IGE487 - Module 8 – Entrepôts de données

Cours IGE487 – Modélisation de bases de données

Chargé de cours Tudor Antohi

Objectifs

- Problématique
- Architecture : Lac de données, Entrepôt, Data Mart, Hub
- Bill Inmon – Information Factory
- Ralph Kimball – Entrepôt de données
- Dimensionnalité et techniques dimensionnelles
- Conseils d’Adamson

Problématique

Problématique

La modélisation basée sur la normalisation doit couvrir d'une façon **performante et avec une bonne mise a echelle** :

- **les concepts temporels (on veut garder l'histoire)**
- les bases de données actives (qui contient de triggers)
- les concepts spatiaux , graph, multimédia
- les sources de données différentes
- la science de données , les systèmes expert etc.
- les données sont structurées , sans structure ou semi-structure

Obs.: Les bases de données temporelles, englobent toutes les applications de base de données qui nécessitent un certain aspect du temps lors de l'organisation de leurs informations.

Problématique – Luc Lavoie

	Exploitation	*Analyse*
Objectif	Soutenir les processus	Mesurer les processus
Fonctions	CRUD	R *(ucd)*
Cible d'optimisation	U (r)	R
Portée	Transaction	Lot de transactions
Nature des requêtes	Prédéfinie et stable (souvent plus de 90%)	Prédéfinie et stable (souvent moins de 50%)
Temporalité	Courante	Historique (surtout passé)
Principes de conception	Consensus : Normalisation (3FN, FNBC, 5FN)	Débat : Dimensionalité, 6FN
Appellations courantes	Système transationnel On Line Transaction System (OLTP) Système source	Système analytique Entrepôt de données Data Mart Data Wharehouse

Architecture de données

Architecture moderne

1. **Bases de données opérationnelles** envoient les données avec **processus ETL** utilisant …

2a. Systèmes de messagerie

2b. Systèmes de transfert de fichiers

3. Vers **lacs de données** et/ou **dépôts de données opérationnelles** et/ou d**ata marts** vers …

4. Vers **entrepôts de données** ou **lacs entrepôts** vers …

5. **Cubes de données** et/ou **data marts**

Les données peuvent être exposées avec des **hubs de données.**

Et plus tard on va mentionner les architectures maillées – data mesh.

Extraction, transformation et chargement (1)

Le processus d'extraction des données des systèmes sources et de leur transfert dans l'entrepôt de données est communément appelé ETL, qui signifie extraction, transformation et chargement.

Extraction des données

La première étape est l'extraction. Au cours de l'extraction, les données sont spécifiquement identifiées et ensuite prélevées à de nombreux endroits différent. Ces données peuvent provenir d'une variété de choses, comme des fichiers, queues de messages, des feuilles de calcul, des bases de données et des applications, etc. Selon les capacités du système source (par exemple, les ressources du système d'exploitation), certaines transformations peuvent avoir lieu pendant ce processus d'extraction. La taille des données extraites varie de quelques centaines de kilo-octets à plusieurs giga-octets, selon le système source et la situation commerciale. C'est également le cas pour la période entre deux extractions ; certaines peuvent varier de jours ou d'heures à presque en temps réel.

Transport des données

Une fois les données extraites, elles doivent être physiquement transportées vers le système cible ou vers un système intermédiaire pour traitement ultérieur. Selon le mode de transport choisi, certaines transformations peuvent également être effectuées au cours de ce processus.

Extraction, transformation et chargement (2)

Transformation des données

La prochaine étape du processus ETL est la transformation. Une fois les données extraites, elles doivent être physiquement transportées vers la destination cible et converties dans le format approprié. Cette transformation de données peut inclure des opérations telles que le nettoyage, l'assemblage et la validation des données.

Chargement des données

La dernière étape du processus ETL consiste à charger les données transformées dans la cible de destination : entrepôts de données, DDOE, lacs de données. Il existe deux méthodes principales pour charger les données dans un entrepôt : **chargement complet et chargement incrémentale**. La méthode du chargement complet implique un déchargement complet des données qui a lieu la première fois que la source est chargée dans l'entrepôt. La charge incrémentale, par contre, a lieu à intervalles réguliers. Ces intervalles peuvent être des **incréments de flux** (meilleurs pour de plus petits volumes de données) ou des **incréments de lots** (meilleurs pour de plus grands volumes de données).

DDOE (ODS)

DDOE – **dépôt de données opérationnel** – terme utilisé pour désigner la forme intermédiaire des bases de données avant qu'elles soient nettoyées, agrégées et transformées en entrepôt de données.

Dépôt de **données a court terme,** peut quand même avoir un certain nettoyage de données. Les actualisations sont transactionnelles.

En général, une étape intermédiaire (optionnelle) entre les SGBDs opérationnelles et les systèmes analytiques.

Utilisé pour **analyser les transactions (analytiques opérationnelles, rapides)**

Lac de données (1)

Un amalgame des différents types de données trouvés dans l'organisation. La qualité, la performance, la scalabilité ne sont pas conformes.

- Données avec structure
- Données texte (sans structure)
- Données IoT

Le lac de données est l'endroit où les entreprises chargent toutes leurs données, systèmes de stockage à faible coût avec une API de fichier qui contient des données dans des formats de fichiers génériques et ouverts, tels **qu'Apache Parquet et ORC.**

L'utilisation de f**ormats ouverts** a également rendu les données du lac de données directement accessibles à un large éventail d'autres moteurs d'analyse, langages et systèmes d'apprentissage automatique.

Lac de données (2)

Format de données propriétaire : ouvert (Parquet, ORC, Avro)

Type de données : toutes les types

Langage : SQL , R, Python, Rest APIs

Transactionnel : en général non

Sécurité : pauvre, se fie sur le SGF et le système qui héberge les fichiers

Performance : pauvre. La partie calcule n'est pas optimisée

Scalabilité : couts bas, quantité énormes de données

Cas d'affaires : en général étapes intermédiaires, stockage longue terme et/ou apprentissage machine

Entrepôt de données (1a) – Elmasri

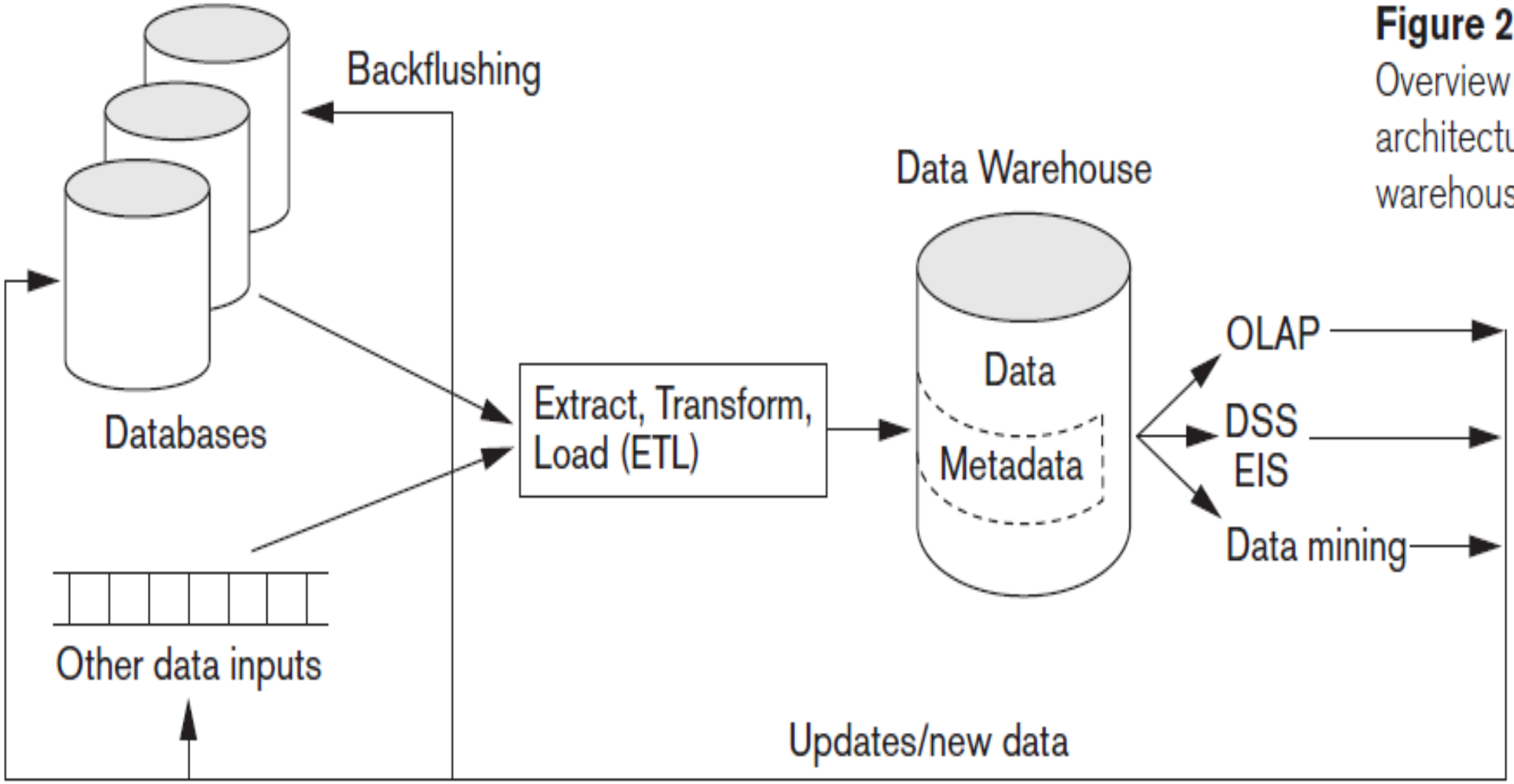
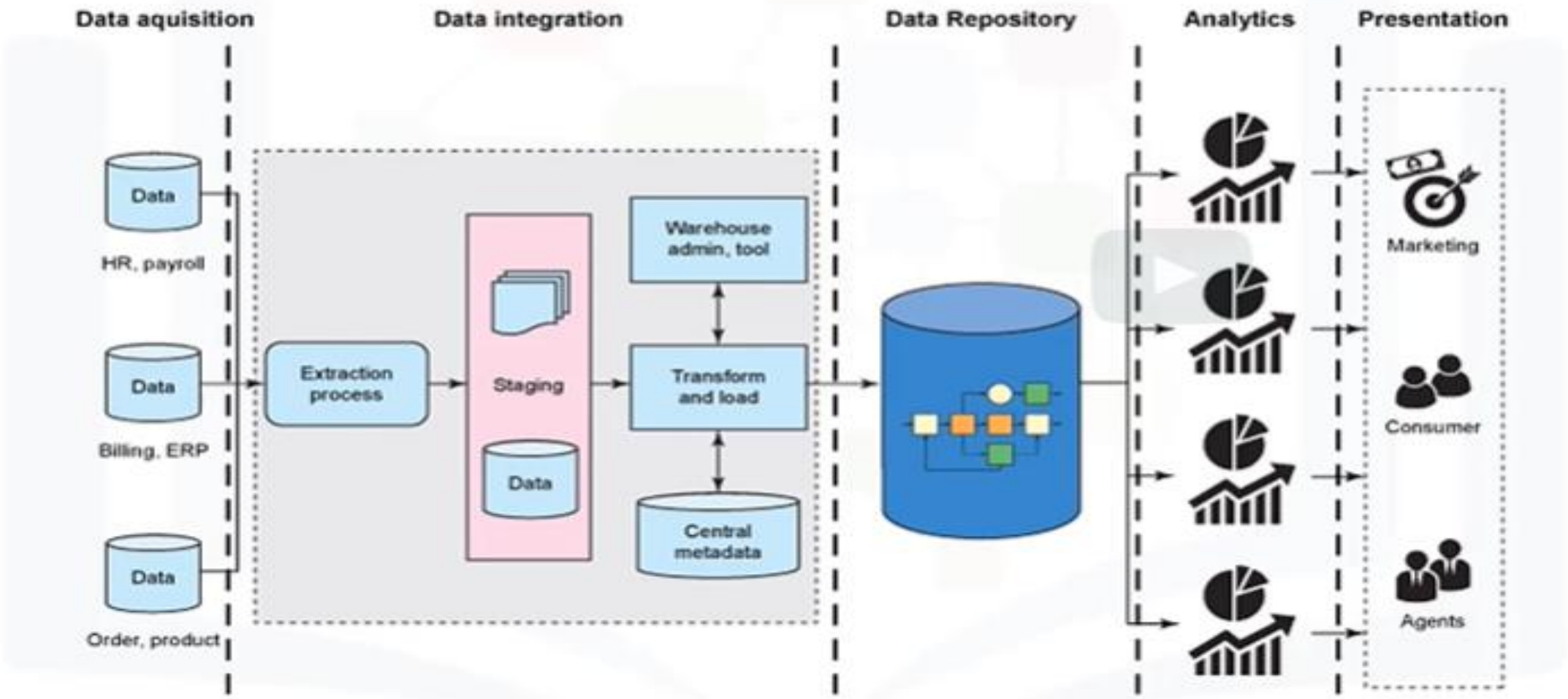
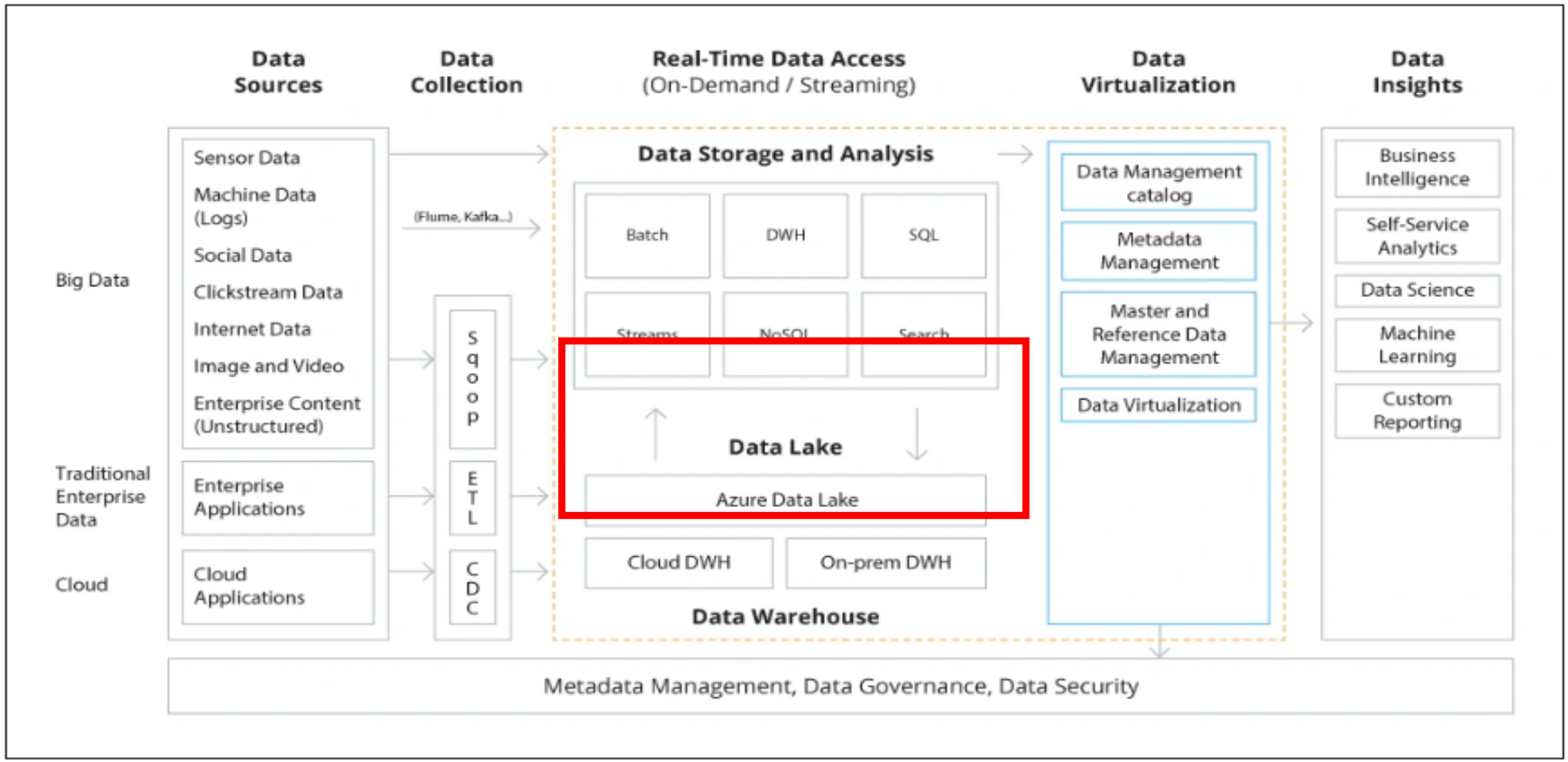


Figure 29.1
Overview of the general architecture of a data warehouse.

Entrepôt de données (1b) – données organisées



Entrepôt de données (1c) – vision moderne



Staging Area – zone de préparation

- ✓intégration
- ✓détection des changements,
- ✓cédule des processus
- ✓nettoyage et validation des données
- ✓agrégation des données et …
- ✓normalisation des données

Évolution : La zone de préparation est sauvée dans le lac de données dernièrement pour éviter perdre le *Dark Data.*

Entrepôt de données (1)

Une collection organisée de données, orientée sujet, intégrée, non volatile et variable dans le temps à l'appui des décisions.

Les bases de données traditionnelles sont transactionnelles (relationnelles, orientées objet, réseau ou hiérarchiques).

- **Les entrepôts de données ont la particularité d'être principalement destinés à des applications d'aide à la décision. Ils sont optimisés pour la récupération des données, et non pour le traitement des transactions.**
- Les **entrepôts de données permettent d'accéder aux données pour des analyses complexes**, la découverte de connaissances et la prise de décision via des **requêtes ad hoc et prédéfinies**.
- Les **requêtes prédéfinies font référence à des requêtes définies a priori** avec des paramètres qui peuvent se reproduire à haute fréquence.

Entrepôt de données (3)

Format de données propriétaire : ex. Oracle , SQL Server

Type de données : structurées avec un support limité semi-structure

Langage : SQL

Transactionnel : ACID

Sécurité : comme une base de données normale

Performance : bonne (partitionnement, infrastructure)

Scalabilité : chère , stockage cher

Cas d'affaires : IA, SQL complexes et support a la décision, science de données

Entrepôt de données (4)

Les entrepôts de données à l'échelle de l'entreprise sont d'énormes projets nécessitant un investissement massif en temps et en ressources.

Comme alternative, **les entrepôts de données virtuels** fournissent des **vues** des bases de données opérationnelles qui sont matérialisées pour un accès efficace.

Les entrepôts de données logiques utilisent des techniques de **fédération, distribution et de virtualisation des données.**

Entrepôt de données (5) - processus

- Stockage des données selon le modèle de données de l'entrepôt
- Création et maintenance des structures de données requises
- Création et maintenance de chemins d'accès appropriés (ex. indexation, statistiques)
- Fournir les actualisations temporelles de dimensions
- Mise à jour des données de l'entrepôt
- Épuration des données

Entrepôt de données – considération de design

- Cas d’utilisation
- Modelé de données (normalisé ou non)
- Sources de données (intégration)
- Métadonnées (catalogue)
- Composants modulaires
- Gestion et maintenance
- Considérations sur l'architecture distribuée et parallélisme

Data Marts sont en general des DDA (ADS)

Les magasins de données (***data marts***) ciblent généralement un sous-ensemble de l'organisation, tel qu'un département, et sont plus étroitement ciblés.

DDA - **dépôt de données analytique** – terme utilisé pour désigner la forme des bases de données organisée pour les analyses.

Les **data marts** sont de DDAs similaires à des entrepôts de données mais ciblés sur des besoins d'affaires spécifiques. Architectures de type *hub and spoke* . Elles peuvent être situées après la création de l'Entrepôt de données en general selon Inmon ou en general avant selon Kimball.

Ex.: DSS (systèmes d'aide à la décision), connus aussi sous le nom d'EIS (ou MIS) - systèmes d'information exécutifs (ou systèmes d'information de gestion) soutiennent les principaux décideurs d'une organisation avec des données (analytiques) pour les décisions complexes et importantes.

Cubes (implantation de data marts) - 1

Les **cubes** sont une représentation logique de **données multidimensionnelles.** Le **bord du cube contient les membres de dimension** et le **corps du cube contient les valeurs** de données.

OLAP (traitement analytique en ligne) est un terme utilisé pour décrire l'analyse de données à partir de l'entrepôt de données.

Les outils OLAP permettent **une interrogation simple et rapide** des données analytiques stockées dans les entrepôts de données et les magasins de données.

- Vue conceptuelle **matrice multidimensionnelle**
- **Dimensions et niveaux d'agrégation illimités**
- **Opérations inter-dimensionnelles illimitées**
- **Gestion dynamique des matrices**
- Dans la **cellule de la matrice = la valeur agrégée**

Cubes (une implantation de data marts) - 2

MOLAP entraîne **le stockage des agrégations de la partition et d'une copie de ses données source dans une structure multidimensionnelle**, optimisée pour maximiser les performances des requêtes

ROLAP entraîne le stockage des agrégations de la partition dans des **vues indexées de la base de données relationnelle** spécifiée dans la source de données de la partition et **n'entraîne pas le stockage** d'une copie des données source.

HOLAP combine les attributs de MOLAP et ROLAP. Comme MOLAP, HOLAP **entraîne le stockage des agrégations** de la partition dans une structure multidimensionnelle. HOLAP **n'entraîne pas le stockage d'une copie des données source**

Data Hub

Les **entrepôts de données** et **les lacs de données** existent principalement pour prendre en charge les charges de travail analytiques.

Les **hubs de données** ne sont pas principalement des structures analytiques - ils permettent l'intégration, le partage et la gouvernance des données.

Ils peuvent comprendre des **services d'exposition de données** , **routage, service de normalisation et qualité.** Ils peuvent être virtualisés. Les hubs peuvent offrir des services d'accès aux ED, Cubes, DataMarts etc.

Data Hub peut exposer un MDM

La **gestion des données de référence (MDM),** un concept populaire au sein des entreprises, est de définir les normes, les processus, les politiques et la gouvernance liés aux entités de données critiques de l'organisation.

Les **tables de dimension** — qui, dans un entrepôt de données, matérialisent les concepts, tels que les clients, les régions et les catégories de produits — **représentent essentiellement les données de base**.

Les dimensions sont partagées entre plusieurs **faits ou datamarts** de rapports, et doivent être nettoyées et harmonisées (concilier les différences définitionnelles et notionnelles entre plusieurs systèmes sources d'où proviennent les données de dimension).

Les structures de table contenant ces dimensions deviennent de bons candidats a partager pour être utilisés dans d'autres environnements.

Trois questions

1. Dans quelle ordre on construit les stockages de données opérationnelles et analytiques **?**

2. Quels types de stockages sont nécessaires selon **les cas d'affaires ?**

3. Quels **modèles de données** selon les stockages et les cas d'affaires ?

Information Factory – Bill Inmon

Usine d'information d'entreprise

Usine d’information d’entreprise

- **L'usine d'information d'entreprise (CIF)** suit une architecture de flux de données de haut niveau préconisée par Bill Inmon et Claudia Imhoff [2001].
- Le **CIF** rassemble des données à partir de sources et les transforme en un **référentiel normalisé dans la couche d'intégration de l'architecture de référence**.
- À partir de là, **les informations sont subdivisées en datamarts** départementaux, fournissant les colonnes et les lignes spécifiques nécessaires à chacun.
- Dans le modèle **CIF**, **les données stockées dans la couche d'intégration doivent être une « version unique de la vérité »** au sein de l'entreprise.

Usine d’information d’entreprise

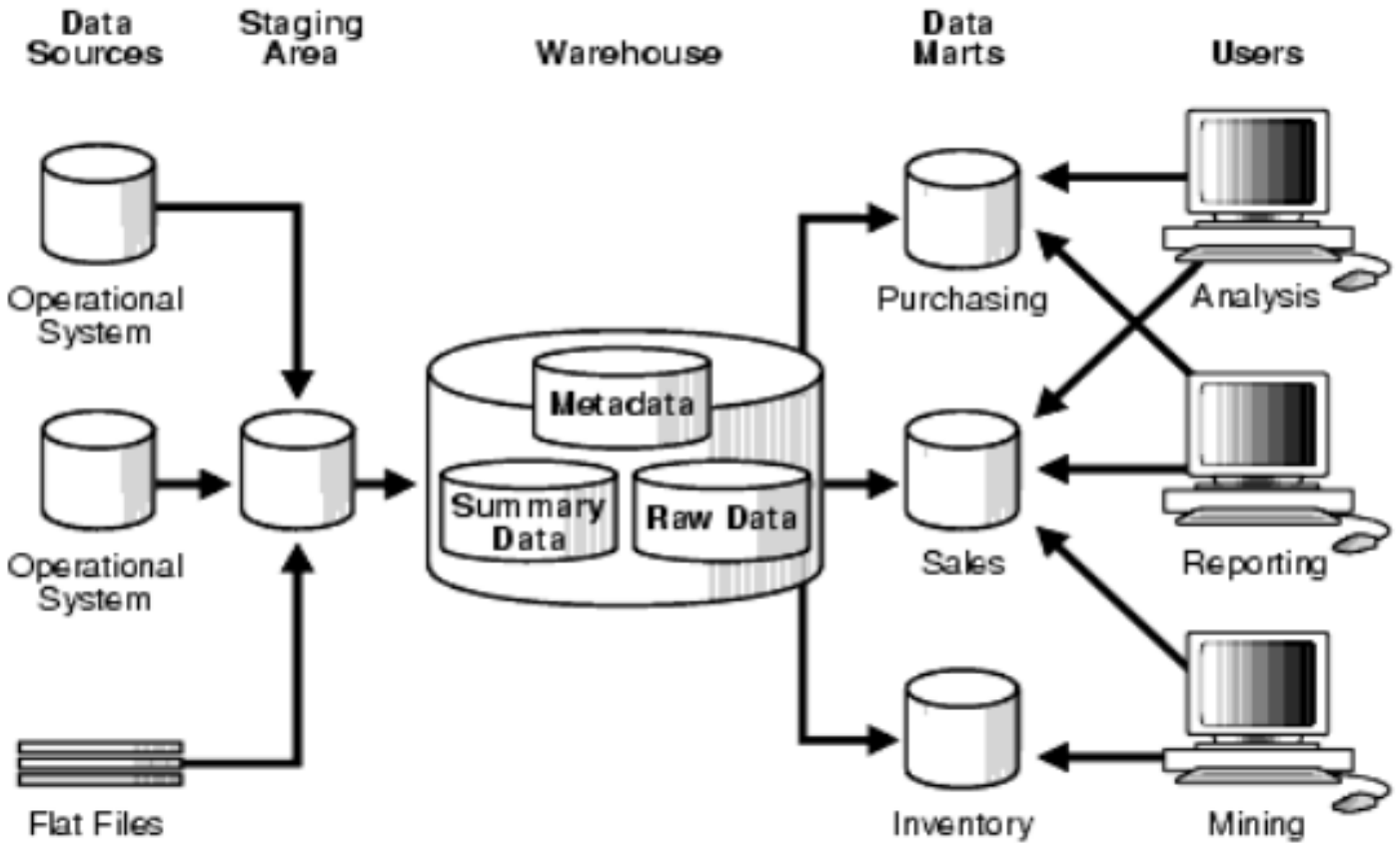


Figure 1.2, Source: Stanford. 2003. “Data Warehousing Concepts”

Avantages

- L'entrepôt de données sert véritablement de **source unique de vérité pour l'entreprise**. C’est un pointe d’intégration flexible.
- Les anomalies de mises à jour des données sont évitées grâce à une très **faible redondance**. Les processus ETL ne sont pas dupliqués.
- Les processus métier peuvent être compris facilement, car **le modèle logique représente les entités métier** détaillées.
- Peut gérer les divers besoins de **rapports centralisés** dans l'entreprise.
- **Maintenance facile**

Désavantages

- Les modèles et la mise en œuvre peuvent devenir complexes au fil du temps car ils impliquent **davantage de tables et de jointures** (impacts sur la performance)
- **Pas de modèle unique mais un concept de model corporatif complexe.** Besoin de ressources en modélisation de données et de l'entreprise elle-même. La configuration initiale et la livraison prendront plus de temps, et la direction doit en être consciente.
- Plus de ETL nécessaire car les datamarts sont construits à partir de l'entrepôt de données.

Évolution de concept Inmon vers le lac-entrepôt de données

Format de données : ouvert

Tous les types de données : structurés, semi-structurés, texte, sans structure

Accès : SQL, Python, R & autres langages

Qualité de données : transactionnel acide

Performance : bonne

Architecture : unifiée

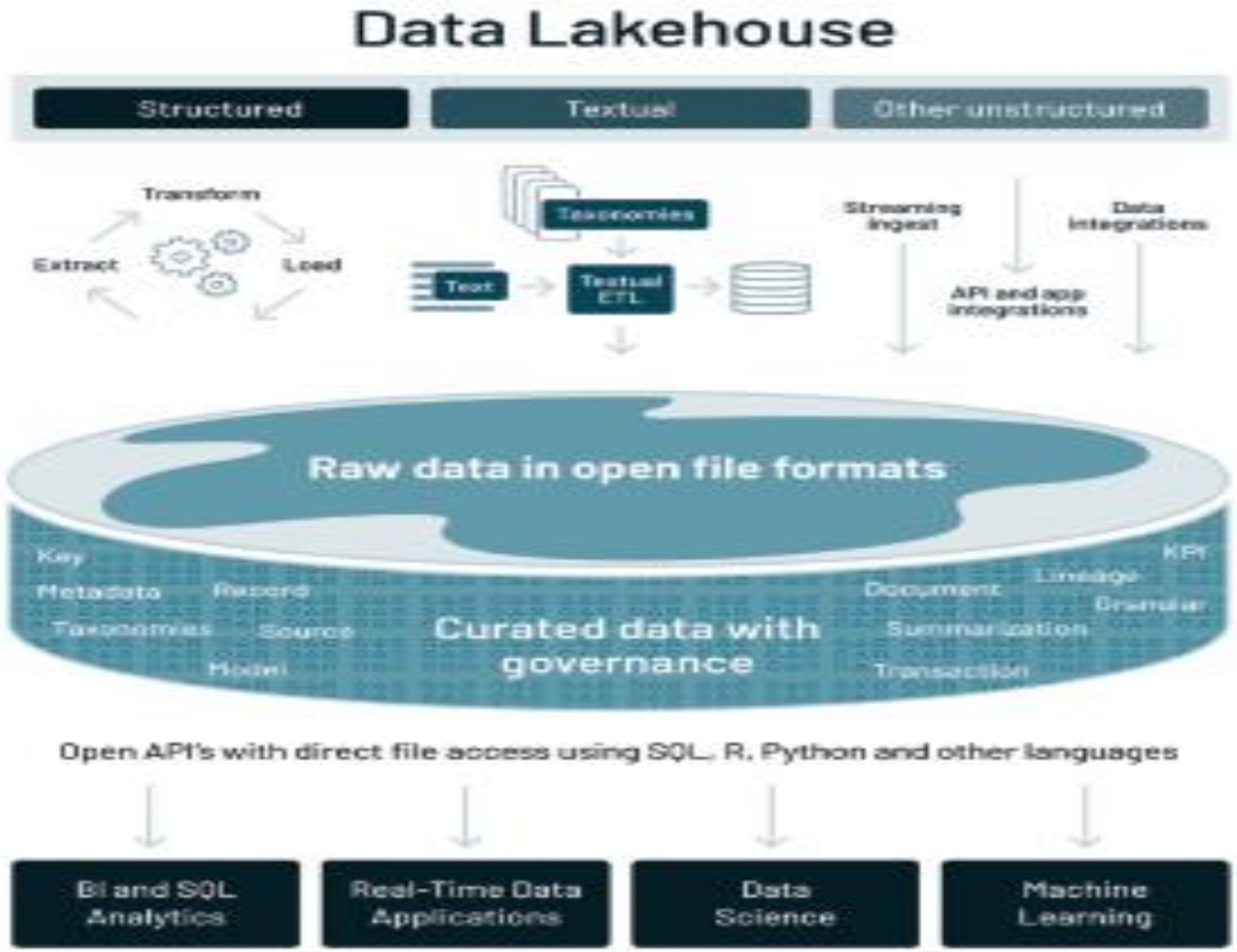


Figure 1-6. The data lakehouse architecture.

Technologies ED – exemple

Azure DataLake – lac de données

Azure Databricks – lac-entrepôt de données

Azure Synapse Analytics – combinaison de tradition avec le concept de lac entrepôt

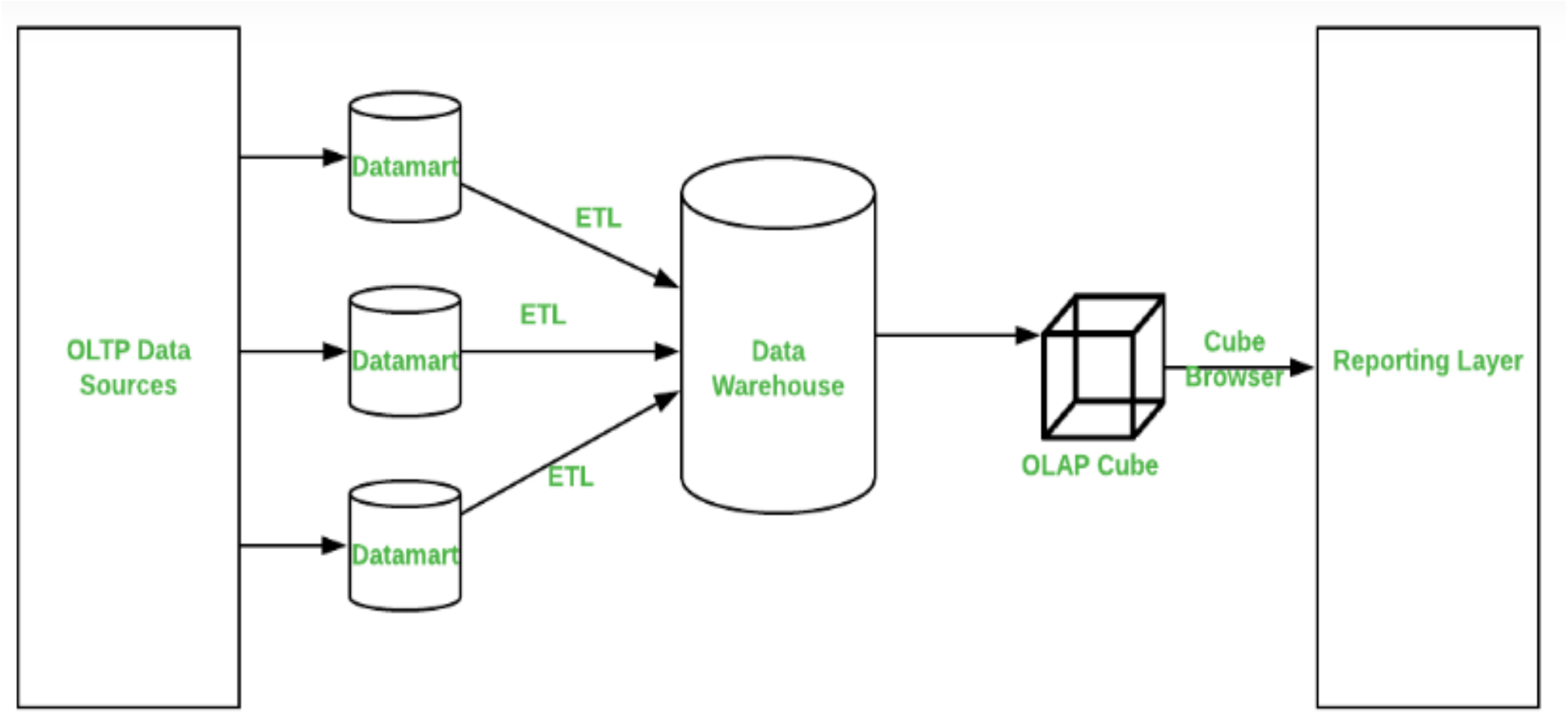
AWS Redshift - entrepôt de données basé sur PostgreSQL

Entrepôts de données – Ralph Kimball

Vers un modèle universel

Effectuez le travail maintenant pour faciliter les requêtes ultérieures

Kimball – ED – diagramme



Kimball – ED

La modélisation dimensionnelle utilise **deux types de tables - Faits et Dimensions.**

La **table de faits contient les faits (les mesures) de l'entreprise** et la **table de dimension contient le contexte** des mesures.

La modélisation **dimensionnelle** de Ralph Kimball adopte une **approche ascendante** des principes de conception de **schéma en étoile et de flocon de neige.**

La **conception est dénormalisée** avec les données réparties sur **les faits** représentant les ensembles de données transactionnelles numériques et les dimensions, qui sont des **données de référence** qui prennent en charge la transaction de fait.

Le **modèle est générique**, et les développeurs ont plus de **facilité a l'implanter.**

Architecture – Luc Lavoie

	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

Modèle de données ED

Le **modèle multidimensionnel** (également appelé modèle dimensionnel) implique deux types de tables : les **tables de dimension et les tables de faits**.

Une **table de dimension** se compose de **tuples d'attributs de la dimension**.

Une **table de faits** peut être considérée comme ayant des **tuples, un tuple par fait enregistré**. Ce fait contient une ou **plusieurs variables mesurées ou observées et les identifie avec des pointeurs vers des tableaux de dimensions**. La table de faits contient les données et les dimensions identifient chaque tuple dans ces données.

Une **table de faits** est comme une vue agglomérée des données de transaction alors que chaque table de dimension représente ce que l'on appelle les "données principales" auxquelles ces transactions appartenaient.

Dans les **systèmes de bases de données multidimensionnelles**, le modèle multidimensionnel a été mis en œuvre sous la forme d'un **système logiciel spécialisé** connu sous le nom de **base de données multidimensionnelle**.

Modèles de données étoile - ETS

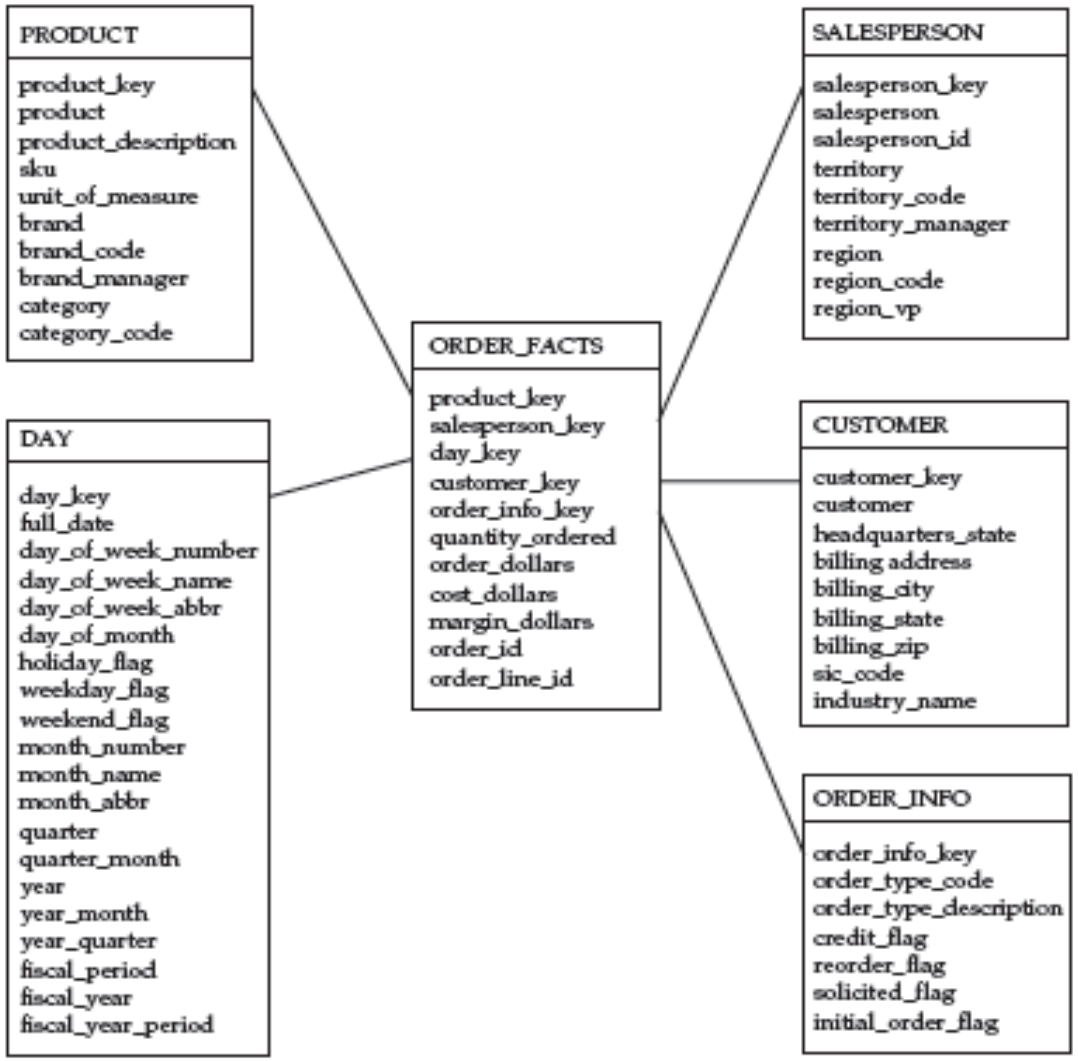


Figure 6-1 A fact table explicitly relates dimension tables

Modèles de données étoile - Elmasri

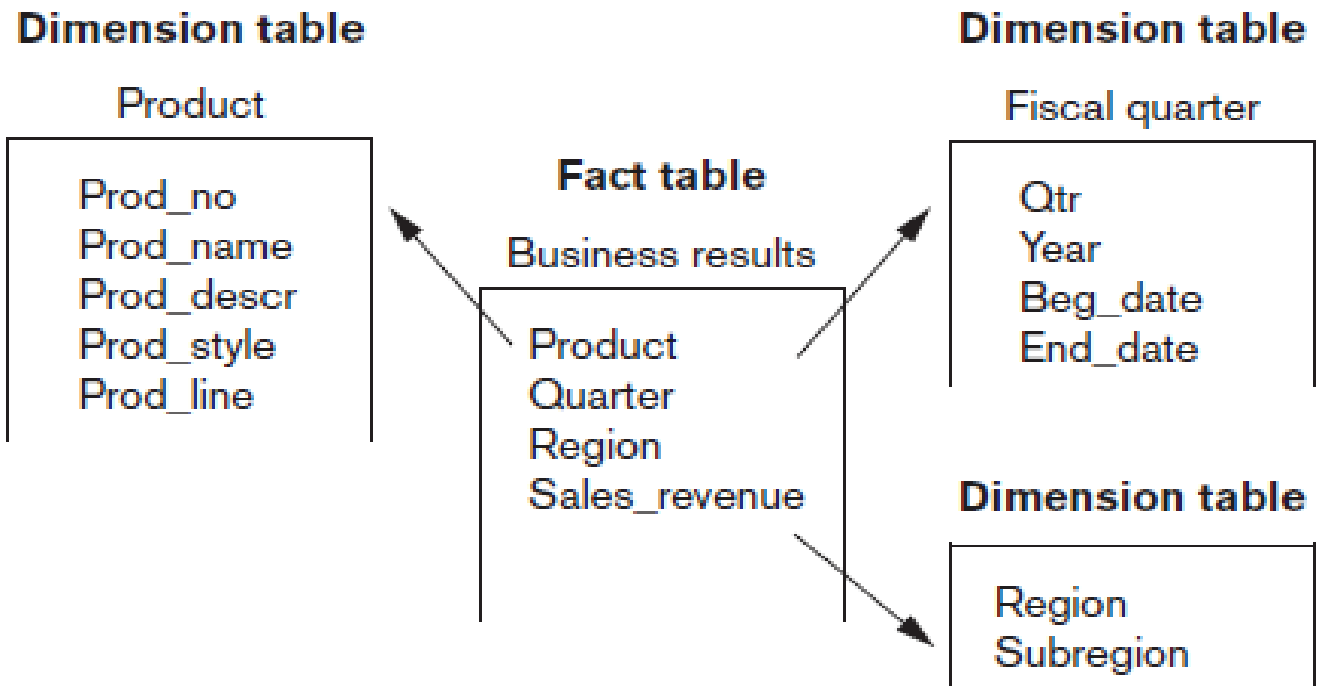
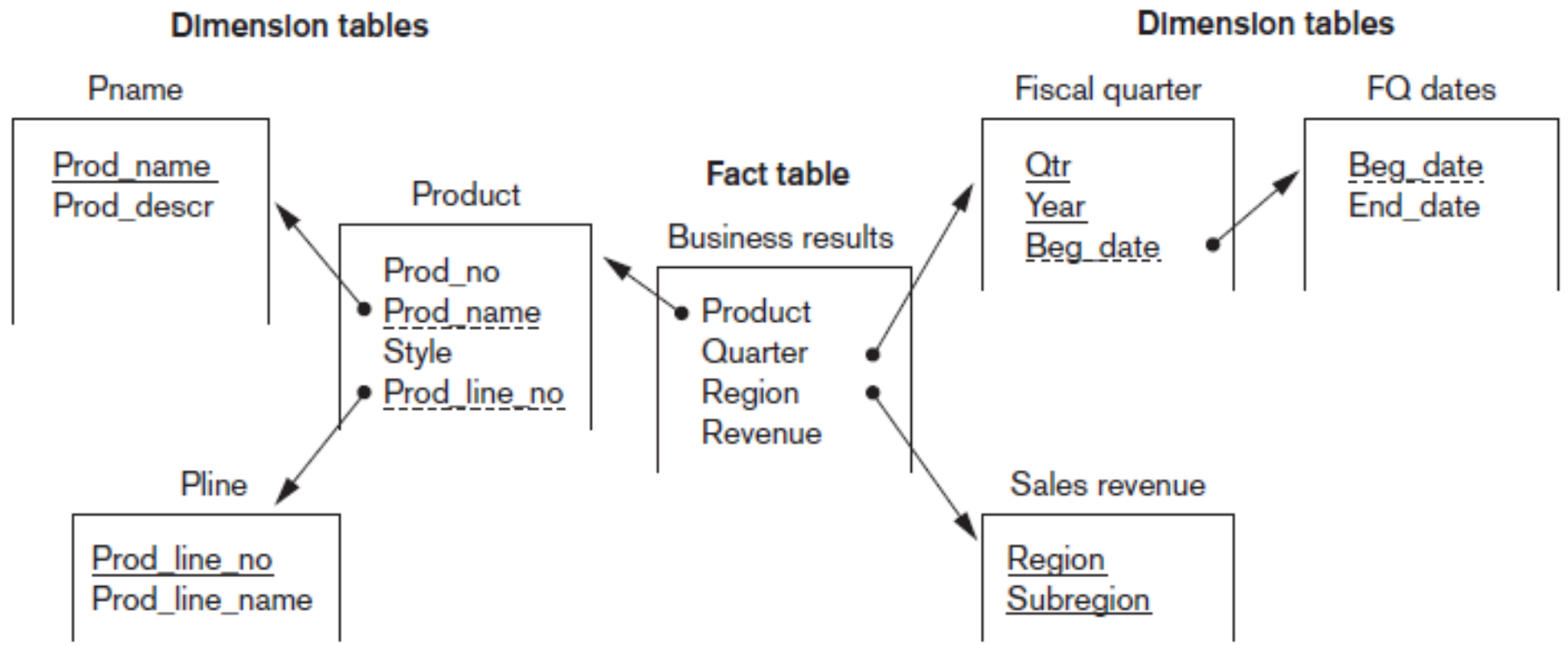


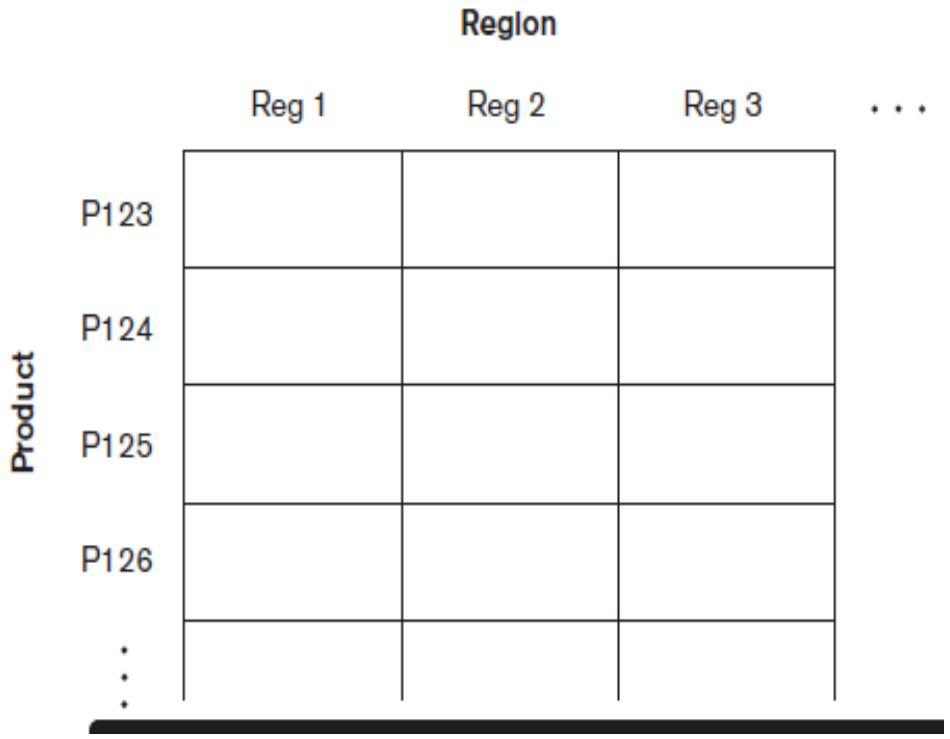
Figure 29.7
A star schema with fact and dimensional tables.

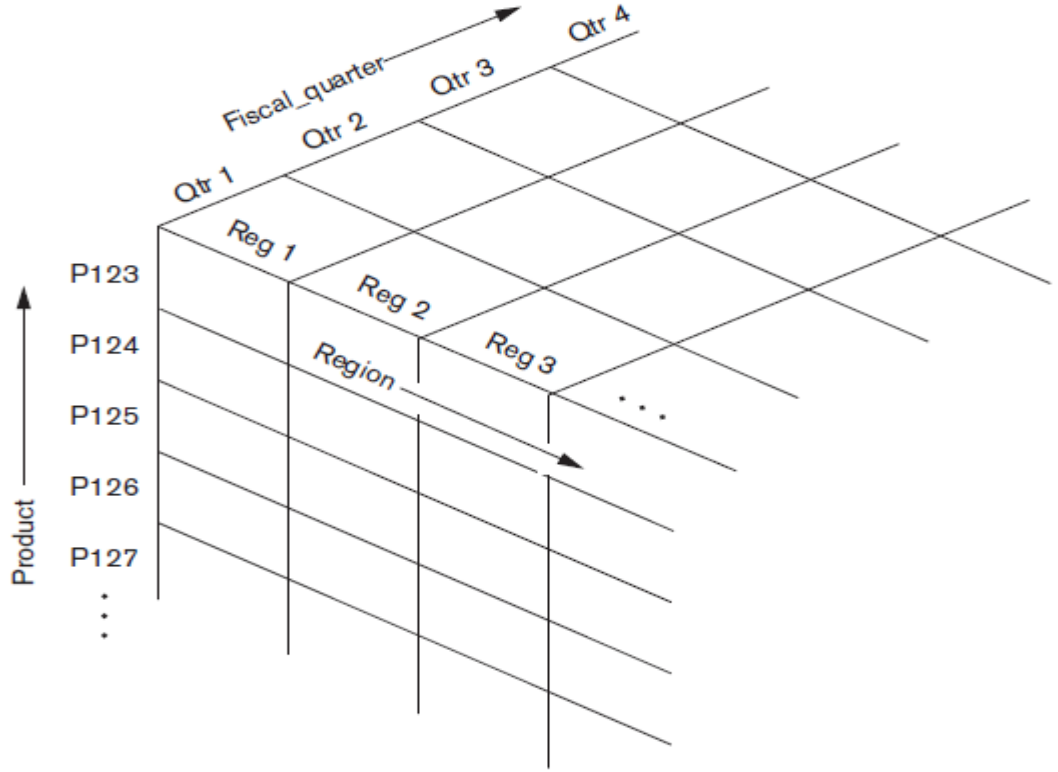
Modèles de données flocon de neige - Elmasri

Figure 29.8
A snowflake schema.



Modèles de données 2D et 3D





ata

Dimensionnalité – Luc Lavoie (1)

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.

Dimensionnalité – Luc Lavoie (2)

TABLES DIMENSIONNELLES

- ***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 ?

Dimensionnalité – Luc Lavoie (3)

- 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

TABLE TEMPORELLE

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).

Dimensionnalité – Luc Lavoie (4)

- 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

Modélisation temporelle de dimensions

Type 0 – Dimension fixe

Aucune modification autorisée, la dimension ne change jamais

Type 1 – Pas d'historique

Mettre à jour l'enregistrement directement, il n'y a pas d'enregistrement des valeurs historiques, uniquement l'état actuel

Type 2 – Gestion des versions de ligne

Suivre les modifications sous forme d'enregistrements de version avec l'indicateur actuel et les dates activation

Type 3 – Colonne Valeur précédente

Suivre les modifications apportées à un attribut spécifique, ajouter une colonne pour afficher la valeur précédente, qui est mise à jour au fur et à mesure que d'autres modifications se produisent

Type 4 – Tableau d'historique

Afficher la valeur actuelle dans le tableau des dimensions, mais suivre toutes les modifications dans un tableau séparé

Type 6 – SCD hybride

Utiliser les techniques des types SCD 1, 2 et 3 pour suivre le changement

Type de dimensions

- Dimensions conformes
- Dimensions rallongées (*outrigge*r)
- Dimensions réduites(*shrunken*)
- Dimensions rôles
- Dimension vers Dimension
- Dimension déchet (*junk*)
- Dimension dégénérée (*degenerate*)
- Dimension échangeable (*swappable*)
- Dimension étape (*step*)

Type de dimensions

1) Une **dimension conforme** est une dimension qui a la même signification pour chaque fait avec lequel elle se rapporte

2) Une **dimension réduite** est un **sous-ensemble d'une entité de dimension, plus détaillée et granulaire**. Dans ce cas, **les attributs communs aux dimensions de sous-ensemble détaillé et réduit ont les mêmes noms d'attribut, définitions et valeurs de domaine**.

3) Une **dimension rallongée est une table de dimension ou une entité jointe à d'autres tables de dimension** dans un schéma en étoile, lorsqu'une **table de dimensions est en flocons**. Les **dimensions rallongées** sont des tables ou des entités partagées par plusieurs dimensions.

Type de dimensions

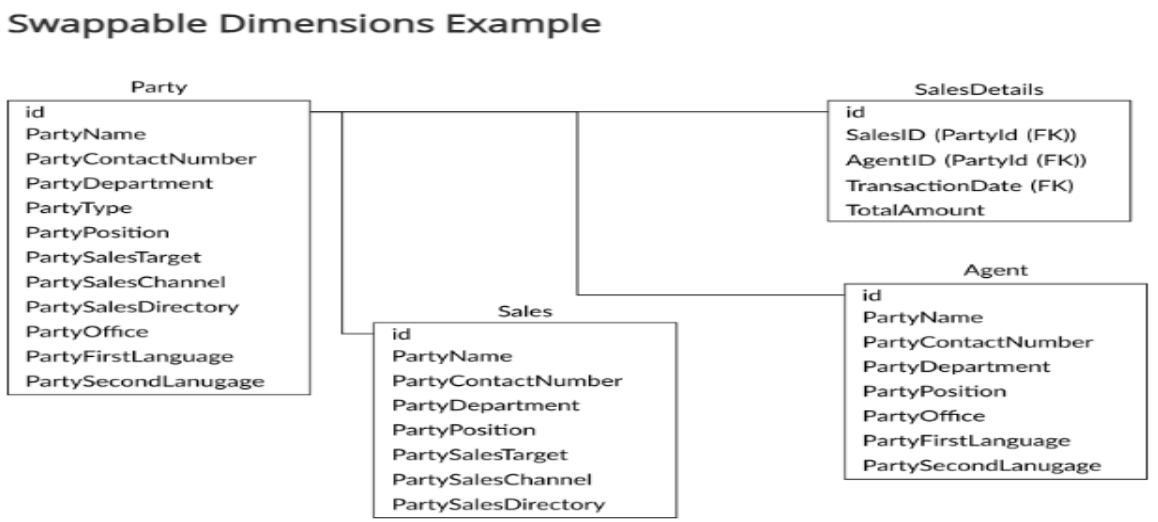
4) Une table avec **plusieurs relations valides entre elle-même et une autre table est appelée dimension rôle**. Cela se voit le plus souvent dans des dimensions telles que le **temps et le client**.

5) Dimension déchets - c'est ce que Kimball appellerait un indicateurs de faible cardinalité. En créant une dimension abstraite, nous supprimons les drapeaux de la table des faits tout en les plaçant dans un cadre dimensionnel utile. L'analogie du tiroir de cuisine est bonne.

Type de dimensions

6) Une **dimension dégénérée** est **une clé de dimension dans la table de faits qui n'a pas sa propre table de dimension**, car tous les attributs intéressants ont été placés dans des dimensions analytiques (cubes).

7) Dimension qui possède plusieurs versions alternatives d'elle-même qui peuvent être échangées au moment de la requête.



Type de dimensions

8) **Étape :** Les processus séquentiels, tels que les événements de page **Web**, ont normalement une ligne distincte dans une table de faits de transaction pour chaque étape d'un processus. Pour indiquer où se situe l'étape individuelle dans la session globale, une dimension indique quel **numéro d'étape** est représenté par l'étape actuelle et combien d'étapes supplémentaires ont été nécessaires pour terminer la session.

Conclusions

Dimensions : tables **plus petits et dénormalisés** contenant des colonnes descriptives d'entreprise que les utilisateurs utilisent pour interroger de manière sélective pour rechercher des correspondances de valeurs de recherche.

Faits : grandes tables **avec des clés primaires formées par la concaténation de colonnes de clés étrangères de tables de dimension associées**, et **possédant également des colonnes non-clés numériquement additives** utilisées pour les calculs lors des requêtes des utilisateurs. Elles doivent être interrogées de manière sélective, car ils ont généralement des centaines de millions à des milliards de lignes.

Démarche a suivre pour un ED – revue

1. Analyse des exigences
2. Identifier les sources de données nécessaires
3. Identifier les faits
4. Définir les dimensions
5. Définir les attributs
6. Redéfinissez les dimensions et les attributs si nécessaire
7. Organiser la hiérarchie des attributs
8. Définir les relations entres les faits et les dimensions
9. Attribuer des identifiants uniques

Techniques d’exploration

Roll-up (également drill-up). Les données sont résumées avec une généralisation croissante (par exemple, hebdomadaire à trimestrielle à annuelle).

Exploration vers le bas (drill-down, drill-across). Des niveaux de détail croissants sont révélés (le complément du roll-up).

Pivot. Une opération de tableau croisé (également appelé rotation).

Sélection. *Les opérations de projection sont effectuées sur les cotes.*

Tri. *Les données sont triées par valeur ordinale.*

Sélection. *Les données sont filtrées par valeur ou plage.*

Attributs dérivés (calculés). *Les attributs sont calculés par des opérations sur des valeurs stockées et dérivées.*

Techniques d’exploration vers haut – Roll-Up et Cube

ROLLUP récapitule par rapport à une hiérarchie de colonnes utilisée dans la clause **GROUP BY** . **CUBE** regroupe toutes les combinaisons de valeurs.

```
SELECT teams.regional_office,
       pipeline.sales_agent,
       SUM(pipeline.close_value)
FROM sales_pipeline AS pipeline, sales_teams AS teams
 WHERE sales_pipeline.sales_agent = sales_teams.sales_agent
 AND sales_pipeline.deal_stage = "Won"
 GROUP BY ROLLUP (teams.regional_office, pipeline.sales_agent)
 ORDER BY teams.regional_office, pipeline.sales_agent
```

regional_office	sales_agent	sum
		10,005,534
Central		3,346,293
Central	Anna Snelling	275,056
Central	Cecily Lampkin	229,800
Central	Darcel Schlecht	1,153,214
Central	Gladys Colclough	345,674
Central	Jonathan Berthelot	284,886
Central	Lajuana Vencill	194,632
Central	Marty Freudenburg	291,195
Central	Moses Frase	207,182
Central	Niesha Huffines	176,961
Central	Versie Hillebrand	187,693
East		3,090,594
East	Boris Faz	261,631
East	Cassey Cress	450,489
East	Corliss Cosme	421,036
East	Daniell Hammack	364,229
East	Donn Cantrell	445,860
East	Garret Kinder	197,773

Techniques d’exploration vers haut – Roll-Up et Cube

ROLLUP récapitule par rapport à une hiérarchie de colonnes utilisée dans la clause **GROUP BY** .
CUBE regroupe toutes les combinaisons de valeurs.

```
SELECT product,
       sales_agent,
       SUM(close_value)
  FROM sales_pipeline
 WHERE sales_pipeline.deal_stage =
"Won"
 GROUP BY CUBE(product, sales_agent)
 ORDER BY product, sales_agent
```

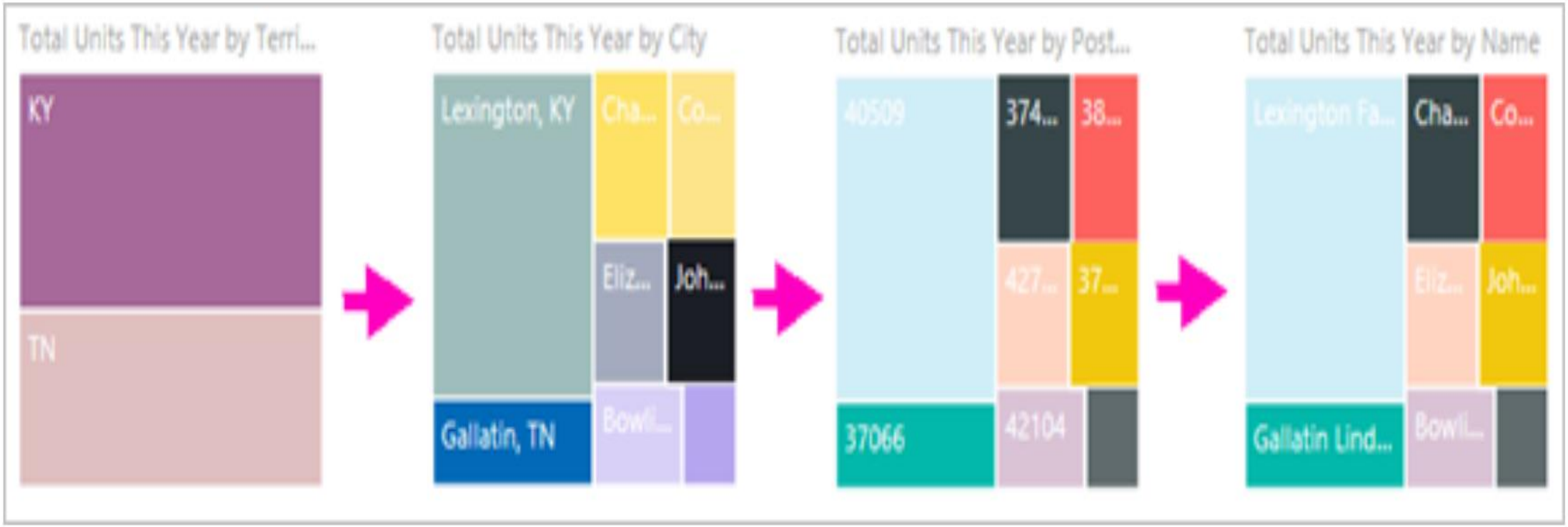
product	sales_agent	sum
		10005534
	Anna Snelling	275056
	Boris Faz	261631
GTK 500		400612
GTK 500	Elease Gluck	184632
GTK 500	Markita Hansen	83528
GTK 500	Rosalina Dieter	132452
GTX Basic		499263
GTX Basic	Anna Snelling	17437
GTX Basic	Boris Faz	9041

Ex.: techniques d'exploration vers haut – Roll-Up

```
CREATE TABLE sales (brand VARCHAR
NOT NULL,    segment VARCHAR NOT
NULL, quantity INT NOT NULL,
PRIMARY KEY (brand, segment));

INSERT INTO sales (brand,
segment, quantity) VALUES
('ABC', 'Premium', 100),
('ABC', 'Basic', 200),
('XYZ', 'Premium', 100),
('XYZ', 'Basic', 300);

SELECT     brand,    segment,
SUM (quantity) FROM     sales
GROUP BY

ROLLUP (brand, segment) ORDER BY
brand,     segment;
```

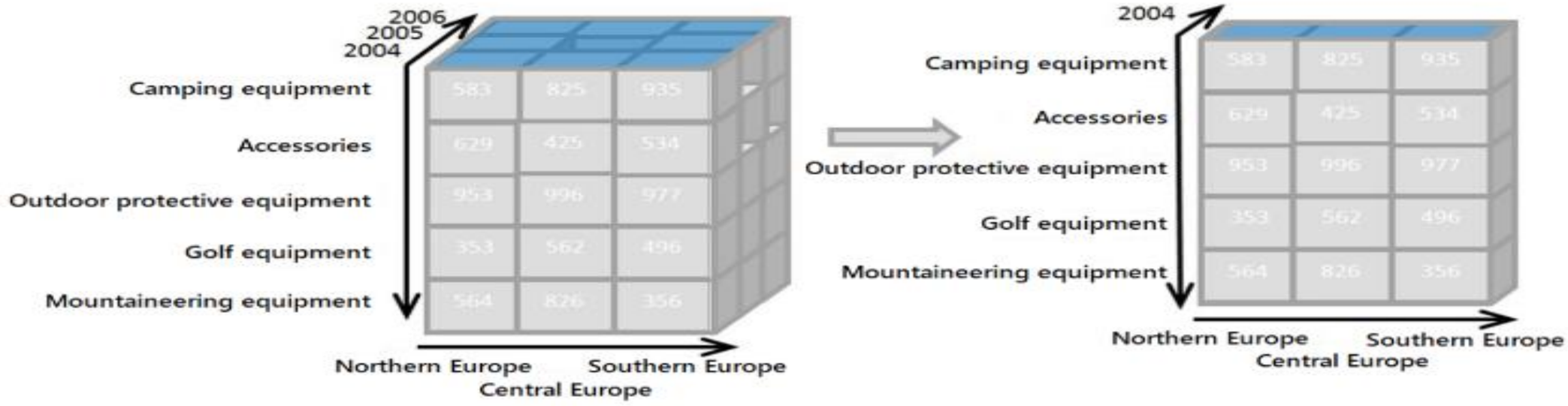
brand	segment	sum
ABC	Basic	200
ABC	Premium	100
ABC	(Null)	300
XYZ	Basic	300
XYZ	Premium	100
XYZ	(Null)	400
(Null)	(Null)	700

Techniques d’exploration vers bas – Drill Down



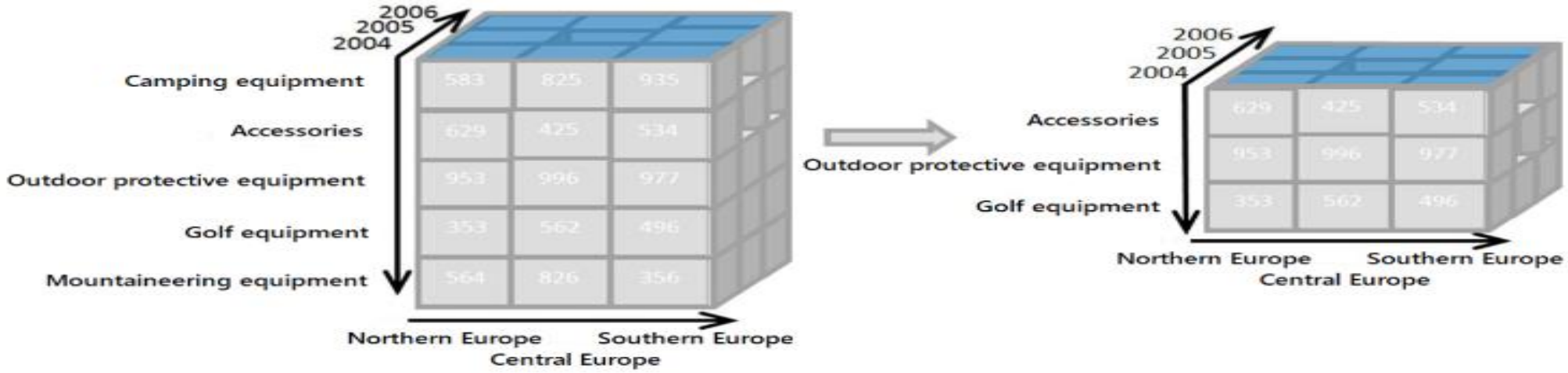
Techniques d’exploration – Slice (tranche)

Le découpage nous permet d'examiner les données en fixant (ou en sélectionnant) une seule valeur d'une dimension. Dans l'exemple ci-dessous, nous avons découpé le cube de données pour examiner les données de 2004 uniquement.



Techniques d’exploration – Dice (des)

L'opération est similaire à la tranche, mais **au lieu de prendre une seule valeur pour une dimension, nous prenons plus d'une valeur dans les dimensions que nous choisissons**. Dans l'exemple ci-dessous, le cube de données est découpé le long de la dimension du produit (sélectionné uniquement « Accessoires », « Équipement de protection extérieure » et « Équipement de golf ») tout en conservant la dimension de la date et la dimension de la région intactes.



Techniques d’exploration – Percer a travers - Drill Across

Les **dimensions conformes** garantissent la compatibilité des informations dans diverses étoiles et data marts; percer à travers est le processus de les rassembler.

Phase 1 : Récupérer les faits de chaque table de faits séparément, en les agrégeant à un niveau de détail commun

Phase 2 : fusionner ces ensembles de résultats intermédiaires en fonction de leurs dimensions communes

Techniques d'exploration – drill across (2)

```
SELECT
  COALESCE (shp.product, rtn.product) as Product,
  quantity_returned / quantity_shipped as ReturnRate
FROM
   ( SELECT product, sum(quantity_shipped)as
quantity_shipped
     FROM shipment_facts, product
     WHERE .....
   ) shp
FULL OUTER JOIN
   ( SELECT product, sum(quantity_returned) as
quantity_returned
     FROM return_facts, product
     WHERE....
   ) rtn
 ON
     shp.product = rtn.product
```

Techniques d'exploration – Pivot

PIVOT – rotation de valeurs uniques d'une colonne dans des colonnes multiples.

```
USE AdventureWorks2014 ;
GO
SELECT DaysToManufacture, AVG(StandardCost) AS AverageCost
FROM Production.Product
GROUP BY DaysToManufacture;
```

```
DaysToManufacture AverageCost
----------------- -----------
0                 5.0885
1                 223.88
2                 359.1082
4                 949.4105
```

```
-- Pivot table with one row and five columns
SELECT 'AverageCost' AS Cost_Sorted_By_Production_Days,
  [0], [1], [2], [3], [4]
FROM
(
  SELECT DaysToManufacture, StandardCost
  FROM Production.Product
) AS SourceTable
PIVOT
(
  AVG(StandardCost)
  FOR DaysToManufacture IN ([0], [1], [2], [3], [4])
) AS PivotTable;
```

```
Cost_Sorted_By_Production_Days 0          1          2          3    
------------------------------ ---------- ---------- ---------- ----
AverageCost                    5.0885     223.88     359.1082   NULL
```

Techniques d'exploration – Unipivot

UNIPIVOT – rotation de colonnes dans des valeurs

```
CREATE TABLE pvt (VendorID INT, Emp1 INT, Emp2 INT,
    Emp3 INT, Emp4 INT, Emp5 INT);
GO
INSERT INTO pvt VALUES (1,4,3,5,4,4);
INSERT INTO pvt VALUES (2,4,1,5,5,5);
INSERT INTO pvt VALUES (3,4,3,5,4,4);
INSERT INTO pvt VALUES (4,4,2,5,5,4);
INSERT INTO pvt VALUES (5,5,1,5,5,5);
```

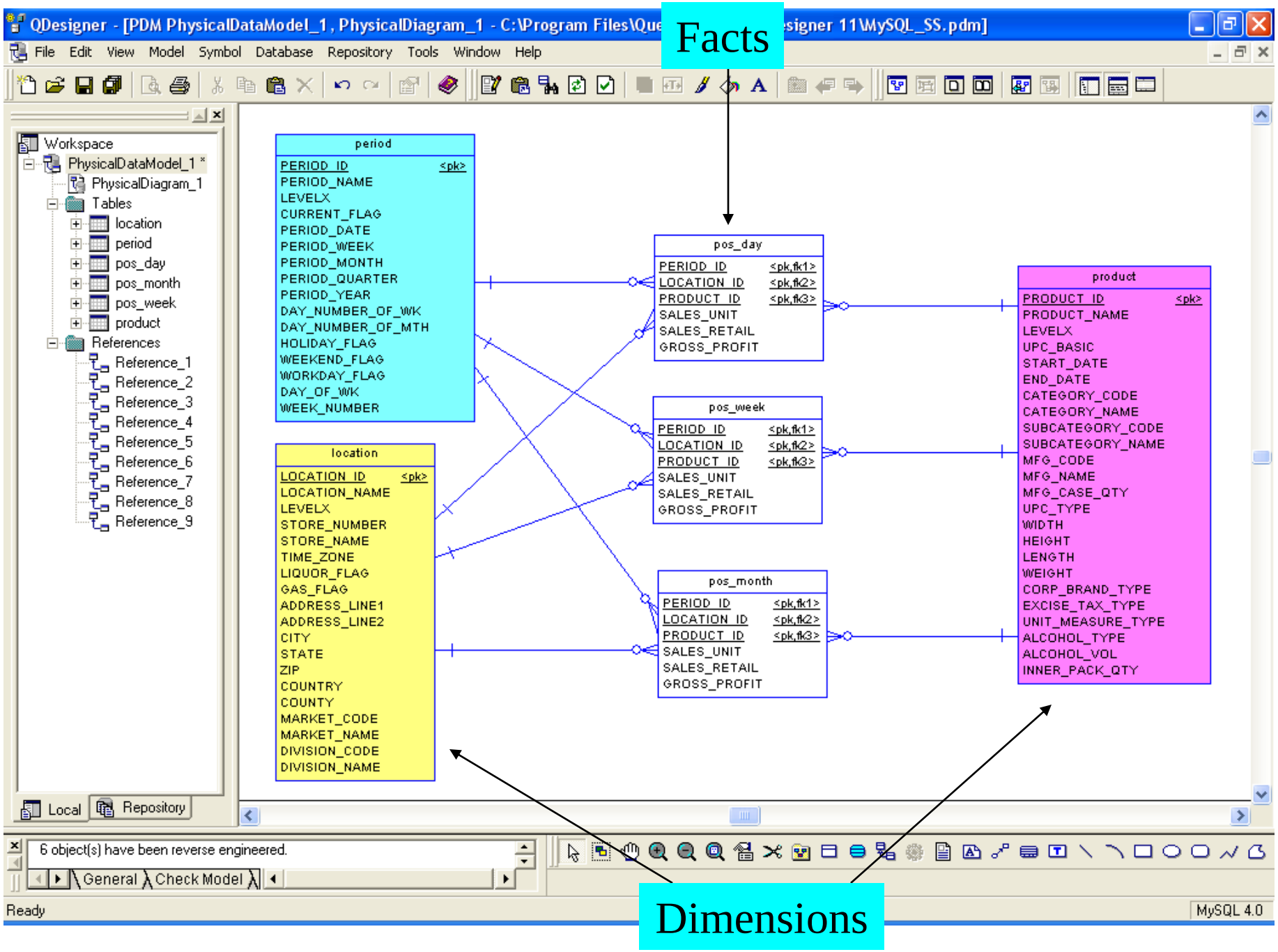
```
SELECT VendorID, Employee, Orders
FROM
   (SELECT VendorID, Emp1, Emp2, Emp3, Emp4, Emp5
   FROM pvt) p
UNPIVOT
   (Orders FOR Employee IN
      (Emp1, Emp2, Emp3, Emp4, Emp5)
)AS unpvt;
```

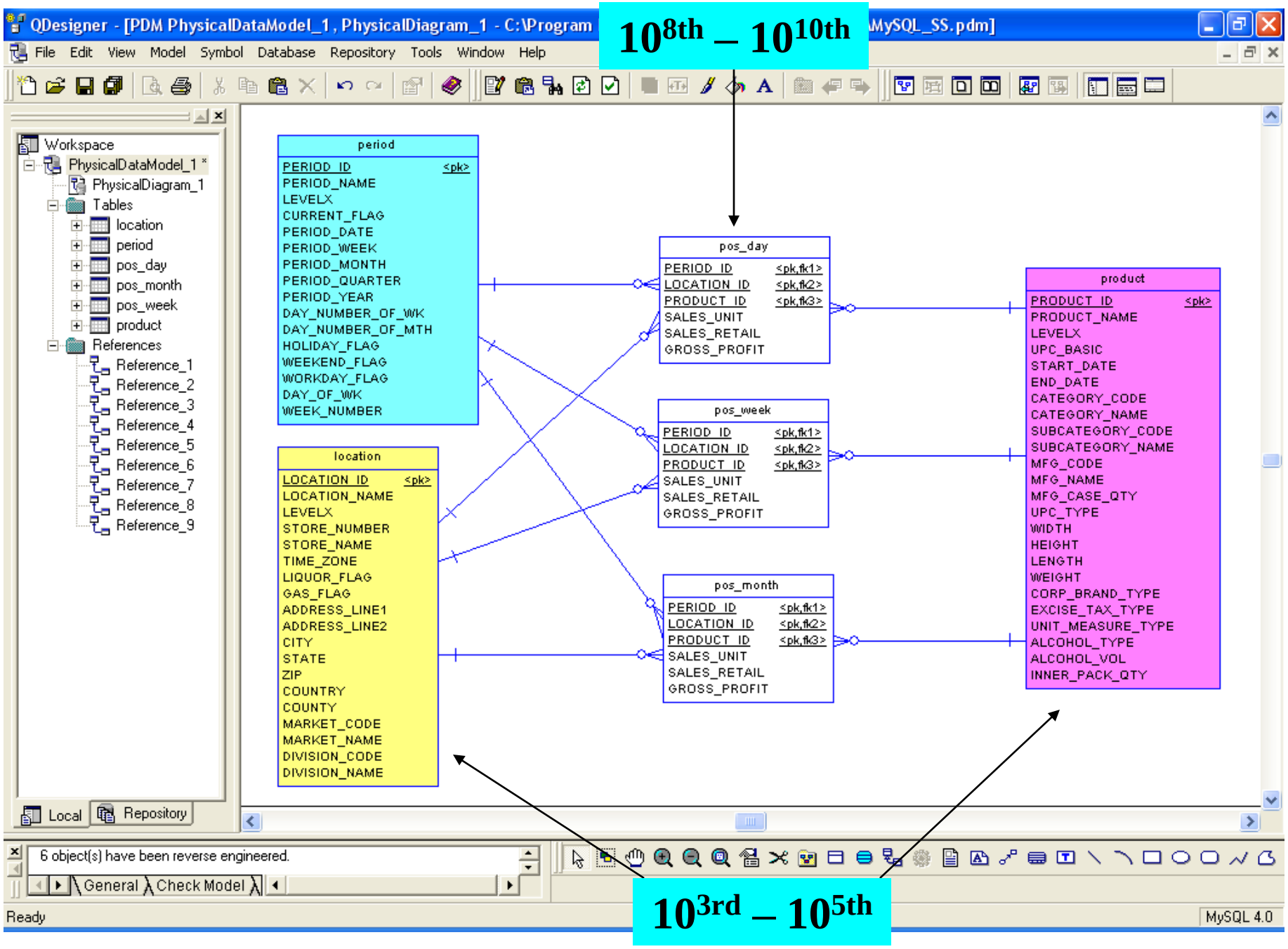
VendorID	Employee	Orders
1	Emp1	4
1	Emp2	3
1	Emp3	5
1	Emp4	4
1	Emp5	4
2	Emp1	4
2	Emp2	1
2	Emp3	5
2	Emp4	5
2	Emp5	5

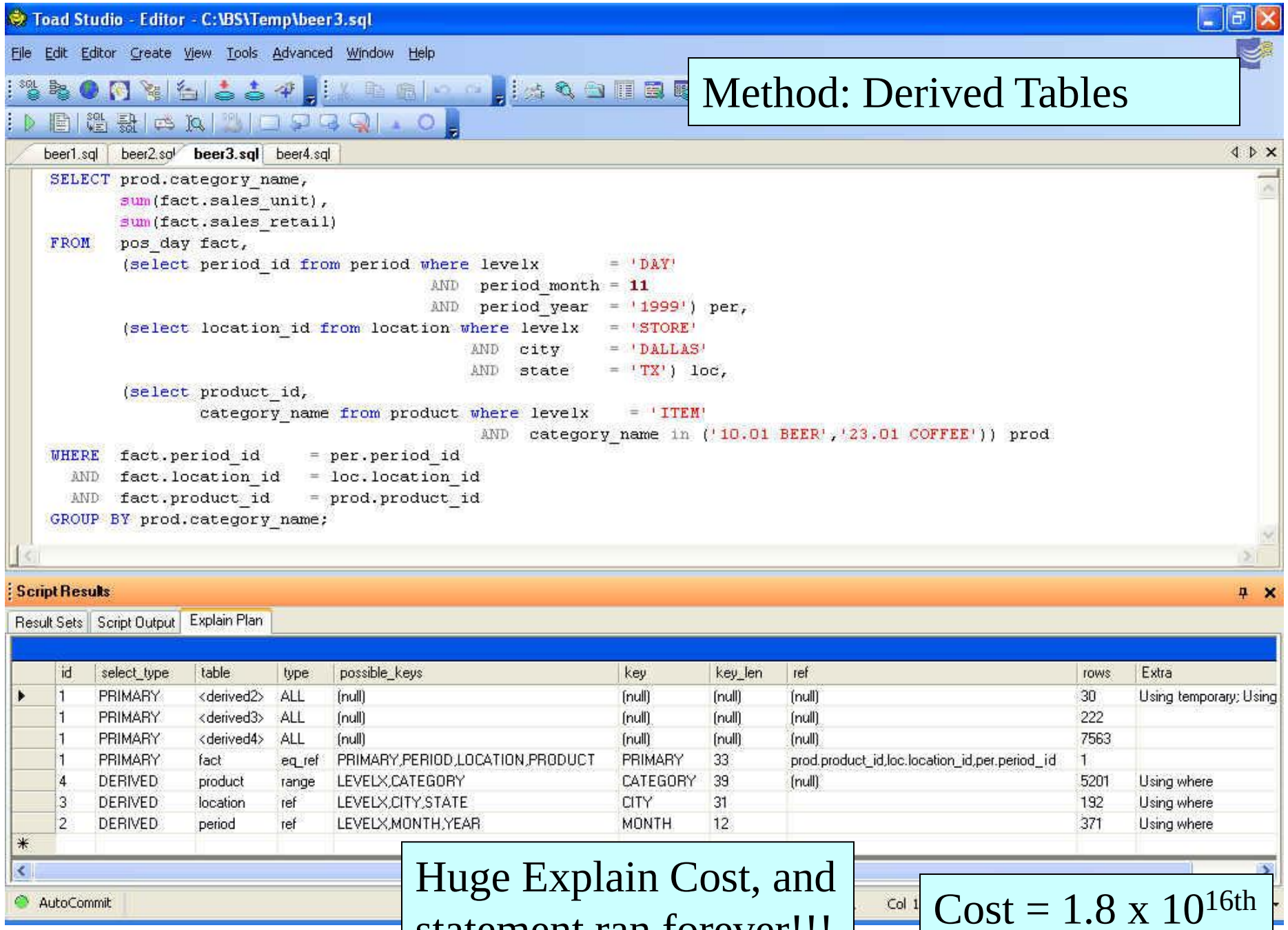
Bert Scalzo

Expert sur la performance d’entrepôts de données

+ les conseils de Adamson slides 38-46 de professeur Lavoie


Facts
QDesigner - [PDM PhysicalDataModel_1, PhysicalDiagram_1 - C:\Program Files\Que
esigner 11\MySQL_SS.pdm]
File Edit View Model Symbol Database Repository Tools Window Help
Workspace
PhysicalDataModel_1 *
PhysicalDiagram_1
Tables
location
period
pos_day
pos_month
pos_week
product
References
Reference_1
Reference_2
Reference_3
Reference_4
Reference_5
Reference_6
Reference_7
Reference_8
Reference_9
Local
Repository
period
PERIOD_ID <pk>
PERIOD_NAME
LEVELX
CURRENT_FLAG
PERIOD_DATE
PERIOD_WEEK
PERIOD_MONTH
PERIOD_QUARTER
PERIOD_YEAR
DAY_NUMBER_OF_WK
DAY_NUMBER_OF_MTH
HOLIDAY_FLAG
WEEKEND_FLAG
WORKDAY_FLAG
DAY_OF_WK
WEEK_NUMBER
location
LOCATION_ID <pk>
LOCATION_NAME
LEVELX
STORE_NUMBER
STORE_NAME
TIME_ZONE
LIQUOR_FLAG
GAS_FLAG
ADDRESS_LINE1
ADDRESS_LINE2
CITY
STATE
ZIP
COUNTRY
COUNTY
MARKET_CODE
MARKET_NAME
DIVISION_CODE
DIVISION_NAME
pos_day
PERIOD_ID <pk,fk1>
LOCATION_ID <pk,fk2>
PRODUCT_ID <pk,fk3>
SALES_UNIT
SALES_RETAIL
GROSS_PROFIT
pos_week
PERIOD_ID <pk,fk1>
LOCATION_ID <pk,fk2>
PRODUCT_ID <pk,fk3>
SALES_UNIT
SALES_RETAIL
GROSS_PROFIT
pos_month
PERIOD_ID <pk,fk1>
LOCATION_ID <pk,fk2>
PRODUCT_ID <pk,fk3>
SALES_UNIT
SALES_RETAIL
GROSS_PROFIT
product
PRODUCT_ID <pk>
PRODUCT_NAME
LEVELX
UPC_BASIC
START_DATE
END_DATE
CATEGORY_CODE
CATEGORY_NAME
SUBCATEGORY_CODE
SUBCATEGORY_NAME
MFG_CODE
MFG_NAME
MFG_CASE_QTY
UPC_TYPE
WIDTH
HEIGHT
LENGTH
WEIGHT
CORP_BRAND_TYPE
EXCISE_TAX_TYPE
UNIT_MEASURE_TYPE
ALCOHOL_TYPE
ALCOHOL_VOL
INNER_PACK_QTY
6 object(s) have been reverse engineered.
General
Check Model
Ready
MySQL 4.0
Dimensions

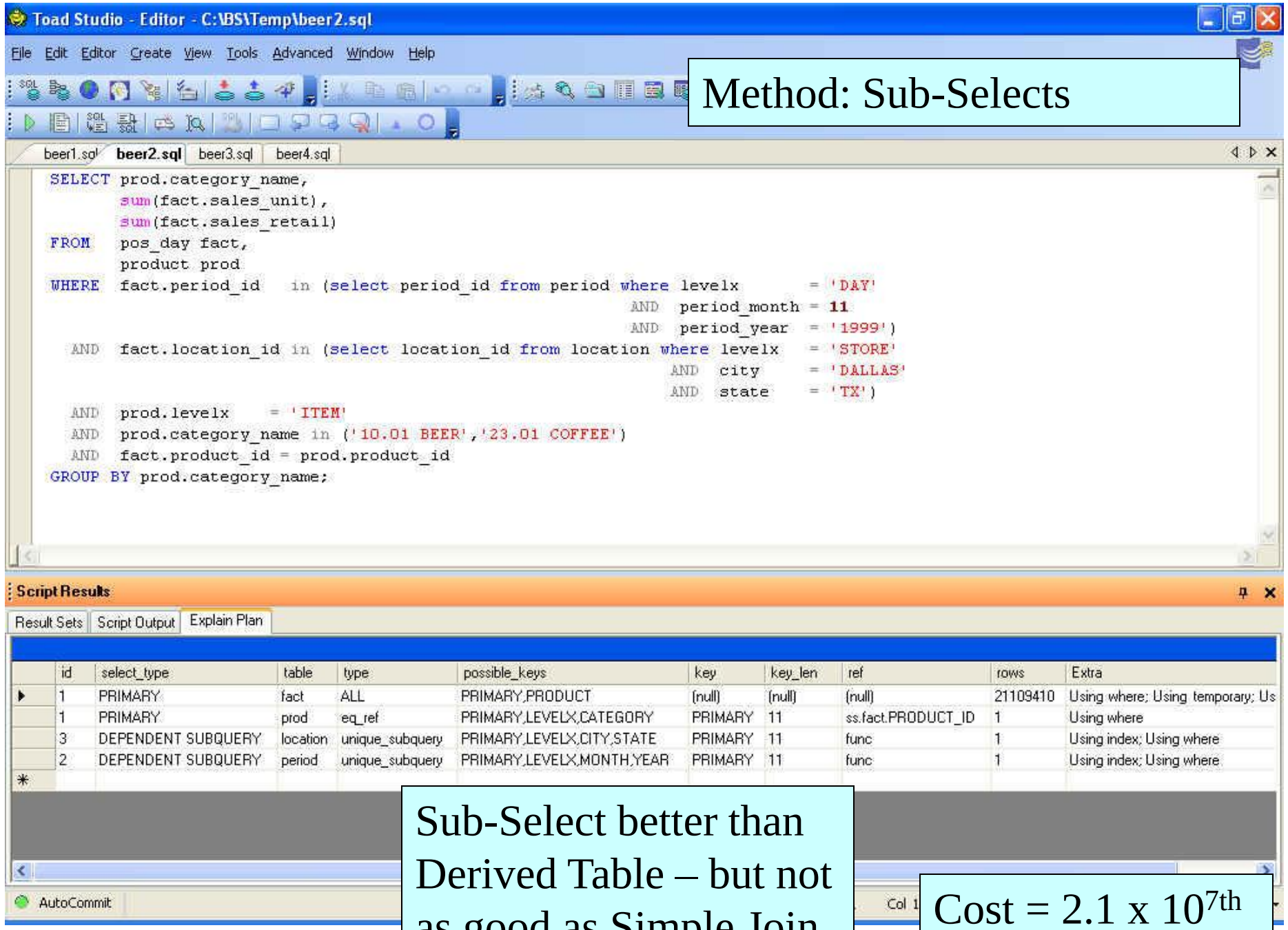

QDesigner - [PDM PhysicalDataModel_1, PhysicalDiagram_1 - C:\Program
MySQL_SS.pdm]
File Edit View Model Symbol Database Repository Tools Window Help
Workspace
PhysicalDataModel_1 *
PhysicalDiagram_1
Tables
location
period
pos_day
pos_month
pos_week
product
References
Reference_1
Reference_2
Reference_3
Reference_4
Reference_5
Reference_6
Reference_7
Reference_8
Reference_9
Local
Repository
period
PERIOD_ID <pk>
PERIOD_NAME
LEVELX
CURRENT_FLAG
PERIOD_DATE
PERIOD_WEEK
PERIOD_MONTH
PERIOD_QUARTER
PERIOD_YEAR
DAY_NUMBER_OF_WK
DAY_NUMBER_OF_MTH
HOLIDAY_FLAG
WEEKEND_FLAG
WORKDAY_FLAG
DAY_OF_WK
WEEK_NUMBER
location
LOCATION_ID <pk>
LOCATION_NAME
LEVELX
STORE_NUMBER
STORE_NAME
TIME_ZONE
LIQUOR_FLAG
GAS_FLAG
ADDRESS_LINE1
ADDRESS_LINE2
CITY
STATE
ZIP
COUNTRY
COUNTY
MARKET_CODE
MARKET_NAME
DIVISION_CODE
DIVISION_NAME
pos_day
PERIOD_ID <pk,fk1>
LOCATION_ID <pk,fk2>
PRODUCT_ID <pk,fk3>
SALES_UNIT
SALES_RETAIL
GROSS_PROFIT
pos_week
PERIOD_ID <pk,fk1>
LOCATION_ID <pk,fk2>
PRODUCT_ID <pk,fk3>
SALES_UNIT
SALES_RETAIL
GROSS_PROFIT
pos_month
PERIOD_ID <pk,fk1>
LOCATION_ID <pk,fk2>
PRODUCT_ID <pk,fk3>
SALES_UNIT
SALES_RETAIL
GROSS_PROFIT
product
PRODUCT_ID <pk>
PRODUCT_NAME
LEVELX
UPC_BASIC
START_DATE
END_DATE
CATEGORY_CODE
CATEGORY_NAME
SUBCATEGORY_CODE
SUBCATEGORY_NAME
MFG_CODE
MFG_NAME
MFG_CASE_QTY
UPC_TYPE
WIDTH
HEIGHT
LENGTH
WEIGHT
CORP_BRAND_TYPE
EXCISE_TAX_TYPE
UNIT_MEASURE_TYPE
ALCOHOL_TYPE
ALCOHOL_VOL
INNER_PACK_QTY
6 object(s) have been reverse engineered.
General
Check Model
Ready
MySQL 4.0
10^8th – 10^10th
10^3rd – 10^5th

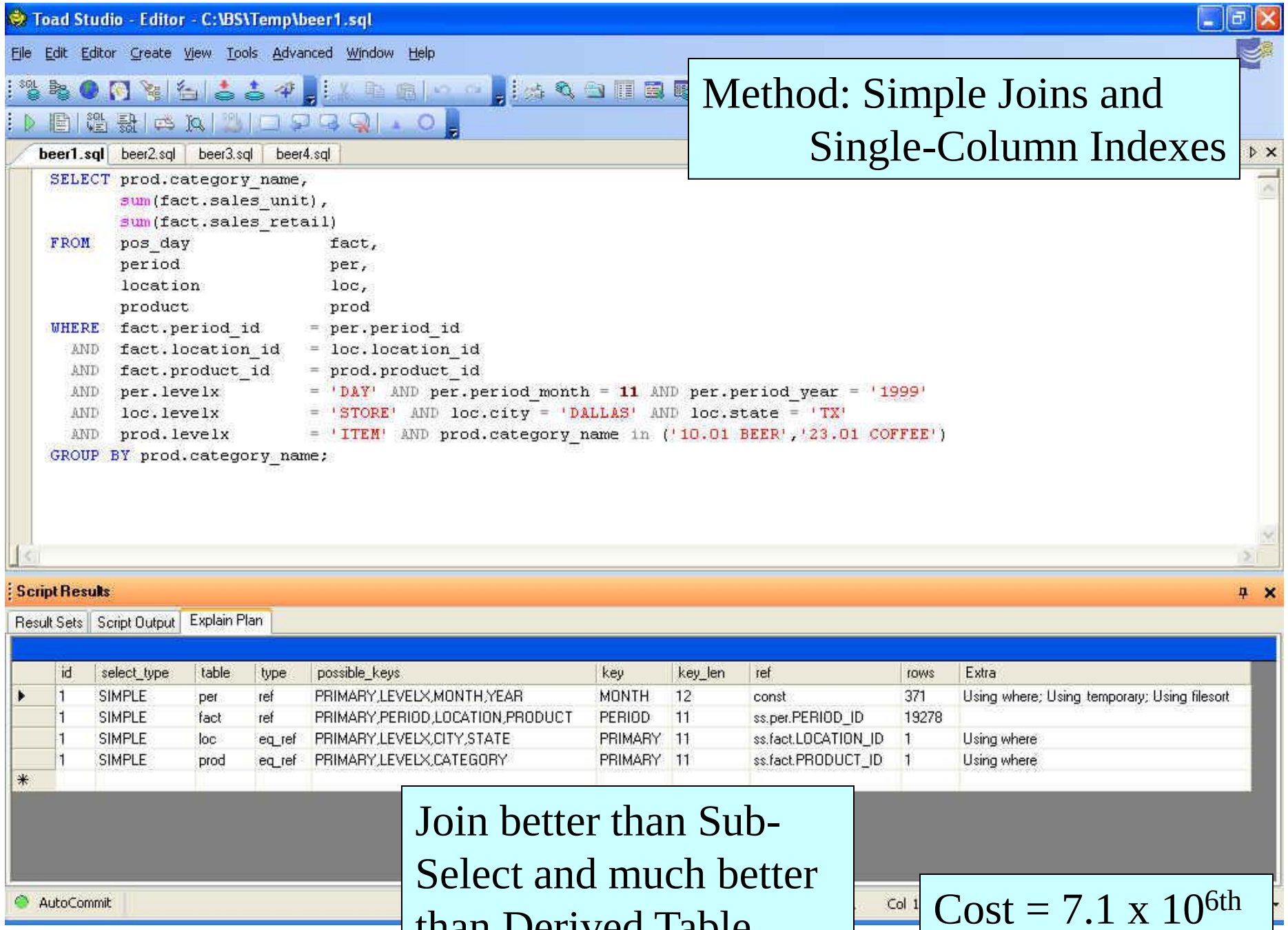


Method: Derived Tables

Huge Explain Cost, and statement ran forever!!!

Cost = 1.8×10^{16th}

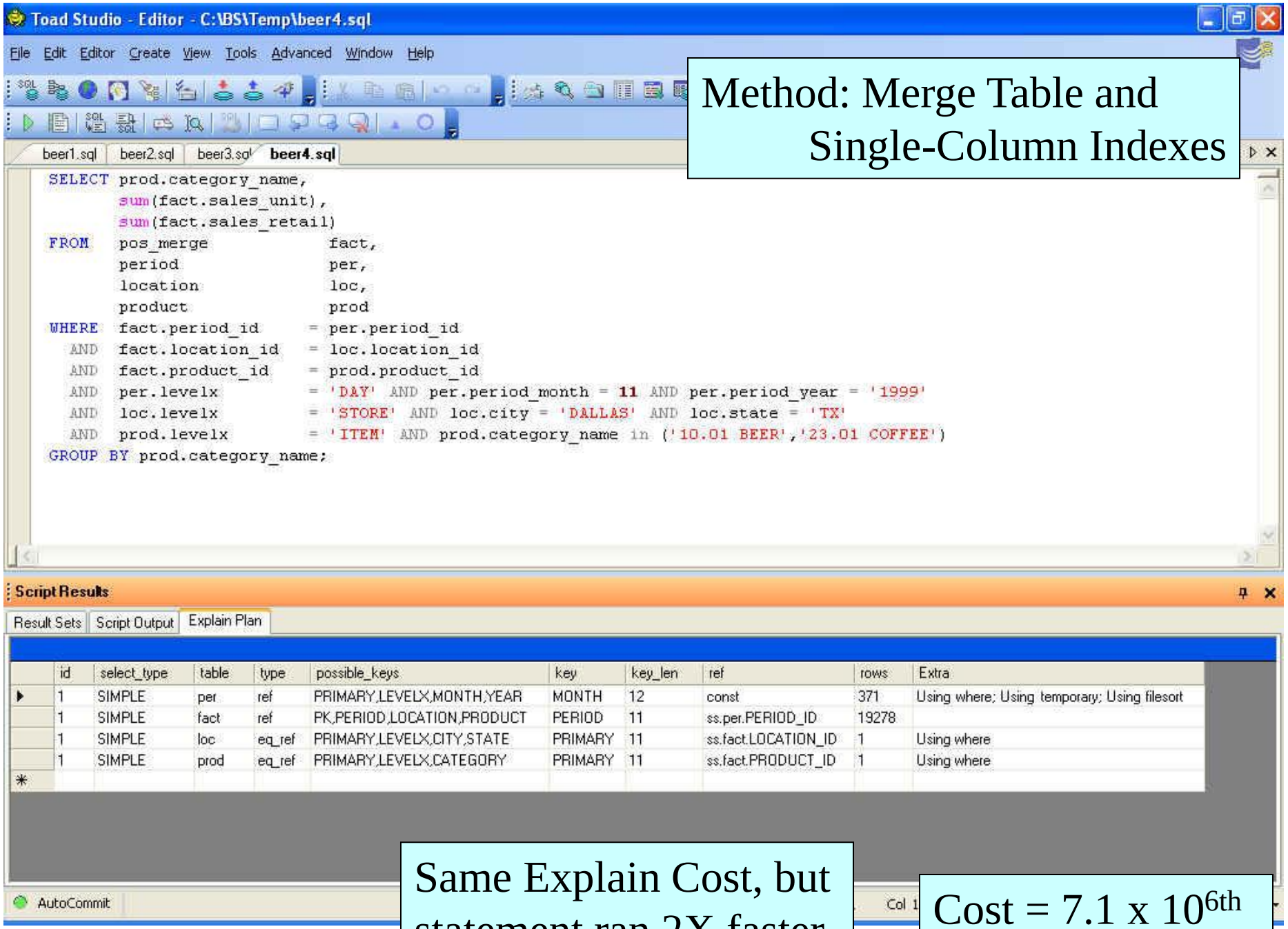

Toad Studio - Editor - C:\BS\Temp\beer2.sql
File Edit Editor Create View Tools Advanced Window Help
Method: Sub-Selects
beer1.sql beer2.sql beer3.sql beer4.sql
SELECT prod.category_name,
sum(fact.sales_unit),
sum(fact.sales_retail)
FROM pos_day fact,
product prod
WHERE fact.period_id in (select period_id from period where levelx = 'DAY'
AND period_month = 11
AND period_year = '1999')
AND fact.location_id in (select location_id from location where levelx = 'STORE'
AND city = 'DALLAS'
AND state = 'TX')
AND prod.levelx = 'ITEM'
AND prod.category_name in ('10.01 BEER','23.01 COFFEE')
AND fact.product_id = prod.product_id
GROUP BY prod.category_name;
Script Results
Result Sets Script Output Explain Plan
id select_type table type possible_keys key key_len ref rows Extra
1 PRIMARY fact ALL PRIMARY,PRODUCT (null) (null) (null) 21109410 Using where; Using temporary; Us
1 PRIMARY prod eq_ref PRIMARY,LEVELX,CATEGORY PRIMARY 11 ss.fact.PRODUCT_ID 1 Using where
3 DEPENDENT SUBQUERY location unique_subquery PRIMARY,LEVELX,CITY,STATE PRIMARY 11 func 1 Using index; Using where
2 DEPENDENT SUBQUERY period unique_subquery PRIMARY,LEVELX,MONTH,YEAR PRIMARY 11 func 1 Using index; Using where
AutoCommit
Col 1
Sub-Select better than Derived Table – but not as good as Simple Join
Cost = 2.1 x 10^7th



Method: Simple Joins and Single-Column Indexes

Join better than Sub-Select and much better than Derived Table

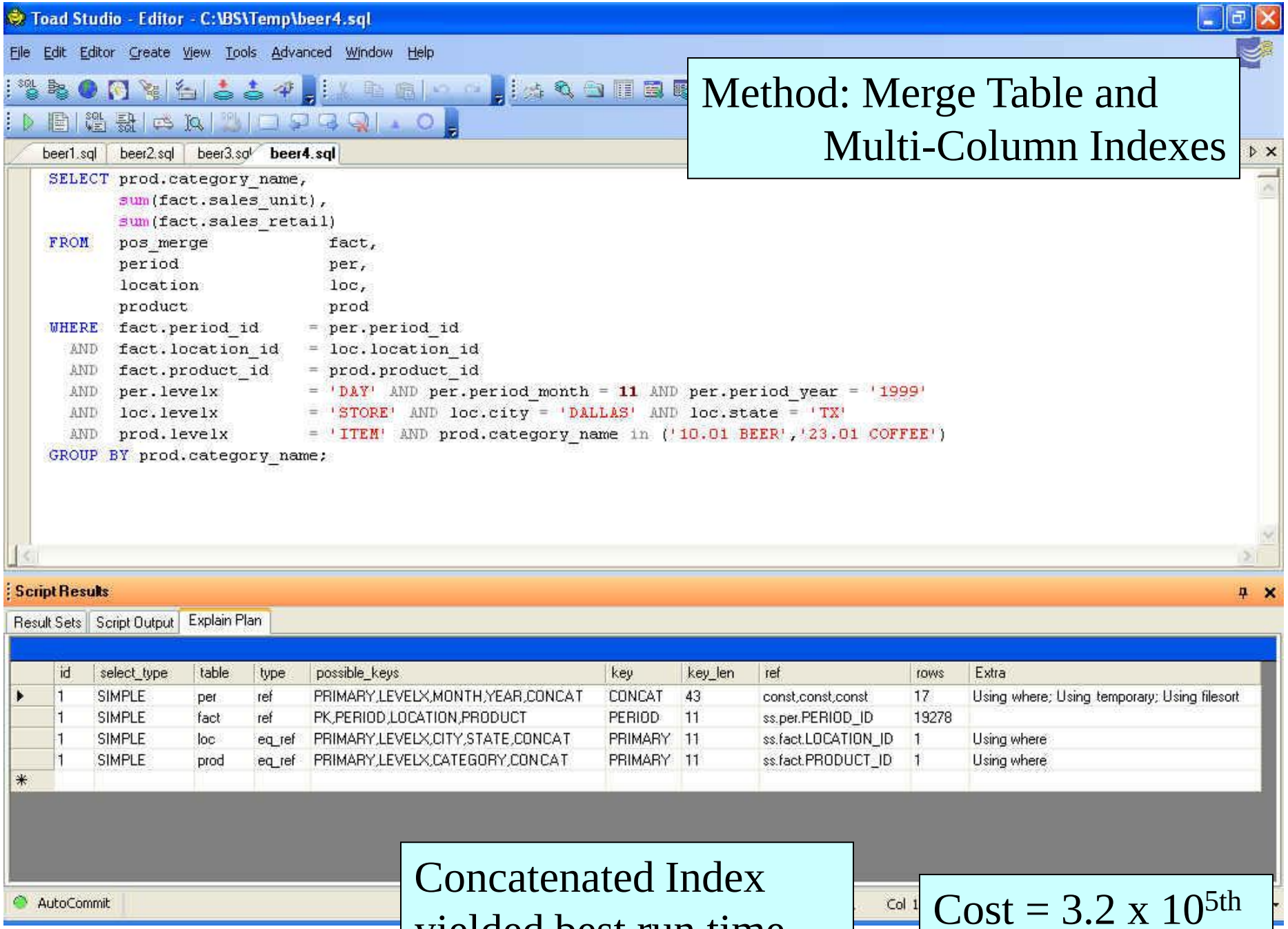
Cost = 7.1×10^{6th}



Method: Merge Table and Single-Column Indexes

Same Explain Cost, but statement ran 2X faster

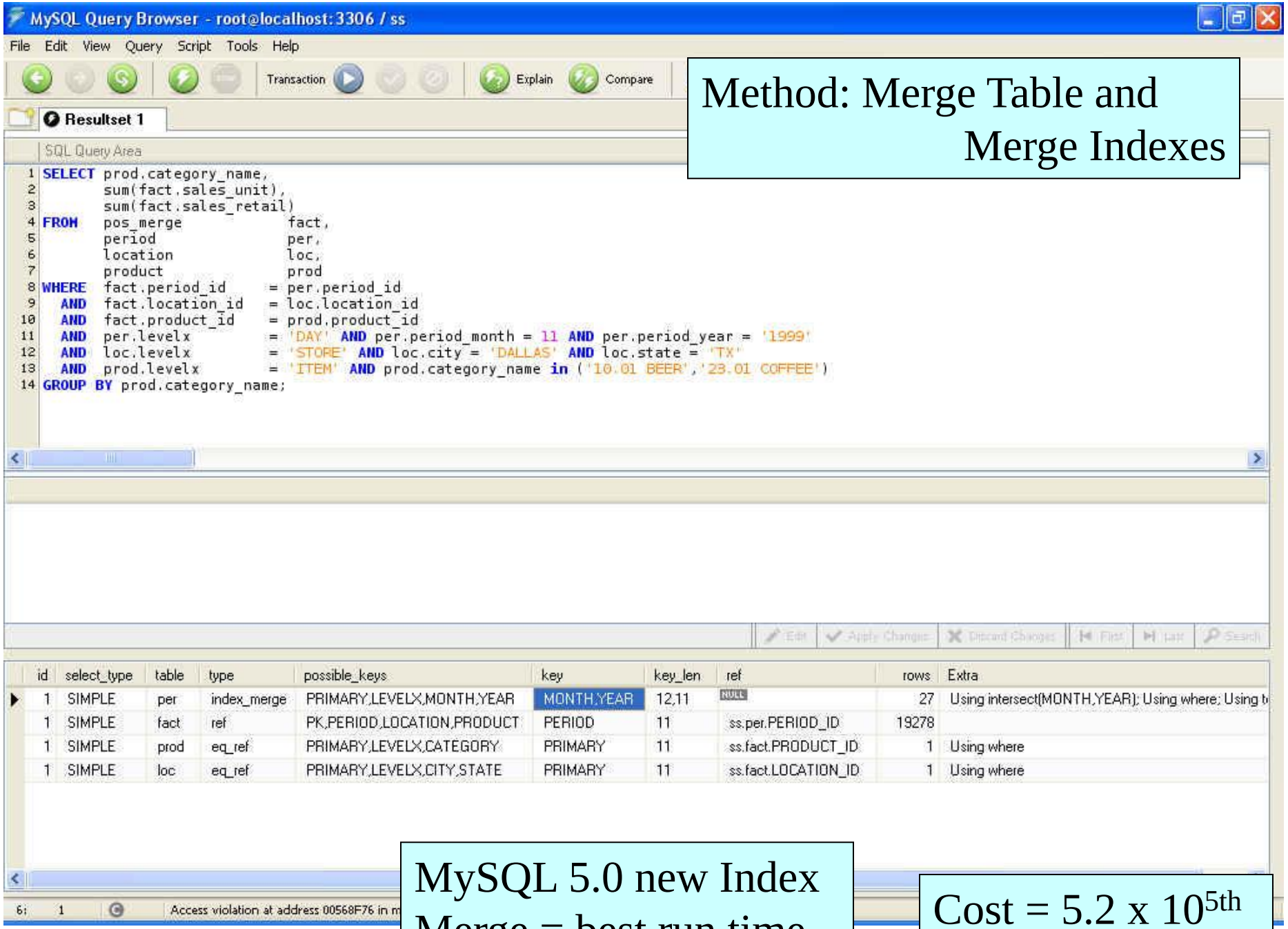
Cost = 7.1×10^{6th}



Method: Merge Table and Multi-Column Indexes

Concatenated Index yielded best run time

Cost = 3.2×10^{5th}



Method: Merge Table and Merge Indexes

MySQL 5.0 new Index Merge = best run time

Cost = 5.2 x 10^{5th}

Question 1

Une dimension est une variable qui catégorise des faits. Comment les statisticiens et les ingénieurs en machine learning appellent-ils les dimensions ?

Choisir :

1. **Mesure numérique**
2. **Variable catégoriques**
3. **Clé référentielle**
4. **Snapshot**

Question 2

Choisissez l'énoncé qui décrit le mieux ce qu'implique le pivotement du cube de données.

1. **Ne pas changer le point de vue**
2. **Changement de point de vue**
3. **Modification du contenu des informations**
4. **Aucune de ces affirmations n'est correcte**