SQL EVIDENCE

```sql
Create TABLE book(
bno INT PRIMARY KEY
CHECK (bno >= 100000 and bno <= 999999),
title VARCHAR(255) NOT NULL,
author VARCHAR(255) NOT NULL,
category VARCHAR(50)
CHECK (category IN ('Science', 'Lifestyle', 'Arts', 'Leisure')),
price NUMERIC NOT NULL,
sales INT DEFAULT 0
);

select bno AS Book_No,
title AS Book_Title,
author, category AS GENRE,
price AS Book_Price,
sales AS Books_Sold From Book;
```

Data Output | Messages | Notifications

book_no integer	book_title character varying (255)	author character varying (255)	genre character varying (50)	book_price numeric	books_sold integer

```sql
Create TABLE customer(
cno INT PRIMARY KEY
CHECK (cno >= 100000 and cno <= 999999),
name VARCHAR(255) NOT NULL,
address VARCHAR(255) NOT NULL,
balance NUMERIC DEFAULT 0
);

select cno AS customer_no,
name AS Customer_Name,
address AS Customer_address,
balance AS Remaining_Dues
From Customer;
```

Data Output | Messages | Notifications

customer_no integer	customer_name character varying (255)	customer_address character varying (255)	remaining_dues numeric

```sql
Create TABLE bookOrder (
cno INT,
bno INT,
orderTime TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
qty INT NOT NULL
);

select cno AS customer_no,
bno AS book_no,
orderTime,
qty AS quantity from bookOrder;
```

Data Output | Messages | Notifications

	customer_no integer	book_no integer	ordertime timestamp without time zone	quantity integer

A) Insert a new book

```
1  Insert into Book (bno, title, author, category, price) values
2  (123456, 'Python Programming', 'John Doe', 'Science', 29.99);
3
```

Data Output | Messages | Notifications

	bno [PK] integer	title character varying (255)	author character varying (255)	category character varying (50)	price numeric	sales integer
1	123456	Python Programming	John Doe	Science	29.99	0

B) Delete a Book

```
1  Delete from Book where bno = 123456;
```

Data Output | Messages | Notifications

	bno [PK] integer	title character varying (255)	author character varying (255)	category character varying (50)	price numeric	sales integer

C) Insert a Customer

```
1  Insert into Customer (cno, name, address, balance) values
2  (123456, 'Alice Smith', '123 Main St, London', 0.00);
```

Data Output | Messages | Notifications

	cno [PK] integer	name character varying (255)	address character varying (255)	balance numeric
1	123456	Alice Smith	123 Main St, London	0.00

D) Delete a Customer

```
1  Delete from customer where cno = 123456;
```

Data Output | Messages | Notifications

cno [PK] integer	name character varying (255)	address character varying (255)	balance numeric

E) Place an order for a customer for a specified number of copies of a book. The copies ordered are assumed to be sold and will have to be paid for. Books are not supplied on 'sale or return' terms business. Check if the book and customer is already in the system. If there is book entry or customer information is not available then create required entries and then perform the operation.

i) Initial values in the Tables Customer and Book.

```
2  select cno as CustomerNo, name, address, balance from Customer;
```

Data Output | Messages | Notifications

customerno integer	name character varying (255)	address character varying (255)	balance numeric

```
2  select bno as BookNo, title as Book_Title, author, category, price, sales from book;
```

Data Output | Messages | Notifications

bookno integer	book_title character varying (255)	author character varying (255)	category character varying (50)	price numeric	sales integer

```sql
Create Or Replace Function handleorder()
Returns Trigger AS $$
DECLARE
   bookPrice NUMERIC;
   bookExists Boolean;
   customerExists Boolean;
BEGIN
   IF NOT EXISTS(Select 1 from Book where bno = NEW.bno)then
      Insert Into Book(bno, title, author, category, price, sales) values
      (New.bno, 'Gulliver Travels', 'Jonathan Swift', 'Leisure', 50.00, NEW.qty);
   End IF;
   SELECT price INTO bookPrice
   FROM Book
   WHERE bno = NEW.bno;
   IF NOT EXISTS(Select 1 from Customer where cno=NEW.cno) then
      Insert Into Customer(cno, name, address, balance) values
      (NEW.cno, 'ABC', '456 Main St, London', 0.00);
   End IF;

   Update customer
   SET balance = balance - (bookPrice*NEW.qty)
   where cno = NEW.cno;

   Return NEW;
END;
$$ LANGUAGE plpgsql;


CREATE Trigger handle_order_trigger
AFTER INSERT ON bookOrder
For EACH ROW
EXECUTE Function handleorder();
```

```sql
Insert into bookOrder (bno, cno, qty) values
(123456, 678901, 10);
Select bno as Book_No, cno as Customer_No, qty as Quantity from bookOrder;
```

Data Output | Messages | Notifications

	book_no integer	customer_no integer	quantity integer
1	123456	678901	10

ii) Updated Book Table and Customer Table after inserting.

```sql
Select bno as Book_No, title as Book_title, author, category, price, sales from book;
```

Data Output | Messages | Notifications

	book_no integer	book_title character varying (255)	author character varying (255)	category character varying (50)	price numeric	sales integer
1	123456	Gulliver Travels	Jonathan Swift	Leisure	50.00	10

```
2  Select cno as Customer_No, name, address, balance from Customer;
```

Data Output | Messages | Notifications

	customer_no integer	name character varying (255)	address character varying (255)	balance numeric
1	678901	ABC	456 Main St, London	-1000.00

F) Record a payment by a customer. The payment is subtracted from the customer's balance.

```
11
12  Create or Replace Function record_payment(customer_Number integer, payment numeric)
13  RETURNS VOID AS $$
14  BEGIN
15     Update customer
16     SET balance = balance + payment
17     WHERE cno = customer_Number;
18  END;
19  $$ LANGUAGE plpgsql;
20
21  SELECT record_payment(678901, 500);
22
23  select cno as Customer_Number,
24  name as Customer_Name,
25  address,
26  balance as Remaining_Payment from customer;
27
28
29
30
31
32
33
34
```

Data Output | Messages | Notifications

	customer_number integer	customer_name character varying (255)	address character varying (255)	remaining_payment numeric
1	678901	ABC	456 Main St, London	-500

G) Find details of customers who have current orders for a book with a given text fragment in the book title. For example, find customers with orders for books with 'Python' in the title. This transaction produces a report with lines showing the full title of a book ordered, the customer name and the customer address relevant to the order. The report is to be sorted by title and then by customer name.

```
1  Select b.title as Book_Title,
2  c.name as Customer_Name,
3  c.address as Customer_Address
4  From Book b
5  Join bookOrder bo on b.bno = bo.bno
6  Join Customer c on bo.cno = c.cno
7  WHERE b.title like '%Python%'
8  group by (c.name, c.address, b.title)
9  order by b.title , c.name;
10
```

Data Output | Messages | Notifications

	book_title character varying (255)	customer_name character varying (255)	customer_address character varying (255)
1	Learn Python Basics	JKL	123 Central St, London
2	Learn Python Basics	ABC	456 Main St, London

H) Find details of books ordered by a specified customer. The report will show the name of the customer followed by, for each book, the book number, title and author, sorted by book number.

```sql
SELECT
    distinct(c.name) AS customer_name,
    b.bno AS book_number,
    b.title AS book_title,
    b.author AS book_author
FROM
    customer c
JOIN
    bookOrder bo ON c.cno = bo.cno
JOIN
    book b ON bo.bno = b.bno
WHERE
    c.cno = 678901
ORDER BY
    b.bno;
```

Data Output | Messages | Notifications

	customer_name character varying (255)	book_number integer	book_title character varying (255)	book_author character varying (255)
1	ABC	133456	Learn Python Basics	John Doe

I) Produce a book report by category. This report shows, for each category, the number of books sold and the total value of these sales. The total value calculation assumes that the currently held price is used and any earlier changes in the price of a book since it was inserted into the database are ignored.

```sql
Select category,
Sum(sales) AS total_books_sold,
Sum(sales * price) AS total_sales_value
FROM book
Group By category
Order By category;
```

Data Output | Messages | Notifications

	category character varying (50)	total_books_sold bigint	total_sales_value numeric
1	Leisure	20	1000.00
2	Science	5	250.00

J) Produce a customer report. This report shows, for each customer, the customer number, customer name and a count of the number of copies of books on order (if any). This report is to be in customer number order.

```sql
SELECT
    c.cno AS customer_number,
    c.name AS customer_name,
    SUM(bo.qty) AS total_copies_on_order
FROM
    customer c
JOIN
    bookOrder bo ON c.cno = bo.cno
GROUP BY
    c.cno, c.name
ORDER BY
    c.cno;
```

Data Output | Messages | Notifications

	customer_number integer	customer_name character varying (255)	total_copies_on_order bigint
1	678901	ABC	4
2	688901	JKL	1