

The following 6 queries are written for Online Marketplace Database that we (Team 8) designed. I have also uploaded the .sql files for creating the database and tables (`main.sql`), populating the database (`populate.sql`) and the following 6 queries in `EE18B001_assgn3.sql`.

1 Question 1

List the contacts along with the names of sellers who have sold products of total value more than 500.

Query

```
SELECT s.name as sellerName, sc.phoneNumber as contactNo
FROM Seller AS s, SellerContacts AS sc
WHERE s.sellerId = sc.sellerId AND
      500 <= ANY (
          SELECT SUM(ct.qty * p.price)
          FROM Orders AS o, CartItem AS ct, Product AS p
          WHERE o.sellerId = s.sellerId AND o.productId = ct.productId
              AND o.itemId = ct.itemId AND ct.productId = p.productId
      );
```

Result Grid		Filter Rows:	Search	Export:
	sellerName	contactNo		
▶	Leexo	557-972-564700		
	Leexo	625-967-0504		

Figure 1: Question 1 query results

2 Question 2

List the top 3 customers with their email IDs who have spent the most by buying products from the online marketplace.

Query

```
DROP VIEW IF EXISTS CustomerOriginalSpent;
CREATE VIEW CustomerOriginalSpent AS (
    SELECT c.customerId AS customerId,
           COALESCE(SUM(tmp.qty * tmp.price), 0) AS originalSpent
    FROM Customer AS c LEFT OUTER JOIN (
        SELECT pm.customerId, ci.qty, p.price
        FROM Payment AS pm, CartItem AS ci, Product AS p
        WHERE pm.cartId = ci.cartId AND pm.customerId = ci.customerId
            AND ci.productId = p.productId
    ) AS tmp ON c.customerId = tmp.customerId
    GROUP BY c.customerId
);
```

```

DROP VIEW IF EXISTS CustomerAdditionalSpent;
CREATE VIEW CustomerAdditionalSpent AS (
    SELECT c.customerId AS customerId,
           COALESCE(SUM(pm.additionalCharges), 0) AS additionalSpent
    FROM Customer AS c LEFT OUTER JOIN Payment as pm
        ON c.customerId = pm.customerId
    GROUP BY c.customerId
);

SELECT cos.customerId AS customerId, c.email AS email,
       originalSpent+additionalSpent AS totalSpent
FROM CustomerOriginalSpent AS cos,
     CustomerAdditionalSpent AS cas,
     Customer AS c
WHERE cos.customerId = cas.customerId AND c.customerId = cos.customerId
ORDER BY totalSpent DESC
LIMIT 3;

```




Result Grid   Filter Rows: <input type="text" value="Search"/> Export: 				
	customerId	email	totalSpent	
▶	15	ewaters@example.net	1170.2799875736237	
	2	rice.shawn@example.net	1040.920016527176	
	24	vida61@example.com	461.8299939632416	

Figure 2: Question 2 query results

3 Question 3

List the tags that appear more than 2 times among products that have been purchased by customers. Consider the same product bought twice by same/different customer during a different time as different as far as counting number of tags is concerned.

Query

```

SELECT tagName
FROM Orders AS o, Product AS p, ProductTags as pt
WHERE o.productId = p.productId AND p.productId = pt.productId
GROUP BY tagName
HAVING COUNT(*) > 2;

```




Result Grid   Filter Rows: <input type="text" value="Search"/> Export: 				
	tagName			
▶	Nail Design			
	Polishes			
	Home Decor			
	Tools & Accessories			

Figure 3: Question 3 query results

4 Question 4

Find the most preferred payment method used by customers when they have bought atleast 4 quantities of the same product.

Query

```
DROP VIEW IF EXISTS ReqPayments;
CREATE VIEW ReqPayments AS (
    SELECT paymentId, paymentMethod
    FROM Payment AS pm
    WHERE 4 <= ANY (
        SELECT qty
        FROM CartItem AS ci
        WHERE ci.cartId = pm.cartId AND ci.customerId = pm.customerId
    )
);

DROP VIEW IF EXISTS PaymentMethodsCount;
CREATE VIEW PaymentMethodsCount AS (
    SELECT paymentMethod, COUNT(paymentMethod) AS paymentMethodCount
    FROM ReqPayments
    GROUP BY paymentMethod
);

SELECT paymentMethod AS mostPreferredPaymentMethod, paymentMethodCount
FROM PaymentMethodsCount
WHERE paymentMethodCount = ANY (
    SELECT MAX(paymentMethodCount)
    FROM PaymentMethodsCount
);
```

Result Grid			Filter Rows:	Search	Export:
	mostPreferredPaymentMethod	paymentMethodCount			
▶	NetBanking	3			

Figure 4: Question 4 query results

5 Question 5

List the postalcodes where the 2 most costliest products are sold.

The first costliest product is the one with the maximum cost among products listed and has the lowest productId among the potential results.

The second costliest product is the one with second maximum cost (strictly less than that of the first costliest product) and has the lowest productId in case of a tie.

Query

```
DROP VIEW IF EXISTS CostliestPdt;
CREATE VIEW CostliestPdt AS (
    SELECT productId, price
    FROM Product
    WHERE price = ANY (
        SELECT MAX(price)
        FROM Product
    )
```

```

    )
    ORDER BY productId
    LIMIT 1
);

DROP VIEW IF EXISTS SecondCostliestPdt;
CREATE VIEW SecondCostliestPdt AS (
    SELECT productId, price
    FROM Product
    WHERE price = ANY (
        SELECT MAX(price)
        FROM Product
        WHERE price < ALL (
            SELECT price
            FROM CostliestPdt
        )
    )
    ORDER BY productId
    LIMIT 1
);
-- INTERSECT using INNER JOIN
SELECT t1.postalcode AS commonPostalcode
FROM (
    SELECT postalcode
    FROM CostliestPdt AS cp, SoldIn AS s
    WHERE cp.productId = s.productId
) AS t1 INNER JOIN
(
    SELECT postalcode
    FROM SecondCostliestPdt AS scp, SoldIn AS s
    WHERE scp.productId = s.productId
) AS t2
ON t1.postalcode = t2.postalcode;

```

Result Grid		Filter Rows:	Search	Export:
	commonPostalcode			
▶	35394			
	57309-4717			

Figure 5: Question 5 query results

6 Question 6

List the number of products ordered in the most recent timestamp. Include products ordered by different customers also if they occurred in the same most recent timestamp (the precision of which is limited to seconds) and include multiple quantities of the same product in the count.

Query

```

SELECT SUM(ci.qty) AS numProducts
FROM (
    SELECT paymentId, cartId, customerId, paymentTimestamp
    FROM Payment

```

```

WHERE paymentTimestamp = ANY (
    SELECT MAX(paymentTimestamp)
    FROM Payment
)
) AS mrp, CartItem AS ci
WHERE mrp.cartId = ci.cartId AND mrp.customerId = ci.customerId;

```

Result Grid		Filter Rows:	Search	Export:
	numProducts			
▶	3			

Figure 6: Question 6 query results (truncated as there are many rows in output)