BASES DE DONNÉES
ENTREPÔTS DE DONNÉES

Architecture et dimensionnalité

BD301
v110b

2020-11-13

Christina KHNAISSER et Luc LAVOIE
Département d'informatique
Faculté des sciences

UNIVERSITÉ DE SHERBROOKE

Christina.Khnaisser@usherbrooke.ca
Luc.Lavoie@usherbrooke.ca
http://info.usherbrooke.ca/llavoie

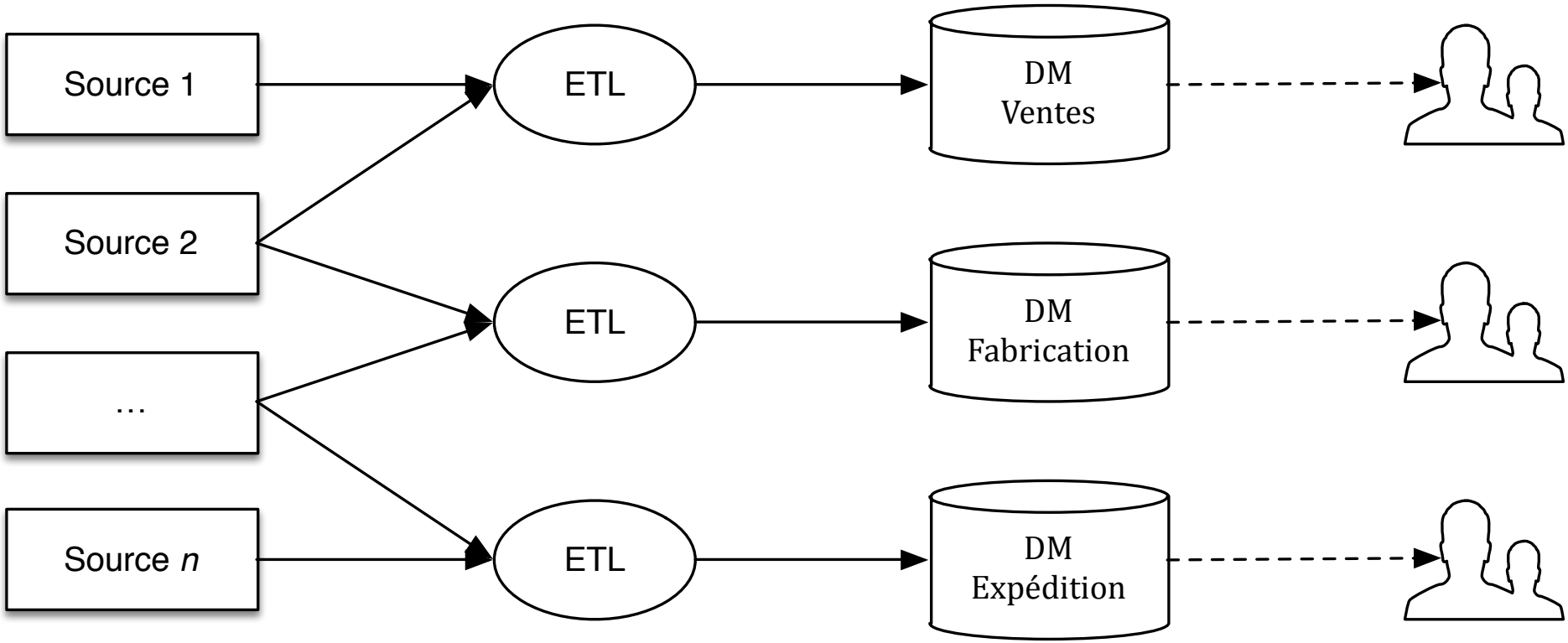
Plan

- Architecture
- Dimensionnalité
- Techniques dimensionnelles
- Conseils d'Adamson
- Questions ouvertes
- Références

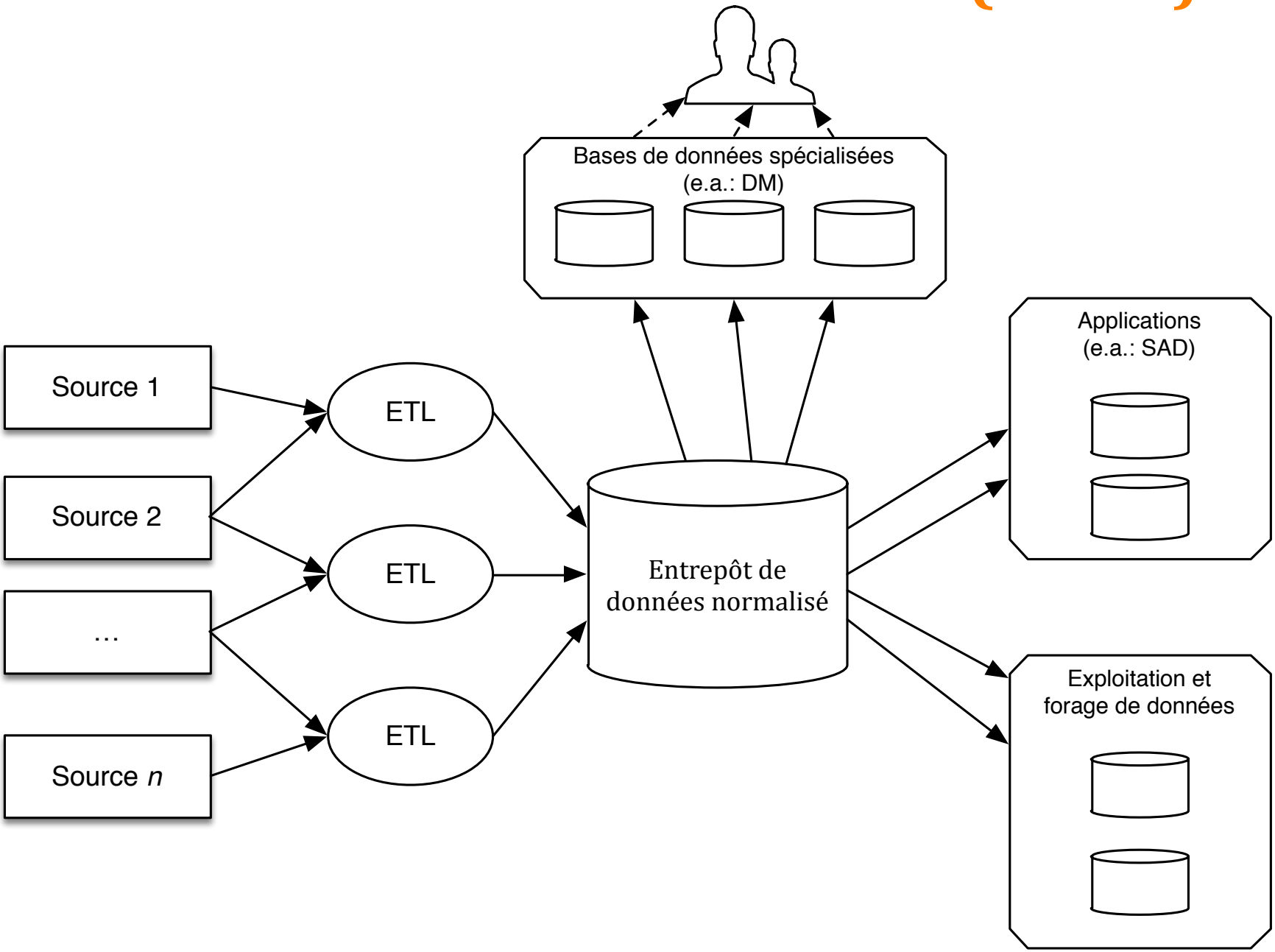
Architecture Propositions

	Data Marts indépendants	*Information Factory*	*Data Warehouse*
Figure de proue	--	Inmon	Kimball
Conception	Variable	Dim+3FN	Dim
Description	Dépôts spécifiques et indépendants	entrepôt commun + dépôts	entrepôt + vues
Appellations	Data Mart	Information Factory	Data Warehouse

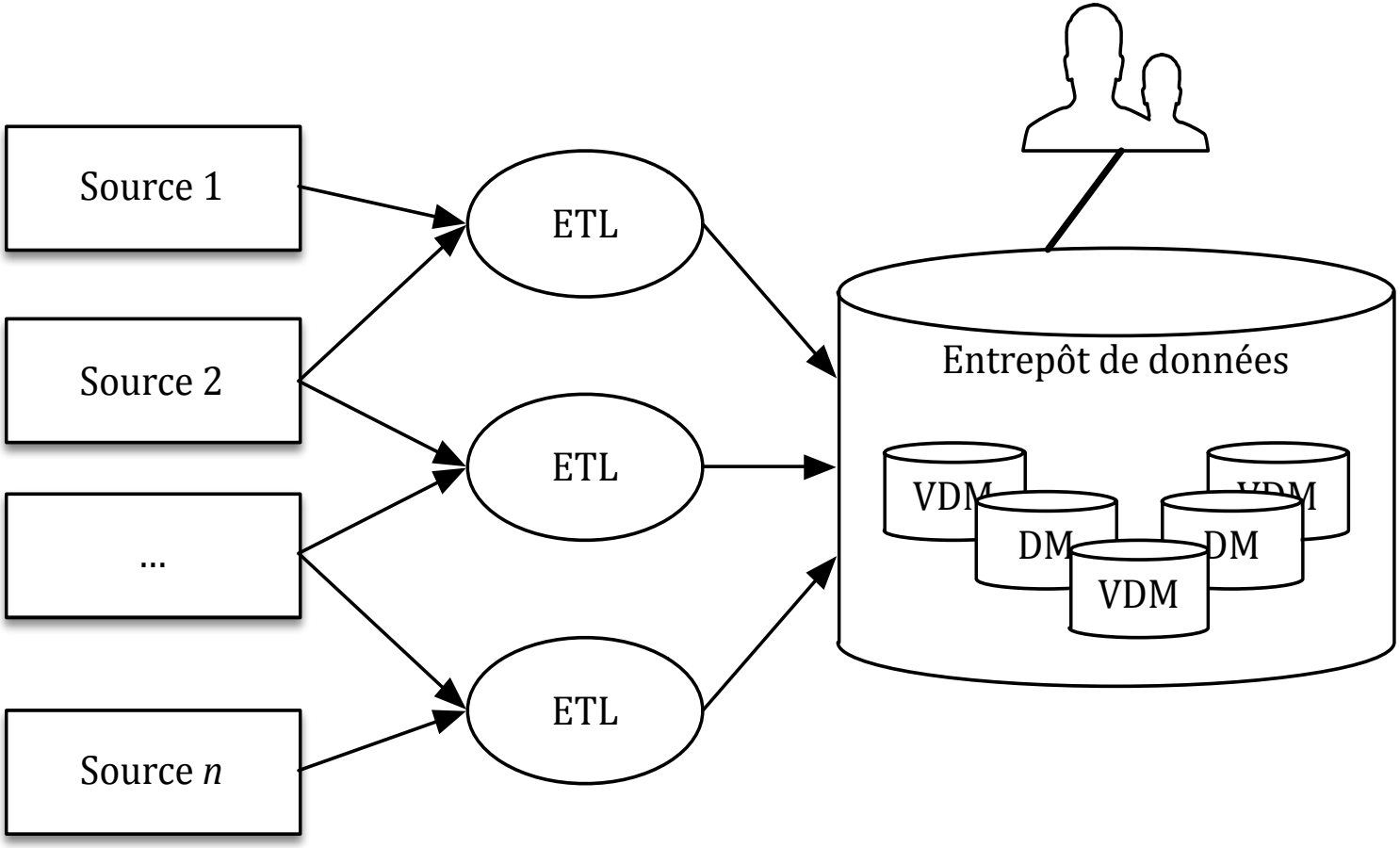
Architecture
Modèle Data Mart « ad hoc »



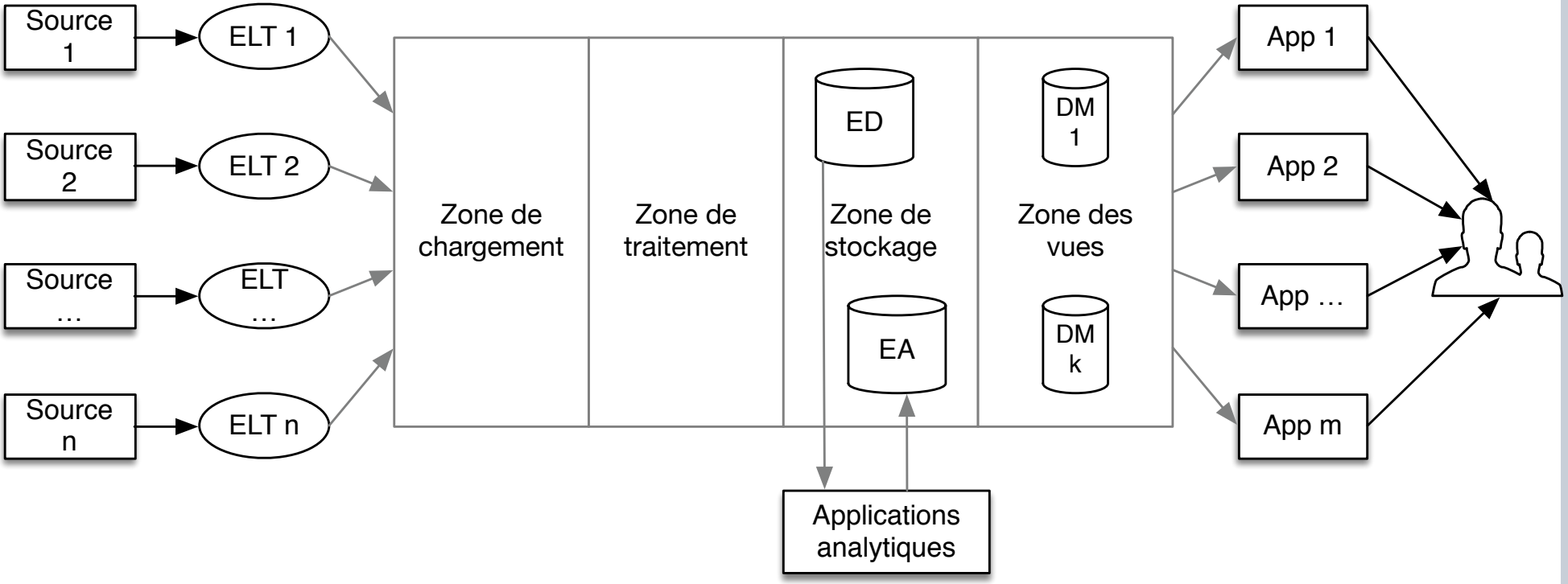
Architecture
Modèle « Information Factory » (Inmon)



Architecture
Modèle « Dimensional DW » (Kimball)



Architecture
Modèle « Dimensionnel » contemporain



Architecture
Synthèse

Architecture	Description	Caractéristiques
Ad hoc * Data Mart * Silo * Stovepipe * Island	Schéma spécifique sans contexte organisationnel	Souvent un schéma dimensionnel (étoiles ou flocons de neige)
Factory * Atomic DW * Enterprise DW	Un schéma organisationnel et plusieurs schémas spécifiques.	Le schéma organisationnel est normalisé.L Les schémas spécifiques sont dimensionnels.
DDW * Enterprise DW * Bus DW * Architected DM * Virtual DM	Un schéma organisationnel comprenant des vues spécifiques.	Le schéma organisationnel est « purement » dimensionnel. Les vues spécifiques sont souvent matérialisées.

Architecture interne
Questions et Propositions

- Quelle structure pour le schéma ?
- Quelle modélisation temporelle pour le schéma ?
- Comment construire le schéma ?
- Comment alimenter (ELT ou ETL) ?
- …

- Quelques propositions
  - Inmon
  - Kimball
  - Jiang
  - Snodgrass
  - Johnston & Weis
  - Date, Darwen & Lorentzos
  - …

ARCHITECTURE INTERNE
APERÇU GLOBAL COMMUN

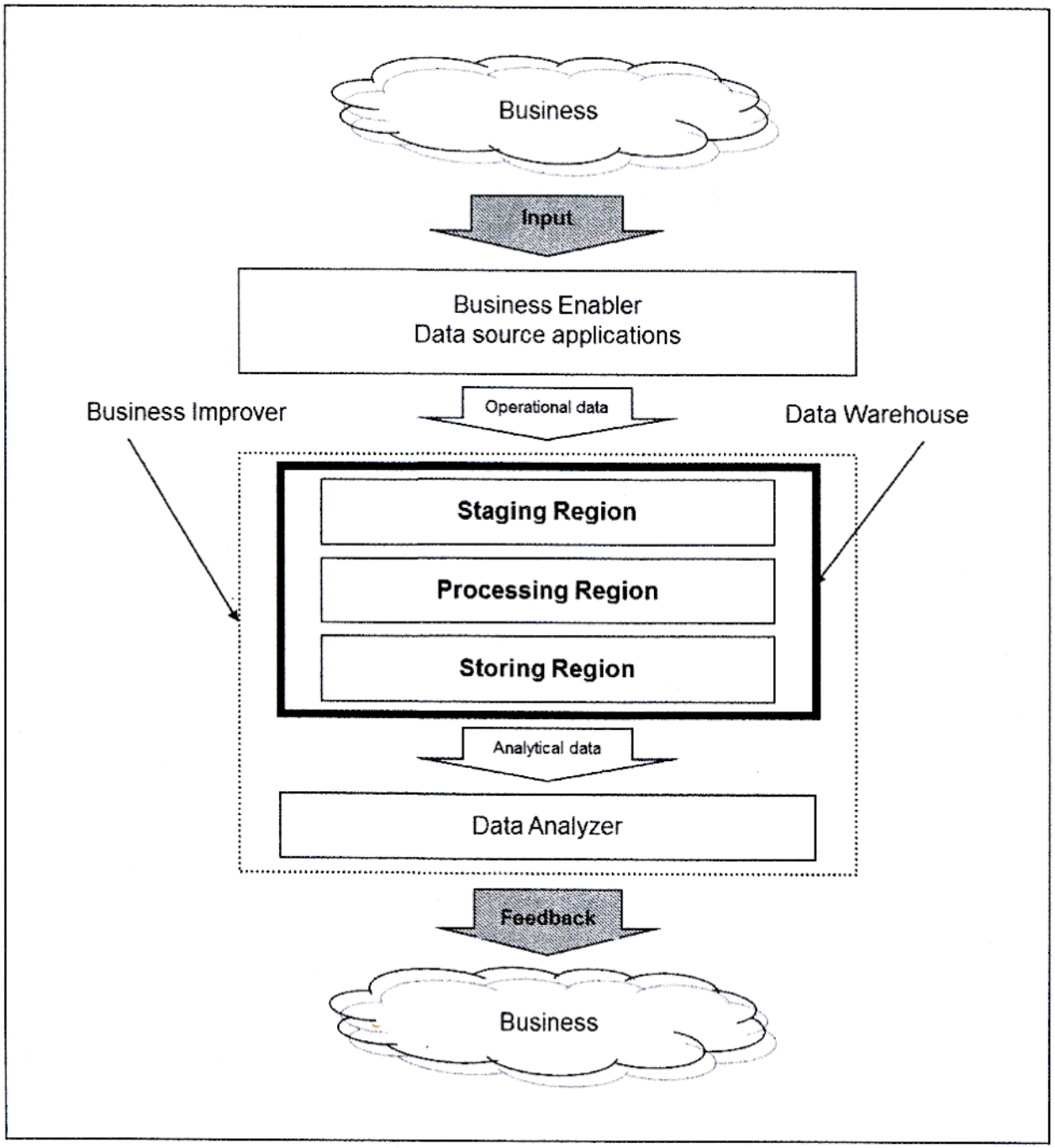


Figure 2.1: Overview of the Reference Data Warehouse Jiang. 2015.

Chaque « Region » réside sur un « Cluster » de serveurs qui lui est propre.

La « Staging Region » est représentée sous forme de tables munie chacune d'une clé artificielle, sans aucune contrainte.

La « Processing Region » est une « constellation ».

La « Storing Region » en est déduite en fonction des besoins d'analyse.

Vue détaillée de la « Staging Region »

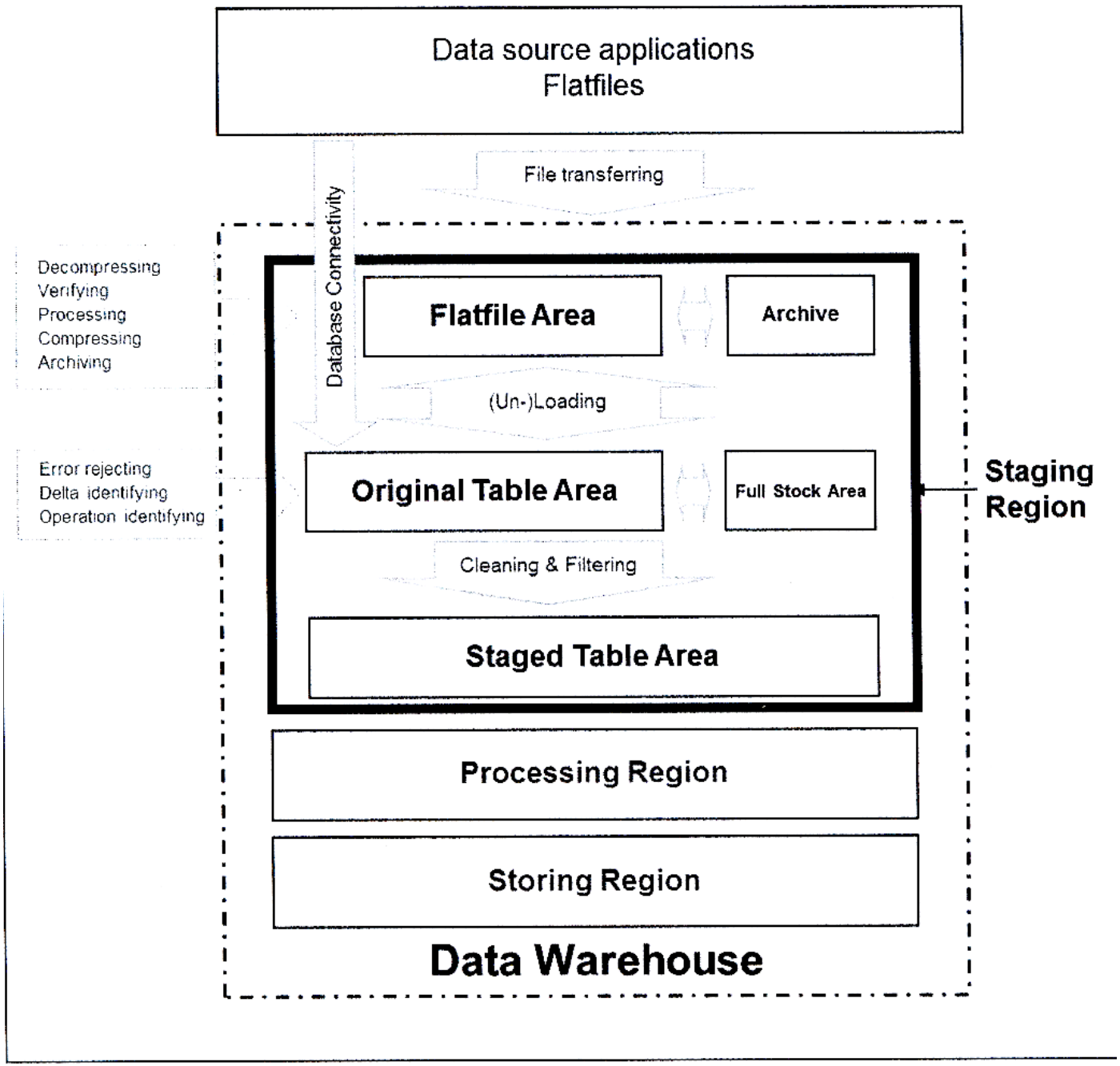


Figure 3.1: Staging Region

Jiang. 2015.

DIMENSIONALITÉ

- La base
- Principe premier
- Tables dimensionnelles
- Tables factuelles
- Évolutivité
- Exemples

Dimensionnalité
Exemple d'Adamson

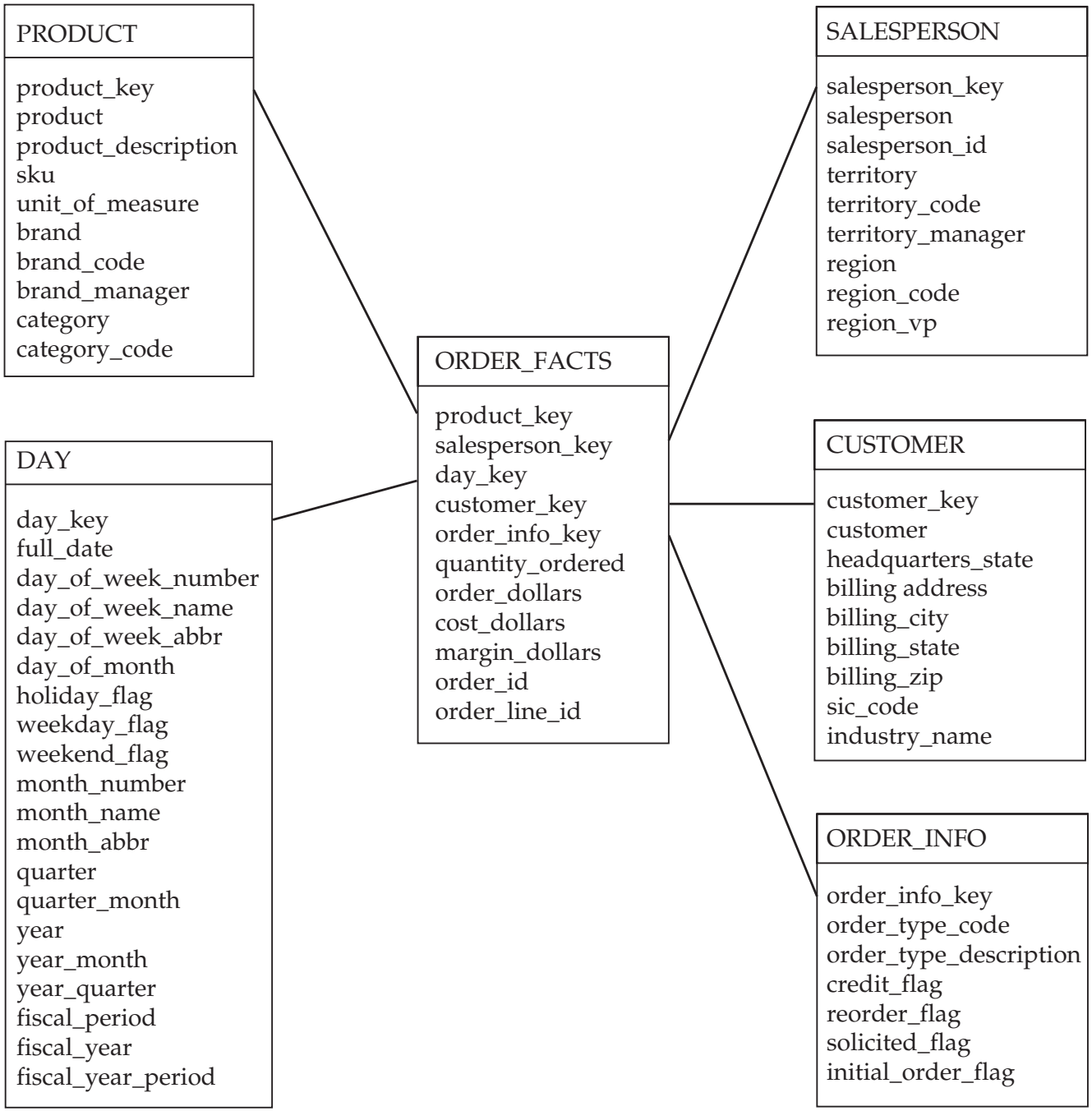


Figure 6-1 A fact table explicitly relates dimension tables

Dimensionnalité
La base (commune à l'étoile et au flocon)

- Un évènement est unique, se produit à un moment précis, met en cause des entités (appelées dimensions) et est caractérisé par des mesures (appelés faits).
- Les faits sont minimalement agrégeables (souvent « additifs »).
- Les dimensions sont déterminantes (elles déterminent des restrictions).
- Principe de la clé primaire artificielle.
- Principe de l'universalité des dimensions (UD) et des faits (UF) en rapport aux évènements décrits par une même (valeur de) relation « étoilée ».

Dimensionnalité
Un constat

- 1. Corolaire : les faits ne sont pas annulables.
- 2. Rappel : les clés ne sont pas annulables.
- 3. Conclusion (1+2) :
  - **aucun** attribut d'une table de faits (TF) n'est annulable.

Dimensionnalité
Quelques synonymes

Évènement	Dimension
Relation évènementielle	Relation dimensionnelle
Table factuelle	Table dimensionnelle
Table de faits	Dimension

Dimensionnalité
Principe premier

- *Principe simpliste* : « *Une table factuelle par processus* ».
- Dangers du principe simpliste
  - incompatibilité des périodes;
  - indépendance des référentiels.
- **Principe efficace : « Deux faits sont dans la même table, si, et seulement s'ils ont même granularité et même synchronicité. »**
- Remarque
  - Granularité et synchronicité sont indépendants.
- Il en découle tout de même un lien étroit entre tables factuelles (TF) et processus :
  - une TF est tout entière définie par un seul processus;
  - un processus peut déterminer plusieurs TF.

- ***Génération obligée*** *de nouvelles* clés artificielles
  - pourquoi ?
- Regroupement des dimensions
  - par affinité
  - les laissés-pour-compte (*junk tables*)
  - pourquoi minimiser le nombre de tables ?
  - tables associatives, auxiliaires et autres ?
- Richesse représentative
  - code et libellé
  - permutations
  - variantes
- Joies (et peines) de la redondance
  - pourquoi limiter les rallonges (*outriggers*) ?
  - pourquoi pas des vues ?

Table temporelle

- Rôle :
  - fixer la granularité de la table de faits
  - établir les différentes catégorisations admises
  - réunir les différentes représentations utilisées
- D'autres solutions existent, fondées sur des modèles unificateurs :
  - BCDM
  - DDLM
  - AV
  - ...

Une table factuelle spécialisée

Les catégorisations et les représentations sont partagées autant que possible entre les différente tables temporelles, mais les clés (granularité de base) sont indépendantes (postulat initial).

- Clé :
  - base :
    $\{dim_1, dim_2, ..., dim_n, dim_T\}$
  - discrimination supplémentaire ?
  - génération d'une clé artificielle synthétique ?
- Granularité représentée par dim_T
- Synchronisme
  (co-occurence temporelle)
- Densité (*sparsity*)
- Dimensions dégénérées
- Additivité et agrégeabilité

- Évolution des dimensions :
  Permise
  - Type 1 (récriture) – *fortement déconseillé*
    - maintien du tuple, modification de la valeur de l'attribut.
  - Type 2 (historique) – *recommandée*
    - insertion d'un nouveau tuple avec les nouvelles valeurs d'attribut.
- Évolution des faits :
  Interdite
  - ... sauf si elle est modélisée comme une table bitemporelle!

Une modification de type 1 est occasionne une perte ***irrémédiable*** *d'information.*

À moyen et long terme, elle est une source d'erreurs et une cause de distorsion du modèle.

Elle doit donc être évitée... à tout prix.

EXEMPLE DE « RICHESSE »

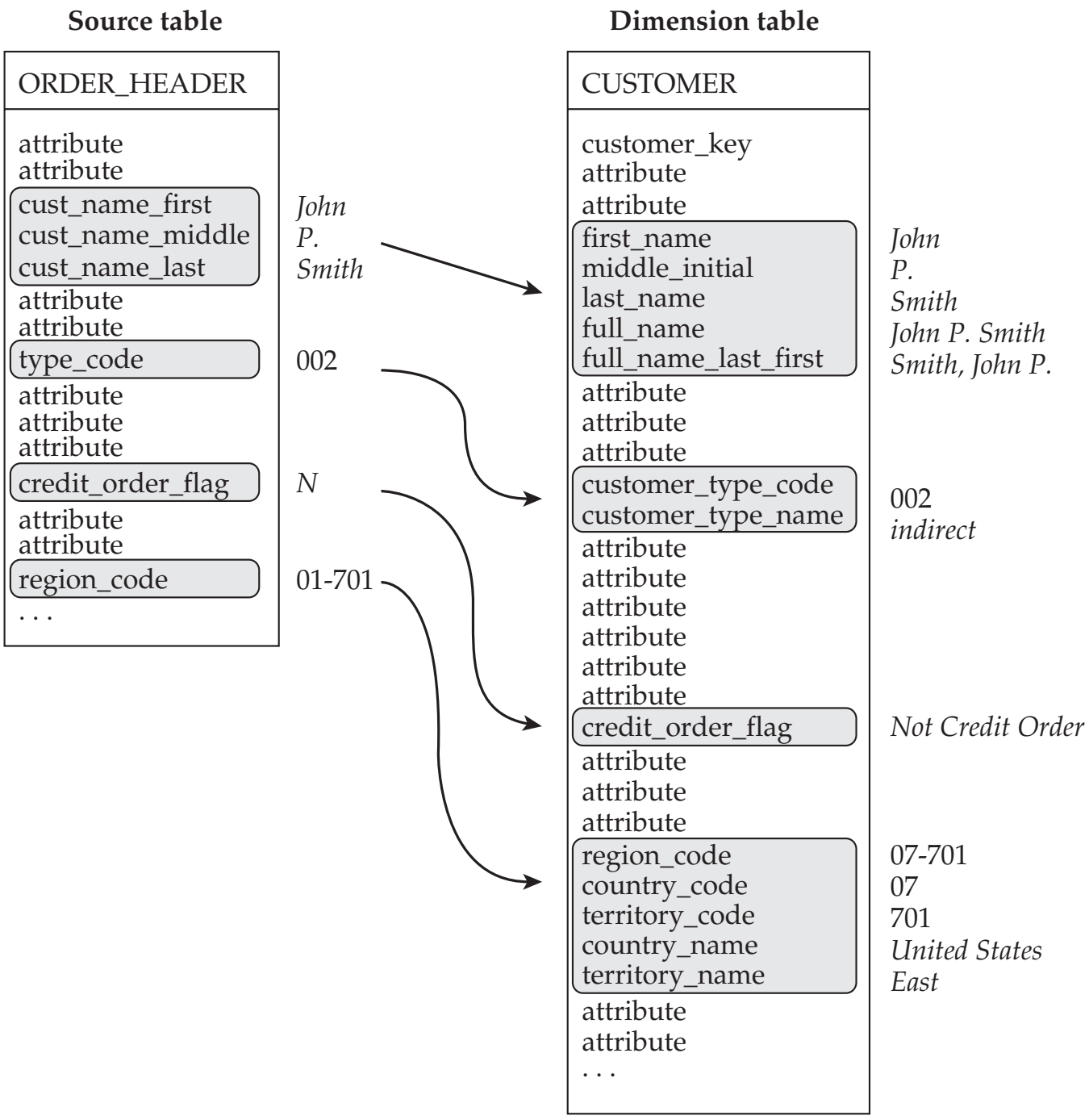


Figure 3-2 Constructing a rich set of dimension attributes

Exemple de dimensions dégénérées

S'agit-il de dimensions dégénérées... ou de discriminants supplémentaires ?

ORDER vs ORDER_INFO

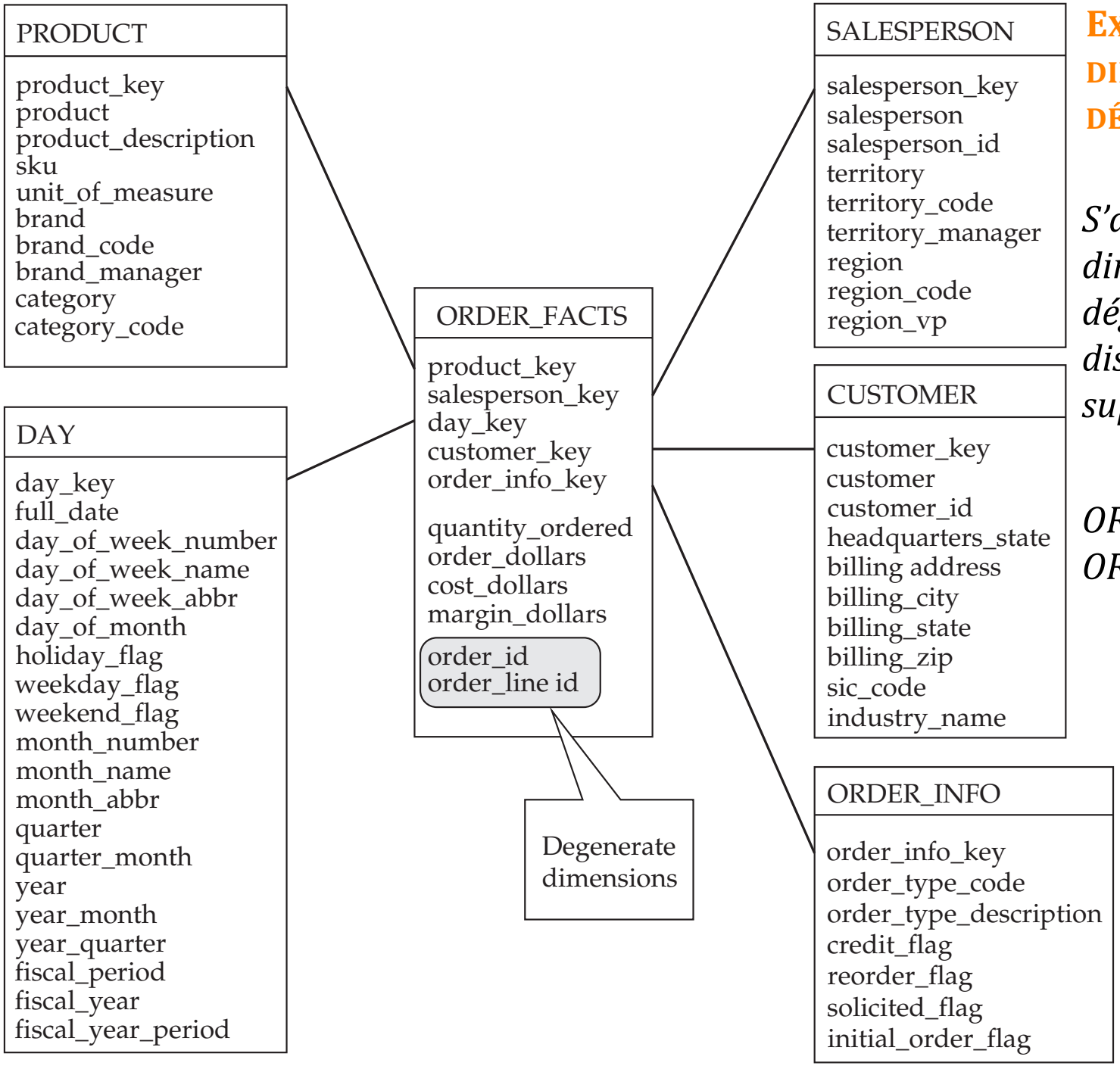


Figure 3-5 Degenerate dimensions define the grain of this fact table

EXEMPLE DE RALLONGES (*OUTRIGGERS*)

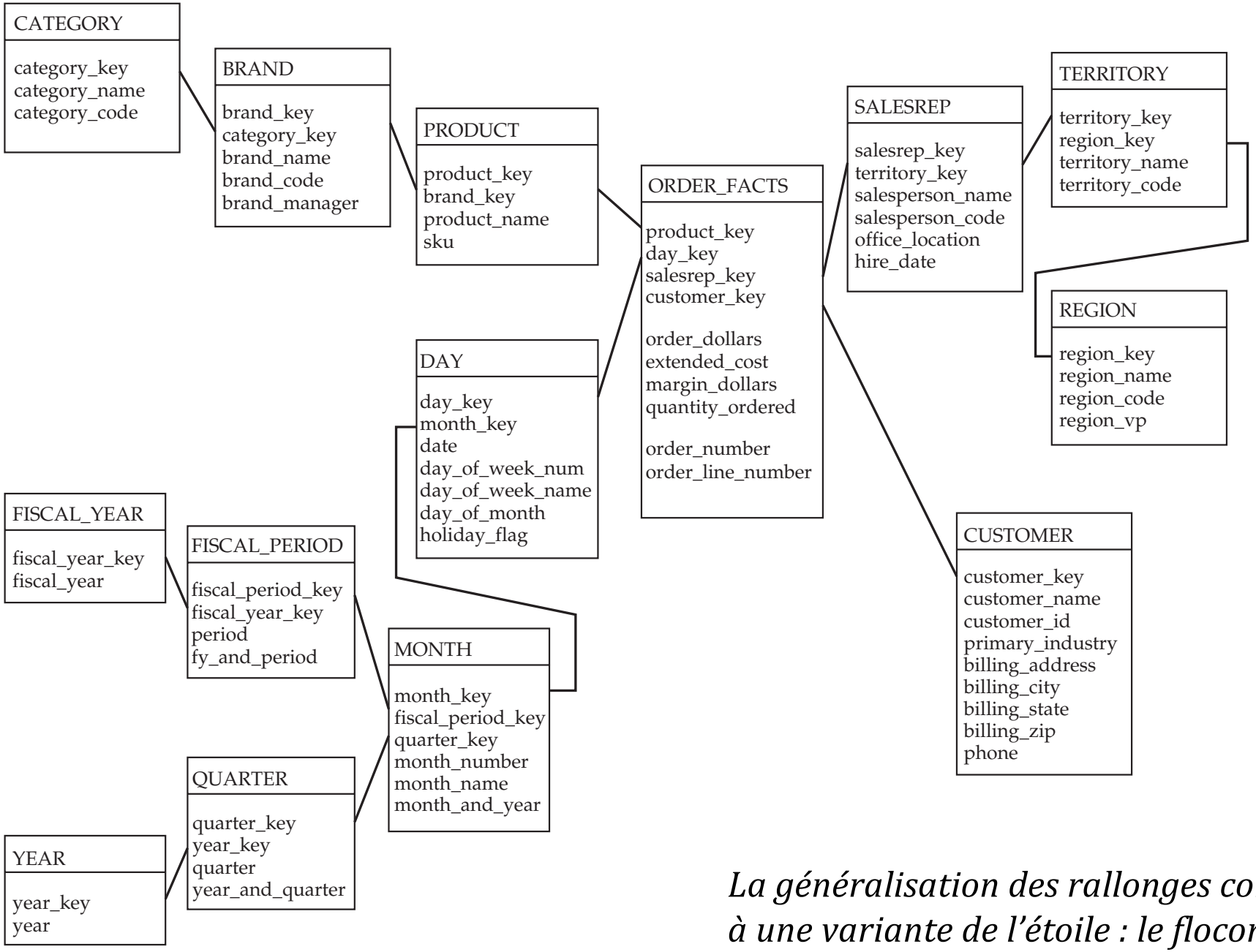


Figure 7-5 A snowflake schema

La généralisation des rallonges conduit à une variante de l'étoile : le flocon.

EXEMPLE NON ADDITIF

Margin Report

Date: January 1, 2009
Product: Gel Pen Red

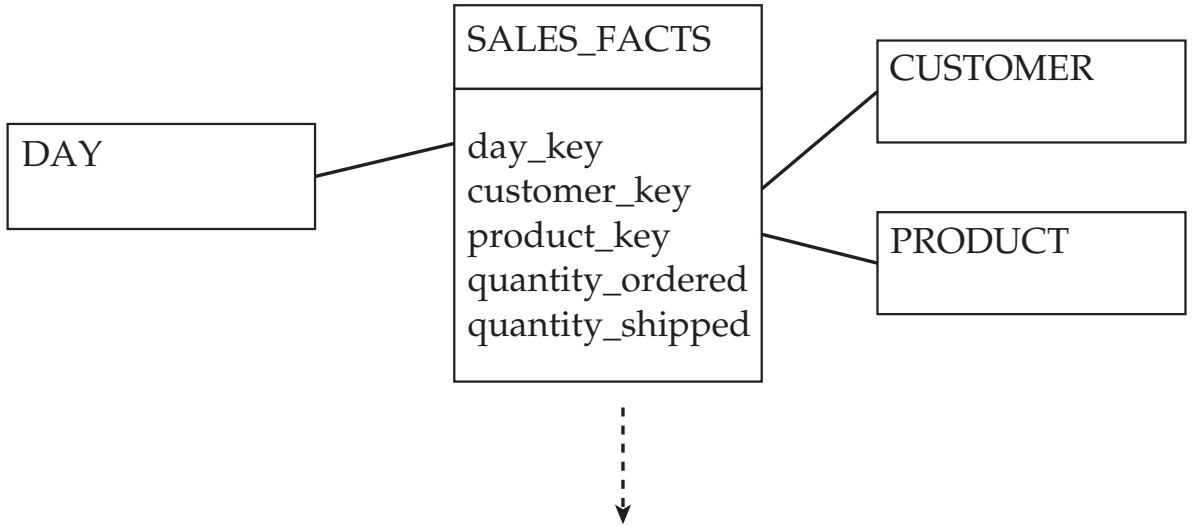
Salesperson	Customer	Margin Dollars	Order Dollars	Margin Rate
Jones	Balter Inc.	192.74	6,382.21	3.02%
	Raytech	39.05	1,293.11	3.02%
	Subtotal:	*231.79*	*7,675.32*	*3.02%*
Baldwin	Venerable Holdings	121.50	4,023.22	3.02%
	eMart LLC	253.44	8,392.00	3.02%
	Shatter & Lose	8.74	289.54	3.02%
	Subtotal:	*383.68*	*12,704.76*	*3.02%*
Sebenik	Comstock Realty	12.06	399.29	3.02%
	RizSpace	58.10	1,923.93	3.02%
	Starcomp	90.36	2,992.11	3.02%
	Subtotal:	*160.52*	*5,315,33*	*3.02%*
Sgamma	Implosion Town	213.88	7,082.22	3.02%
	DemiSpace	113.92	3,772.11	3.02%
	Subtotal:	*327.80*	*10,854.33*	*3.02%*
Grand Total:		***1,103.80***	***36,549.74***	***3.02%***

Margin Rate is a nonadditive fact.

Summary row is computed as a ratio of the subtotals, not by summing margin rates for the salesperson.

Figure 3-4 Nonadditive facts are computed as the ratio of additive facts

Exemple non synchrone



SALES_FACTS

day_key	customer_ key	product_ key	quantity_ ordered	quantity_ shipped
123	777	111	100	0
123	777	222	200	0
123	777	333	50	0
456	777	111	0	100
456	777	222	0	75
789	777	222	0	125

These zeros will cause trouble

Figure 4-1 Facts with different timing in a single table

```
SELECT
   product_key,
   SUM(quantity_shipped)
FROM
   sales_facts
GROUP BY
   product_key
HAVING
   SUM(quantity_shipped) > 0
```

Ceci entraine une complexité insidieuse et croissante (*boiling the frog*).

Fusionner les faits dans un seul fait général en ajoutant une dimension discriminante n'est pas une solution.

La meilleure solution demeure de construire deux tables de faits.

TECHNIQUES DIMENSIONNELLES

- Drill across
- Hiérarchisation et « conformed dimensions »
- Drill through
- Skip drill

« Drill across »

Le problème : jointure des tables de faits

ORDER_FACTS

day_key	customer_ key	product_ key	quantity_ key
123	777	111	100
123	777	222	200
123	777	333	50

SHIPMENT_FACTS

day_key	customer_ key	product_ key	quantity_ shipped
456	777	111	100
456	777	222	75
789	777	222	125

```
SELECT
   product.product,
   SUM( order_facts.quantity ordered ),
   SUM( shipment_facts.quantity_shipped )
FROM
   product,
   day,
   order_facts,
   shipment_facts
WHERE
   order_facts.product_key = product.product_key AND
   order_facts.day_key = day.day_key AND
   shipment_facts.product_key = product.product_key AND
   shipment_facts.day_key = day.day_key AND
   ...additional qualifications on date...
GROUP BY
   product.product
```

The order for product 222 is double counted

```
product        sum(quantity_ ordered)   sum(quantity_ shipped)
-----------    ----------------         -----------------

Product 111    100                      100

Product 222    400                      200
```

The order for product 333 does not appear

Figure 4-10 Joining two fact tables leads to trouble

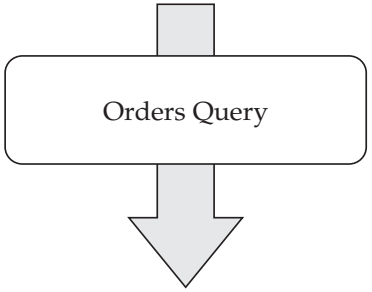
« Drill across » L'exemple

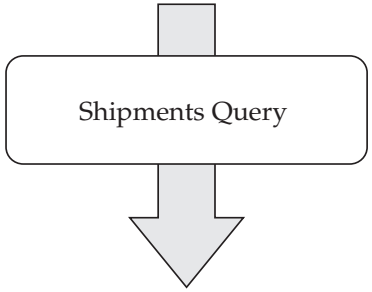
ORDER_FACTS

day_key	customer_ key	product_ key	quantity_ ordered
123	777	111	100
123	777	222	200
123	777	333	50

SHIPMENT_FACTS

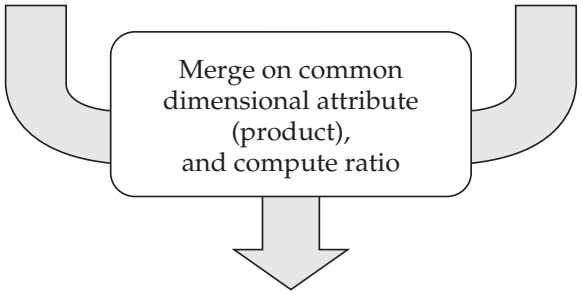
day_key	customer_ key	product_ key	quantity_ shipped
456	777	111	100
456	777	222	75
789	777	222	125






```
product          quantity ordered
===========      ================
Product 111      100
Product 222      200
Product 333      50
```

```
product          quantity shipped
===========      ================
Product 111      100
Product 222      200
```

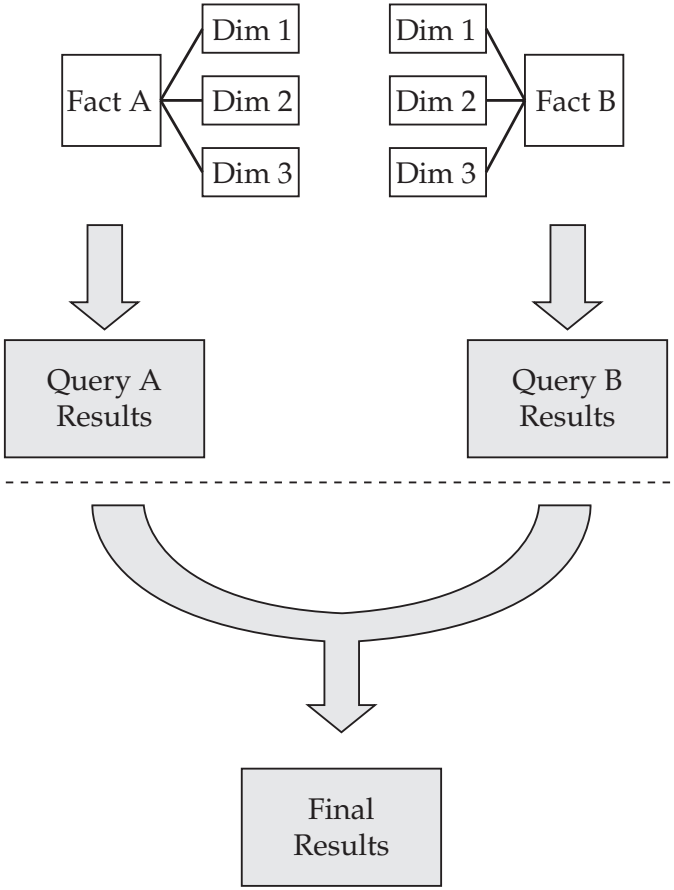



```
product       quantity ordered   quantity shipped   ratio
===========   ================   ================   ======
Product 111         100                100           100%
Product 222         200                200           100%
Product 333          50                               0%
```

Les résultats obtenus sont corrects, mais incomplets

Phase 1: Issue a separate query for each fact table

- Qualify each query as needed
- Get same dimensions in each query
- Summarize facts by chosen dimensions



Phase 2: Combine the result sets

- Perform a full outer join based on common dimensions
- Compute comparisons or ratios of facts if desired

Figure 4-12 Drilling across

« Drill across » La recette

Étapes

1. *Traiter séparément*
2. *Fusionner*
3. *Compléter*
4. *Épurer*

La fusion peut prendre différentes formes (jointure interne ou externe) selon qu'il y a des calculs ou non sur les attributs agrégés.

Ne regrettez-vous pas l'absence des opérateurs de semi-jointure (matching) et semi-différence (not matching) en SQL ?

« Drill across »
Le code (pour deux « étoiles » A et B)

SELECT
  COALESCE (A.clé, B.clé, ...) AS clé,
  A.quantité$_1$, ..., A.quantité$_{nA}$,
  B.quantité$_1$, ..., B.quantité$_{nB}$,
  f_1(A.quantité$_1$, ..., A.quantité$_{nA}$, B.quantité$_1$, ..., B.quantité$_{nB}$) AS résultat$_1$, ...
  f_{nR}(A.quantité$_1$, ..., A.quantité$_{nA}$, B.quantité$_1$, ..., B.quantité$_{nB}$) AS résultat$_{nR}$
FROM
  (
  SELECT
    A.clé,
    agg$_1$(att$_1$) AS quantité$_1$, ..., agg$_{nA}$ (att$_{nA}$) AS quantité$_{nA}$, ...
  FROM
    FA JOIN DA$_1$ ON (FA.dim$_1$=DA$_1$.clé) ... JOIN DA$_{nA}$ (FA.dim$_{nA}$=DA$_{nA}$.clé)
  WHERE
    ... autres restrictions, e.a. sur les dates ...
) AS A
FULL OUTER JOIN
  (...) AS B
ON A.clé = B.clé

- La « recette » ne peut être appliquée aveuglément, elle doit aussi respecter certaines contraintes
  - conformité des entités mises en cause;
  - si le résultat du drill across doit être utilisé comme « table de fait synthétique », deux autres contraintes sont applicables
    - granularité identique
    - synchronicité

- Pour les exprimer, nous avons besoin des concepts de hiérarchisation des entités et de celle des processus

Techniques
Hiérarchisation par entités

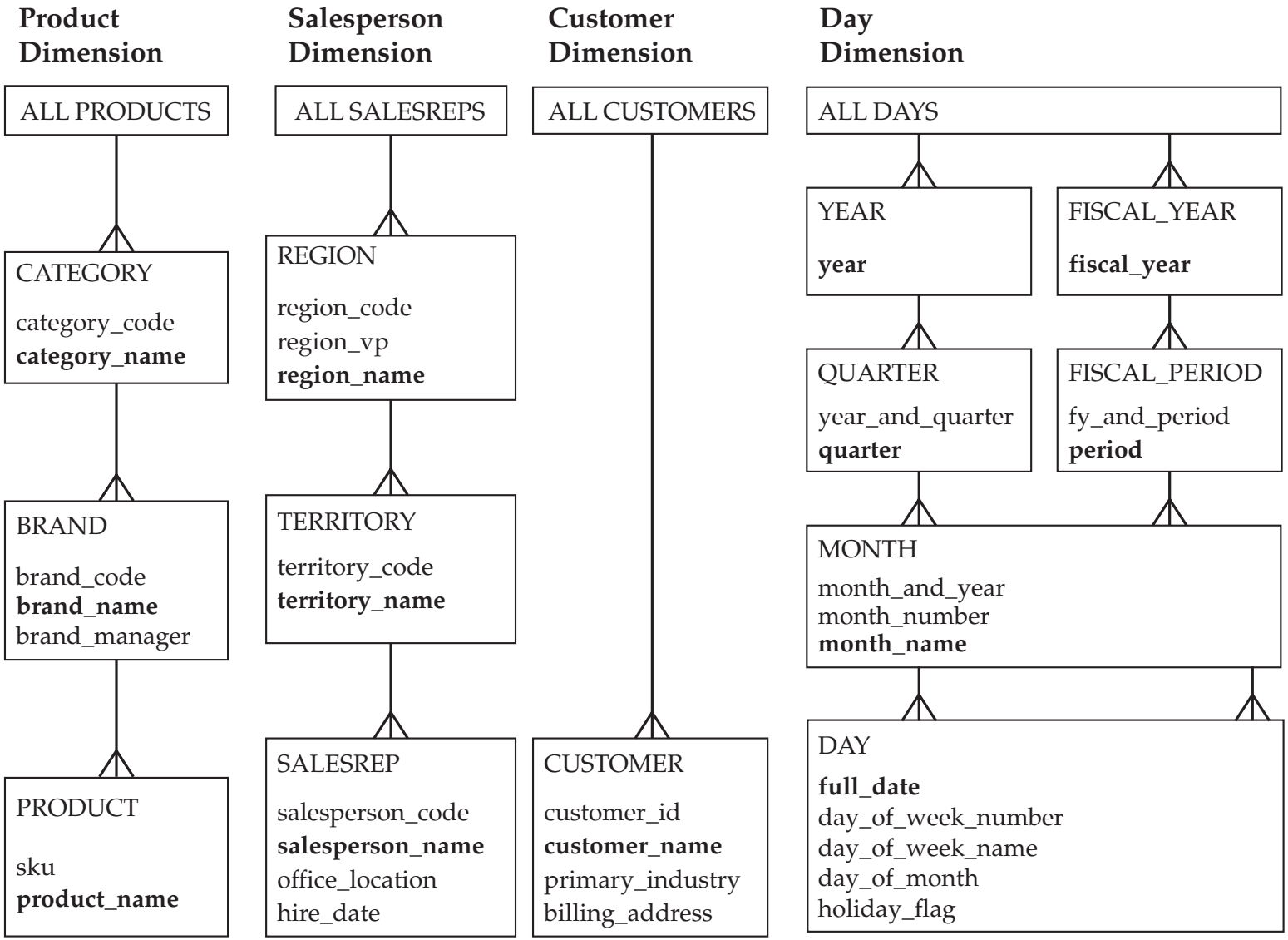
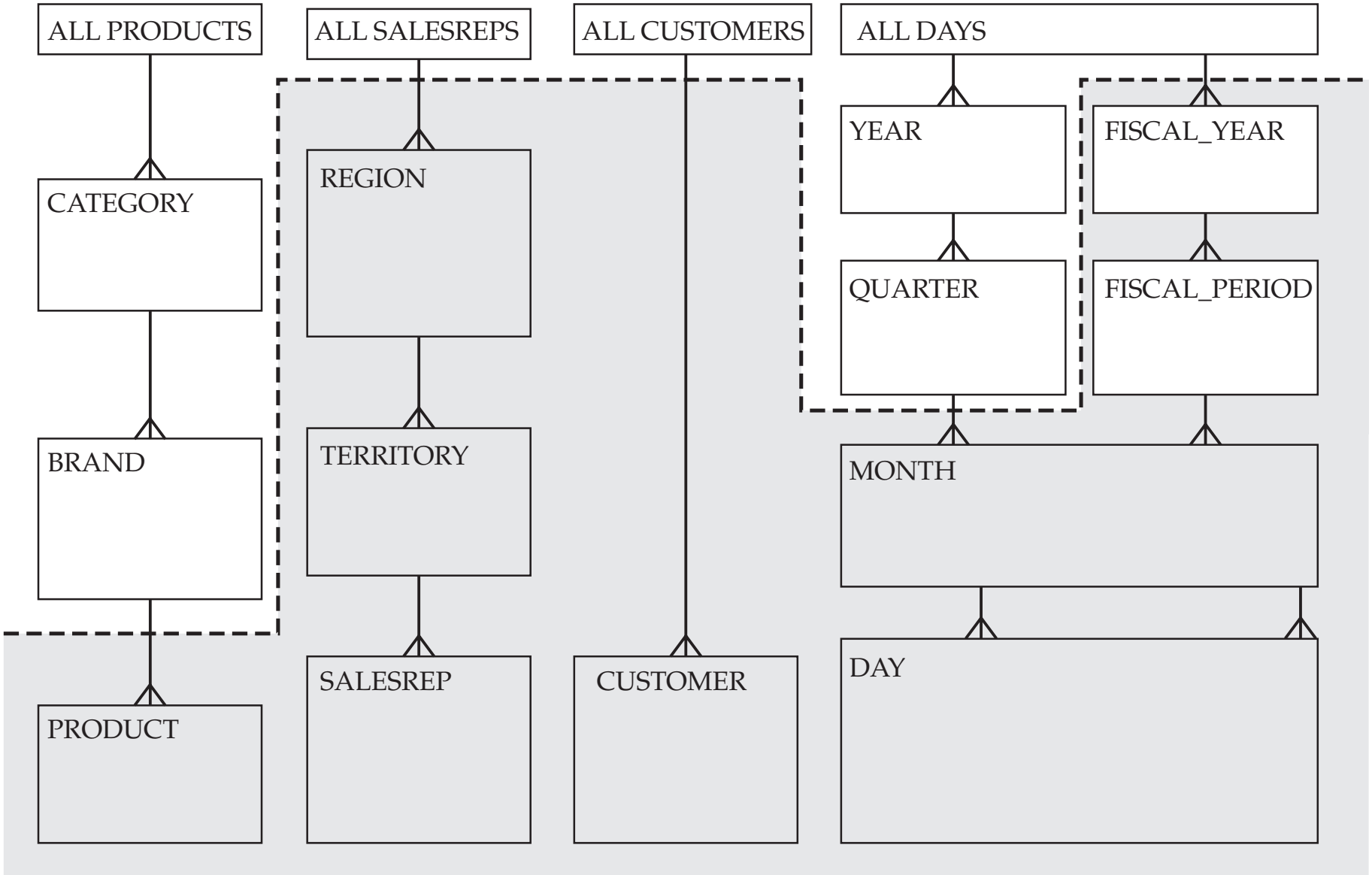


Figure 7-3 Documenting attribute hierarchies

Techniques
Construction de cubes



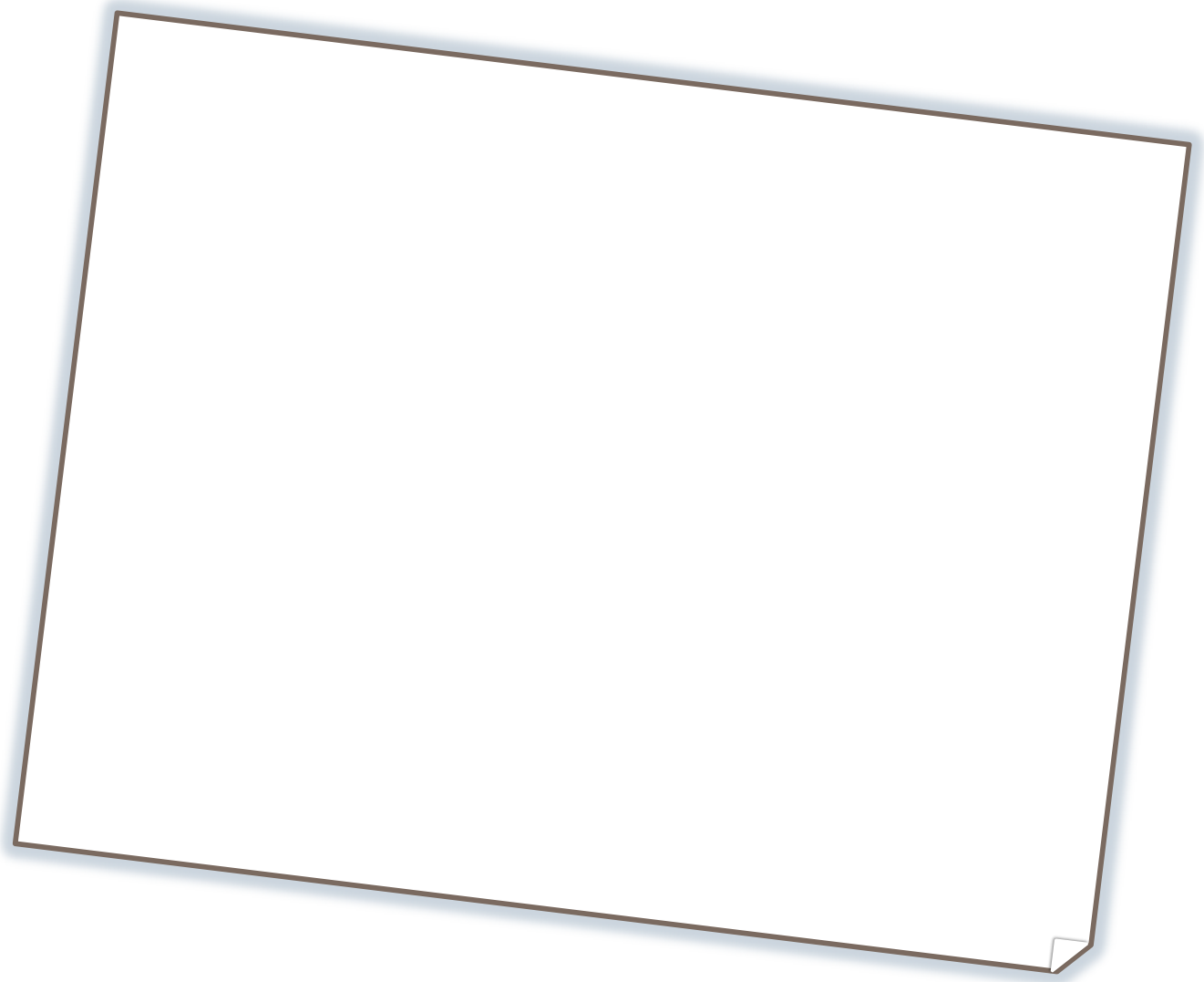
Techniques
Hiérarchisation par processus

- Les processus pouvant être hiérarchisés, il est tentant de faire de même pour les tables de faits pour obtenir des « tables de faits synthétiques ».
- Certaines précautions doivent toutefois être prises
  - la mise en relation de deux tables de faits, même si elles sont en relation de sous-processus, doit passer par un drill across;
  - un tel drill across doit donc répondre à des contraintes supplémentaires de granularité et de synchronicité en plus de celle de conformité des dimensions.

Techniques
Bridging

- Parce que les relations n x n... ça existe!
- Par exemple
  - multi dignostic
- Il faut donc adapter (généraliser) le « drill across » à n tables de fait.
- Voir exemple dans Adamson.

Les conseils d'Adamson



Dimensionnalité
Les conseils d'Adamson (1)

- *TIP-031*
  Assign each dimension table a surrogate key. This single column will be used to uniquely identify each row in the table.
- *TIP-032*
  Provide a rich and comprehensive set of dimension attributes. Each new attribute dramatically increases the number of analytic possibilities.
- *TIP-035*
  It is not always clear whether a numeric data element is a fact or a dimension. When in doubt, pay close attention to how it will be used. If the element values are used to filter queries, order data, control aggregation, or drive master–detail relationships, it is most likely a dimension.

Dimensionnalité
Les conseils d'Adamson (2)

- *TIP-037*
  Do not use the principles of normalization to guide dimension table design. Analytic databases do not benefit from these techniques. Situations that call for snowflakes and outriggers are the exception rather than the rule.
- *TIP-043*
  Avoid overusing degenerate dimensions. If an attribute is not a transaction identifier, consider placing it in a junk dimension instead.

Dimensionnalité
Les conseils d'Adamson (3)

- *TIP-048*
  Use type 1 changes carefully. They restate the context for associated facts. Confusion can be minimized by educating systems analysts and business users.
- *TIP-052*
  For each dimension attribute, choose and document the appropriate slow change response. If you are uncertain, the type 2 response is safest. When a source system captures the reason for a change, a single attribute may drive either type of response.

Dimensionnalité
Les conseils d'Adamson (4)

- *TIP-073*
  Never attempt to join two fact tables, either directly or through a common dimension. This can produce inaccurate results.
- *TIP-081*
  When available tools cannot drill across, or when drill-across reports suffer from poor performance, design and build a merged fact table that summarizes the processes at a common level of detail. This derived table performs the drill-across operation when the warehouse tables are loaded, instead of performing it at query time.

Dimensionnalité
Les conseils d'Adamson (5)

- *TIP-118*
  When two dimension attributes share a natural affinity, and are only related in one context, they belong in the same dimension table. When their relationships are determined by transactions or activities, and they can occur in multiple contexts, they should be placed in separate dimension tables.
- *REC-122*
  Relocate Free-Form Text Fields to an outrigger. Excessive row length is often a result of the inclusion of several free-form text fields in the dimension table.

Dimensionnalité
Les conseils d'Adamson (6)

- *TIP-130*
  When a fact table and dimension table have multiple relationships, it is not necessary to build multiple copies of the dimension. Each role can be accessed by joining views or aliases of the dimension to the appropriate foreign keys in the fact table.

Dimensionnalité
Les conseils d'Adamson (7)

- *TIP-134*
  Do not allow the storage of NULLs in dimension columns. Instead, choose a value that will be used whenever data is not available.
- *TIP-136*
  Avoid allowing NULL values in foreign key columns. They require alternative join syntax and create NULL instance values for dimension columns even when NULLs are not stored.
- *TIP-140*
  Special-case rows can be added to dimensions to deal with incorrect or missing information. This avoids the need to exclude facts from the warehouse. The star should record sufficient transaction identifiers to allow the anomalous record to be identified and corrected in the future.

Questions ouvertes
Temporalisation

- La conception des tables de faits semble conduire naturellement à une temporalisation « par tuple » excluant le type 2, d'où la recommandation (forte) de ne pas temporaliser, car l'utilisation type 1 invaliderait tous les rapports antérieurs.
- En conclusion, si la possibilité de corriger les faits (donc leur temporalisation) est nécessaire, la dimensionnalité n'apporte pas de solution.

Questions ouvertes
Le retour des jointures

- Il en découle que plusieurs requêtes devront faire appel à des jointures :
  - pour lier des faits de TF différentes « drilling across »;
  - pour lier des dimensions différentes « (dimension) merging »;
  - pour réduire le poids de dimensions obèses, la hiérarchisation et en particulier les flocons (« snow flake »);
  - en général, pour traiter les requêtes non anticipées.
- Pourquoi donc avoir tout fait pour les exclure au départ ?

Principales références

- Adamson, C. 2010.
  The complete reference star schema.
  McGraw-Hill, New York, NY, USA.
- Adamson, C. 2008-2015.
  http://blog.oaktonsoftware.com
- Inmon, W.H. 2005.
  Building the data warehouse.
  John Wiley, Indianapolis, IN, USA.
- Jiang, B. 2015.
  Constructing Data Wharehouses with Metadriven Generic Operators, and more.
  2nd ed., Createspace.
- Kimball, R. 2013.
  The data warehouse toolkit: the definitive guide to dimensional modeling.
  John Wiley, Indianapolis, IN, USA.