Entity	Attributes	Candidate Identifiers
Customer	customer_id, first_name, last_name, email, phone, shipping_address, created_at	customer_id, email
Order	order_id, customer_id, order_date, status, total_amount, shipping_method	order_id
Product	product_id, sku, product_name, description, price, stock_quantity, category	product_id, sku
OrderItem	order_id, product_id, quantity, unit_price_at_sale	(order_id, product_id)

Choose Primary Keys (PK)

1. Customer: customer_id
   - Selection: Surrogate Key (Auto-incrementing Integer or UUID).
   - Justification: While email is a natural key, it is unstable (users change emails). Using a surrogate customer_id ensures that even if a user updates their email, their order history remains linked.
2. Order: order_id
   - Selection: Surrogate Key.
   - Justification: Orders don't have a reliable natural identifier that is simple. A unique ID generated at the moment of purchase is the industry standard for stability.
3. Product: product_id
   - Selection: Surrogate Key.
   - Justification: Although SKU (Stock Keeping Unit) is a strong natural key, internal product IDs are preferred to protect against supply chain changes where a manufacturer might reuse an SKU for a slightly different item.
4. OrderItem: (order_id, product_id)
   - Selection: Composite Key.

o Justification: This represents the intersection of an order and a product. See Step 4 for the detailed analysis.

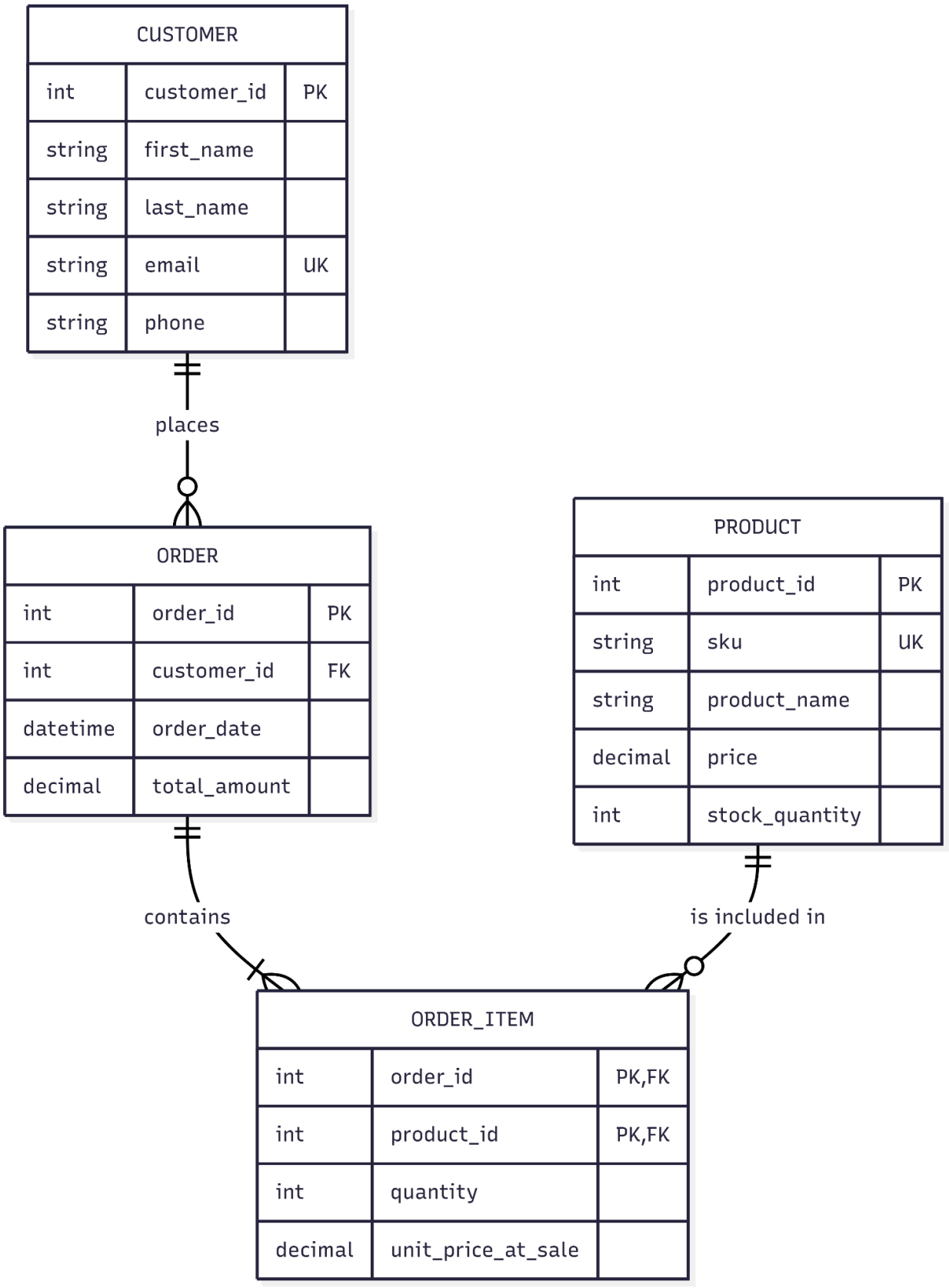
Step 3: Define Relationships & Foreign Keys (FK)

- Order → Customer
  - Relationship: Many-to-One (Many orders can belong to one customer).
  - Foreign Key: Order.customer_id references Customer.customer_id.
  - Required: Yes. An order cannot exist without an associated customer record.
- OrderItem → Order
  - Relationship: Many-to-One (An order contains multiple items).
  - Foreign Key: OrderItem.order_id references Order.order_id.
  - Required: Yes. If an order is deleted, the line items must also be handled (cascading).
- OrderItem → Product
  - Relationship: Many-to-One (A product can be part of many different orders).
  - Foreign Key: OrderItem.product_id references Product.product_id.
  - Required: Yes. Every line item must point to a valid product in the catalog.

Step 4: Composite Key Analysis

In the OrderItem table, a single column is insufficient to identify a record uniquely.

- Why both are needed: The order_id tells us which checkout happened, and the product_id tells us what was bought. Together, they identify a specific "line" on a receipt.
- What breaks with only order_id? You could only have one product per order. If a customer bought a shirt and a hat, the second item would overwrite the first because the PK must be unique.
- What breaks with only product_id? You could only sell each product once in the history of the store.
- The "Same Product Twice" Problem: Usually, we don't allow the same product_id twice in the same order as separate rows. Instead, we increment the quantity column. Therefore, the combination of (order_id, product_id) is the perfect unique anchor.


CUSTOMER
int
customer_id
PK
string
first_name
string
last_name
string
email
UK
string
phone
places
ORDER
int
order_id
PK
int
customer_id
FK
datetime
order_date
decimal
total_amount
PRODUCT
int
product_id
PK
string
sku
UK
string
product_name
decimal
price
int
stock_quantity
contains
is included in
ORDER_ITEM
int
order_id
PK,FK
int
product_id
PK,FK
int
quantity
decimal
unit_price_at_sale