**A case study: Internet Book Store**



CREATE TABLE Books (

isbn CHAR ( 10) ,

title CHAR(80) ,

author CHAR(80),

qty\_in-stock INTEGER,

price REAL,

year\_published INTEGER,

PRIMARY KEY (isbn))

CREATE TABLE Orders (

isbn CHAR (10) ,

cid INTEGER,

cardnum CHAR (16) ,

qty INTEGER,

order\_date DATE,

ship\_date DATE,

PRIMARY KEY (isbn,cid),

FOREIGN KEY (isbn) REFERENCES Books,

FOREIGN KEY (cid) REFERENCES Customers)

CREATE TABLE Customers (

cid INTEGER,

cname CHAR(80),

address CHAR(200),

PRIMARY KEY (cid)

Reviewer: What if a customer places two orders for the same book in one day?

Designer: The first order is handlecl by creating a new Orders relationship and the second order is handled by updating the value of the quantity attribute in this relationship.

Reviewer: What if a customer places two orders for different books in one day?

Designer: No problem. Each instance of the Orders relationship set relates the customer to a different book.

Reviewer: Ah, but what if a customer places two orders for the same book on different days?

Designer : We can use the attribute order date of the orders relationship to distinguish the two orders.

Reviewer : Oh no you can't. The attributes of Customers and Books must jointly contain a key for Orders. So this design does not allow a customer to place orders for the same book on different days.

Designer: you are right. Adding order\_date to the key can solve this problem, i.e., PRIMARY KEY (isbn, cid, order\_date),

The owner of B&N now brings up some additional requirements:

* Customers should be able to purchase several different books in a *single* order: for example, three copies of 'The English Teacher' and two copies of 'The Character of Physical Law’.
* As soon as we have enough copies of an ordered book we ship it: the three copies of 'The English Teacher' are shipped today because we have five copies in stock, but 'The Character of Physical Law' is shipped tomorrow, because we currently have only one copy in stock and another copy arrives tomorrow.
* In addition, customers could place more than one order per day, and they want to be able to identify the orders they placed.

To accommodate these requirements, add a new attribute into the Orders table called ordernum, which uniquely identifies an order and therefore the customer placing the order. However, since several books could be purchased in a single order, ordernum and isbn are both needed to determine qt.y and ship\_date in the Orders table.

CREATE TABLE Orders (

ordernum INTEGER,

isbn CHAR(10),

cid INTEGER,

cardnum CHAR (16) ,

qty INTEGER,

order\_date DATE,

ship\_date DATE,

PRIMARY KEY (ordernum, isbn),

FOREIGN KEY (isbn) REFERENCES Books

FOREIGN KEY (cid) REFERENCES Customers)

In addition, employees should not see customer’s credit card information, the latter is

restricted to B&N's Accounting division. To address this concern, we create the following view:

