Design and Structure of Amazon's E-commerce Database

1st Juan David Quiroga
Universidad Distrital Francisco Jose de Caldas, Bogotá, Colombia
2nd Luis Alejandro Morales
Universidad Distrital Francisco Jose de Caldas, Bogotá, Colombia

Abstract

This document provides a concise overview of how one of the most recognized Amazon web pages is constructed using the entity-relationship model (DER). It highlights components such as purchasing, sales, suppliers, among others, which will be discussed below.

Index Terms—Amazon, database, DER.

0.1 INTRODUCTION

Amazon is one of the world's largest e-commerce platforms, founded in 1994 by Jeff Bezos. Over the years, Amazon has built a reputation based on convenience, speed of delivery, and a wide variety of products, positioning it as a global leader in online commerce. Amazon's success is largely attributed to its ability to handle millions of daily transactions efficiently, which necessitates a robust and scalable database system. This system enables the storage, organization, and access to large volumes of product, customer, and transaction information, ensuring data is processed quickly and accurately. This paper aims to describe Amazon's database design in the context of its e-commerce operation, focusing on the main entities and relationships that facilitate the management of vast amounts of data handled by the platform.

0.2 METHOD AND MATERIALS

0.2.1 Taking a Look at the DER

The method used to describe and model the Amazon database is the entity-relationship diagram (DER), a widely recognized technique in software and database engineering. The DER graphically represents the logical structure of a database, showing the entities that compose it, the attributes of those entities, and the relationships between them.

In an entity-relationship diagram, the entities represent the main objects to be stored in the database, such as “Customers”, “Products”, and “Orders”. Each entity contains specific attributes that describe its properties, such as name, address, or order date. The relationships between entities define how they interact within the system, for example, a customer may place multiple orders, or an order may be associated with multiple products.

Using a DER to model Amazon’s database simplifies understanding its internal structure, ensuring that data is organized efficiently and that the relationships between entities accurately reflect real e-commerce operations. This is critical for optimizing database queries and overall system performance.

1 DEFINE COMPONENTS

The basis for this is born between:

- **Users**
  - Customers: Interact with the site to search, evaluate, and purchase products. They can leave reviews and ratings.
  - Sellers: Offer products on the platform.
- **Shopping Cart System**
  - User Interaction: Allows users to add products and view their selection before proceeding to checkout.
- **Payment System**

  - Payment Method Interaction: Communicates with various payment gateways (such as credit cards, PayPal) to process transactions.

- **Order Management**

  - Status Update: Allows users to track the status of their orders (processing, shipped, delivered).

2 DEFINE ENTITIES

The following entities are defined:

- Customer (Usuario)
- Product (Producto)
- Category (Categoría de Producto)
- Order (Pedido)
- ShoppingCart (Carrito de Compras)
- PaymentMethod (Método de Pago)
- Review (Reseña)
- Shipping (Envio)
- Offer (Oferta)
- Seller (Vendedor)
- SearchHistory (Historial de Búsquedas)
- ProductRecommendations (Recomendaciones de Productos)
- Returns (Devoluciones)
- Coupons (Cupones)

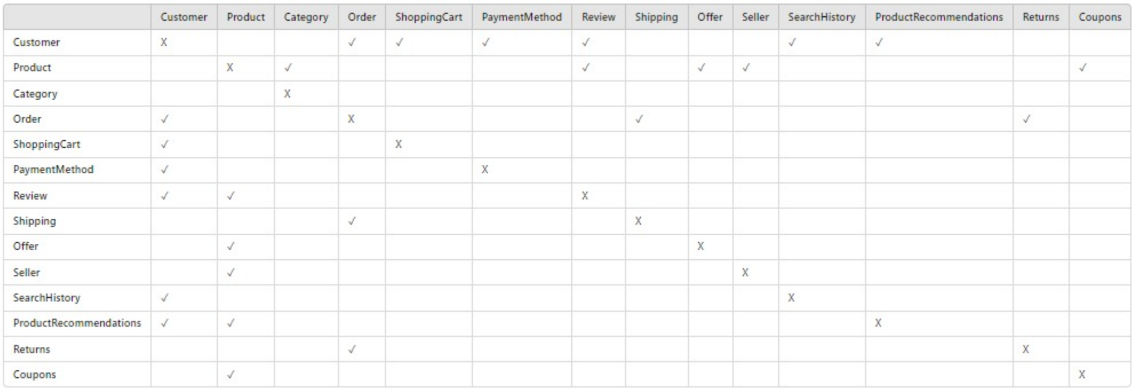
3 DEFINE ATTRIBUTES FOR EACH ENTITY

Each entity is associated with specific attributes that define its characteristics. The key attributes for each entity are as follows:

- **Customer:** User ID (PK), Full Name, Email, Shipping Address, Phone, Registration Date.
- **Product:** Product ID (PK), Product Name, Description, Price, Quantity Available, Category ID (FK), Seller ID (FK).
- **Category:** Category ID (PK), Category Name, Description.
- **Order:** Order ID (PK), Order Date, Customer ID (FK), Total Amount, Order Status, Payment Method ID (FK), Shipping ID (FK).
- **Shopping Cart:** Cart ID (PK), Customer ID (FK).
- **Payment Method:** Payment Method ID (PK), Payment Type, Customer ID (FK).
- **Review:** Review ID (PK), Rating, Comment, Review Date, Customer ID (FK), Product ID (FK).
- **Shipping:** Shipping ID (PK), Shipping Company, Shipping Date, Estimated Delivery Date, Shipping Cost.
- **Offer:** Offer ID (PK), Discount, Start Date, End Date, Product ID (FK).
- **Seller:** Seller ID (PK), Seller Name, Seller Type, Seller Rating.
- **Search History:** Search ID (PK), Search Term, Search Date, Customer ID (FK).
- **Product Recommendations:** Recommendation ID (PK), Customer ID (FK), Recommended Product ID (FK).
- **Returns:** Return ID (PK), Return Date, Return Reason, Return Status, Order ID (FK).
- **Coupons:** Coupon ID (PK), Discount Code, Discount Value, Expiration Date, Applicable Product ID (FK).

4 DEFINE RELATIONSHIPS

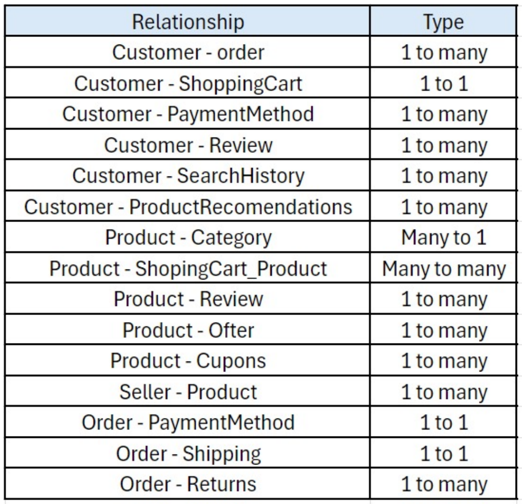
The following table shows the relationships that the entities have with each other, which we can then see what type they are related to.



	Customer	Product	Category	Order	ShoppingCart	PaymentMethod	Review	Shipping	Offer	Seller	SearchHistory	ProductRecommendations	Returns	Coupons
Customer	X			✓	✓	✓	✓				✓	✓		
Product		X	✓				✓		✓	✓				✓
Category			X											
Order	✓			X				✓					✓	
ShoppingCart	✓				X									
PaymentMethod	✓					X								
Review	✓	✓					X							
Shipping				✓				X						
Offer		✓							X					
Seller		✓								X				
SearchHistory	✓										X			
ProductRecommendations	✓	✓										X		
Returns				✓									X	
Coupons		✓												X

Figure 1: Entity Relationship Table

5 DEFINE TYPES OF RELATIONSHIPS



Relationship	Type
Customer - order	1 to many
Customer - ShoppingCart	1 to 1
Customer - PaymentMethod	1 to many
Customer - Review	1 to many
Customer - SearchHistory	1 to many
Customer - ProductRecomendations	1 to many
Product - Category	Many to 1
Product - ShopingCart_Product	Many to many
Product - Review	1 to many
Product - Ofter	1 to many
Product - Cupons	1 to many
Seller - Product	1 to many
Order - PaymentMethod	1 to 1
Order - Shipping	1 to 1
Order - Returns	1 to many

Figure 2: Types of Relationships

6 FIRST VIEW OF THE DIAGRAM

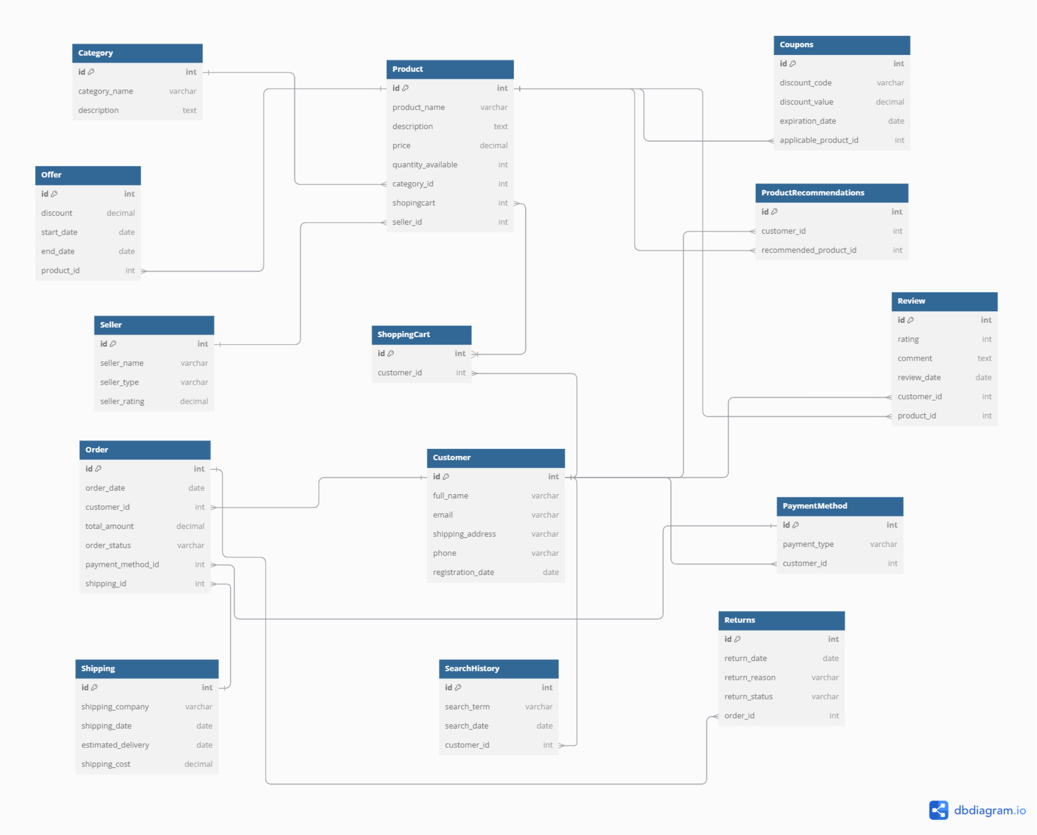


Figure 3: First View of the Entity-Relationship Diagram

7 DIVIDE MANY-TO-MANY RELATIONSHIPS

Within our first design (DER), we find that between the entities ”Products” and ”ShoppingCart” there is a many-to-many relationship since a shopping cart can contain many products and a product can be in many carts. We must eliminate this relationship since it can generate problems in the queries. That is why we implemented a new entity called ”ShoppingCart_Product” with the following attributes:

- **cart_id**: A foreign key referencing the ShoppingCart entity to identify which shopping cart the product belongs to.
- **product_id**: A foreign key referencing the Product entity to identify which product is being added to the shopping cart.
- **quantity**: An integer attribute indicating the number of units of the product that the customer wants to purchase.

In this way, the relationships will be one-to-many.

8 SECOND VIEW OF THE DIAGRAM

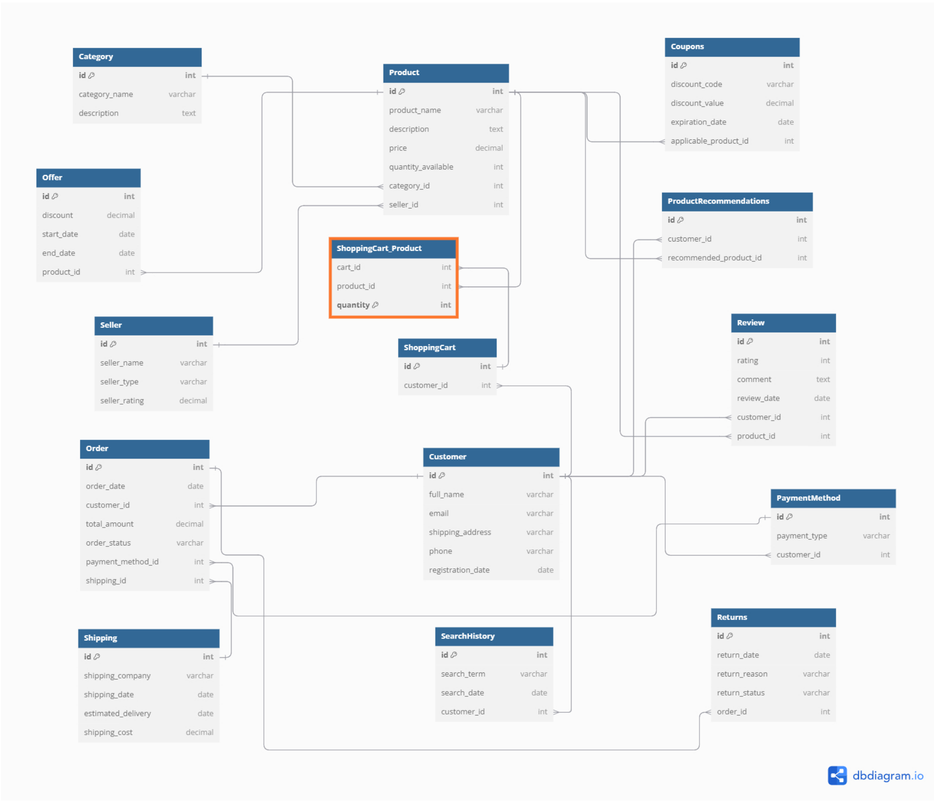


Figure 4: Second View of the Entity-Relationship Diagram

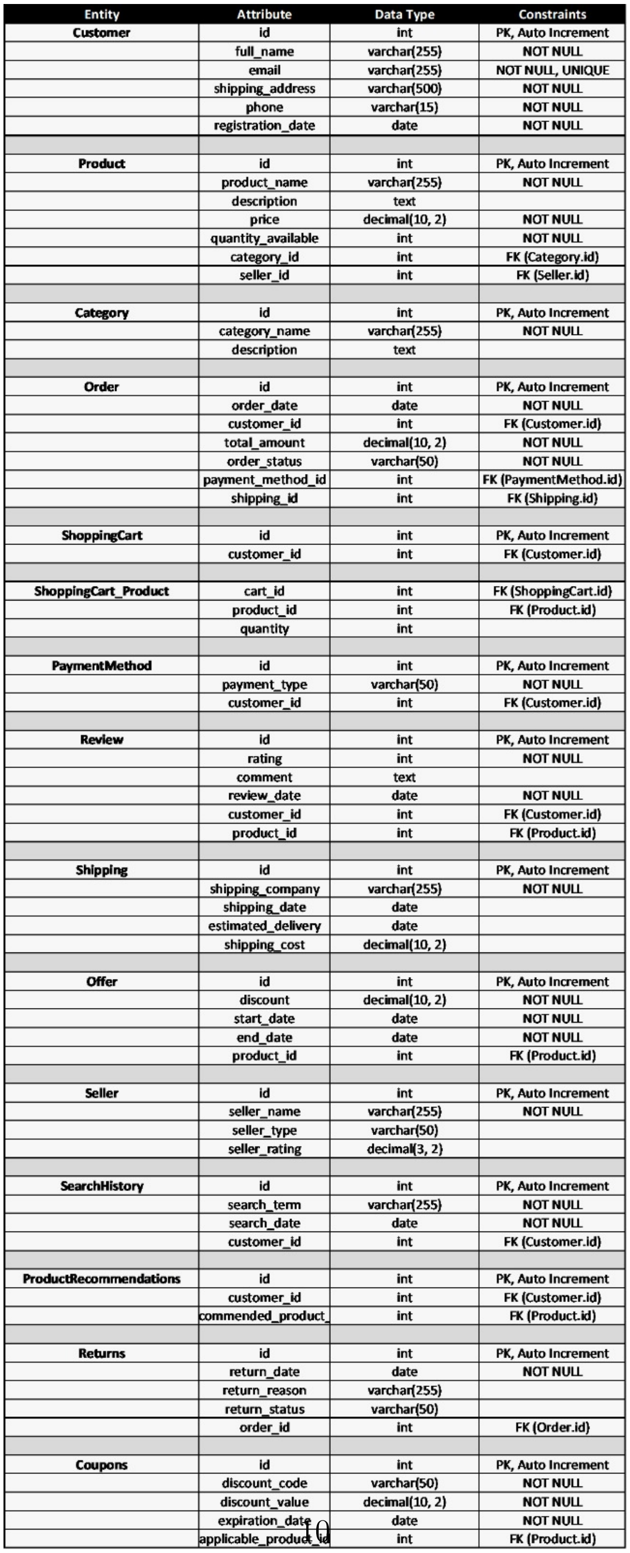
9 OBTAIN THE DATA STRUCTURE

- **Customer** (Cliente):
  - id (int, PK), full_name (varchar), email (varchar), shipping_address (varchar), phone (varchar), registration_date (date)
- **Product** (Producto):
  - id (int, PK), product_name (varchar), description (text), price (decimal), quantity_available (int), category_id (int, FK → Category.id), seller_id (int, FK → Seller.id)
- **Category** (Categoría):
  - id (int, PK), category_name (varchar), description (text)
- **Order** (Pedido):
  - id (int, PK), order_date (date), customer_id (int, FK → Customer.id), total_amount (decimal), order_status (varchar), payment_method_id (int, FK → PaymentMethod.id), shipping_id (int, FK → Shipping.id)
- **ShoppingCart** (Carrito de Compras):
  - id (int, PK), customer_id (int, FK → Customer.id)
- **ShoppingCart_Product**:
  - cart_id (int, FK → ShoppingCart.id), product_id (int, FK → Product.id), quantity (int), PK: (cart_id, product_id)
- **PaymentMethod** (Método de Pago):
  - id (int, PK), payment_type (varchar), customer_id (int, FK → Customer.id)
- **Review** (Opinión):
  - id (int, PK), rating (int), comment (text), review_date (date), customer_id (int, FK → Customer.id), product_id (int, FK → Product.id)

- **Shipping** (Envío):
  - id (int, PK), shipping_company (varchar), shipping_date (date), estimated_delivery (date), shipping_cost (decimal)
- **Offer** (Oferta):
  - id (int, PK), discount (decimal), start_date (date), end_date (date), product_id (int, FK → Product.id)
- **Seller** (Vendedor):
  - id (int, PK), seller_name (varchar), seller_type (varchar), seller_rating (decimal)
- **SearchHistory** (Historial de Búsquedas):
  - id (int, PK), search_term (varchar), search_date (date), customer_id (int, FK → Customer.id)
- **ProductRecommendations** (Recomendaciones de Productos):
  - id (int, PK), customer_id (int, FK → Customer.id), recommended_product_id (int, FK → Product.id)
- **Returns** (Devoluciones):
  - id (int, PK), return_date (date), return_reason (varchar), return_status (varchar), order_id (int, FK → Order.id)
- **Coupons** (Cupones):
  - id (int, PK), discount_code (varchar), discount_value (decimal), expiration_date (date), applicable_product_id (int, FK → Product.id)

10 DEFINE DATA AND COMPONENT PROPERTIES

Finally, each attribute and relationship in the database needs to be defined with its respective properties, such as constraints (e.g., primary keys, foreign keys) and data types (e.g., integer, string, date).



Entity	Attribute	Data Type	Constraints
Customer	id	int	PK, Auto Increment
	full_name	varchar(255)	NOT NULL
	email	varchar(255)	NOT NULL, UNIQUE
	shipping_address	varchar(500)	NOT NULL
	phone	varchar(15)	NOT NULL
	registration_date	date	NOT NULL
Product	id	int	PK, Auto Increment
	product_name	varchar(255)	NOT NULL
	description	text	
	price	decimal(10, 2)	NOT NULL
	quantity_available	int	NOT NULL
	category_id	int	FK (Category.id)
	seller_id	int	FK (Seller.id)
Category	id	int	PK, Auto Increment
	category_name	varchar(255)	NOT NULL
	description	text	
Order	id	int	PK, Auto Increment
	order_date	date	NOT NULL
	customer_id	int	FK (Customer.id)
	total_amount	decimal(10, 2)	NOT NULL
	order_status	varchar(50)	NOT NULL
	payment_method_id	int	FK (PaymentMethod.id)
	shipping_id	int	FK (Shipping.id)
ShoppingCart	id	int	PK, Auto Increment
	customer_id	int	FK (Customer.id)
ShoppingCart_Product	cart_id	int	FK (ShoppingCart.id)
	product_id	int	FK (Product.id)
	quantity	int	
PaymentMethod	id	int	PK, Auto Increment
	payment_type	varchar(50)	NOT NULL
	customer_id	int	FK (Customer.id)
Review	id	int	PK, Auto Increment
	rating	int	NOT NULL
	comment	text	
	review_date	date	NOT NULL
	customer_id	int	FK (Customer.id)
	product_id	int	FK (Product.id)
Shipping	id	int	PK, Auto Increment
	shipping_company	varchar(255)	NOT NULL
	shipping_date	date	
	estimated_delivery	date	
	shipping_cost	decimal(10, 2)	
Offer	id	int	PK, Auto Increment
	discount	decimal(10, 2)	NOT NULL
	start_date	date	NOT NULL
	end_date	date	NOT NULL
	product_id	int	FK (Product.id)
Seller	id	int	PK, Auto Increment
	seller_name	varchar(255)	NOT NULL
	seller_type	varchar(50)	
	seller_rating	decimal(3, 2)	
SearchHistory	id	int	PK, Auto Increment
	search_term	varchar(255)	NOT NULL
	search_date	date	NOT NULL
	customer_id	int	FK (Customer.id)
ProductRecommendations	id	int	PK, Auto Increment
	customer_id	int	FK (Customer.id)
	commended_product_	int	FK (Product.id)
Returns	id	int	PK, Auto Increment
	return_date	date	NOT NULL
	return_reason	varchar(255)	
	return_status	varchar(50)	
	order_id	int	FK (Order.id)
Coupons	id	int	PK, Auto Increment
	discount_code	varchar(50)	NOT NULL
	discount_value	decimal(10, 2)	NOT NULL
	expiration_date	date	NOT NULL
	applicable_product_id	int	FK (Product.id)

Figure 5: Second View of the Entity-Relationship Diagram

11 CONCLUSIONS

The entity-relationship model is a powerful tool for representing the underlying structure of a database, as shown in Amazon's e-commerce platform. By clearly defining entities, attributes, and relationships, we can create a robust framework that can effectively manage the vast amounts of data generated by the platform. This paper provides a solid foundation for understanding how Amazon's database operates, facilitating further research and development in the field of e-commerce.

12 REFERENCES

1. Amazon, 2024. *Annual Report.*

2. R. Elmasri and S. Navathe, 2016. *Fundamentals of Database Systems.* 7th ed. Pearson.

3. M. Blaha and J. Premerlani, 2004. *Object-Oriented Modeling and Design with UML.* 3rd ed. Pearson.

4. J. L. V. P. de Jesus, M. S. V. A. D. de Lima, 2021. "A Review of E-commerce Models: A Case Study of Amazon," *Journal of Retailing and Consumer Services*, vol. 58, p. 102279.