Analytics Avenue for Research and Development

*-In the journey of empowering the digital minds*

**WORKSHEET-2**

1. **Basic SELECT Query Table: customers**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| customer\_id | INT | Unique ID for each customer |
| customer\_name | VARCHAR(100) | Name of the customer |
| phone\_number | VARCHAR(15) | Phone number of the customer |
| city | VARCHAR(50) | City where the customer lives |
| balance | DECIMAL(10,2) | Account balance of the customer |

**Question:**

Write an SQL query to retrieve all customer names and their phone numbers from the customers table.

ANSWER:

SELECT customer\_name, phone\_number

FROM customers

1. **WHERE Clause Table: orders**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| order\_id | INT | Unique ID for each order |
| customer\_id | INT | ID of the customer who placed the order |
| order\_date | DATE | Date when the order was placed |
| total\_amount | DECIMAL(10,2) | Total amount of the order |

**Question:**

Write an SQL query to find all orders placed after January 1, 2023, with a total amount greater than $100 from the orders table.

ANSWER:

SELECT order\_id

FROM ORDERS

WHERE total amount > 100 AND order date > ‘2023-01-01’;

1. **Aliases Table: products**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| product\_id | INT | Unique ID for each product |
| product\_name | VARCHAR(100) | Name of the product |
| price | DECIMAL(8,2) | Price of the product |
| stock\_quantity | INT | Available stock of the product |

**Question:**

Write an SQL query to display the product name as Item and the price as Cost from the products table.

ANSWER:

SELECT product\_name as Item, price and Cost

FROM products

1. **DISTINCT Keyword Table: payments**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| payment\_id | INT | Unique ID for each payment |
| customer\_id | INT | ID of the customer who made the payment |
| payment\_date | DATE | Date of the payment |
| payment\_method | VARCHAR(50) | Method used for payment (e.g., 'Credit Card', 'Cash', 'PayPal') |

**Question:**

Write an SQL query to find all unique payment methods from the payments table.

ANSWER:

Select DISTINCT(payment\_method) as Payment\_method

From payments

1. **ORDER BY**

**Table: customers**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| customer\_id | INT | Unique ID for each customer |
| customer\_name | VARCHAR(100) | Name of the customer |
| balance | DECIMAL(10,2) | Account balance of the customer |

**Question:**

Write an SQL query to retrieve customer names and their account balance from the customers table and sort the result by balance in descending order.

ANSWER:

SELECT customer\_name as name, balance

FROM customers

ORDER BY balance desc;

1. **COUNT and GROUP BY**

**Table: orders**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| order\_id | INT | Unique ID for each order |
| customer\_id | INT | ID of the customer who placed the order |
| order\_date | DATE | Date when the order was placed |
| total\_amount | DECIMAL(10,2) | Total amount of the order |

**Question:**

Write an SQL query to find the number of orders placed by each customer from the orders table.

ANSWER:

SELECT customer\_id, COUNT(order\_id) as Total\_orders

FROM customers

Group BY customer\_id;

1. **HAVING Clause Table: order\_items**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| order\_item\_id | INT | Unique ID for each order item |
| order\_id | INT | ID of the order that contains the item |
| product\_id | INT | ID of the product that was ordered |
| quantity | INT | Number of units ordered |
| price\_per\_unit | DECIMAL(8,2) | Price per unit of the product |

**Question:**

Write an SQL query to find products that have been ordered more than 20 times in total from the order\_items table.

ANSWER:

SELECT Product\_id, COUNT(order\_id)  
FROM customers

HAVING COUNT(order\_id) > 20;

1. **Aggregate Functions (SUM, AVG, MAX, MIN) Table: orders**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| order\_id | INT | Unique ID for each order |
| customer\_id | INT | ID of the customer who placed the order |
| order\_date | DATE | Date when the order was placed |
| total\_amount | DECIMAL(10,2) | Total amount of the order |

**Question:**

Write an SQL query to find the total, average, maximum, and minimum order amount from the orders table.

ANSWER:

SELECT SUM(total\_amount), AVG(total\_amount), Min(total\_amount), MAX(total\_amount)  
FROM ORDERS;

1. **LIMIT and OFFSET**

**Table: products**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| product\_id | INT | Unique ID for each product |
| product\_name | VARCHAR(100) | Name of the product |
| price | DECIMAL(8,2) | Price of the product |

**Question:**

Write an SQL query to retrieve the top 3 most expensive products from the products table.

ANSWER:

SELECT product\_name, price

FROM products

ORDER BY price DESC  
LIMIT 3;

1. **LIKE Operator Table: customers**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| customer\_id | INT | Unique ID for each customer |
| customer\_name | VARCHAR(100) | Name of the customer |
| phone\_number | VARCHAR(15) | Phone number of the customer |

**Question:**

Write an SQL query to find all customers whose name starts with 'J' in the customers table.

ANSWER:

SELECT customer\_name

FROM customers

Where customer\_name LIKE ‘j%’;

1. **BETWEEN Operator**

**Table: orders**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| order\_id | INT | Unique ID for each order |
| customer\_id | INT | ID of the customer who placed the order |
| order\_date | DATE | Date when the order was placed |
| total\_amount | DECIMAL(10,2) | Total amount of the order |

**Question:**

Write an SQL query to retrieve all orders placed between January 1, 2023, and March 31, 2023, from the orders table.

ANSWER:

SELECT order\_id

FROM orders

Where order\_date Between ‘2023-01-01’ and ‘2023-03-31’;

1. **ORDER BY with LIMIT and OFFSET**

**Table: employees**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| employee\_id | INT | Unique ID for each employee |
| employee\_name | VARCHAR(100) | Name of the employee |
| salary | DECIMAL(8,2) | Salary of the employee |
| department | VARCHAR(50) | Department where the employee works |

**Question:**

Write an SQL query to retrieve the 2nd and 3rd highest salaries from the employees table.

SELECT salary

FROM employees

ORDER BY salary DESC

OFFSET 1 LIMIT 2;

1. **LIKE with Wildcards Table: customers**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| customer\_id | INT | Unique ID for each customer |
| customer\_name | VARCHAR(100) | Name of the customer |
| email | VARCHAR(100) | Email address of the customer |

**Question:**

Write an SQL query to retrieve all customers whose email address ends with '@gmail.com'.

Select customer\_name

FROM customers

WHERE email LIKE ‘%@%.%’;

1. **NOT LIKE Operator Table: products**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| product\_id | INT | Unique ID for each product |
| product\_name | VARCHAR(100) | Name of the product |
| price | DECIMAL(8,2) | Price of the product |

**Question:**

Write an SQL query to find products whose names do not contain the word 'Pro' in the products table.

SELECT Product name

FROM products

WHERE product\_name NOT LIKE ‘%pro%’;

1. **Using Functions (UPPER, LOWER) Table: employees**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| employee\_id | INT | Unique ID for each employee |
| employee\_name | VARCHAR(100) | Name of the employee |
| email | VARCHAR(100) | Email of the employee |
| position | VARCHAR(50) | Employee's position in the company |

**Question:**

Write an SQL query to display the email addresses of all employees in lowercase letters.

SELECT LOWER(employee\_name)

FROM employees

1. **WHERE with Multiple Conditions (AND) Table: products**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| product\_id | INT | Unique ID for each product |
| product\_name | VARCHAR(100) | Name of the product |
| price | DECIMAL(8,2) | Price of the product |
| stock\_quantity | INT | Available stock of the product |

**Question:**

Write an SQL query to find all products with a price greater than $50 and a stock quantity less than 100.

ANSWERS:

SELECT product\_name

FROM products

WHERE price > 50 and stock\_quantity <100;

1. **WHERE with Multiple Conditions (OR) Table: orders**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| order\_id | INT | Unique ID for each order |
| customer\_id | INT | ID of the customer who placed the order |
| order\_date | DATE | Date when the order was placed |
| total\_amount | DECIMAL(10,2) | Total amount of the order |

**Question:**

Write an SQL query to retrieve all orders where the total amount is less than $100 or the order was placed before January 1, 2023.

SELECT order\_id

From Orders

WHERE total\_amount <100 or ORDER\_date < ‘2023-01-01’;

1. **IS NULL**

**Table: employees**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| employee\_id | INT | Unique ID for each employee |
| employee\_name | VARCHAR(100) | Name of the employee |
| salary | DECIMAL(8,2) | Salary of the employee |
| department | VARCHAR(50) | Department where the employee works |
| bonus | DECIMAL(8,2) | Bonus amount for the employee (can be NULL) |

**Question:**

Write an SQL query to find all employees who have not been assigned a bonus (i.e., bonus is NULL).

Select employee\_name

FROM employees

WHERE is NULL;

1. **IS NOT NULL**

**Table: employees**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| employee\_id | INT | Unique ID for each employee |
| employee\_name | VARCHAR(100) | Name of the employee |
| salary | DECIMAL(8,2) | Salary of the employee |
| department | VARCHAR(50) | Department where the employee works |
| bonus | DECIMAL(8,2) | Bonus amount for the employee (can be NULL) |

**Question:**

Write an SQL query to find all employees who have been assigned a bonus.

Select employee\_name

FROM employees

WHERE bonus IS NOT NULL

1. **Mathematical Operations Table: products**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| product\_id | INT | Unique ID for each product |
| product\_name | VARCHAR(100) | Name of the product |
| price | DECIMAL(8,2) | Price of the product |
| discount | DECIMAL(5,2) | Discount percentage on the product |

**Question:**

Write an SQL query to calculate the final price of each product after applying the discount.

Answer:

Select product\_name, price, discount, price\* (1-(discount/100)) as after\_discount

From products

1. **ROUND Function Table: products**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| product\_id | INT | Unique ID for each product |
| product\_name | VARCHAR(100) | Name of the product |

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| price | DECIMAL(8,2) | Price of the product |
| discount | DECIMAL(5,2) | Discount percentage on the product |

**Question:**

Write an SQL query to round the price of each product to the nearest whole number.

Answer:

Select round(price)

FROM products

1. **SUM with WHERE Clause Table: sales**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| sale\_id | INT | Unique ID for each sale |
| product\_id | INT | ID of the product sold |
| sale\_date | DATE | Date when the sale was made |
| sale\_amount | DECIMAL(10,2) | Amount of the sale |

**Question:**

Write an SQL query to find the total sales amount for sales made in 2023.

Select SUM(sale\_amount)

FROM sales

WHERE YEAR(sale\_date) = ‘2023’;

1. **AVG Function Table: employees**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| employee\_id | INT | Unique ID for each employee |
| employee\_name | VARCHAR(100) | Name of the employee |
| salary | DECIMAL(8,2) | Salary of the employee |
| department | VARCHAR(50) | Department where the employee works |

**Question:**

Write an SQL query to calculate the average salary of employees from the employees table.

SELECT employee\_name, AVG(salary)

FROM employees

1. **MAX Function Table: products**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| product\_id | INT | Unique ID for each product |
| product\_name | VARCHAR(100) | Name of the product |
| price | DECIMAL(8,2) | Price of the product |

**Question:**

Write an SQL query to find the most expensive product from the products table.

ANSWER:

Select product\_name, MAX(price)

FROM products

GROUP BY product\_name

Limit 1;

1. **MIN Function Table: products**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| product\_id | INT | Unique ID for each product |
| product\_name | VARCHAR(100) | Name of the product |
| price | DECIMAL(8,2) | Price of the product |

**Question:**

Write an SQL query to find the least expensive product from the products table.

ANSWER:

Select product\_name, MIN(price)

FROM products

GROUP BY product\_name

LIMIT 1;

1. **TRIM Function Table: employees**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| employee\_id | INT | Unique ID for each employee |
| employee\_name | VARCHAR(100) | Name of the employee |
| phone\_number | VARCHAR(15) | Phone number of the employee (may contain extra spaces) |

**Question:**

Write an SQL query to remove any leading and trailing spaces from the phone\_number column.

SELECT TRIM(phone\_number)

FROM employees;

1. **COUNT with WHERE**

**Table: products**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| product\_id | INT | Unique ID for each product |
| product\_name | VARCHAR(100) | Name of the product |
| price | DECIMAL(8,2) | Price of the product |

**Question:**

Write an SQL query to count how many products have a price greater than $100.

ANSWER:

SELECT COUNT(\*)

FROM products

WHERE price > 100;

1. **LENGTH Function Table: customers**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| customer\_id | INT | Unique ID for each customer |
| customer\_name | VARCHAR(100) | Name of the customer |

**Question:**

Write an SQL query to find the length of each customer's name.

SELECT customer\_name, LENGTH(customer\_name)

FROM customers

1. **MOD Function Table: employees**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| employee\_id | INT | Unique ID for each employee |
| employee\_name | VARCHAR(100) | Name of the employee |
| salary | DECIMAL(8,2) | Salary of the employee |

**Question:**

Write an SQL query to find if the salary of each employee is an even or odd number using the MOD function.

Select employee\_name, salary, CASE WHEN MOD(salary,2) = 0 THEN ‘EVEN’ ELSE ‘ODD’ END as logic

FROM employees

1. **UPPER Function Table: employees**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Data Type** | **Description** |
| employee\_id | INT | Unique ID for each employee |
| employee\_name | VARCHAR(100) | Name of the employee |
| position | VARCHAR(50) | Position of the employee |

**Question:**

Write an SQL query to display all employee names and their positions in uppercase.

Select UPPER(employee\_name) , UPPER(position)

FROM employees

**Questions on Subqueries**

1. **Basic Subquery**
   1. **Table**: employees
   2. **Columns**:
      1. employee\_id (INT): Unique ID for each employee
      2. employee\_name (VARCHAR(100)): Name of the employee
      3. salary (DECIMAL(10, 2)): Salary of the employee
   3. **Question**: Write an SQL query to find the names of employees whose salary is greater than the average salary of all employees.

Select name, salary

FROM employees

WHERE Salary > (Select AVG(salary) from Employees);

1. **Correlated Subquery**

* **Table**: orders
* **Columns**:
  + order\_id (INT): Unique ID for each order
  + customer\_id (INT): ID of the customer who placed the order
  + total\_amount (DECIMAL(10, 2)): Total amount of the order
* **Question**: Write an SQL query to find the orders with a total amount greater than the total amount of orders placed by the same customer.

Select total\_amount , customer\_id, ORDER\_id

FROM Orders o1

WHERE o1.total\_amount > (select AVG(o2.total\_amount)

FROM orders o2

JOIN o1.customer\_id = o2.customer\_id);

1. **Subquery in FROM Clause**

* **Table**: products
* **Columns**:
  + product\_id (INT): Unique ID for each product
  + product\_name (VARCHAR(100)): Name of the product
  + price (DECIMAL(8, 2)): Price of the product
* **Question**: Write an SQL query to find the average price of products that have a price greater than 50.

SELECT AVG(PG.price) AS average\_high\_price

FROM

( SELECT price

FROM Products

WHERE price > 50) AS PG;

1. **Subquery with COUNT**

* **Table**: orders
* **Columns**:
  + order\_id (INT): Unique ID for each order
  + customer\_id (INT): ID of the customer who placed the order
* **Question**: Write an SQL query to find the customers who have placed more than 3 orders.

Select customer\_id, count(order\_id)

FROM orders

Group by customer\_id

Having count(order\_id) > 3;

1. **Subquery with MIN**

* **Table**: products
* **Columns**:
  + product\_id (INT): Unique ID for each product
  + price (DECIMAL(8, 2)): Price of the product
* **Question**: Write an SQL query to find the products that have a price lower than the minimum price of any product in the electronics category.

Select product\_id

From products

Where price <(

Select Min(price) from products where category = ‘electronics’);

1. **Subquery with GROUP BY**

* **Table**: sales
* **Columns**:
  + sale\_id (INT): Unique ID for each sale
  + product\_id (INT): ID of the product sold
  + quantity (INT): Quantity sold
* **Question**: Write an SQL query to find products that have sold more than the average quantity sold.

Select product\_id, SUM(quantity)

From sales

Group by product\_id

Having SUM(quantity) > ( Select AVG (total\_quantity)

From (

Select product\_id,SUM(quantity) as total\_quantity

From sales

Group by product ID) as product\_sales);

1. **Subquery in the SELECT Clause**

* **Table**: students
* **Columns**:
  + student\_id (INT): Unique ID for each student
  + student\_name (VARCHAR(100)): Name of the student
  + grade (DECIMAL(4, 2)): Grade of the student
* **Question**: Write an SQL query to find each student's name along with the average grade of all students.

Select student\_name, Round((select AVG(grade) FROM reading.students),2)

From reading.students;

1. **Nested Subquery**

* **Table**: products
* **Columns**:
  + product\_id (INT): Unique ID for each product
  + product\_name (VARCHAR(100)): Name of the product
  + category (VARCHAR(50)): Category of the product
  + price (DECIMAL(8, 2)): Price of the product
* **Question**: Write an SQL query to find the products in the category with the highest average price.

ANSWER:

Select distinct(ItemName), ItemCategory, price

from restaurant.restaurantorders

Where ItemCategory = (

Select ItemCategory From restaurant.restaurantorders group by ItemCategory HAVING AVG(Price) = (

Select MAX(avgprice) mavg from (

Select AVG(Price) as avgprice from restaurant.restaurantorders Group by ItemCategory) as avg\_price));

1. **Subquery with JOIN**

* **Table**: orders
* **Columns**:
  + order\_id (INT): Unique ID for each order
  + customer\_id (INT): ID of the customer who placed the order
  + product\_id (INT): ID of the product ordered
* **Question**: Write an SQL query to find the products that were ordered by customers from a specific city (e.g., 'New York').

 **orders Table**

* order\_id (INT)
* customer\_id (INT)
* product\_id (INT)

 **customers Table (Assumed)**

* customer\_id (INT)
* customer\_name (VARCHAR)
* city (VARCHAR)

ANSWER:

Select o.product\_id

From order o

Join customers c on o.customer\_id = c.customer\_id

Where city = ‘New York’

1. **Subquery with UPDATE**

* **Table**: products
* **Columns**:
  + product\_id (INT): Unique ID for each product
  + price (DECIMAL(8, 2)): Price of the product
  + category (VARCHAR(50)): Category of the product
* **Question**: Write an SQL query to increase the price of all products in the 'Furniture' category by 10%.

Update products

SET price = Price\*1.10

Where category = 'Furniture';