>location</i>	<i>sales</i>	<i>count</i>	<i>sales</i>	<i>count</i>	<i>sales</i>	<i>count</i>
Asia	15	300	120	1000	135	1300
Europe	12	250	150	1200	162	1450
North_America	28	450	200	1800	228	2250
<i>all_regions</i>	45	1000	470	4000	525	5000



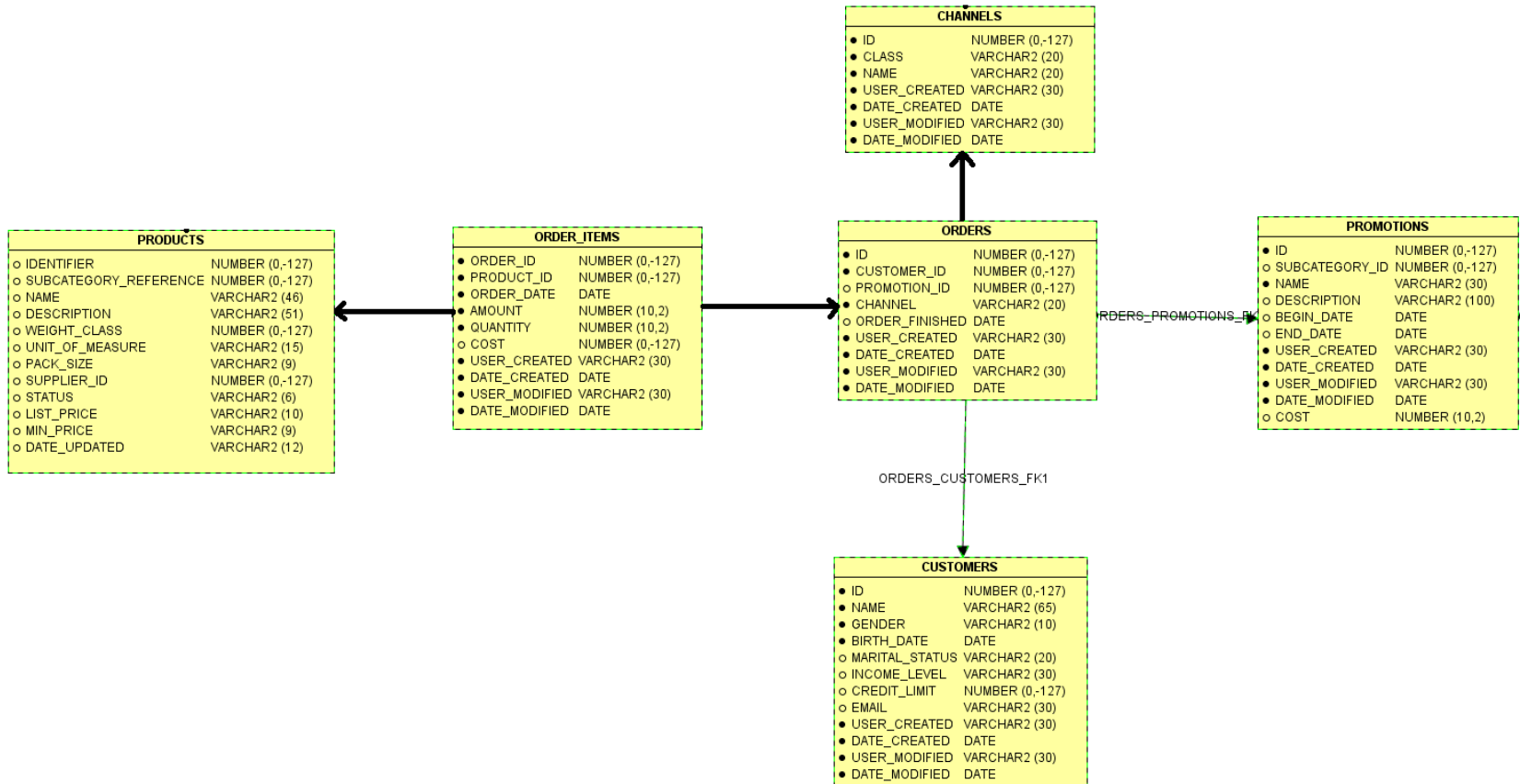


# Οπτικοποίηση



# Παράδειγμα

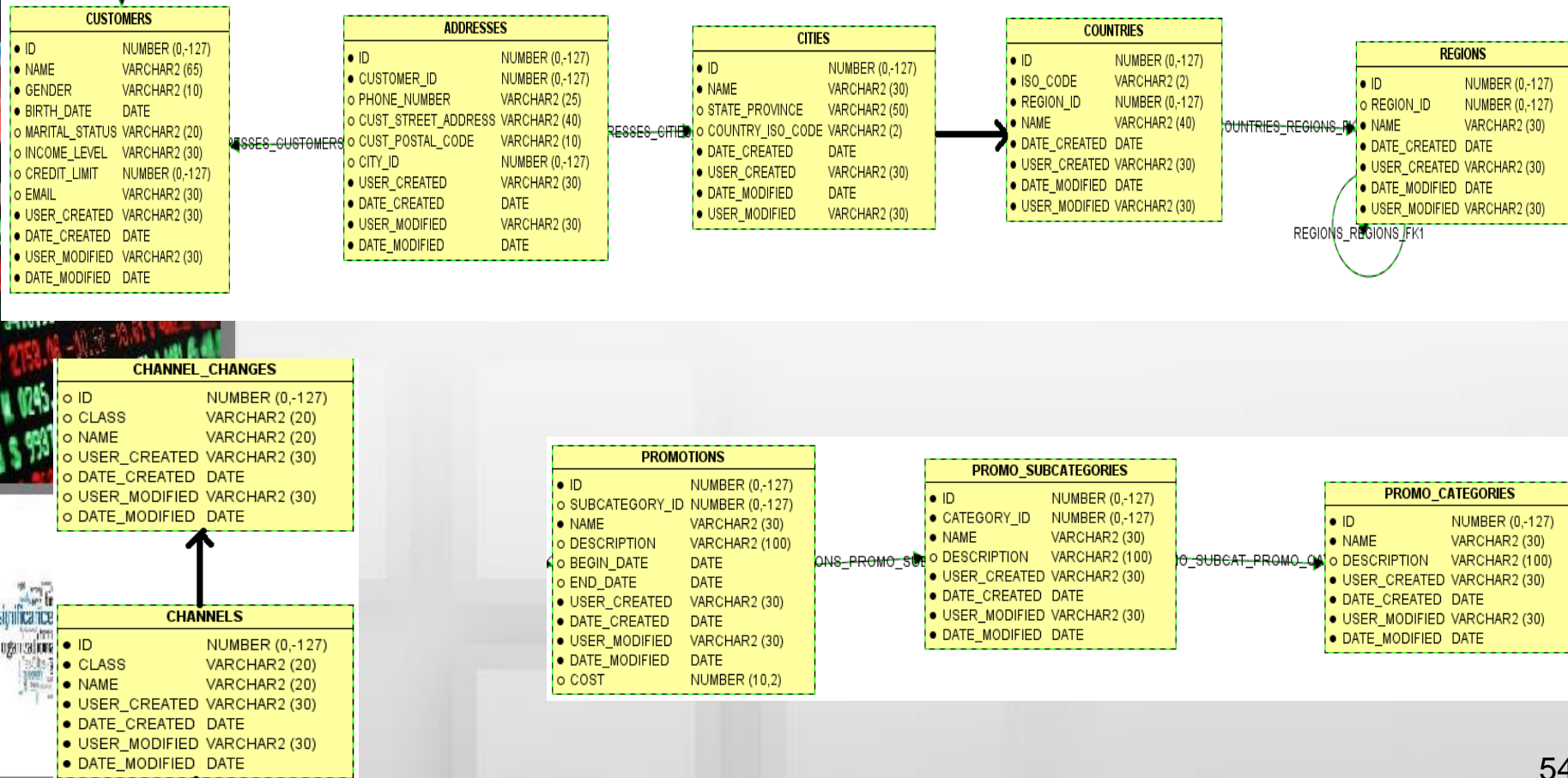
- Μια εταιρία που πουλάει ηλεκτρονικά διαθέτει δεδομένα που αφορούν τις πωλήσεις σε μια transactional dbms
- Το σχήμα της λειτουργικής ΒΔ περιλαμβάνει πληροφορίες για τις παραγγελίες, τους πελάτες, τα κανάλια διανομής και τους προωθητές





# Διαστάσεις

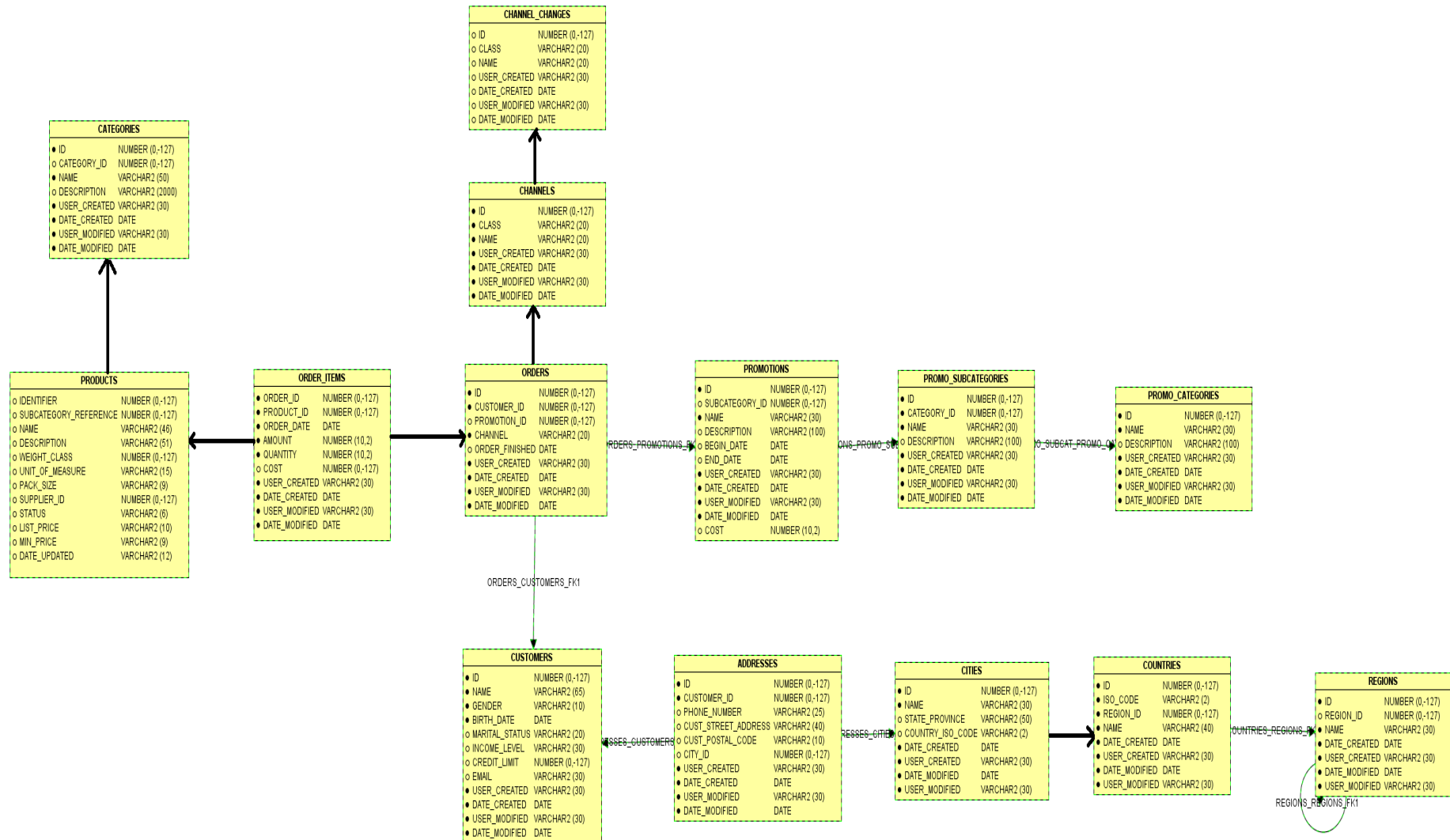
- Σε κάθε κατηγορία δεδομένων έχει ορισμένες επιπλέον πληροφορίες: διεύθυνση πελάτη, κατηγορία προϊόντων, τύπος καναλιού διάθεσης κλπ







# Τελικό ER



# Δημιουργία του DW (star) schema

- Διαστάσεις
  - CHANNELS: Πίνακας που θα περιέχει τα κανάλια διανομής των παραγγελιών
  - CUSTOMERS: Πίνακας που δείχνει ποιος αγόρασε προϊόντα καθώς και τη γεωγραφική κατανομή
  - PRODUCTS: Πίνακας που δείχνει τα προϊόντα που πουλά η εταιρία (και τις κατηγορίες τους)
  - TIMES: Πίνακας που περιέχει χρονικές περιόδους πώλησης προϊόντων
- Μετρικές
  - SALES\_FACT: Περιέχει τις αγορές σε δολάρια, ποσότητα και τιμή, ανά κανάλι διανομής, προϊόν, ημέρα και πελάτη



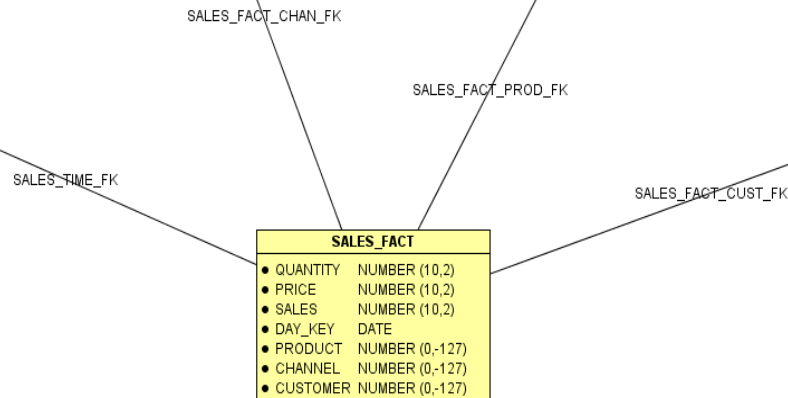
# Τελικό star schema

TIMES	
# DAY_KEY	DATE
o CALENDAR_YEAR_TIME_SPAN	NUMBER (0,-127)
• CALENDAR_YEAR_ID	VARCHAR2 (30)
o CALENDAR_YEAR_NAME	VARCHAR2 (40)
o CALENDAR_YEAR_END_DATE	DATE
o CALENDAR_QUARTER_NAME	VARCHAR2 (40)
o CALENDAR_QUARTER_TIME_SPAN	NUMBER (0,-127)
o CALENDAR_QUARTER_END_DATE	DATE
o CALENDAR_QUART_NUMBER_IN_YEAR	NUMBER (0,-127)
• CALENDAR_QUARTER_ID	VARCHAR2 (30)
o MONTH_TIME_SPAN	NUMBER (0,-127)
o MONTH_NAME	VARCHAR2 (40)
o MONTH_END_DATE	DATE
• MONTH_ID	VARCHAR2 (30)
o MONTH_NUMBER_IN_YEAR	NUMBER (0,-127)
o NUMBER_IN_WEEK	NUMBER (0,-127)
o NUMBER_IN_MONTH	NUMBER (0,-127)
o DAY_DESCRIPTION	VARCHAR2 (40)
o DAY_END_DATE	DATE
o DAY_TIME_SPAN	NUMBER (0,-127)
o FISCAL_YEAR_ID	VARCHAR2 (30)
o FISCAL_YEAR_TIME_SPAN	NUMBER (0,-127)
o FISCAL_YEAR_NAME	VARCHAR2 (40)
o FISCAL_YEAR_END_DATE	DATE
o FISCAL_QUARTER_NUMBER_IN_YEAR	NUMBER (0,-127)
o FISCAL_QUARTER_ID	VARCHAR2 (30)
o FISCAL_QUARTER_NAME	VARCHAR2 (40)
o FISCAL_QUARTER_END_DATE	DATE
o FISCAL_QUARTER_TIME_SPAN	NUMBER (0,-127)
o RETAIL_YEAR_TIME_SPAN	NUMBER (0,-127)
o RETAIL_YEAR_END_DATE	DATE
o RETAIL_YEAR_ID	VARCHAR2 (30)
o RETAIL_YEAR_NAME	VARCHAR2 (40)
o RETAIL_SEASON_END_DATE	DATE
o RETAIL_SEASON_NAME	VARCHAR2 (40)
o RETAIL_SEASON_TIME_SPAN	NUMBER (0,-127)
o RETAIL_SEASON_ID	VARCHAR2 (30)
o RETAIL_QUARTER_END_DATE	DATE
o RETAIL_QUARTER_NAME	VARCHAR2 (40)
o RETAIL_QUARTER_TIME_SPAN	NUMBER (0,-127)
o RETAIL_QUARTER_NUMBER_IN_YEAR	NUMBER (0,-127)
o RETAIL_QUARTER_ID	VARCHAR2 (30)
o RETAIL_MONTH_END_DATE	DATE
o RETAIL_MONTH_NAME	VARCHAR2 (40)
o RETAIL_MONTH_ID	VARCHAR2 (30)
o RETAIL_MONTH_TIME_SPAN	NUMBER (0,-127)
o RETAIL_WEEK_END_DATE	DATE
o RETAIL_WEEK_TIME_SPAN	NUMBER (0,-127)
o RETAIL_WEEK_ID	VARCHAR2 (30)
o RETAIL_WEEK_NAME	VARCHAR2 (40)

CHANNELS	
# CHANNEL_KEY	NUMBER (0,-127)
o CHANNEL_NAME	VARCHAR2 (20)
o CHANNEL_TYPE	VARCHAR2 (30)
• CLASS_KEY	NUMBER (0,-127)
o CLASS_NAME	VARCHAR2 (20)

PRODUCTS	
o DEPARTMENT_SHORT_DESC	VARCHAR2 (100)
o DEPARTMENT_NAME	VARCHAR2 (100)
• DEPARTMENT_KEY	NUMBER (0,-127)
o CATEGORY_SHORT_DESC	VARCHAR2 (100)
o CATEGORY_NAME	VARCHAR2 (100)
o CATEGORY_MANAGER	VARCHAR2 (100)
• CATEGORY_KEY	NUMBER (0,-127)
o TYPE_NAME	VARCHAR2 (100)
o TYPE_SHORT_DESC	VARCHAR2 (100)
• TYPE_KEY	NUMBER (0,-127)
o SUBTYPE_SHORT_DESC	VARCHAR2 (100)
o SUBTYPE_NAME	VARCHAR2 (100)
• SUBTYPE_KEY	NUMBER (0,-127)
o DATE_INTRODUCED	DATE
# ITEM_KEY	NUMBER (0,-127)
o ITEM_NAME	VARCHAR2 (100)
o ITEM_SHORT_DESC	VARCHAR2 (100)
o BUYER	VARCHAR2 (100)
o VENDOR_KEY	NUMBER (0,-127)
o VENDOR_NAME	VARCHAR2 (100)
o VENDOR_SHORT_DESC	VARCHAR2 (100)

CUSTOMERS	
# CUSTOMER_KEY	NUMBER (0,-127)
o CUSTOMER_NUMBER	NUMBER (0,-127)
o AGE	NUMBER (0,-127)
o AGE_RANGE	VARCHAR2 (30)
o BIRTHDAY	DATE
o COMMUTE_DISTANCE	NUMBER (0,-127)
o CREDIT_BALANCE	NUMBER (0,-127)
o CREDIT_BALANCE_RANGE	VARCHAR2 (20)
o DISTANCE_RANGE	VARCHAR2 (20)
o EDUCATION	VARCHAR2 (40)
o EMAIL	VARCHAR2 (425)
o FIRST_NAME	VARCHAR2 (200)
o FULL_TIME	VARCHAR2 (40)
o GENDER	VARCHAR2 (6)
o HIGH_VALUE	NUMBER (1,0)
o HIGH_VALUE_DESCRIPTION	VARCHAR2 (5)
o HOUSEHOLD_SIZE	NUMBER (0,-127)
o HOUSEHOLD_SIZE_RANGE	VARCHAR2 (20)
o INCOME	NUMBER (0,-127)
o INCOME_LEVEL	VARCHAR2 (20)
o INSUFF_FUNDS_INCIDENTS	NUMBER (0,-127)
o JOB_TYPE	VARCHAR2 (200)
o LAST_NAME	VARCHAR2 (200)
o LATE_MORT_RENT_PMTS	NUMBER (0,-127)
o MARITAL_STATUS	VARCHAR2 (8)
o MORTGAGE_AMT_RANGE	VARCHAR2 (25)
o NUM_CARS	NUMBER (0,-127)
o NUM_MORTGAGES	NUMBER (0,-127)
o MORTGAGE_AMT	NUMBER (0,-127)
o PET	VARCHAR2 (40)
o POSTAL_CODE	VARCHAR2 (10)
o PROMOTION_RESPONSE	NUMBER (0,-127)
o RENT_OWN	VARCHAR2 (40)
o STREET_ADDRESS	VARCHAR2 (400)
o WORK_EXPERIENCE	NUMBER (0,-127)
o WORK_EXPERIENCE_RANGE	VARCHAR2 (20)
o YRS_CURRENT_EMPLOYER_RANGE	VARCHAR2 (20)
o YRS_CUSTOMER	NUMBER (0,-127)
o YRS_CUSTOMER_RANGE	VARCHAR2 (20)
o YRS_CURRENT_EMPLOYER	NUMBER (0,-127)
o YRS_RESIDENCE	NUMBER (0,-127)
o YRS_RESIDENCE_RANGE	VARCHAR2 (20)
o COMMENTS	CLOB
o CITY_KEY	NUMBER (0,-127)
o CITY_NAME	VARCHAR2 (100)
o STATE_PROVINCE_KEY	NUMBER (0,-127)
o STATE_PROVINCE_NAME	VARCHAR2 (400)
o COUNTRY_KEY	NUMBER (0,-127)
o COUNTRY_NAME	VARCHAR2 (100)
o ISO_CODE	VARCHAR2 (2)
o REGION_KEY	NUMBER (0,-127)
o REGION_NAME	VARCHAR2 (100)



# Πωλήσεις

SALESTRACK (attached RW)

Dimensions

Cubes

SALES\_CUBE

Measures

SALES

QUAL

Calculate

Mappings

Views

Cube Sc

Data Security

Create Measure...

Maintain Measure SA

View Data SALES...

Delete Measure SAL

Measure Data Viewer

File

Agency FB

8

B

U

Στοιχεία σελίδας

GEOGRAPHY All Regions

PRODUCT All Products

SALES

All Channels

All Years

417.515.017,27

Agency FB

8

B

U

400M

300M

200M

100M

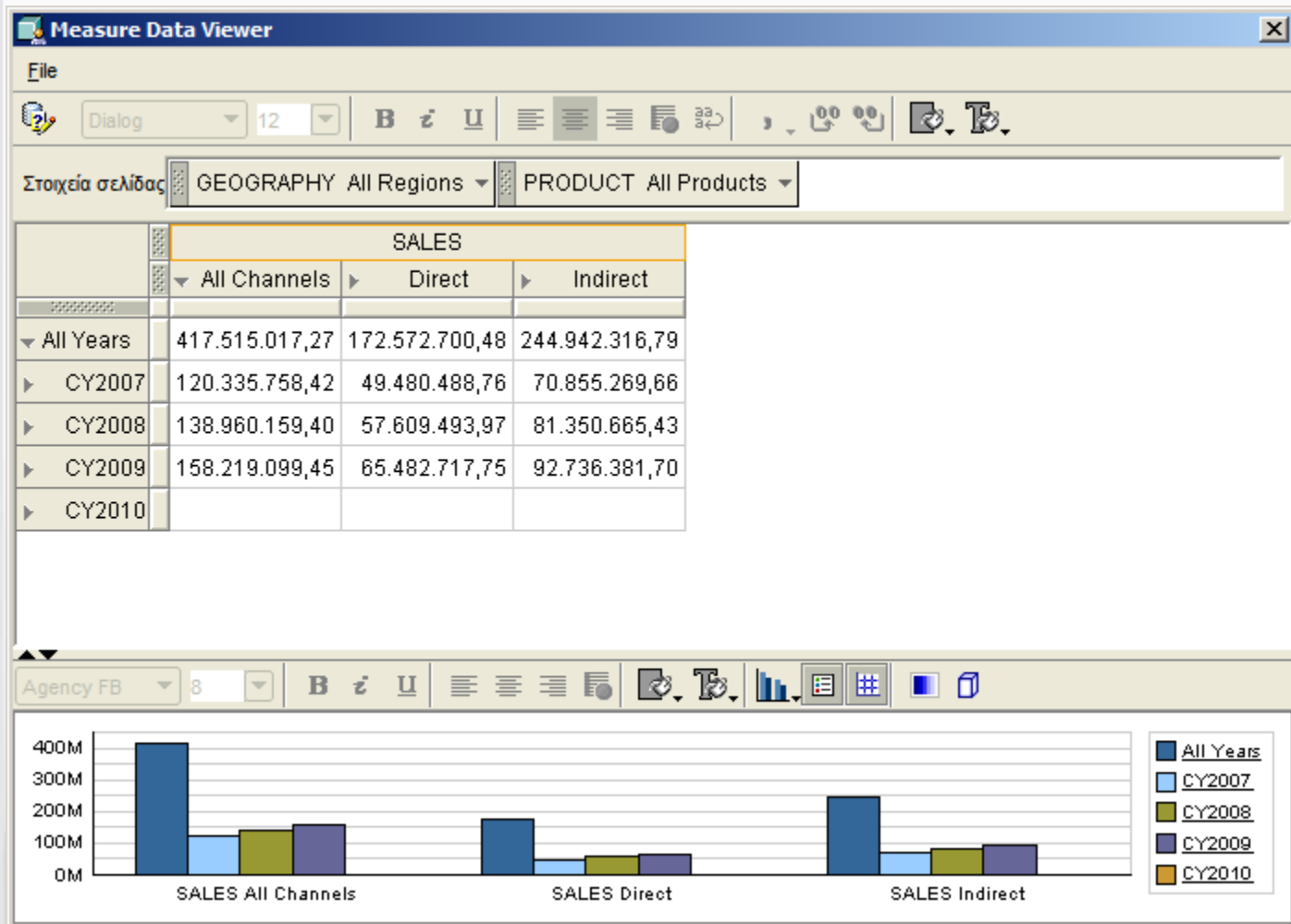
0M

SALES All Channels

All Years



# Drill down





# Προσθήκη διάστασης

