StoreFlow — A1.2 Data Model & SQL Specification (Part 2)

6. Product Catalog Domain – Overview

The Product Catalog Domain is responsible for storing and serving all product-related information that a merchant uses across its stores. The design supports merchants with one or multiple stores, allowing products to exist on a merchant-wide level while still supporting store-specific overrides or configurations as needed.

This domain includes the following entities:

• products

• categories (implicit or explicit)

• customization_groups

• customization_options

Key architectural principles:

• Products always belong to a merchant.

• Products may or may not belong to a specific store.

• Product customization is fully hierarchical.

• Every product can have shipping attributes (weight & dimensions).

• Designed to scale for menus, retail, food ordering, and item customization.

7. Product Table – Full Specification

TABLE: products

---

id BIGINT PK

merchant_id BIGINT FK → merchants.id

store_id BIGINT NULL FK → stores.id (optional store-specific product)

category_id BIGINT NULL (future explicit category table)

name VARCHAR(255)

description TEXT

price_cents INT NOT NULL

image_path VARCHAR(255) NULL

is_active BOOLEAN DEFAULT TRUE

SHIPPING FIELDS:

weight_grams INT DEFAULT 0

length_cm INT DEFAULT 0

width_cm INT DEFAULT 0

height_cm INT DEFAULT 0

is_shippable BOOLEAN DEFAULT TRUE

TIMESTAMPS:

created_at TIMESTAMP

updated_at TIMESTAMP

Indexes:

• idx_products_merchant (merchant_id)

• idx_products_store (store_id)

• idx_products_active (merchant_id, store_id, is_active)

• idx_products_category (category_id)

Notes:

• store_id is NULL for merchant-wide products.

• A future feature may allow store-level price overrides.

• Shipping fields are required to support dynamic shipping engine.

8. Product Categories – Specification

STORELEVEL OR MERCHANTLEVEL CATEGORIES:

StoreFlow allows for both implicit and explicit category use-cases.

OPTIONAL TABLE (future use):

TABLE: categories

id BIGINT PK

merchant_id FK

store_id FK NULL

name VARCHAR(255)

sort_order INT

Notes:

• Not required for MVP.

• Category filtering can be implemented implicitly by tag systems or product grouping.

• If categories are enabled later, products.category_id will be a FK.

9. Customization Groups – Full Specification

Customization groups allow products to include selectable options such as:

“Size”, “Add-ons”, “Sauces”, “Toppings”, etc.

TABLE: customization_groups

---

id BIGINT PK

product_id BIGINT FK → products.id

name VARCHAR(255)

min_select INT DEFAULT 0

max_select INT DEFAULT 1

required BOOLEAN DEFAULT FALSE

TIMESTAMPS:

created_at

updated_at

Indexes:

• idx_customization_groups_product (product_id)

Notes:

• required = TRUE means customer must select at least one option.

• min_select / max_select allow flexible choice rules.

10. Customization Options – Full Specification

TABLE: customization_options

---

id BIGINT PK

group_id BIGINT FK → customization_groups.id

name VARCHAR(255)

price_delta_cents INT DEFAULT 0

max_quantity INT DEFAULT 1

TIMESTAMPS:

created_at

updated_at

Indexes:

• idx_customization_options_group (group_id)

Notes:

• max_quantity > 1 allows repeatable add-ons (e.g., extra cheese ×2).

• price_delta_cents may be negative for discounts or free modifiers.

11. Entity Relationships – Detailed ER Notes

Product 1---N Customization Groups

Customization Group 1---N Customization Options

A product can have:

• Zero customization groups (simple item)

• One group (e.g., "Size")

• Many groups (e.g., "Size", "Extras", "Sauces")

Customization groups have complex rules:

• min_select and max_select enforce business logic

• required ensures incomplete orders are rejected

Example Structure:

Product "Burger"

Group "Size": min 1, max 1

- Small

- Large (+200)

Group "Extras": min 0, max 4

- Cheese (+100)

- Bacon (+250)

- Jalapeños (+120)

12. Shipping Attributes – Deep Technical Explanation

Shipping attributes exist on products to support the dynamic shipping engine.

Fields:

• weight_grams

• length_cm

• width_cm

• height_cm

• is_shippable

Use Cases:

1. Food ordering: use weight only; dimensions ignored.

2. Retail items: full use of dimensions required to calculate volume.

3. Non-shippable items: set is_shippable = FALSE; shipping engine automatically excludes them.

Performance Considerations:

• shipping engine uses product weight only for MVP.

• dimensions are reserved for volumetric weight shipping providers.

• computed “cart weight” = SUM(weight_grams * qty).

13. Cascading Rules & Referential Integrity

CASCADE BEHAVIOR:

If a product is deleted:

• customization groups are deleted (ON DELETE CASCADE)

• customization options are deleted (ON DELETE CASCADE)

• product cannot be deleted if referenced by existing orders (restrict for safety)

If a customization group is deleted:

• options are deleted (ON DELETE CASCADE)

If a customization option is deleted:

• allowed only if no orders reference it.

IN PRODUCTION:

Soft deletes may be enabled on:

✓ products

✓ customization_groups

✓ customization_options

This prevents breaking references in existing orders.

14. Example SQL Queries – Product Domain

Query 1 — Get active products for a store:

```
SELECT * FROM products
WHERE merchant_id = ?
AND (store_id = ? OR store_id IS NULL)
AND is_active = 1;
```

Query 2 — Get customization tree for product:

```
SELECT g.*, o.*
FROM customization_groups g
LEFT JOIN customization_options o ON o.group_id = g.id
WHERE g.product_id = ?;
```

Query 3 — Get shippable items in cart:

```
SELECT * FROM products WHERE id IN (…) AND is_shippable = 1;
```

Query 4 — Calculate total cart weight:

```
SELECT SUM(weight_grams * qty)
FROM products p
JOIN cart_items c ON c.product_id = p.id
WHERE c.cart_id = ?;
```