Partie 3
Relations avancées et performance

Chapitre 3-1
Optimiser les relations faits et dimensions

1. Gestion des clés primaires et clés étrangères

Dans un modèle multidimensionnel, les relations entre faits et dimensions jouent un rôle prépondérant dans la précision analytique et la performance des requêtes.

Que ce soit dans un modèle en étoile ou en flocon, la qualité de ces relations repose sur une gestion efficace des clés.

Les dimensions peuvent utiliser des clés entreprises, qu'elles soient simples, composites ou encore des clés artificielles (*surrogate keys*). Ces clés doivent faire référence à une et une seule ligne de la dimension.

La table de fait quant à elle va utiliser cette clé sous forme d'une clé étrangère pour garantir un lien exact entre l'agrégat et les jointures. C'est-à-dire que chaque ligne de la table de fait devra faire référence à une clé primaire valide dans la table de dimension. Cette correspondance garantira l'exactitude des résultats analytiques.

Voici les types de clés primaires dans une dimension.

1.1 Clés entreprises simples

Une clé entreprise simple correspond à l'identifiant naturel donné par le système source pour identifier, de façon unique, une entité.

Exemple

La table source des produits utilise un code produit unique alphanumérique fourni par l'ERP de l'entreprise.

Code_Produit	Nom_Produit	Catégorie	Marque
P12345	Ordinateur Portable	Informatique	Acer
P12346	Souris Bluetooth	Accessoires	Logitech

Dans ce cas-là, le code produit sera utilisé dans la dimension puis reproduit dans la table de fait en tant que clé étrangère.

Cependant, d'un point de vue architectural, s'appuyer structurellement sur une clé naturelle présente de lourdes limites :

– **Problématique d'historisation (SCD)** : la traçabilité historique devient impossible à gérer. Si un produit change de catégorie ou de nom mais conserve le même code métier, il est impossible de créer une nouvelle ligne d'historique dans la dimension.

– **Risque de collision multi-sources** : si l'entrepôt intègre demain un second système source, le même code naturel pourrait désigner deux produits différents.

– **Performances et volumétrie**: l'utilisation de chaînes de caractères dans les jointures ralentit considérablement l'exécution des requêtes et augmente l'espace de stockage. La bonne pratique : l'entrepôt génère sa propre clé technique (entier) pour les jointures. Le code naturel métier est simplement conservé en tant qu'attribut descriptif dans la dimension pour les utilisateurs.

Donc, s'appuyer sur une simple clé naturelle montre rapidement ses limites dans un entrepôt de données. Pour garantir l'unicité de chaque ligne d'une dimension, en particulier face à des contextes multi-sources ou d'historisation, l'architecte doit mettre en place des stratégies d'identification plus avancées. Cela passe généralement par la combinaison de plusieurs attributs, ou pas la génération d'identifiants purement techniques.

1.2 Clés entreprises composites

Une clé composée est formée par plusieurs champs qui, ensemble, garantissent l'unicité de la ligne.

Exemple

Imaginons que le produit puisse être référencé plusieurs fois dans des catégories différentes selon les pays (multinational). En plus de cela, nous devons historiser la dimension en type 2.

On devra alors combiner `Code_Produit`, le `Pays` ainsi que la `date_début` pour garantir l'unicité d'une ligne.

Code_Produit	Pays	Nom_Produit	Catégorie	Date_Début	Date_Fin
P12345	France	Ordinateur Portable	Informatique	2020-01-01	2022-06-30
P12345	France	Ordinateur Portable	Électronique	2022-07-01	9999-12-31
P12345	Canada	Laptop	Électronique	2020-01-01	9999-12-31

La clé primaire devient alors : `Code_Produit` + `Pays` + `Date_Début`.

Clé entreprise composite	Code_ Produit	Pays	Nom_ Produit	Catégorie	Date_ Début	Date_ Fin
P12345_France_ 20200101	P12345	France	Ordinateur Portable	Informa-tique	2020-01-01	2022-06-30
P12345_France_ 20220701	P12345	France	Ordinateur Portable	Électronique	2022-07-01	9999-12-31

Clé entreprise composite	Code_ Produit	Pays	Nom_ Produit	Catégorie	Date_ Début	Date_ Fin
P12345_Canada_20200101	P12345	Canada	Laptop	Électronique	2020-01-01	9999-12-31

Ainsi, nous pouvons répondre à la problématique d'historisation mais nous conservons un risque important de collision multi-sources et de performances et volumétrie.

1.3 Clés artificielles (surrogate key)

La clé artificielle est une valeur générée en interne (souvent un entier) qui sert de clé primaire dans la dimension, indépendamment du système source.

Exemple

Appliquer les clés artificielles à l'exemple précédent permettra de garder les avantages d'une clé composite tout en répondant aux limites générées en termes de lourdeur de jointure et complexité du modèle OLAP.

Code_Produit	Pays	Nom_Produit	Catégorie	Date_Début	Date_Fin
P12345	France	Ordinateur Portable	Informatique	2020-01-01	2022-06-30
P12345	France	Ordinateur Portable	Électronique	2022-07-01	9999-12-31
P12345	Canada	Laptop	Électronique	2020-01-01	9999-12-31

Une clé artificielle composée peut être générée de deux manières :

– Une clé primaire auto-incrémentée, générée automatiquement à chaque insertion dans la table de dimension, permet d'identifier les lignes de manière simple et performante. En revanche, elle impose une gestion spécifique dans les processus ETL pour assurer la correspondance avec les données de la table de faits, ce qui peut devenir coûteux en mémoire et en ressources de calcul, surtout lorsque les volumes de données sont importants.
– Il est possible de générer un hash à partir de trois colonnes afin de garantir l'unicité des lignes dans la dimension. Ce hash peut être converti en entier à l'aide d'un algorithme comme SHA1, SHA2 ou MD5. L'avantage, c'est qu'il peut être reproduit côté table de faits, ce qui facilite l'alignement entre les deux tables sans avoir à gérer une clé artificielle classique et gérer de lourds processus ETL.

Un exemple en SQL pour générer un Hash entier ne dépassant pas le bigint, à partir de l'algorithme SHA1.

```
-- SQL Server (T-SQL)
SELECT ABS(CAST(HASHBYTES('SHA1', CONCAT('P12345', 'France',
'2020-01-01')) AS bigint)) AS bigint_hash;
```

Artificial key	Code_ Produit	Pays	Nom_ Produit	Catégorie	Date_ Début	Date_ Fin
1632989580715717 2764	P12345	France	Ordinateur Portable	Informatique	2020-01-01	2022-06-30
1414194802738638 7409	P12345	France	Ordinateur Portable	Électronique	2022-07-01	9999-12-31
9755626434983725 597	P12345	Canada	Laptop	Électronique	2020-01-01	9999-12-31

Cette clé peut être générée de la même manière aussi bien du côté de la dimension que celui de la table de fait. Ce qui permet d'avoir une unicité et de chaque ligne de la dimension et son lien vers les lignes de la table de fait.

Il est important de bien gérer l'intégrité référentielle, en assurant que chaque clé étrangère dans la table de fait corresponde à une clé primaire existante dans la dimension associée. Ceci doit être géré par des ETL robustes : une clé étrangère pointant vers «nulle part», appelé aussi une clé orpheline, peut fausser les agrégats ou causer des erreurs de requête.

Une pratique courante est d'avoir un membre «Inconnu» ou «Non défini» dans chaque dimension pour gérer les faits arrivant avant leur dimension, évitant ainsi les rejets.

Remarque

Dans SSAS Multidimensionnel, la propriété UnknownMember de la dimension permet de gérer automatiquement les faits orphelins sans ligne dédiée dans la table de dimension.

2. Relations entre plusieurs tables de faits

2.1 Drill-across et gestion des niveaux de granularité

Dans les structures décisionnelles élaborées, on rencontre souvent plusieurs tables de faits, chacune étant associée à des processus d'affaires spécifiques ou à différents niveaux de détail.

L'approche du *drill-across* autorise un utilisateur à examiner plusieurs tables de faits simultanément, en les liant par le biais de leurs dimensions partagées.

À la différence du *drill-down* qui explore en profondeur une même table, le *drill-across* permet de naviguer horizontalement d'une table de faits à une autre.

Exemple

Supposons une entreprise de distribution qui possède deux tables de faits :

– Table des ventes quotidiennes :

Date	Magasin	Produit	Quantité_Vendue	Chiffre_Affaires
2025-06-01	Paris	P12345	5	250
2025-06-01	Lyon	P67890	3	90

– Table des retours produits mensuels :

Mois	Catégorie	Nombre_Retours
2025-06	Informatique	15
2025-06	Accessoires	4

Pour pouvoir croiser les données de la table des ventes quotidiennes avec la table des retours produits mensuels, il faudra utiliser une dimension commune de la même granularité (ici, une dimension temps au mois) afin d'uniformiser les niveaux de détail des deux tables.

Pour le faire, nous devons agréger la table la plus détaillée (ici la table des ventes quotidiennes) au mois.

```
TSQL :
SELECT
   FORMAT(Date, 'yyyy-MM') AS Periode,
   Catégorie,
   SUM(Quantité_Vendue) AS Total_Ventes,
   SUM(Chiffre_Affaires) AS CA
FROM Ventes
JOIN Produit ON Ventes.Produit = Produit.Code_Produit
GROUP BY FORMAT(Date, 'yyyy-MM'), Catégorie;
```

Periode	Catégorie	Quantité_Vendue	Chiffre_Affaires
2025-06	Informatique	5	250
2025-06	Accessoires	3	90

En résultat, nous pouvons donc faire apparaître les retours de commandes ainsi que les ventes d'articles par catégorie dans un seul tableau.

Periode	Catégorie	Quantité_ Vendue	Chiffre_ Affaires	Nombre_ Retours
2025-06	Informatique	5	250	15
2025-06	Accessoires	3	90	4

Le drill-across est une fonctionnalité très puissante pour les analyses mais reste très dépendante d'une modélisation stricte et peut représenter des risques d'erreurs si les granularités ne sont pas documentées ou si des contrôles de cohérence automatisés ne sont pas mis en place (en ETL).

2.2 Cas des modèles multi-faits

Dans plusieurs cas un modèle multi-faits peut être utilisé, ce dernier peut intégrer plusieurs tables de faits dans un même schéma de base de données décisionnelle.Ces modèles permettent de couvrir des processus métiers variés tout en centralisant les analyses.

Par exemple, au paragraphe Drill-across et gestion des niveaux de granularité de ce chapitre, nous avons vu une table de fait pour les ventes d'articles informatiques ainsi qu'une autre table de fait pour les retours de commandes.

L'optimisation de ces modèles passe par une standardisation des dimensions partagées, la création de vues matérialisées pour faciliter les analyses croisées ou une modélisation des dépendances temporelles entre faits (comme expliqué dans le paragraphe précédent).

2.2.1 Standardisation des dimensions partagées

Nous pouvons avoir deux tables de dimensions issues de deux sources différentes mais concernent la même entité métier à quelques différences près, l'objectif d'une standardisation est d'avoir une même dimension commune pour les deux sources. Il est même possible de regrouper plusieurs entités métier ayant presque les mêmes caractéristiques dans une seule dimension.

Exemple

Une entreprise technologique commercialise des licences de logiciels ainsi que des prestations IT, elle utilise deux ERP différents pour ces deux types de produits.

Les deux sources contiennent chacune une table client, l'idée est de fusionner les deux tables client dans une seule dimension en standardisant les champs de cette dernière.

ERP 1 est dédié à la commercialisation des licences et contient une table client et une table pour les ventes de licences.

Table de client :

ClientCode	RaisonSociale	Pays
C101	TechVision SA	France
C102	NetSecure Ltd.	France
C103	Alpha Consulting	Belgique

Table des ventes de licences :

DateVente	ClientCode	Produit	Montant
2025-06-01	C101	Logiciel CRM Pro	4 200 €
2025-06-10	C102	Antivirus Pro Pack	1 500 €
2025-06-15	C103	Suite Comptabilité	3 000 €

ERP 2 est dédié aux prestations IT et contient une table client et une table de ventes des prestations.