Part 3.

|  |  |
| --- | --- |
| NAME | ID |
| Ibrahim alqurshi | 2046430 |
| Moath mosleh alsolami | 1935160 |
| Omer saeed al-zahrani | 2046430 |
| Fahad Hamad alsifri | 1743998 |
| Abdullah justaniah | 2037017 |
| Section: ZR | |

**The aim of this part is to do the following**

1. **Practicing writing queries in relational algebra.**
2. **Query your database by using the SQL query language**

**What to do as Part 3.**

1. **Based on the 10 queries that you wrote in the first part of this project, try to express them in algebra form.**
2. **Write those queries by using the SQL query language.**
3. **Write new queries that contains more complex witch should include the following**

1. **Selection from different tables at one time.**
2. **Aggregation by using group by and having statement.**
3. **Nesting with aggregation.**
4. **Nesting involving NOT IN or NOT EXISTS**
5. **Outer join**

# Some of the queries from Part 1 cannot be expressed in Algebra form . here I wrote queries that can be convert to algebra

-Select **EmpID** from Employee

π *empid* *employee*

-Select Fname from Employee where salary >9000

π *fname* σ *salary* > 9000 *employee*

-Select \* from Employee where address = ‘Jeddah’;

σ *address* = "Jeddah" *employee*

-Select Fname from Employee where Address = ‘Jeddah’ AND Salary > 8000;

π *fname* σ *address* = "Jeddah" *AND* *salary* > 8000 *employee*

* select \* from warehouse where capacity < 30;

σ *capacity* < 30 *warehouse*

* Select \* from management department where income > 10000 AND Expense <

9000;

σ *income* > 10000 *AND* *expense* < 9000

ρ *department* *management*

Select Receipt, PaymentID from payment

π *receipt*, *paymentid* *payment*

***Writing Queries using Sql query language***

- insert into Employee values( 20001 , ‘ibrahim’ ,’ayman’ , ‘alqurshi’ , ‘157706123’, ‘Jeddah’ , ‘054994705’ , 10000);

- insert into Employee values( 20002 , ‘Omar’ ,’mohamed’ , ‘alzharani’ , ‘299847123’, ‘Jeddah’ , ‘0555355705’ , 9000);

- Select EmpID from Employee;

- Select Fname from Employee where salary >9000;

- Select \* from Employee where address = ‘Jeddah’;

- Select Fname from Employee where Address = ‘Jeddah’ AND Salary > 8000;

-Insert into product values(1002 , ‘Vintage Hoodie M’ , 150 );

-Insert into product values(1011 , ‘Air Jordan Shoes 43’ , 400 );

-Insert into product values(10021 , ‘Vintage T-shirt M’ , 90 );

-update product price = 120 where PID = 10021;

-Describe Customers;

-select \* from warehouse where capacity < 30;

-Describe Warehouse;

-Select \* from management department where income > 10000 AND Expense < 9000;

-Select Receipt, PaymentID from payment;

Complex queries

* 1. **Selection from different tables at one time.**

Shows information of employees who have dependents, as well as their dependent's information

select employee.emp\_id AS "Employee ID" , emp\_fname AS "First Name" , emp\_lname AS "Last Name" , dependent\_fname AS "Dependent First Name" , dependent\_lname AS "Dependent Last Name" from employee, dependent where employee.emp\_id = dependent.emp\_id;

Shows each Order’s information: Customer Phone Number, Order Number , Product(s) ID , as well as quantity of each order

select customer\_phone AS "Customer Phone" , order.orderID AS "Order Num." , productID as "Product ID" , quantity as "Quantity" from order, productinorder where order.orderID = productinorder.orderID;

* 1. **Aggregation by using group by and having statement.**

Shows the item quantity of each order

select orderID AS "Order Number" , count(productID) AS "Item Quantity" from productinorder group by orderID;

Count Number of employees in each warehouse

select count(emp\_id) AS "Employee Count" , emp\_warehouseID AS "Warehouse ID" from employee group by emp\_warehouseID;

Selects employees whose salary falls above 10000

select emp\_id as "Employee ID", max(emp\_salary) as "Salary (Above 10,000)" from employee group by emp\_id having max(emp\_salary) > 10000;

Selects employees whose salary falls below 7000

select emp\_id as "Employee ID", max(emp\_salary) as "Salary (Below 7000)" from employee group by emp\_id having max(emp\_salary) <= 7000;

Shows Orders which have a quantity of 3 or more of the same product, and lists the product ID

select orderID AS "Order Number", productID AS "Product ID", max(quantity) AS "Quantity (Over 3 Items)" from productinorder group by orderID, productID having max(quantity) >= 3;

1. **Nesting with aggregation**.

Shows highest salary average between warehouses

select max(avg(emp\_salary)) AS "Highest Sal. Avg. (Warehouse)" from employee group by emp\_warehouseID;

Shows which warehouse has the highest count of products stored in it

select max(count(productID)) AS "MAX PRDCT COUNT (Warehouse)" from product group by warehouseID;

Shows the lowest number of items a supplier has supplied

select min(count(supID)) AS "Lowest Item # Supplied" from suppliersupplies group by supID;

1. **Nesting involving NOT IN or NOT EXISTS**

Shows information of employees who have no dependents

select employee.emp\_id AS "Employee ID" , emp\_fname AS "First Name" , emp\_lname AS "Last Name", emp\_ssn AS "SSN (No dependent)" from employee where not exists(select emp\_id from dependent where employee.emp\_id = dependent.emp\_id);

1. **Outer join**

Display each warehouse and its finance department

select warehouse.WhID AS "Warehouse ID" , warehouse.capacity AS "Warehouse Capacity", ManagmnetdeptBudgetsWarehouse.dept\_id as "Managment DeptID" from warehouse, ManagmnetdeptBudgetsWarehouse where warehouse.WhID(+) = ManagmnetdeptBudgetsWarehouse.WarehouseID;

Display customer which have made payments

select payment.customer\_phone AS "Customer Phone" , payment.paymentid AS "Payment ID" , customer.customer\_fname as "First Name", customer.customer\_mname as "Middle Name" , customer.customer\_lname AS "Last Name" from payment, customer where payment.customer\_phone = customer.customer\_phone(+);