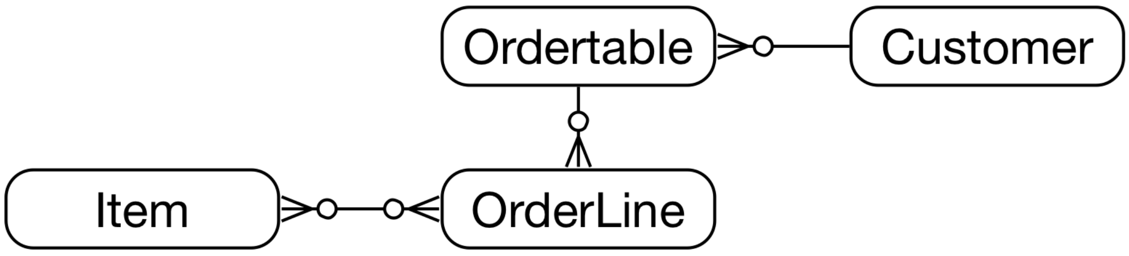
Example: Data Model

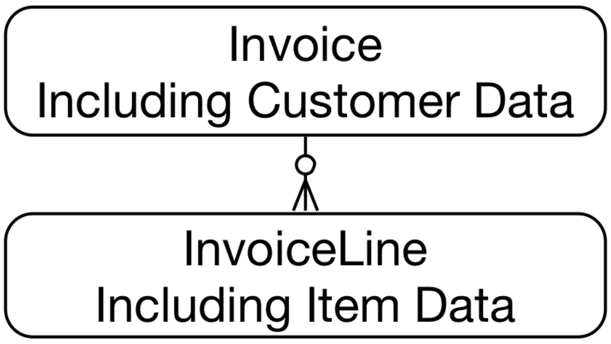
In this lesson, we'll look at the upcoming coding example from the perspective of its data model.

Data model for the database



Data Model in the System microservice-kafka-order

The database of the order microservice (see the drawing above) contains a table for the orders (`Ordertable`) and the individual items in the orders (`OrderLine`). Goods (`Item`) and customers (`Customer`) also have their own tables.



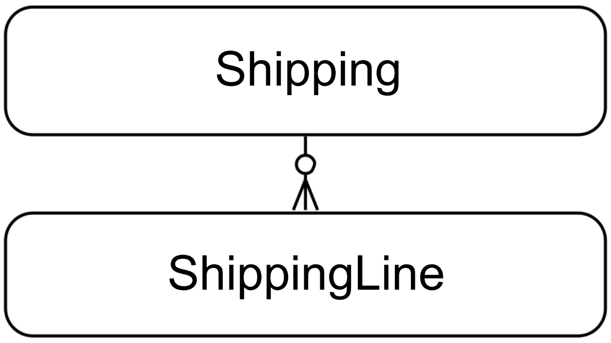
Data Model in the System microservice-kafka-order

In the microservice **microservice-kafka-invoice**, the tables for customers and items are missing. The customer data is stored as part of `invoice`, and the items are missing. The customer data is stored as part of `invoice`, and the item data as part of `invoiceLine` (see the drawing above). The data in the

tables are copies of the customers' and items' data at the time when the order was transferred to the system.

This means that **if a customer changes their data or a product changes its price, it does not affect the previous invoices**. That is correct from a domain logic perspective. After all, a price change should not affect invoices that have already been written.

Otherwise, getting the correct price and customer information at the time of the invoice can be implemented only with a complete history of the data, which is quite complex. With the model used here, it is **also very easy to transfer discounts or special offers to the invoice**. It is necessary to send a different price for a product.



Data Model in the System microservice-kafka-shipping

For the same reason, the microservice **microservice-kafka-shipping** has only the database tables `Shipping` and `ShippingLine`. Data for customers and items is copied to these tables so that the data is stored there as it was when the delivery was triggered.

This example illustrates how *bounded context* simplifies the domain models.

Inconsistencies

The example also shows another effect: the information in the system can be inconsistent. Orders without invoices or orders without deliveries can occur, but such conditions are not permanent. At some point the Kafka topic will be read out with the new orders, and the new orders will then generate an invoice and a delivery.

QUIZ

1 How many tables does the order microservice’s database schema have?

COMPLETED 0%

1 of 3

In the next lesson, we’ll look at this coding example in detail.