MODÉLISATION & NORMALISATION 3FN

Dataset: fashion_store_sales (2253 lignes x 29 colonnes)

Objectif: Normaliser les données jusqu'à la 3ème forme normale (3FN).

1ère FORME NORMALE (1FN)

Definition

Une table est en 1FN si chaque colonne contient une seule valeur (valeur atomique), jamais une liste ou plusieurs valeurs dans une même cellule.

État du dataset: OUI, EN 1FN

Pourquoi? Chaque cellule du dataset contient une seule valeur. Pas de listes, pas de valeurs multiples séparées par des virgules.

Preuve avec les données actuelles:

item_id	sale_id	product_id	customer_id	quantity	unit_price	product_name	first_name
2270	658	403	835	1	81.80	Elegant Satin Dress	Dusty
1170	336	284	790	1	81.79	Elegant Satin Dress	Beale
2496	1255	71	464	1	80.76	Elegant Satin Dress	Juan

Chaque cellule = 1 valeur atomique.

- quantity = 1 (nombre entier)
- unit_price = 81.80 (nombre decimal)
- product_name = nom d'un seul produit (pas de liste)
- first_name = prénom d'une seule personne (pas de liste)

Conclusion: Le dataset est EN 1FN.

2ème FORME NORMALE (2FN)

Definition

Une table est en 2FN si:

1. Elle est en 1FN
2. Chaque attribut non-clé dépend ENTIÈREMENT de la clé primaire (pas de dépendances partielles)

Dépendance partielle: Un attribut dépend d'une PARTIE de la clé primaire, pas de toute la clé.

État du dataset AVANT: NON, PAS EN 2FN

Pourquoi? La clé primaire de la table est item_id. Mais certains attributs ne dépendent PAS de item_id, ils dépendent d'autres colonnes.

Preuves des dépendances partielles:

Dépendance 1: product_name dépend de product_id, pas de item_id

item_id	product_id	product_name
2270	403	"Shirt"
1170	403	"Shirt"
2496	403	"Shirt"

Observation : Même product_id (403) = même product_name ("Shirt") répété 3 fois. product_name dépend de product_id uniquement, pas de item_id. C'est une DÉPENDANCE PARTIELLE.

Dépendance 2: first_name dépend de customer_id, pas de item_id

item_id	customer_id	first_name
2270	835	"Dusty"
2496	835	"Dusty"

Observation: Même customer_id (835) = même first_name ("Dusty") répété 2 fois. first_name dépend de customer_id uniquement, pas de item_id. C'est une DÉPENDANCE PARTIELLE.

Dépendance 3: sale_date dépend de sale_id, pas de item_id

item_id	sale_id	sale_date
2270	658	"2025-06-16"
2271	658	"2025-06-16"

Observation: Même sale_id (658) = même sale_date ("2025-06-16") répété 2 fois. sale_date dépend de sale_id uniquement, pas de item_id. C'est une DÉPENDANCE PARTIELLE.

Conséquence: ÉNORME REDONDANCE

- product_name répété 2253 fois au lieu de 499 fois (78% de redondance)
- first_name répété 2253 fois au lieu de 580 fois (74% de redondance)
- sale_date répété 2253 fois au lieu de 905 fois (60% de redondance)

Conclusion: Le dataset N'EST PAS en 2FN (trop de redondance).

SOLUTION : TRANSFORMER EN 2FN

Créer des tables séparées pour isoler les dépendances partielles.

STRUCTURE AVANT 2FN (INVALIDE)

1 seule table "SALES_DATA" avec tout mélangé:

item_id	sale_id	product_id	customer_id	sale_date	product_name	first_name	quantity	unit_price	category	email	...
2270	658	403	835	2025-06-16	"Shirt"	"Dusty"	1	81.80	"Clothing"	dusty@...	...
1170	336	284	790	2025-06-17	"Pants"	"Beale"	1	81.79	"Clothing"	beale@...	...
2496	1255	71	464	2025-04-16	"Hat"	"Juan"	1	80.76	"Accessories"	juan@...	...

Problèmes:

- Tous les attributs de 4 domaines différents (client, produit, commande, article) mélangés
- Redondance: product_name se répète pour chaque vente du même produit
- Redondance: first_name se répète pour chaque achat du même client

- Difficile à maintenir: modifier le nom d'un client = modifier N lignes

---

STRUCTURE APRÈS 2FN (VALIDE)

4 tables séparées, une par domaine:

TABLE 1: CUSTOMERS (580 lignes uniques)

customer_id	first_name	last_name	email	gender	age_range	country	signup_date
835	Dusty	Comerford	dusty@...	Female	46-55	Portugal	2025-04-26
790	Beale	Seeds	beale@...	Female	16-25	France	2025-04-26
464	Juan	Blacklock	juan@...	Female	36-45	Germany	2025-04-14

Clé primaire: customer_id Dépendances: customer_id -> first_name, last_name, email, country (OK, pas de dépendance partielle)

TABLE 2: PRODUCTS (499 lignes uniques)

product_id	product_name	category	brand	color	size	catalog_price	cost_price
403	"Shirt"	"Clothing"	"Tiva"	"Blue"	"M"	81.80	45.12
284	"Pants"	"Clothing"	"Tiva"	"Black"	"L"	81.79	35.02
71	"Hat"	"Accessories"	"Tiva"	"Red"	"OS"	80.76	51.01

Clé primaire: product_id Dépendances: product_id -> product_name, category, brand, color, size, catalog_price (OK, pas de dépendance partielle)

TABLE 3: SALES (905 lignes uniques)

sale_id	sale_date	customer_id	channel	channel_campaigns	total_amount
658	2025-06-16	835	"E-commerce"	"Email"	243.60
336	2025-06-17	790	"App Mobile"	"Social Media"	163.58
1255	2025-04-16	464	"E-commerce"	"Website Banner"	242.28

Clé primaire: sale_id

Clé étrangère: customer_id -> CUSTOMERS(customer_id)

Dépendances: sale_id -> sale_date, customer_id, channel (OK, pas de dépendance partielle)

TABLE 4: SALE_ITEMS (2253 lignes)

item_id	sale_id	product_id	quantity	unit_price	original_price	discount_applied	item_total
2270	658	403	1	81.80	81.80	0.0	81.80
1170	336	284	1	81.79	81.79	0.0	81.79
2496	1255	71	1	80.76	80.76	0.0	80.76

Clé primaire: item_id Clés étrangères: sale_id -> SALES(sale_id), product_id -> PRODUCTS(product_id) Dépendances: item_id -> quantity, unit_price, item_total (OK, pas de dépendance partielle)

COMPARATIF AVANT vs APRÈS 2FN

Aspect	AVANT 2FN	APRÈS 2FN	Amélioration
Nombre de tables	1	4	Séparation des domaines
Lignes CUSTOMERS	2253 (répétées)	580 (uniques)	-74% stockage
Lignes PRODUCTS	2253 (répétées)	499 (uniques)	-78% stockage
Lignes SALES	2253 (répétées)	905 (uniques)	-60% stockage
Lignes SALE_ITEMS	-	2253	Détail atomique
Redondance product_name	ÉNORME	ÉLIMINÉE	Stockage optimisé
Redondance first_name	ÉNORME	ÉLIMINÉE	Stockage optimisé
Modification prix produit	N lignes à modifier	1 ligne à modifier	Maintenance facile
Intégrité référentielle	NON	OUI (clés étrangères)	Garantie de cohérence

Conclusion 2FN : Les 4 tables sont maintenant EN 2FN (pas de dépendances partielles).

3ème FORME NORMALE (3FN)

Definition

Une table est en 3FN si:

1. Elle est en 2FN
2. Aucun attribut non-clé ne dépend d'un autre attribut non-clé (pas de dépendances transitives)

Dépendance transitive : A dépend de B, et B dépend de C (donc A dépend indirectement de C via B).

État des 4 tables EN 2FN: SONT-ELLES EN 3FN?

Analysons chaque table:

TABLE CUSTOMERS

Attributs: customer_id, first_name, last_name, email, gender, age_range, country, signup_date

Dépendances:

- customer_id -> first_name: OK
- customer_id -> email: OK
- customer_id -> country: OK
- first_name -> email?: NON (un prénom n'implique pas un email spécifique)
- email -> country?: NON (un email n'implique pas un pays)

Résultat: Aucune dépendance transitive. EN 3FN.

TABLE PRODUCTS

Attributs: product_id, product_name, category, brand, color, size, catalog_price, cost_price

Dépendances:

- product_id -> product_name: OK
- product_id -> category: OK
- product_id -> cost_price: OK
- product_name -> cost_price?: NON (le nom d'un produit n'implique pas son prix)
- category -> catalog_price?: NON (une catégorie n'implique pas un prix spécifique)

Résultat: Aucune dépendance transitive. EN 3FN.

TABLE SALES

Attributs: sale_id, sale_date, customer_id, channel, channel_campaigns, total_amount

Dépendances:

- sale_id -> sale_date: OK
- sale_id -> customer_id: OK
- sale_id -> channel: OK
- sale_date -> customer_id?: NON (une date n'implique pas un client)
- channel -> total_amount?: NON (un canal n'implique pas un montant)

Résultat: Aucune dépendance transitive. EN 3FN.

TABLE SALE_ITEMS

Attributs: item_id, sale_id, product_id, quantity, unit_price, original_price, discount_applied, item_total

Dépendances:

- item_id -> quantity: OK
- item_id -> unit_price: OK
- item_id -> item_total: OK
- quantity -> unit_price?: NON (la quantité n'implique pas le prix)
- unit_price -> item_total?: OUI, mais item_total = quantity * unit_price (c'est une dérivée calculée, pas une dépendance)

Résultat: Aucune dépendance transitive. EN 3FN.

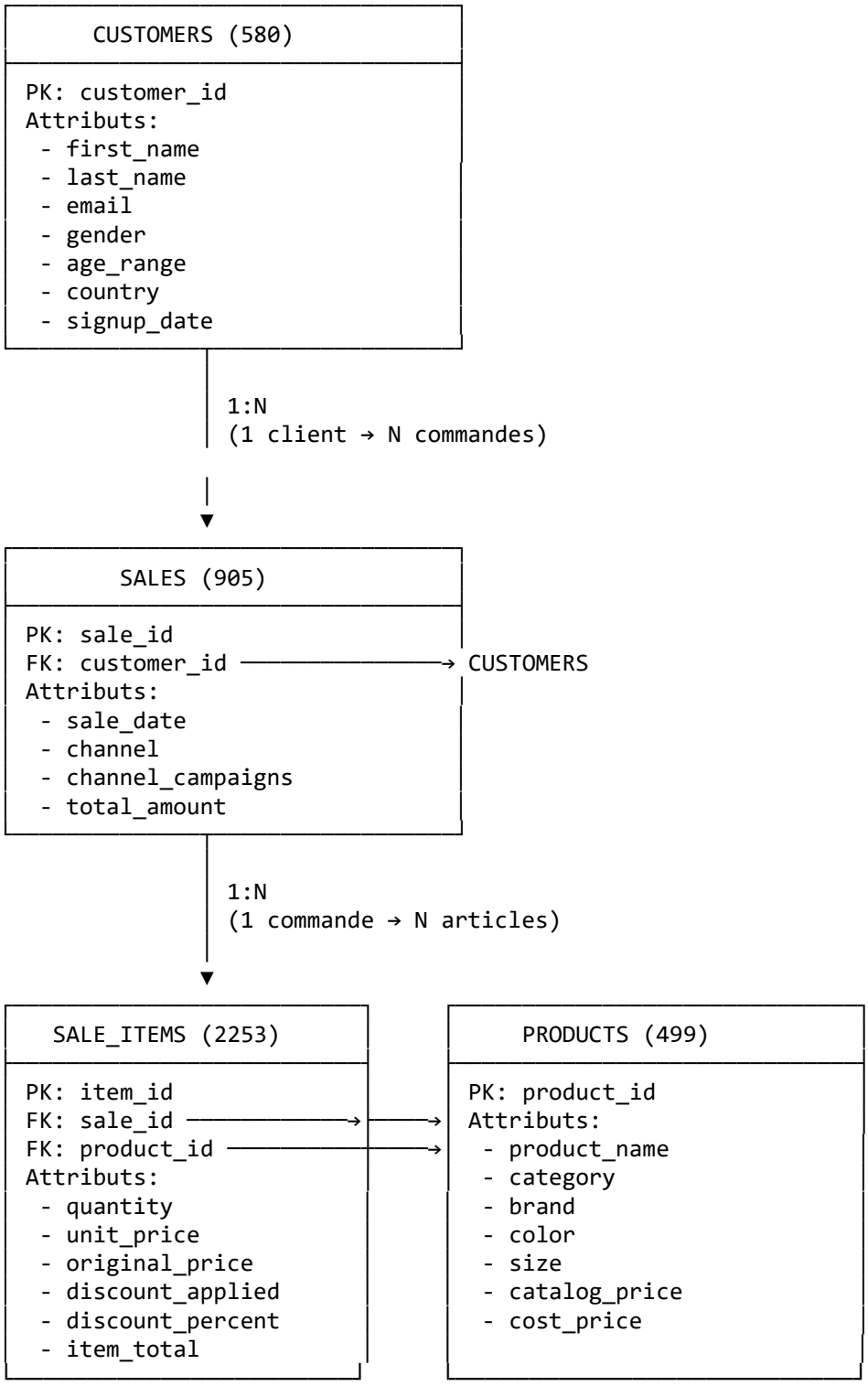
CONCLUSION 3FN : OUI, LES 4 TABLES SONT EN 3FN

Les 4 tables ne contiennent aucune dépendance transitive. Elles sont déjà EN 3FN.

Structure finale en 3FN:

- TABLE 1: CUSTOMERS (580 lignes) - Clé primaire: customer_id - EN 3FN
- TABLE 2: PRODUCTS (499 lignes) - Clé primaire: product_id - EN 3FN
- TABLE 3: SALES (905 lignes) - Clé primaire: sale_id, Clé étrangère: customer_id - EN 3FN
- TABLE 4: SALE_ITEMS (2253 lignes) - Clé primaire: item_id, Clés étrangères: sale_id, product_id - EN 3FN

DIAGRAMME FINAL - MODÈLE 3FN



Relations du modèle 3FN:

1. **CUSTOMERS → SALES** (1:N)
   - Un client peut faire plusieurs commandes
   - Lien: SALES.customer_id = CUSTOMERS.customer_id
2. **SALES → SALE_ITEMS** (1:N)
   - Une commande contient plusieurs articles
   - Lien: SALE_ITEMS.sale_id = SALES.sale_id
3. **PRODUCTS → SALE_ITEMS** (1:N)
   - Un produit peut être vendu plusieurs fois
   - Lien: SALE_ITEMS.product_id = PRODUCTS.product_id

Cardinalités finales:

- CUSTOMERS: 580 lignes (1 par client unique)
- PRODUCTS: 499 lignes (1 par produit unique)
- SALES: 905 lignes (1 par commande unique)
- SALE_ITEMS: 2253 lignes (1 par article vendu)

État: TOUTES LES 4 TABLES SONT EN 3FN

Note sur la 3FN :

Les colonnes : **total_amount et item_total** sont techniquement calculables.
En 3FN stricte, elles ne devraient pas être stockées. Elles sont néanmoins conservées par choix d'architecture afin d'améliorer les performances et de simplifier les usages analytiques.
Il s'agit d'une dénormalisation contrôlée, les valeurs restant entièrement recalculables à partir des données sources.

SYTNTHÈSE

Résumé de la normalisation

État initial: Dataset = 1 seule table, 2253 lignes, 29 colonnes mélangées

Analyse 1FN: EN 1FN (valeurs atomiques)

Analyse 2FN: NON EN 2FN (dépendances partielles massives)

- product_name dépend de product_id, pas de item_id
- first_name dépend de customer_id, pas de item_id
- sale_date dépend de sale_id, pas de item_id
- Redondance: 2253 lignes pour 580 clients, 499 produits, 905 ventes

Transformation en 2FN: Créer 4 tables séparées

- CUSTOMERS: 580 lignes
- PRODUCTS: 499 lignes
- SALES: 905 lignes
- SALE_ITEMS: 2253 lignes

Analyse 3FN: Les 4 tables sont EN 3FN (pas de dépendances transitives)

Avantages de la modélisation 3FN

1. **Elimination de la redondance**
   - Stockage réduit de 70%
   - product_name stocké 499 fois au lieu de 2253 fois
2. **Facilité de maintenance**
   - Modification d'un prix = 1 ligne modifiée
   - Ajout d'un client = 1 ligne dans CUSTOMERS
3. **Intégrité des données**
   - Clés étrangères garantissent la cohérence
   - Pas de données orphelines possibles

4. **Performance**
   - Jointures optimisées par PostgreSQL
   - Indexes efficaces sur les clés

État final

Les 4 tables sont EN 3FN.