-- 1. Simple Query: Retrieve all customers who are seeking insurance, showing all their details.

SELECT \*

FROM Explore\_Secure\_TravelInsurance.Customer;

-- Output: All customer details (customer\_id, name, gender, date\_of\_birth, occupation, travel\_purpose, risk\_level)

-- 2. Aggregate Query: Get the total number of customers in the database.

SELECT COUNT(\*) AS total\_customers

FROM Explore\_Secure\_TravelInsurance.Customer;

-- Output: The total number of customers in the `Customer` table.

-- 3. Inner Join: Get the list of all customers along with the policies they have purchased.

SELECT Customer.name, InsurancePolicy.policy\_name

FROM Explore\_Secure\_TravelInsurance.Customer

INNER JOIN Explore\_Secure\_TravelInsurance.CustomerPolicy

ON Customer.customer\_id = CustomerPolicy.customer\_id

INNER JOIN Explore\_Secure\_TravelInsurance.InsurancePolicy

ON CustomerPolicy.policy\_id = InsurancePolicy.policy\_id;

-- Output: List of customer names and the policies they have purchased.

-- 4. Left Outer Join: Get a list of all customers and their claims (if they have any).

SELECT Customer.name, Claim.claim\_id

FROM Explore\_Secure\_TravelInsurance.Customer

LEFT JOIN Explore\_Secure\_TravelInsurance.Claim

ON Customer.customer\_id = Claim.customer\_id;

-- Output: List of all customers with their claim IDs, including customers with no claims.

-- 5. Nested Query: Get customers who have a policy with a premium greater than 200.

SELECT name

FROM Explore\_Secure\_TravelInsurance.Customer

WHERE customer\_id IN (

SELECT customer\_id

FROM Explore\_Secure\_TravelInsurance.CustomerPolicy

WHERE policy\_id IN (

SELECT policy\_id

FROM Explore\_Secure\_TravelInsurance.InsurancePolicy

WHERE premium > 200

)

);

-- Output: List of customer names who have purchased policies with a premium greater than 200.

-- 6. Correlated Subquery: Get the names of customers whose claim amounts are higher than the average claim amount.

SELECT name

FROM Explore\_Secure\_TravelInsurance.Customer C

WHERE EXISTS (

SELECT 1

FROM Explore\_Secure\_TravelInsurance.Claim Cl

WHERE C.customer\_id = Cl.customer\_id

AND Cl.claim\_amount > (SELECT AVG(claim\_amount) FROM Explore\_Secure\_TravelInsurance.Claim)

);

-- Output: List of customer names who have made a claim greater than the average claim amount.

-- 7. Using `ANY`: Get customers who have made a claim of any amount greater than 1200.

SELECT name

FROM Explore\_Secure\_TravelInsurance.Customer

WHERE customer\_id = ANY (

SELECT customer\_id

FROM Explore\_Secure\_TravelInsurance.Claim

WHERE claim\_amount > 1200

);

-- Output: List of customer names who have made a claim greater than 1000.

-- 8. Set Operations - Union: Get all the distinct insurance policy names that have been sold through agents or travel platforms.

SELECT policy\_name

FROM Explore\_Secure\_TravelInsurance.InsurancePolicy

UNION

SELECT policy\_name

FROM Explore\_Secure\_TravelInsurance.Partnership

INNER JOIN Explore\_Secure\_TravelInsurance.InsurancePolicy

ON Partnership.policy\_id = InsurancePolicy.policy\_id;

-- Output: A list of distinct policy names that have been sold through agents or travel platforms (no duplicates).

-- 9. Subqueries in `SELECT`: Get the name of each customer along with the total premium they have paid.

SELECT name,

(SELECT SUM(premium)

FROM Explore\_Secure\_TravelInsurance.InsurancePolicy

INNER JOIN Explore\_Secure\_TravelInsurance.CustomerPolicy

ON InsurancePolicy.policy\_id = CustomerPolicy.policy\_id

WHERE CustomerPolicy.customer\_id = Explore\_Secure\_TravelInsurance.Customer.customer\_id) AS total\_premium\_paid

FROM Explore\_Secure\_TravelInsurance.Customer;

-- Output: A list of customer names along with the total premium they have paid for their purchased policies.

-- 10. Subqueries in `FROM`: Get the average claim amount for each policy type.

SELECT PolicyType.type\_name, AVG(Claim.claim\_amount) AS average\_claim\_amount

FROM (

SELECT InsurancePolicy.policy\_id, InsurancePolicy.policy\_name, InsurancePolicy.policy\_type\_id

FROM Explore\_Secure\_TravelInsurance.InsurancePolicy

) AS PolicyDetails

INNER JOIN Explore\_Secure\_TravelInsurance.Claim

ON Claim.policy\_id = PolicyDetails.policy\_id

INNER JOIN Explore\_Secure\_TravelInsurance.PolicyType

ON PolicyDetails.policy\_type\_id = PolicyType.policy\_type\_id

GROUP BY PolicyType.type\_name;

-- Output: A list of policy types along with the average claim amount for each policy type.

NO SQL :

//agent name start with M

db.Agent.aggregate([

{ $unwind: "$agents" },

{ $match: { "agents.name": { $regex: "^M", $options: "i" } } },

{ $replaceRoot: { newRoot: "$agents" } }

])

//

………………….

//

//

//

db.Customers.aggregate([

{ $unwind: "$customers" }, // Split each element of the customers array

{ $match: { "customers.risk\_level": "High" } }, // Filter for risk\_level = "High"

{ $replaceRoot: { newRoot: "$customers" } } // Output just the customer sub‑document

])

//customer with risk level high

……………….

//sum of all the commissions of the agents by count of sales

db.AgentSales.aggregate([

{

$group: {

\_id: "$agent\_id", // Group by agent\_id

total\_commission: { $sum: "$commission" }, // Sum of all commissions

total\_sales: { $sum: 1 } // Count of sales

}

},

{

$project: {

\_id: 0,

agent\_id: "$\_id", // Rename \_id back to agent\_id

total\_commission: 1,

total\_sales: 1

}

},

{ $sort: { total\_commission: -1 } }

])