The Bike Peddler

Construction of Conceptual and Logical Data Models

Eoin Fennessy - 20100018

Higher Diploma in Computer Science, SETU

22 November 2022

Table of Contents

System Description

Business Overview

The Bike Peddler is a large cycling store based in Waterford City that specialises in the sale of a broad range of bicycles, clothing, and cycling accessories to customers throughout the country. The company also offers bike repair & maintenance services, as well as a bespoke bicycle assembly service.

Due to the recent spike in popularity that cycling has experienced in Ireland, the business has grown rapidly, and the owners would now like to introduce a database system to keep records of the likes of customers, orders, products, employees, and their relationships.

Object Relationships

Each **employee** of the company works for one **department** at a time. An employee can change departments and may end up working for the same department more than once during their employment with the company. All past employee-department data should be retained for analysis.

All **employees** are **salaried**. Employees should see increases in their salaries over time and will never have their salaries cut. All past employee-salary data should be stored.

An **employee** may manage **other employees**, and an employee may or may not have one (and only one) manager.

An **employee** will be assigned (at time of order) to pick an **order**. Each order is picked by one employee only, and each employee will pick zero or many orders.

Customers may or may not be designated **employees** to support them in their interactions with the company. Employees are designated zero, one, or many customers to support.

Each **customer** will have made at least one **order** with the company, and every order is made by a single customer.

A **customer** can make **enquiries** about individual **products**, and the company would like to record these enquiries and the time at which they happen.

An **order** is made up of one or more **products**, and products may appear on multiple orders. A new product may not yet have been ordered by any customers.

A **product** may be composed of **other products**. A product may be used to compose part of many other products.

Object Attributes

Employee details to be recorded include full name, gender (optional), address, email address, and one, two, or three phone numbers (a multi-valued attribute). Additionally, the company would like to record each employee's PPS number, contract type, all the departments they have worked in, including the start & finish dates for each (a many-to-many association with history), and the identity of their current manager if they have one (a recursive one-to-many relationship).

Customer details to be recorded include full name, gender (optional), address, email address, and one, two, or three phone numbers. The company would also like to keep track of any organisation/business the customer is associated with, their VAT number (if applicable), and their loyalty status (bronze, silver, or gold). The identity of the support employee should also be stored if the customer has been designated one. A large number of the business' employees are also loyal customers, so care should be taken to avoid possible duplication of data (by inheriting properties from a super-type).

For each **order**, the database should keep a record of the order number, total price (a derived/calculated field), date and time of order, and order status (pending, picked, out for delivery, delivered, or cancelled). The identity of the customer who placed the order should be recorded, as well as all the products (including quantity for each) the order contains, and the employee assigned to pick the order.

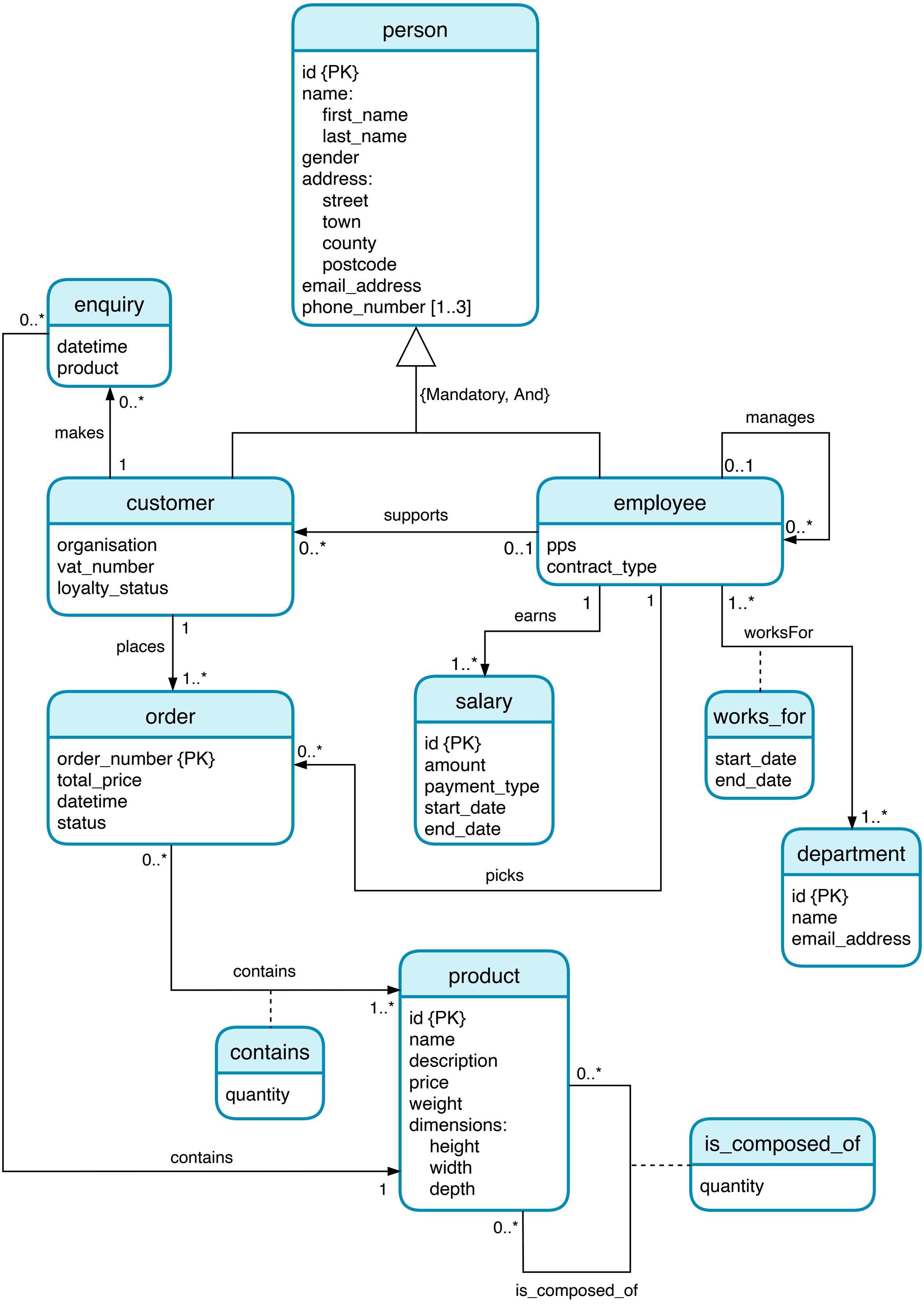
The **product** entity should contain a product ID, name, short description, price, weight, and all dimensions for the product. Any sub-products a product is composed of as well as the quantity of each sub-product should also be stored (a recursive many-to-many relationship).

Each **enquiry** a customer makes should be identified by the customers id and the date & time of the enquiry. The product the customer has enquired about should also be recorded. This is a weak entity.

Department details to be recorded include the department's id, name, and email address.

Each **salary** has a unique id, amount, payment type (weekly, fortnightly, or monthly), start date and end date. Each entry should also include the identity of the person who earns or has earned the salary.

Enhanced ER Diagram



Logical Model

Relationships

person(id, first_name, last_name, gender, street, town, county, postcode, email_address)
Primary key: id

phone(phone_number, person_id)
Primary key: phone_number
Foreign key: person_id, references person(id)

employee(id, pps, contract_type, manager_id)
Primary key: id
Foreign key: id, references person(id)
Foreign key: manager_id, references employee(id)

customer(id, organisation, vat_number, loyalty_status, support_employee_id)
Primary key: id
Foreign key: id, references person(id)
Foreign key: support_employee_id, references employee(id)

enquiry(customer_id, datetime, product_id)
Primary key: customer_id, datetime
Foreign key: customer_id, references customer(id)
Foreign key: product_id, references product(id)

order(order_number, total_price, datetime, status, customer_id, picker_id)
Primary key: order_number
Foreign key: customer_id, references customer(id)
Foreign key: picker_id, references employee(id)

order_product(order_number, product_id, product_quantity)
Primary key: order_number, product_id
Foreign key: order_number, references order(order_number)
Foreign key: product_id, references product(id)

product(id, name, description, price, weight, height, width, depth)
Primary key: id

product_subproduct(product_id, subproduct_id, subproduct_quantity)
Primary key: product_id, subproduct_id
Foreign key: product_id, references product(id)
Foreign key: subproduct_id, references product(id)

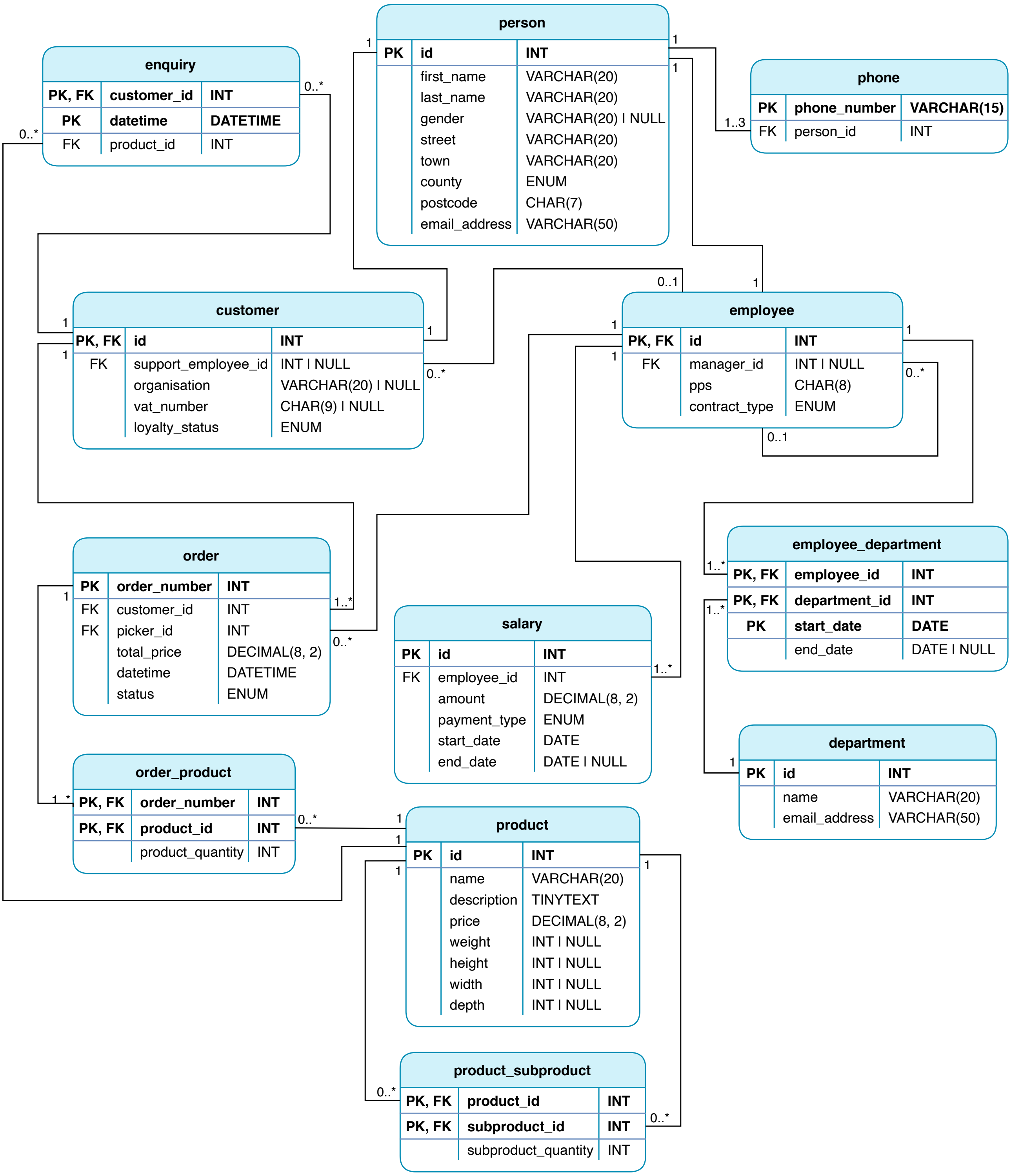
department(id, name, email_address)
Primary key: id

employee_department(employee_id, department_id, start_date, end_date)
Primary key: employee_id, department_id, start_date
Foreign key: employee_id, references employee(id)
Foreign key: department_id, references department(id)

salary(id, amount, payment_type, start_date, end_date, employee_id)
Primary key: id
Foreign key: employee_id, references employee(id)

CONTINUED ON NEXT PAGE

Logical ER Diagram



person

PK	**id**	**INT**
	first_name	VARCHAR(20)
	last_name	VARCHAR(20)
	gender	VARCHAR(20) I NULL
	street	VARCHAR(20)
	town	VARCHAR(20)
	county	ENUM
	postcode	CHAR(7)
	email_address	VARCHAR(50)

enquiry

PK, FK	**customer_id**	**INT**
PK	**datetime**	**DATETIME**
FK	product_id	INT

phone

PK	**phone_number**	**VARCHAR(15)**
FK	person_id	INT

customer

PK, FK	**id**	**INT**
FK	support_employee_id	INT I NULL
	organisation	VARCHAR(20) I NULL
	vat_number	CHAR(9) I NULL
	loyalty_status	ENUM

employee

PK, FK	**id**	**INT**
FK	manager_id	INT I NULL
	pps	CHAR(8)
	contract_type	ENUM

order

PK	**order_number**	**INT**
FK	customer_id	INT
FK	picker_id	INT
	total_price	DECIMAL(8, 2)
	datetime	DATETIME
	status	ENUM

employee_department

PK, FK	**employee_id**	**INT**
PK, FK	**department_id**	**INT**
PK	**start_date**	**DATE**
	end_date	DATE I NULL

salary

PK	**id**	**INT**
FK	employee_id	INT
	amount	DECIMAL(8, 2)
	payment_type	ENUM
	start_date	DATE
	end_date	DATE I NULL

department

PK	**id**	**INT**
	name	VARCHAR(20)
	email_address	VARCHAR(50)

order_product

PK, FK	**order_number**	**INT**
PK, FK	**product_id**	**INT**
	product_quantity	INT

product

PK	**id**	**INT**
	name	VARCHAR(20)
	description	TINYTEXT
	price	DECIMAL(8, 2)
	weight	INT I NULL
	height	INT I NULL
	width	INT I NULL
	depth	INT I NULL

product_subproduct

PK, FK	**product_id**	**INT**
PK, FK	**subproduct_id**	**INT**
	subproduct_quantity	INT