You are required to design a database for a book publishing company. The database needs to store a table of authors and books. An author has many books.

Table definitions are as follows:

|  |  |
| --- | --- |
| authors | id SERIAL, name varchar(50) |
| books | Id SERIAL, title varchar(100), author\_id |
| Reviews | Id SERIAL, rating INTEGER, reviewer\_id INTEGER, book\_is  Note: reviewer\_id is nothing but authors id |

Write a query that will join together these two tables.

1. For each book, print the title of the book and name of the author.

SELECT title,name

FROM books

JOIN authors on books.author\_id=authors.id;

1. Return title of each book, along with the name of the author. All authors should be included, even if they don’t have a book associated with them.

**(You need to provide two possible solutions using Joins)**

1. SELECT title,name

FROM books

RIGHT JOIN authors on books.author\_id=authors.id;

1. SELECT title,name

FROM authors

LEFT JOIN books on authors.id=books.author\_id;

1. Return title of each book, along with the name of the author, and the rating of a review. Only show rows where the author of the book is also the author of the review.

SELECT title, name, rating

FROM reviews

JOIN books ON reviews.bool\_id=books.id

JOIN authors ON books.author\_id=authors.id

WHERE author\_id=reviewer-id;

1. Prints an author’s id and the number of books they have authored.

**(Check do you need two tables for this query).**

SELECT author\_id,Count(\*) AS numberOfBooks

FROM books

GROUP BY author\_id;

1. Print an author’s name and the number of books they have authored.

SELECT authors.id,count(\*)

FROM authors

JOIN books on authors.id=books.author\_id

GROUP BY authors.id;

Table details are as follows:

|  |  |
| --- | --- |
| Phones | name Varchar(50), manufacturer varchar(100), price INTEGER, units\_sold INTEGER |

1. Write a query that prints the name of manufacturers and total revenue (price \* units\_sold) for all phones. Only print the manufacturers who have revenue greater than 2,000,00 for all phones they sold.

SELECT manufacturer,total\_revenue

FROM (SELECT SUM (price \* units\_sold) AS total\_revenue,manufacturer

FROM phones

GROUP BY manufacturer

HAVING SUM (price \*units\_sold)>100000 ) AS FIND\_REV

1. Write a query that shows the names of only the second and third most expensive phones.

SELECT name FROM phones

ORDER BY price DESC

LIMIT 2

1. Write a query that will print the manufacturer of phones where the phone’s price is less than 170. Also print all manufaturer that have created more than two phones.

SELECT manufacturer,numPhonesLessThan170

FROM (SELECT manufacturer,count(\*) AS numPhonesLessThan170

FROM (SELECT \* FROM phones

WHERE price<170) AS priced\_find

GROUP BY manufacturer) AS count\_find

1. Write a query that prints the name and price for each phone. In addition, print out the ratio of the phones price against max of all prices. Rename this third column to price\_ratio.

SELECT name, price

(CAST(price AS double precision) / (SELECT max(price) FROM phones)) AS price\_ratio

FROM phones

1. Write a query that will print the name and price of all phones that sold greater than 5000 units.

SELECT name, price

FROM phones

WHERE units\_sold>5000

1. Write a query that will select the name and manufacturer for all phones created by Apple or Samsung.

SELECT name, price, manufacturer

FROM phones

WHERE manufacturer= ‘Apple’ OR manufacturer=’Samsung’

1. Write a query that will print the name and total\_revenue of all phones with a total\_revenue greater than 1,000,00.

SELECT name,total\_revenue

FROM ( SELECT SUM ( price \* units\_sold) AS total\_revenue, name

FROM phones

GROUP BY name

HAVING SUM (price \* units\_sold) >1000000) AS find\_revene

Create a new dataset from the shared product-orders dataset sheet.

Write the following queries:

1. Print the number of paid and unpaid orders.

SELECT paid,count(\*)

FROM orders

GROUP BY paid

1. Print the first\_name and last\_name of each user along with the whether they have paid for their order.

SELECT first\_name, last\_name,paid

FROM users

JOIN orders ON users.id=orders.user\_id