Relational Databases

Entity-Relationship Diagram (ERD)

8 February 2024

克明峻德，格物致知

Quick Recap: Subqueries

Key Concepts:

- **Breaking Down Complex Queries:** Subqueries, particularly with WITH (CTEs), simplify complex SQL queries by breaking them into manageable parts, enhancing both readability and efficiency.
- **Enhanced Data Analysis Capabilities:** Subqueries allow for advanced data analysis such as conditional filtering and comparative analysis within a database, crucial for extracting detailed insights.

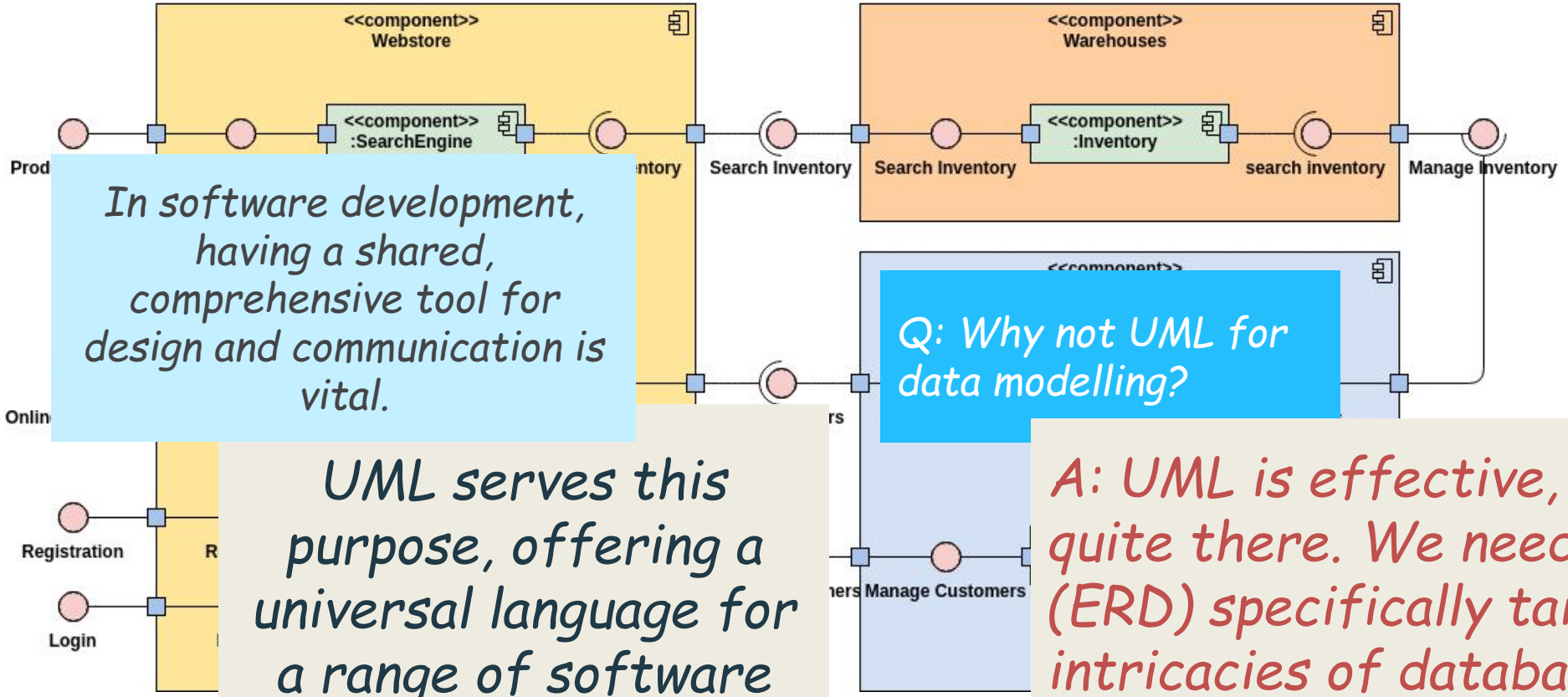
SQL Query Example

```sql
WITH MaxRatings AS (
    SELECT Genre, MAX(Rating) AS MaxRating
    FROM BookRatings
    GROUP BY Genre
)
SELECT B.Title, B.Genre, MR.MaxRating
FROM Books B
INNER JOIN BookRatings BR
ON B.BookID = BR.BookID
INNER JOIN MaxRatings MR
ON B.Genre = MR.Genre AND BR.Rating = MR.MaxRating;
```

- **Output Table:**

```
| Title                | Genre       | MaxRating |
|----------------------|-------------|-----------|
| SQL Essentials       | Technology  | 5         |
| The Garden's Secrets | Hobby       | 4         |
| Data Science Today   | Technology  | 4         |
```

- **Purpose:** Identify the highest-rated book in each genre.



In software development, having a shared, comprehensive tool for design and communication is vital.

UML serves this purpose, offering a universal language for a range of software projects.

Q: Why not UML for data modelling?

A: UML is effective, but still not quite there. We need a design tool (ERD) specifically tailored for the intricacies of database design.

Think also in terms of 'Trade-Off'.

LANGUAGE™

Information System

Information Systems (IS)

- Central structures for organizing, storing, and retrieving information.

Components

- **Inputs:** Data entered by users during their interaction with the system (e.g., product searches, item selections).
- **Processes:** Operations performed by the e-commerce platform (e.g., updating shopping cart, processing checkout).
- **Outputs:** Information generated as a result of processes (e.g., order confirmations, payment receipts).

Database Role

- The core repository where data is stored, linked directly to both the processing and output components.

Database Design

Develop a Unified Data Vocabulary

- Establishing a common set of terms and definitions used across the database.
- Ensuring consistency in how entities, attributes, and relationships are named and understood.

Define and Enforce Business Rules

- Capturing the logic and constraints that govern data relationships and operations.
- Embedding these rules within the ERD to guide database structure and functionality.

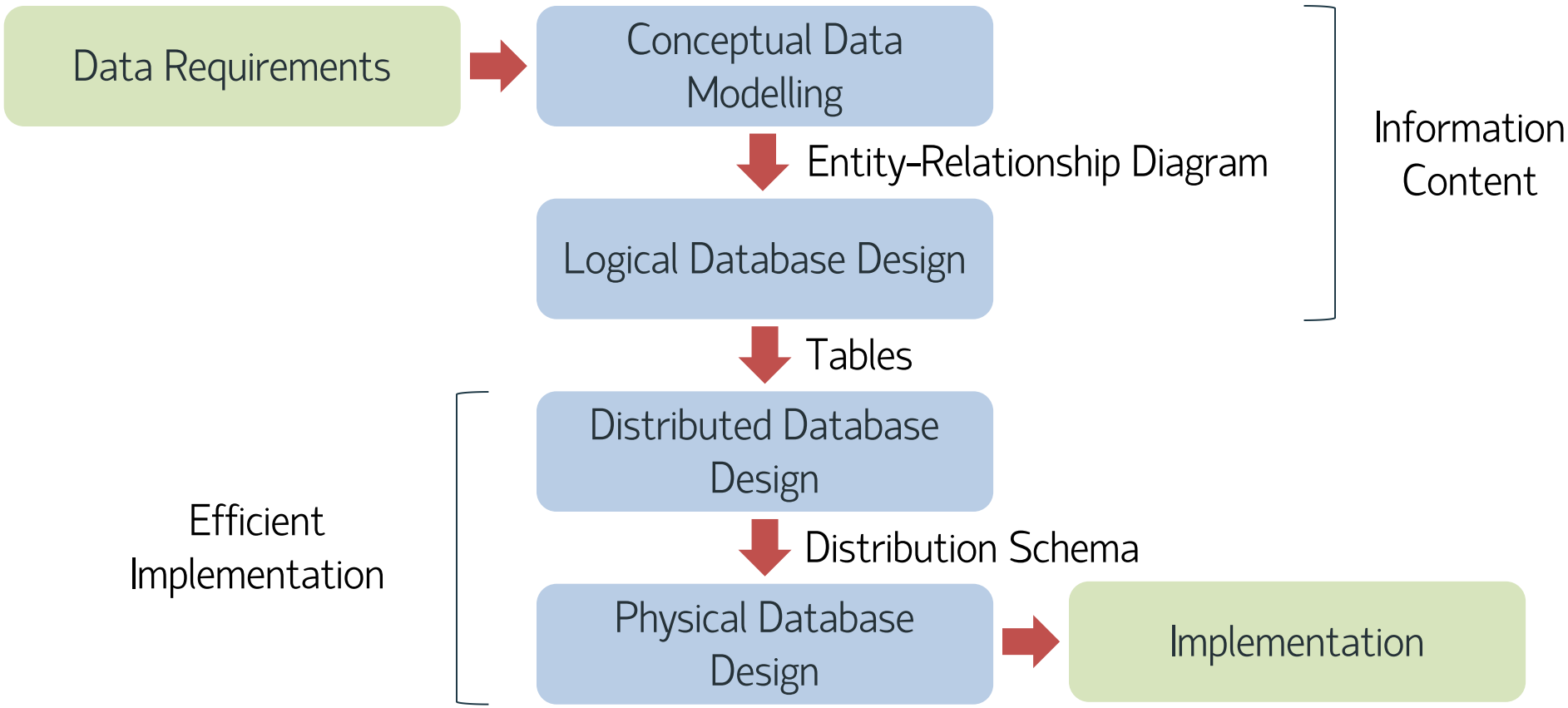
Ensure Data Integrity and Quality

- Implementing measures to ensure accuracy, consistency, and reliability of data.
- Designing the ERD to support validation, data constraints, and referential integrity.

Facilitate Efficient Database Implementation

- Streamlining the path from ERD to physical database design.
- Focusing on scalable, maintainable, and performance-optimized database solutions.

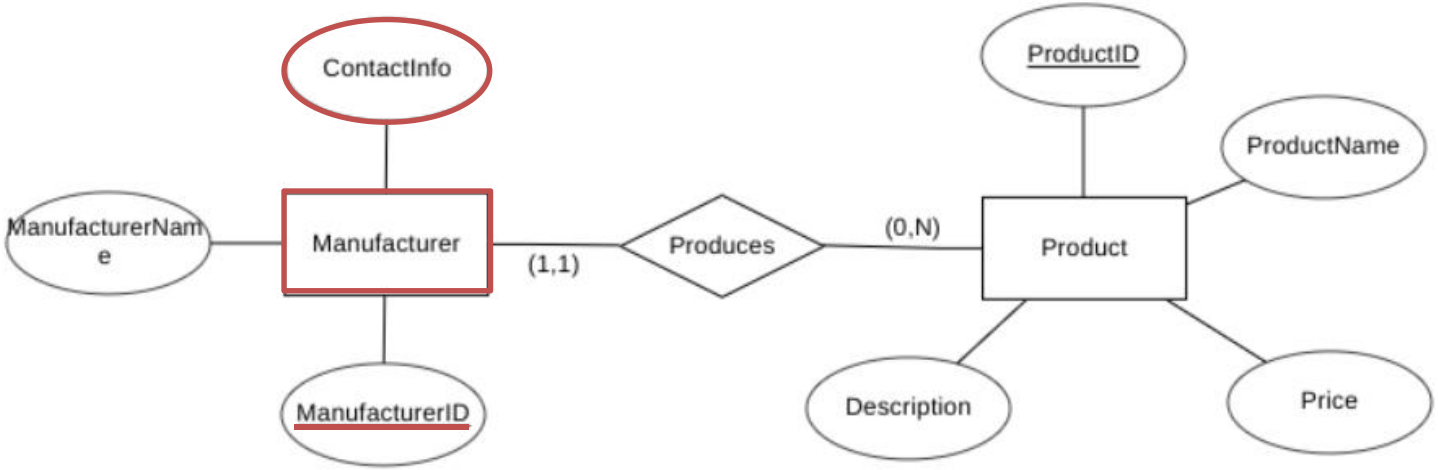
Development Phase



Entitites

Entity Definition

- An entity represents a real-world object or concept.
- In ERDs, entities are typically shown as rectangles.



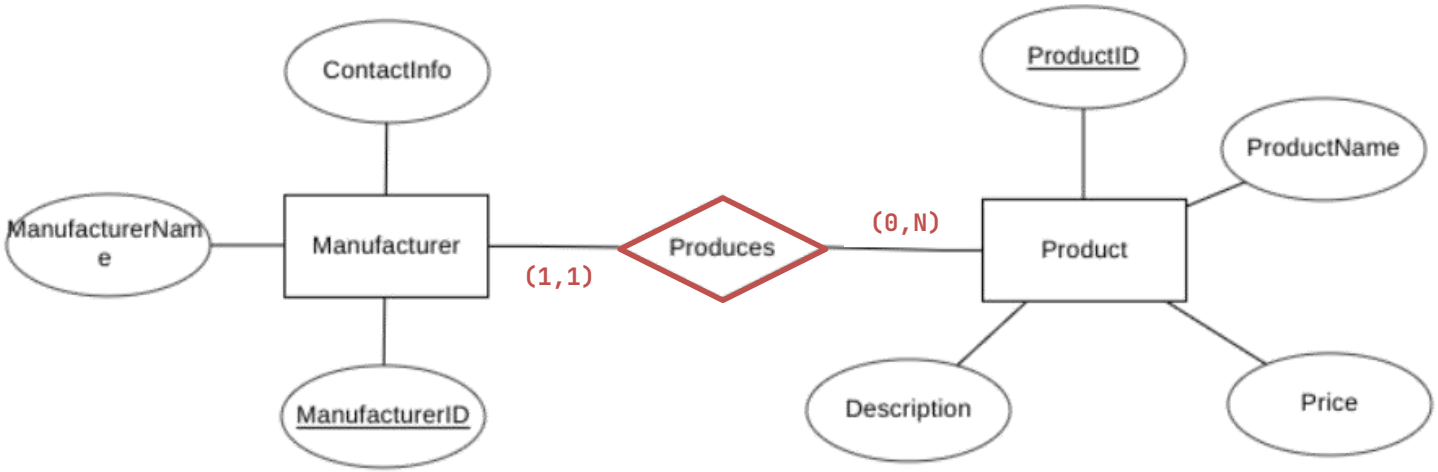
Attributes

- Attributes are characteristics or properties of an entity. In ERDs, attributes are represented as ovals connected to their entity.
- Primary Key, e.g. ManufacturerID in Manufacturer, is denoted by underlying the attribute name in the diagram.

Relationships

Relationship Definition

- A relationship depicts how two entities are related to each other.
- In ERDs, relationships are shown as lines connecting entities with a diamond or label.

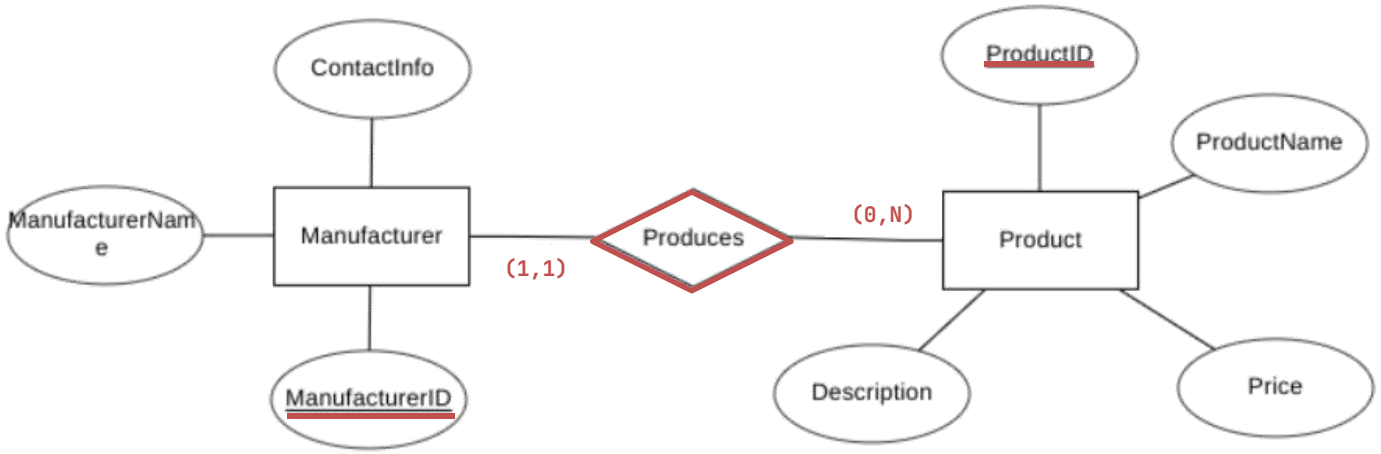


Cardinality Notation

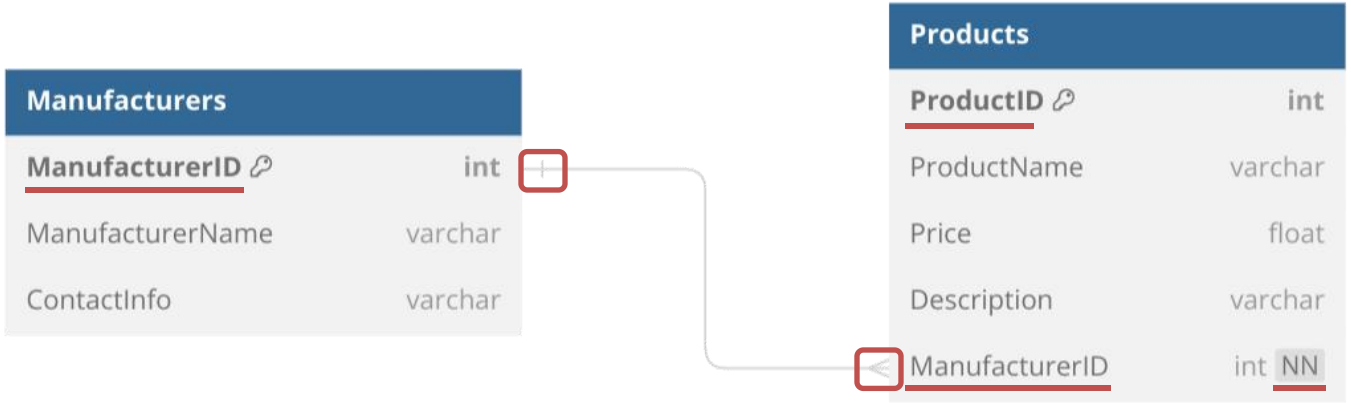
- Indicates the number of instances of one entity related to one instance of another entity.
- Produces: One-to-Many (1:M)
  - 1 (Manufacturer) to Many (Products)
  - Each Product is producted by one Manufacturer. A Manufacturer can produce multiple products.
- 0 (Minimum) indicates Optional. There can be a Manufacturer which does not product any products.
- 1 (Minimum) indicates Mandatory. If there is a Product, then it must be produced by a Manufacturer.

Comparison To Relational Database Schema

ER Diagram:



Relational Database Schema:

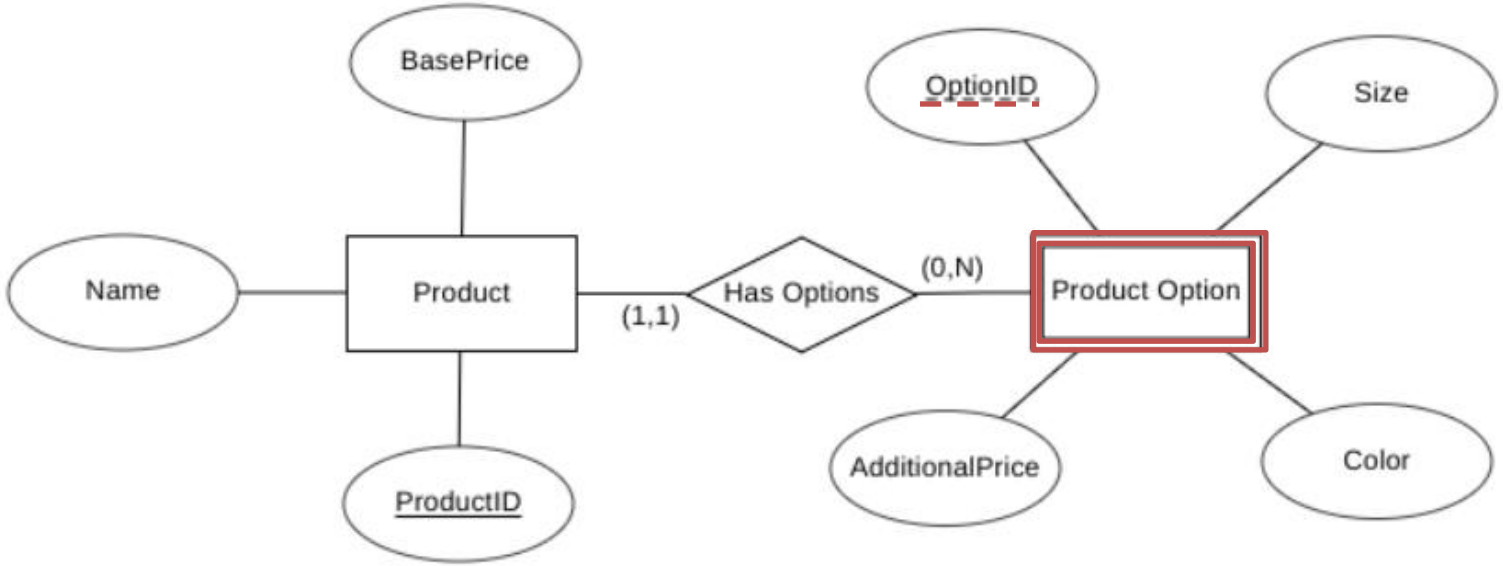


In ERD's notation, the existence of a relationship between two entities implies a foreign key relation in the relational database schema, but these keys are not shown as separate attributes in the ERD.

Weak Entities

Weak Entity Definition

- A weak entity is an entity that cannot be uniquely identified by its own attributes alone. It typically depends on a 'strong' or 'parent' entity for its existence and identification.
- In ERDs, weak entities are often represented with a double border.

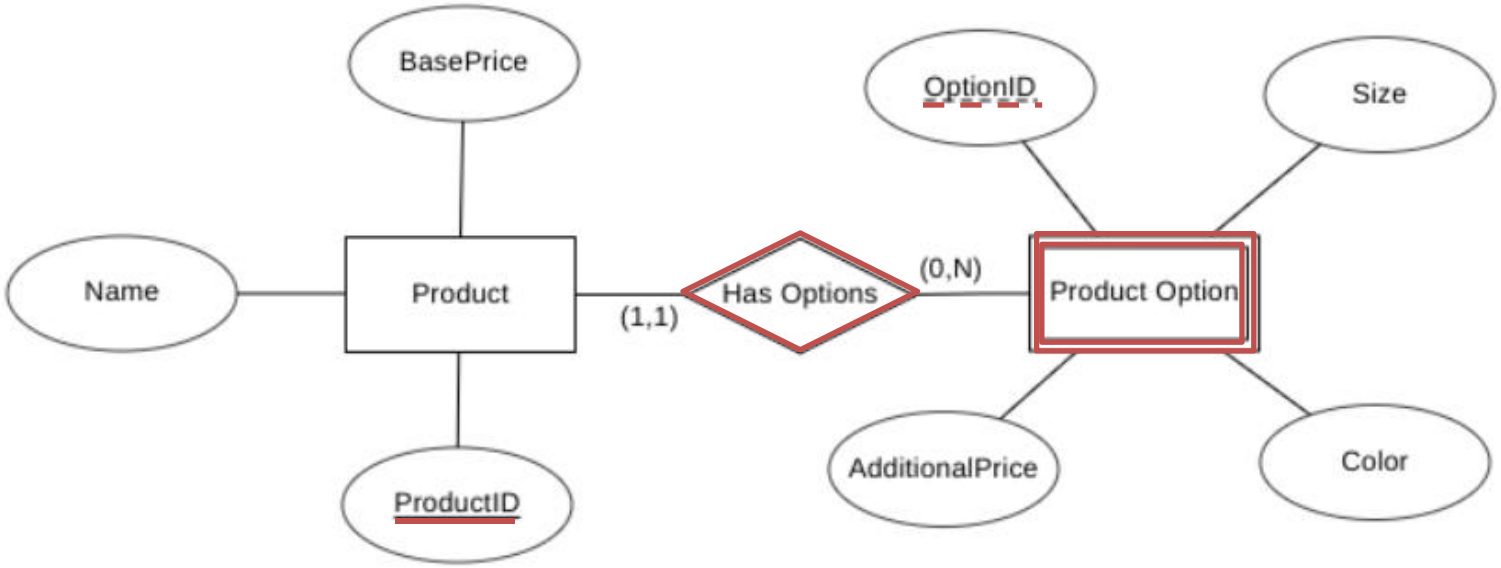


Characteristics of Weak Entities

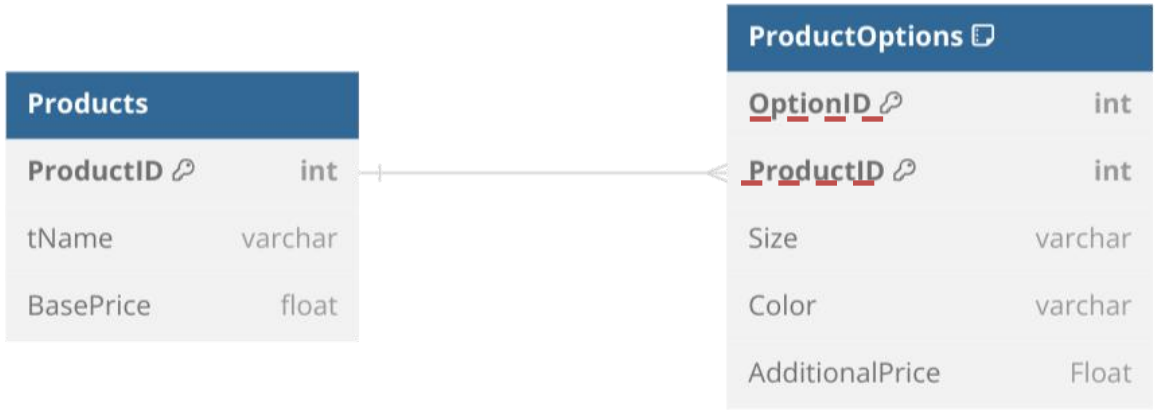
- **Dependence:** Weak entities depend on another entity (strong entity) to ensure their existence and uniqueness.
- **Discriminator (Partial Key):** They often have a partial key, which is an attribute that can uniquely identify weak entities only in conjunction with the key of the strong entity.

Comparison To Relational Schema Diagram

ER Diagram:



Relational Database Schema:



Composite Primary Key:

- (OptionID, ProductID)
- ProductID is also a Foriegn Key.

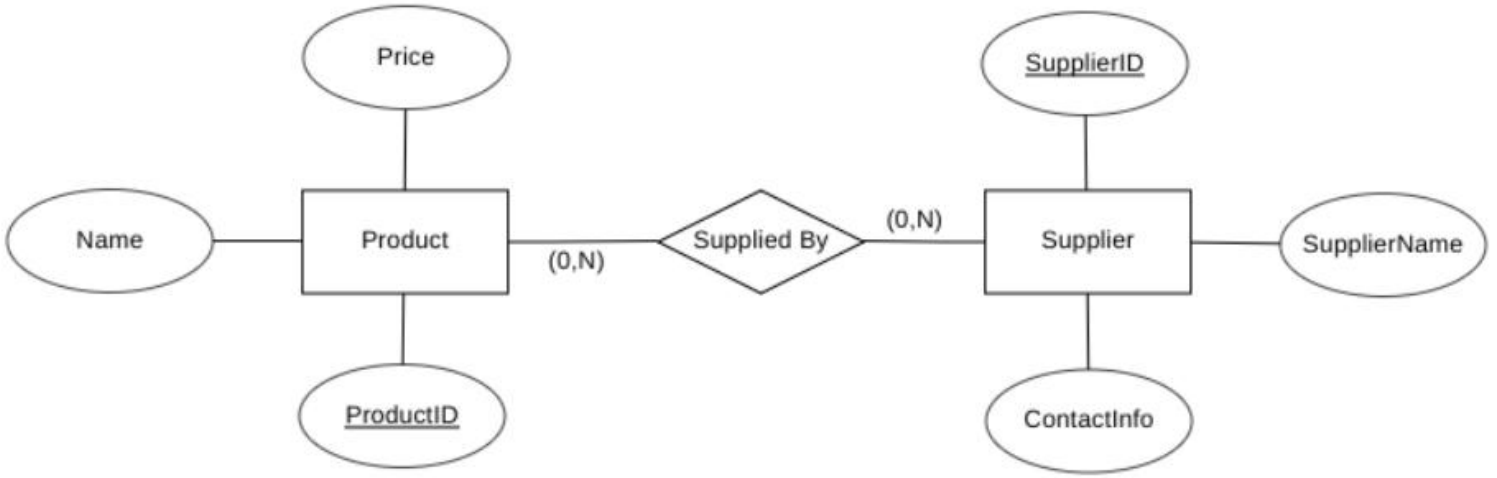
Types of Cardinality

Cardinality Type	Notation	Description	Example
One-to-One (1:1)	1 - 1	Each entity in the relationship will associate with only one entity in the other set.	One User Account is linked to one Customer Profile.
One-to-Many (1:M)	1 - N	An entity on one side of the relationship can be associated with many entities, but those on the other side are associated with only one.	One Category can include many Products.
Many-to-One (M:1)	N - 1	Many entities in one set are associated with a single entity in the other set. (Inverse of 1:M)	Many Products are from one Supplier.
Many-to-Many (M:N)	N - N	Entities in one set can have relationships with multiple entities in the other set and vice versa.	Customers can place many Orders, and each Order can be placed by many Customers.

More Relationship Examples: Many-To-Many (M:N)

What is a Many-to-Many Relationship?

- Defined as a relationship where multiple records in one entity can be associated with multiple records in another entity.
- In ERDs, represented by a line connecting two entities, often with a notation indicating multiplicity.



Example: Products and Suppliers

- **Cardinality:** Many Products can be linked to many Suppliers and vice versa.
- **Business Context:** Reflects real-world e-commerce scenarios of sourcing and inventory diversity.

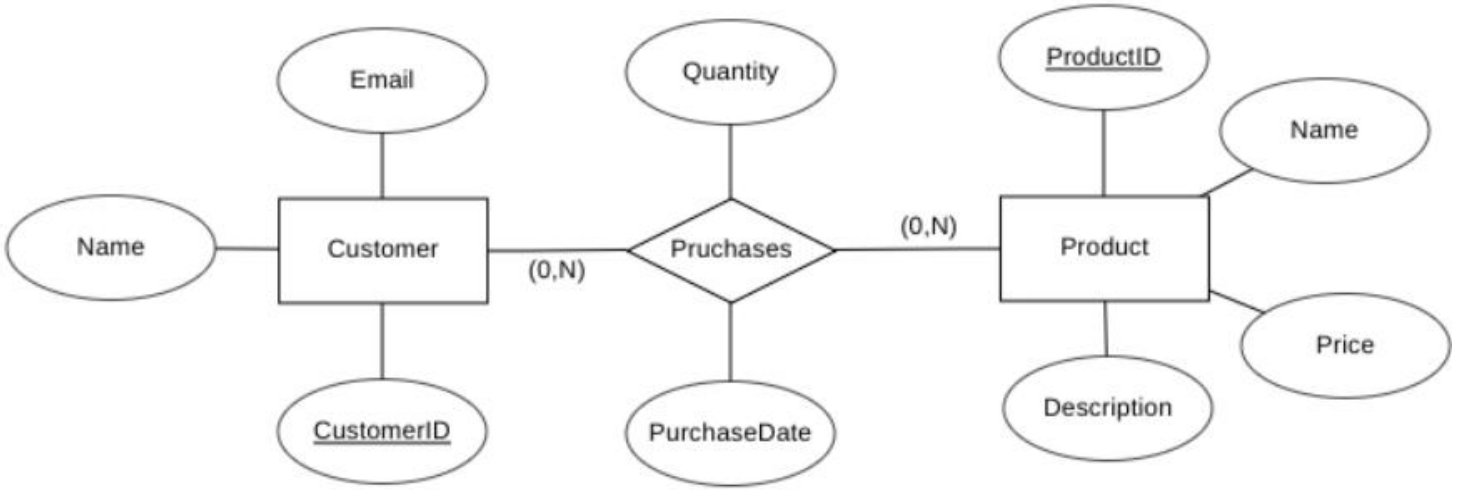
Implementing in Databases

- Often requires an associative entity or junction table to manage these complex relationships effectively.

M-N Relationships with Attributes

Complex Many-to-Many Relationships

- Occur when two entities have a reciprocal relationship involving multiple records on both sides.
- Can include additional attributes to provide more context to the relationship.



Example: Customers and Products

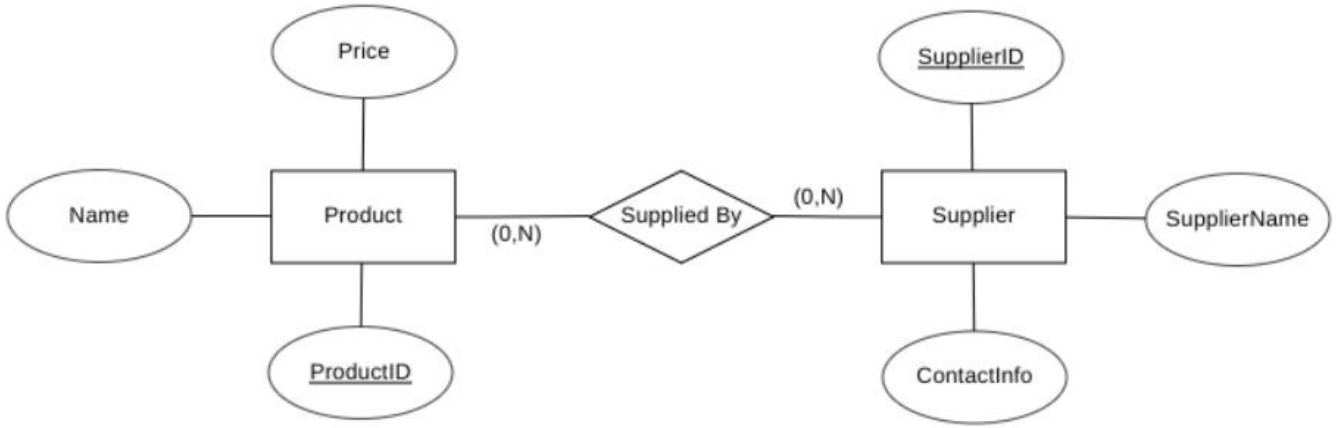
- **Relationship:** Purchases with attributes like Quantity, PurchaseDate, ReviewStatus.
- **Significance:** Enables detailed tracking of customer purchases, important for sales analysis and customer relationship management.

Implementing in Databases

- Managing these relationships can be complex, typically involving a linking table with additional attributes.

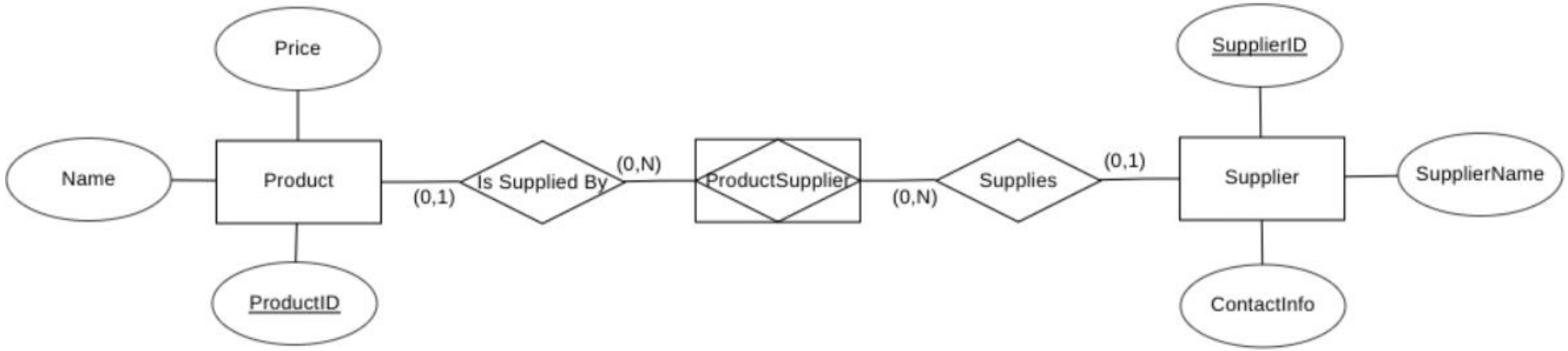
M-N Relationship Equivalency Rule

- M-N Relationship occurs when multiple records in one entity can be associated with multiple records in another entity.



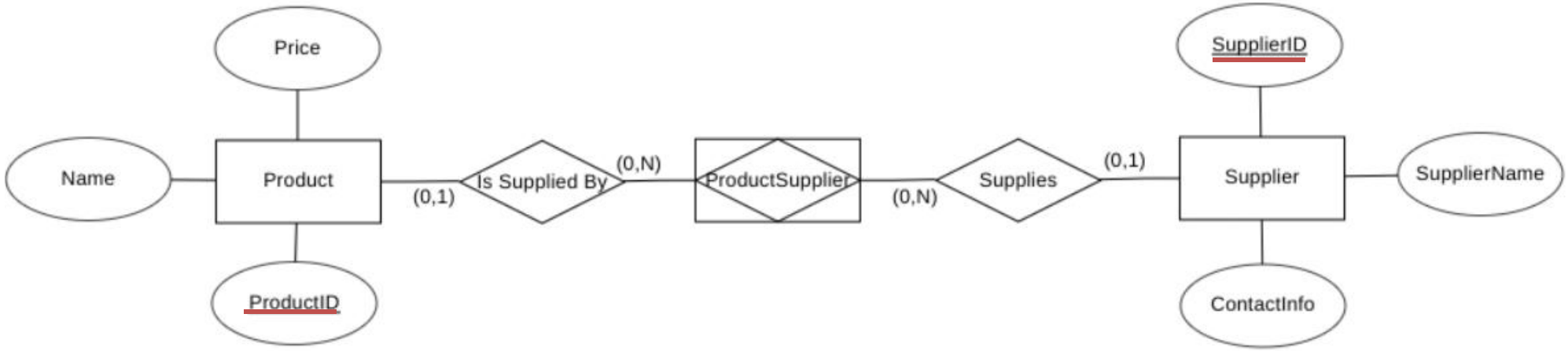
Equivalency Rule for M-N Relationships

- M-N relationships are broken down into two one-to-many (1-M) relationships using an associative entity.
- This associative entity holds foreign keys from both related entities, often with additional attributes.



Comparison To Relational Schema Diagram

ER Diagram:



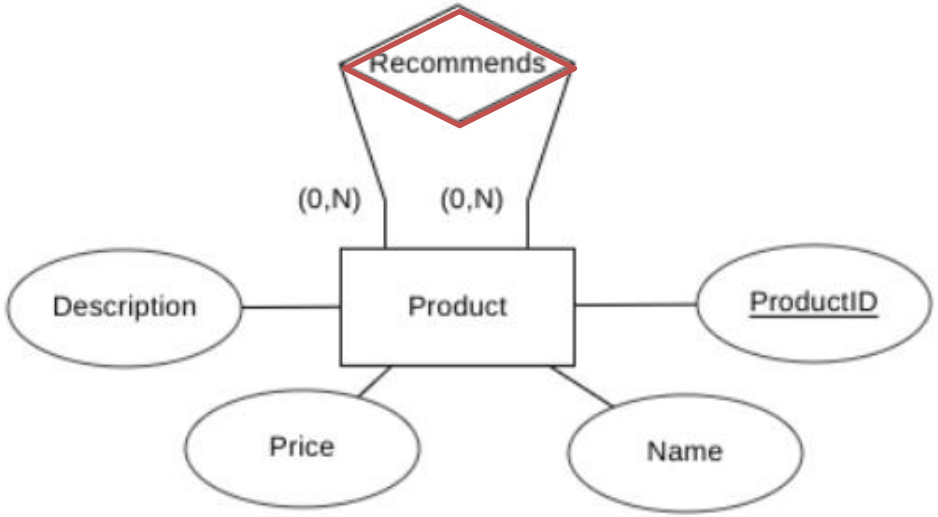
Relational Database Schema:

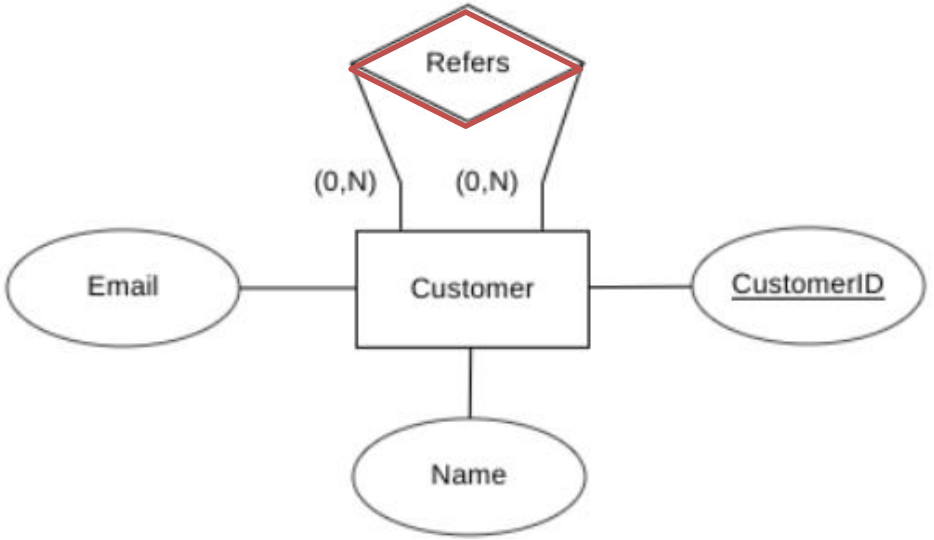
- Creation: An associative entity is created as a new table in the database.
- Key Attributes: Includes foreign keys referencing the primary keys of the entities it connects.

Self-Referencing Relationships

Self-Referencing Relationships Overview

- **Definition:** Occurs when an entity is related to itself, typically seen in scenarios involving hierarchical or peer-to-peer associations.





Product Recommendations

- **Description:** Products recommend other products, creating a network of recommendations enhancing customer experience.

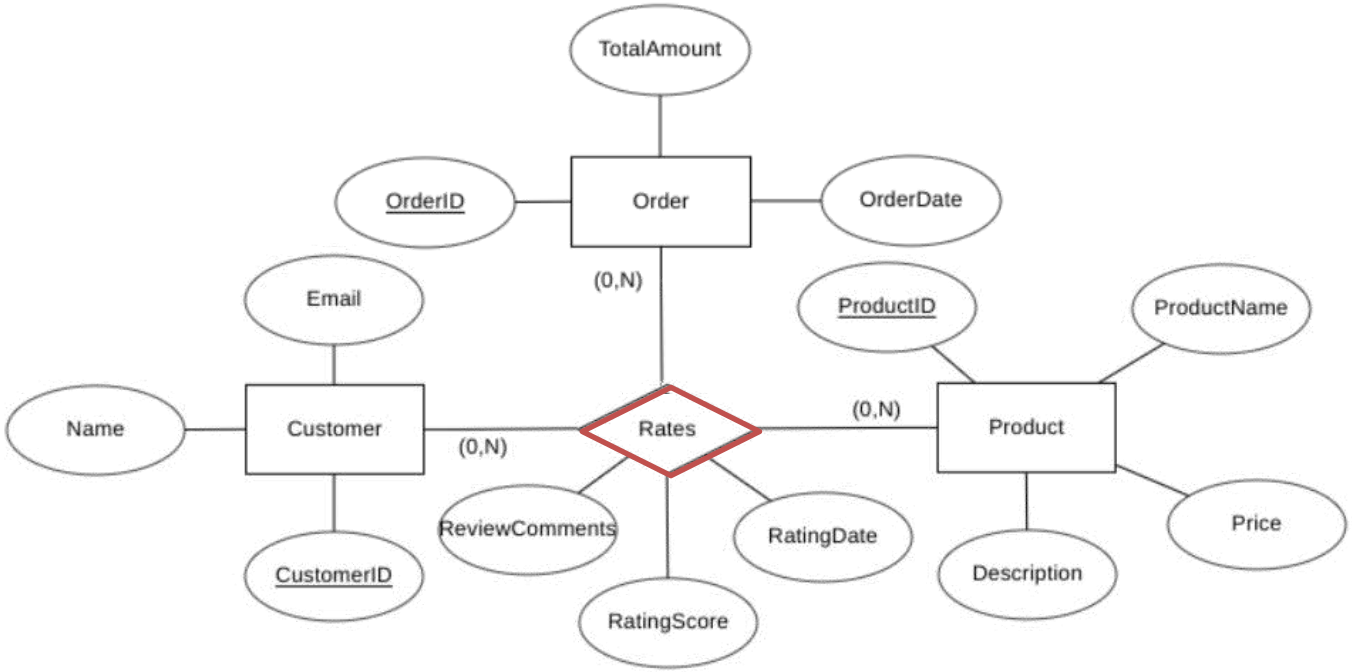
Customer Referral Program

- **Description:** Customers refer new customers, establishing a chain of referrals critical in marketing strategies.

M-Way Relationships

M-Way Relationship Basics

- Definition: An M-Way (or Ternary) relationship involves three different entities simultaneously.
- In ERDs, these relationships capture complex associations that can't be simplified into multiple binary relationships.



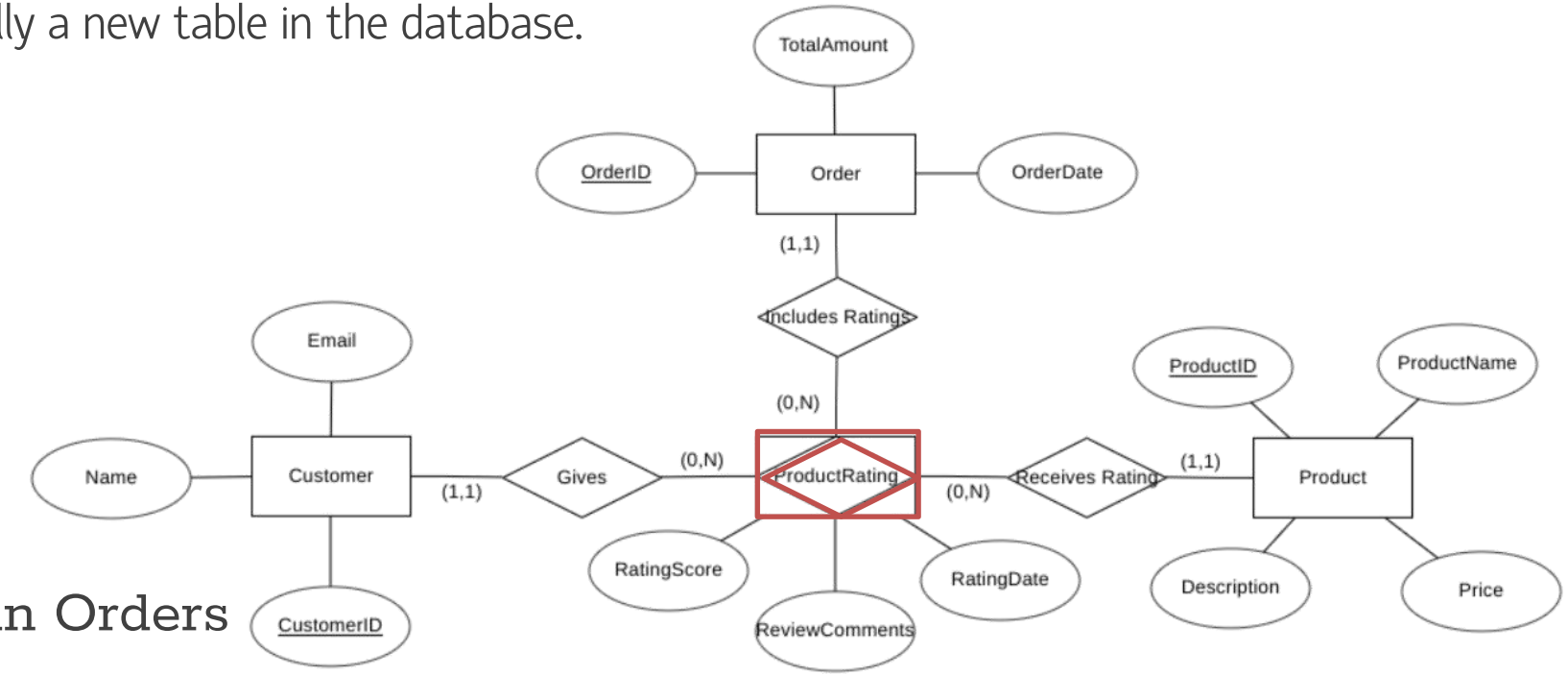
Customer Product Ratings in Orders

- Captures detailed customer feedback specific to each purchase.
- Provides insights into product performance and customer satisfaction within individual orders.

Associative Entity Types for M-Way Relationships

From Conceptual to Physical: Associative Entities

- M-Way relationships in ERDs are conceptual. For database implementation, they are represented using an associative entity that captures the complex interaction between three or more entities.
- This associative entity is essentially a new table in the database.

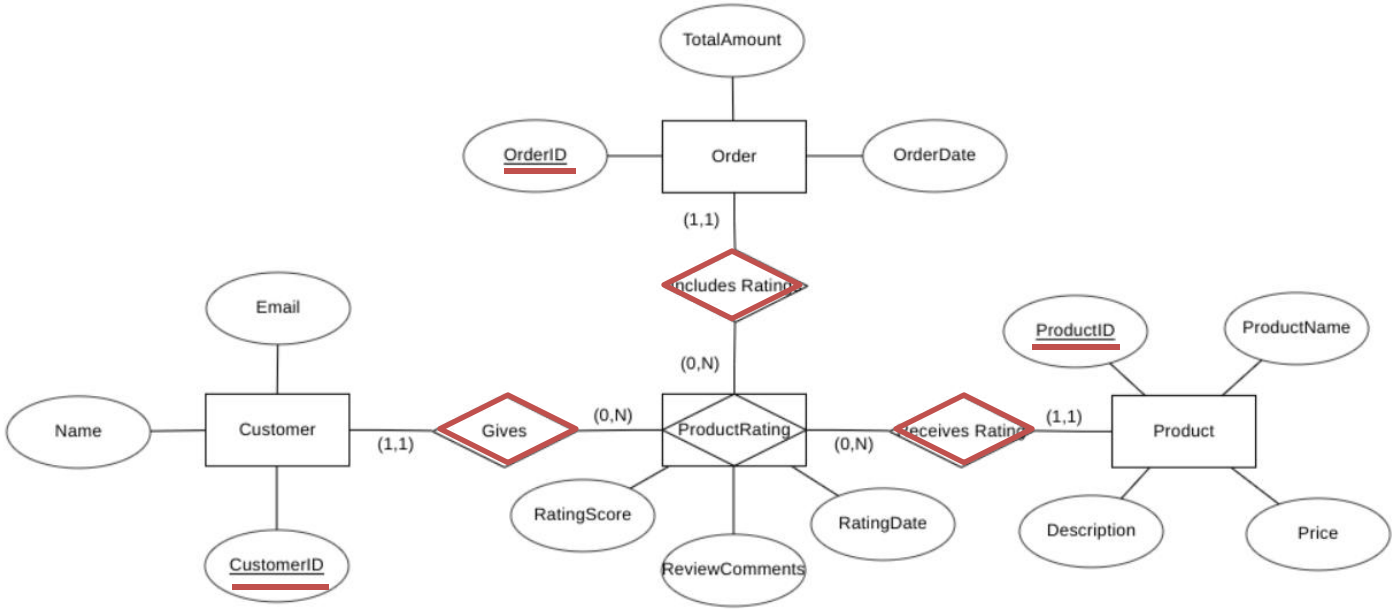


Customer Product Ratings in Orders

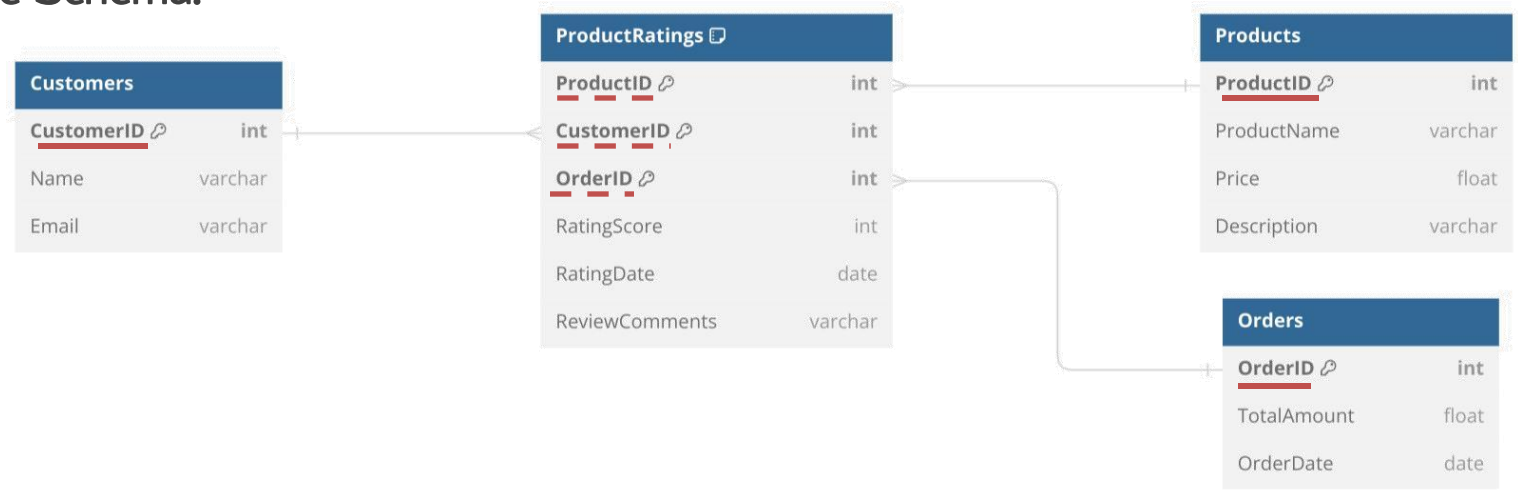
- **Associative Entity:** ProductRating
- **Role:** Acts as a bridge connecting Customer, Product, and Order, while holding unique data for each rating instance.
- The ProductRating table in the database will have columns for each attribute and foreign key.

Comparison To Relational Schema Diagram

ER Diagram:



Relational Database Schema:

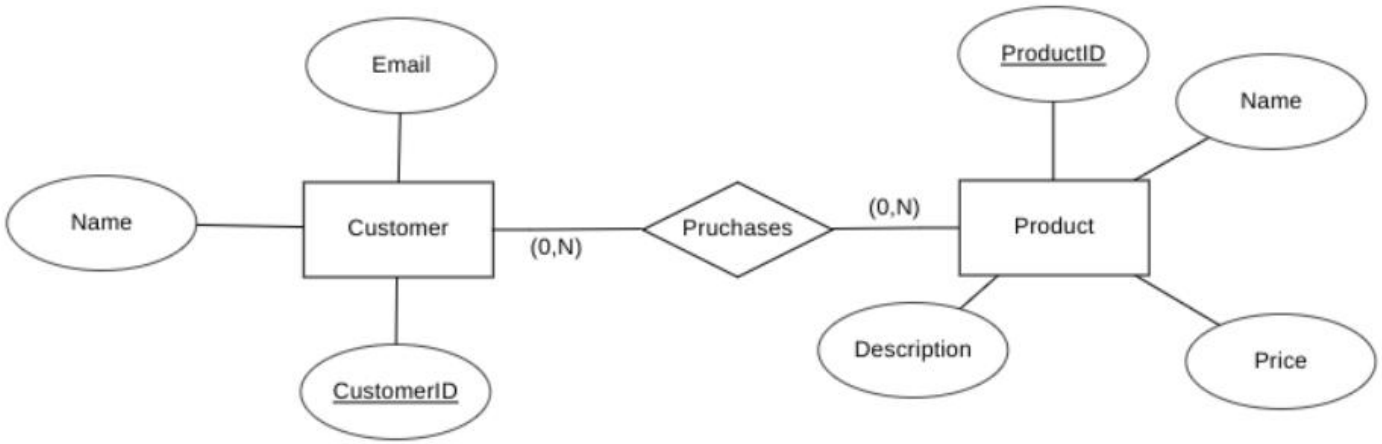


Practice Problems: Basic ER Diagram Creation

Question 1: Basic E-commerce Structure

- **Entities**: Customer, Product
- **Attributes**:
  - Customer: CustomerID, Name, Email
  - Product: ProductID, Name, Price, Description
- **Relationship**: Purchases
- **Instructions**: Create an ER diagram that represents customers and the products they can purchase.

Practice Problems: Basic ER Diagram Creation (Solution)



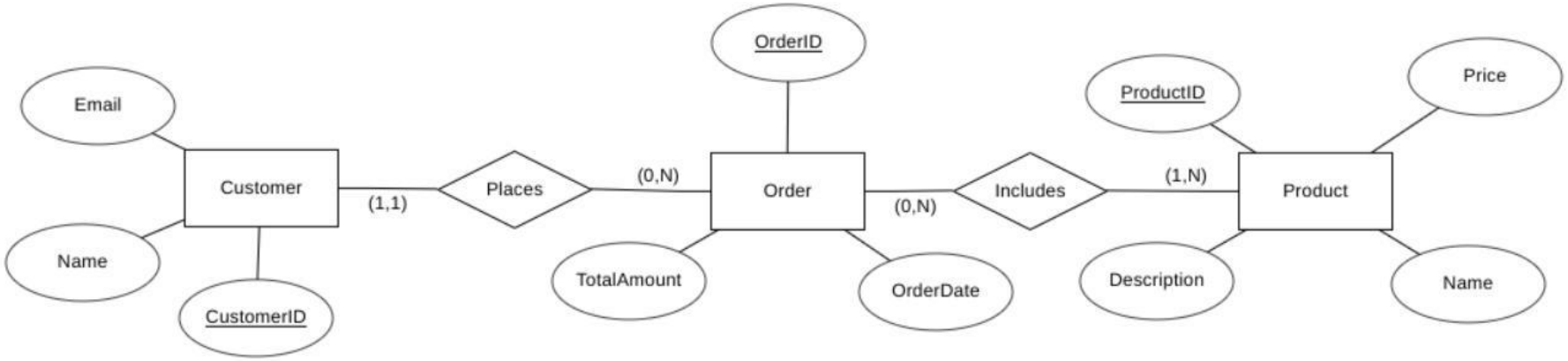
- Notice the Purchases relationship doesn't detail the order process. How can we track order dates, quantities, or multiple products in a single transaction?
- Consider the limitations of representing Purchases without an Order entity. What additional information might be needed for each purchase?

Practice Problems: Introducing Associative Entity

Question 2: Adding Order Processing

- **New Entity**: Order
- **New Attributes**:
  - Order: OrderID, OrderDate, TotalAmount
- **New Relationship**: Places between Customer and Order, Includes between Order and Product
- **Instructions**: Expand on the ER diagram from Question 1 to include orders. Show how customers place orders and how orders include products.

Practice Problems: Introducing Associative Entity (Solution)



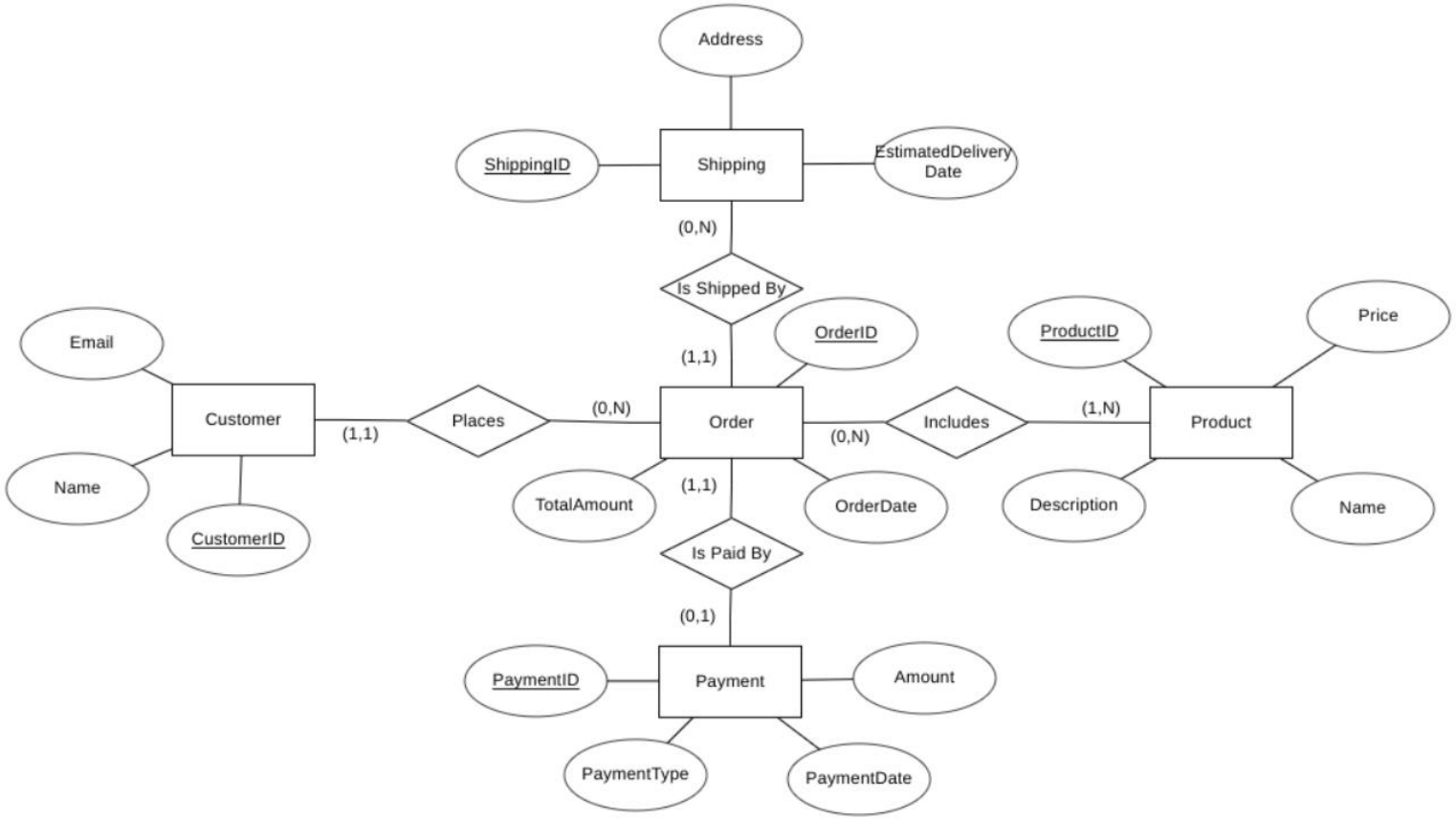
- Currently, the diagram shows orders but lacks details on how payments are processed. How might payments be integrated into the system?
- Consider how orders are fulfilled. What information about shipping or delivery could be added to enhance the model?

Practice Problems: Incorporting Complex Relationships

Question 3: Incorporating Payment and Shipping

- **New Entities**: Payment, Shipping
- **New Attribute**s:
  - Payment: PaymentID, PaymentType, PaymentDate, Amount
  - Shipping: ShippingID, Address, EstimatedDeliveryDate
- **New Relationships**: Is Paid By between Order and Payment, Is Shipped By between Order and Shipping
- **Instructions**: Further expand the ER diagram to include payment and shipping details for each order.

Practice Problems: Incorporting Complex Relationships (Solution)



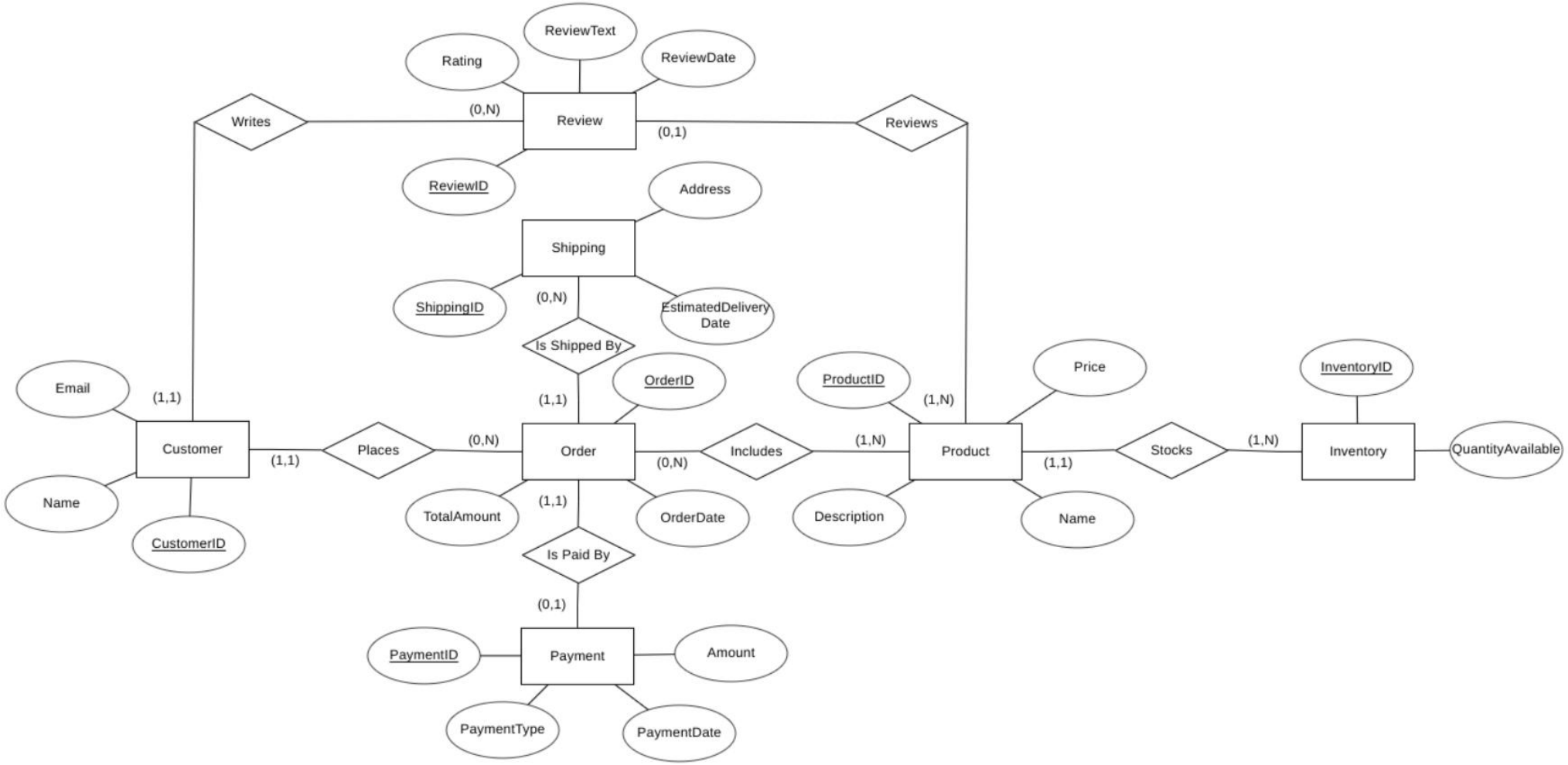
- The diagram now includes payment and shipping, but how can we capture customer feedback on products? Think about integrating product reviews.
- Reflect on inventory management. How does the current diagram track product availability or stock levels?

Practice Problems: Adding Hierarchies and Complex Attributes

Question 4: Advanced Features - Reviews and Inventory

- **New Entities**: Review, Inventory
- **New Attributes**:
  - Review: ReviewID, Rating, ReviewText, ReviewDate
  - Inventory: InventoryID, QuantityAvailable
- **New Relationships**: Writes between Customer and Review (associated with Product), Stocks between Product and Inventory
- **Instructions**: Complete the ER diagram by adding product reviews written by customers and inventory management for products.

Practice Problems: Adding Hierarchies and Complex Attributes (Solution)



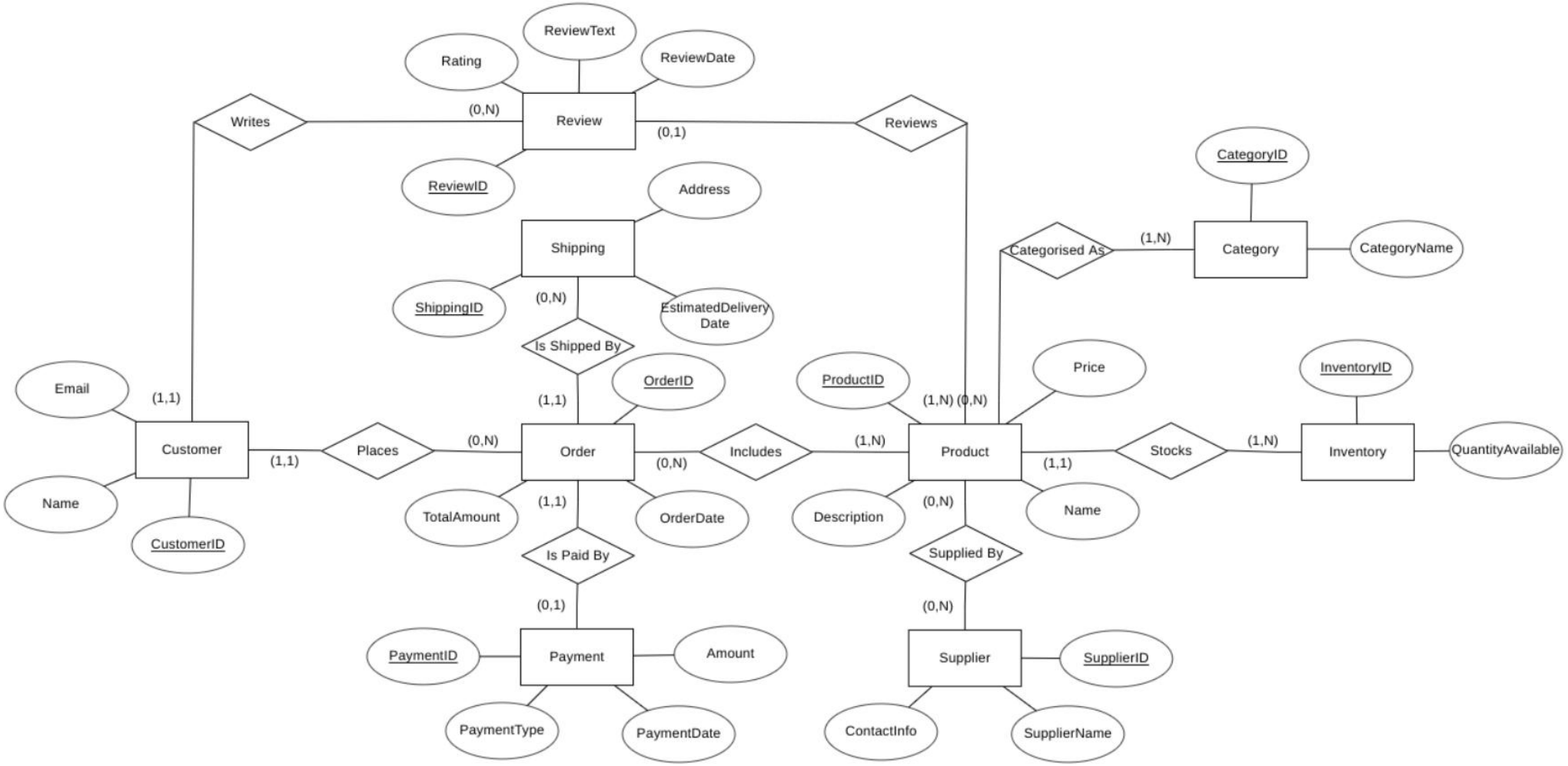
- Our current diagram efficiently handles transactions and inventory, but how are products sourced? Consider the role of suppliers in the e-commerce ecosystem.
- Think about product categorization. How can we represent the various categories a single product might belong to?

Practice Problems: Advanced Hierarchies and Associations

Question 5: Integrating Supplier and Category Management

- **New Entities**: Supplier, Category
- **New Attributes**:
  - Supplier: SupplierID, SupplierName, ContactInfo
  - Category: CategoryID, CategoryName
- **New Relationships**: Supplied By between Product and Supplier, Categorized As between Product and Category
- **Instructions**: Expand the ER diagram to include product suppliers and categories. Show how each product can be supplied by multiple suppliers and belong to multiple categories.

Practice Problems: Advanced Hierarchies and Associations (Solution)



- With suppliers and categories added, consider other real-world aspects. How might we handle returns or exchanges in our e-commerce system?
- Reflect on the user experience. How could the system support personalized product recommendations or wish lists?

ER Diagrams as a Communication Tool

Visual Communication Tool

- ER diagrams provide a clear graphical view of data structures, essential for shared understanding across technical and non-technical stakeholders.

Collaborative Design Process

- Facilitates inclusive feedback, ensuring database design aligns with diverse stakeholder needs and expectations.

Simplifying Complex Data Models

- Breaks down intricate data relationships into understandable components, making complex database concepts accessible to all involved parties.

Standardized Documentation

- Offers a standardized, consistent approach to documenting database structures, aiding in design clarity and future modifications.

ER Diagrams in Decision Making

Incorporating Domain Knowledge

- ER diagrams incorporate domain knowledge, which includes business rules and real-world correlations, ensuring that the database design aligns with organizational needs and real-world scenarios.

Facilitating Informed Decisions

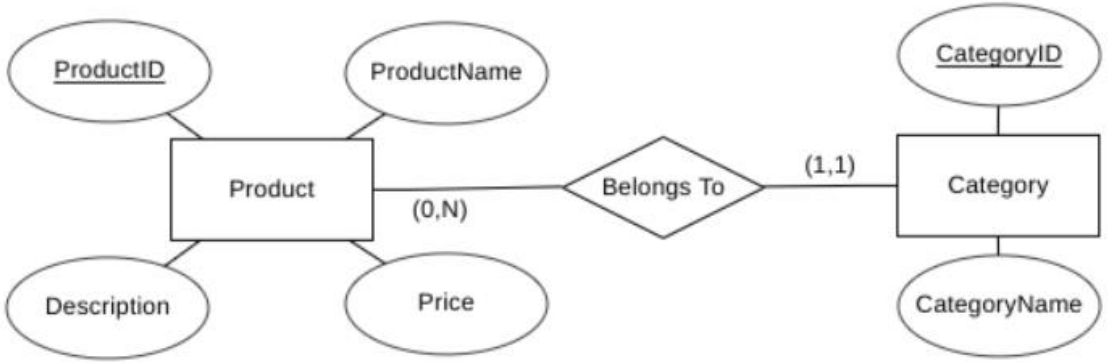
- They help stakeholders visualize complex data structures, making it easier to understand and make informed decisions about database design and functionality.

Reflecting Business Rules

- ER diagrams represent the rules and constraints of the business domain, playing a crucial role in ensuring that the database supports business processes effectively.

Recap and Key Takeaways

- ERDs are essential for visualizing and designing database structures, highlighting entities, attributes, and inter-entity relationships, facilitating clear communication in development.
- Cardinality defines relationship types between entities, while primary and foreign keys in ERDs ensure data uniqueness and consistent referencing, critical for database integrity.



- **Purpose:** This relationship allows for a flexible categorization system where each product isn't limited to a single category, reflecting the real-world scenario.

Preparing for Design Theory:

- Next Lecture Preview:
  - Explore 'What Makes A Good Design?'.
  - Understand Modification Anomalies and how to systematically identify them using Functional Dependencies.

How would we modify an ERD for a bookstore to include both online and in-store sales, considering aspects like inventory management and customer preferences?