# Storage Layer Assignment

1. **Table Definitions:**

CREATE TABLE authors(

id SERIAL,

name VARCHAR(50)

);

CREATE TABLE books(

id SERIAL,

title VARCHAR(100),

author\_id INT

);

CREATE TABLE reviews(

id SERIAL,

rating INT,

reviewer\_id INT,

book\_id INT

);

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

SELECT books.title, authors.name

FROM books, authors

WHERE 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.**

SELECT books.title, authors.name

FROM books

LEFT JOIN authors

ON books.author\_id = authors.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 books.title, authors.name, reviews.rating

FROM books

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

JOIN reviews ON books.id = reviews.book\_id;

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

SELECT authors.id, COUNT(books.id)

FROM authors, books

GROUP BY books.author\_id

WHERE authors.id = books.author\_id;

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

SELECT authors.name, COUNT(books.id)

FROM authors, books

GROUP BY books.author\_id

WHERE authors.id = books.author\_id;

1. **Table Definitions:**

CREATE TABLE phones(

name VARCHAR(50),

manufacturer VARCHAR(100),

price INT,

units\_sold INT

);

* 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, (price \* units\_sold) as total\_revenue

FROM phones

WHERE (price \* units\_sold) > 20000;

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

SELECT name

FROM phones

ORDER BY price

LIMIT 2 OFFSET 1;

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

SELECT manufacturer  
FROM phones

WHERE price < 170

OR HAVING COUNT (manufacturer > 2)

GROUP BY manufacturer;

* 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, (price::float / (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, manufacturer

FROM phones

WHERE manufacturer=“Samsung” OR manufacturer=“Apple”;

* 1. **Write a query that will print the name and total\_revenue of all phones with a total\_revenue greater than 100,000.**

SELECT name, (price\*units\_sold) as total\_revenue

FROM phones

WHERE price\*units\_sold > 100000;

1. **Dataset**
   1. **Print the number of paid and unpaid orders.**

SELECT COUNT(\*) filter (where paid) AS paid, COUNT(\*) filter (where NOT paid) AS unpaid

FROM orders;

* 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 orders.user\_id = users.id;