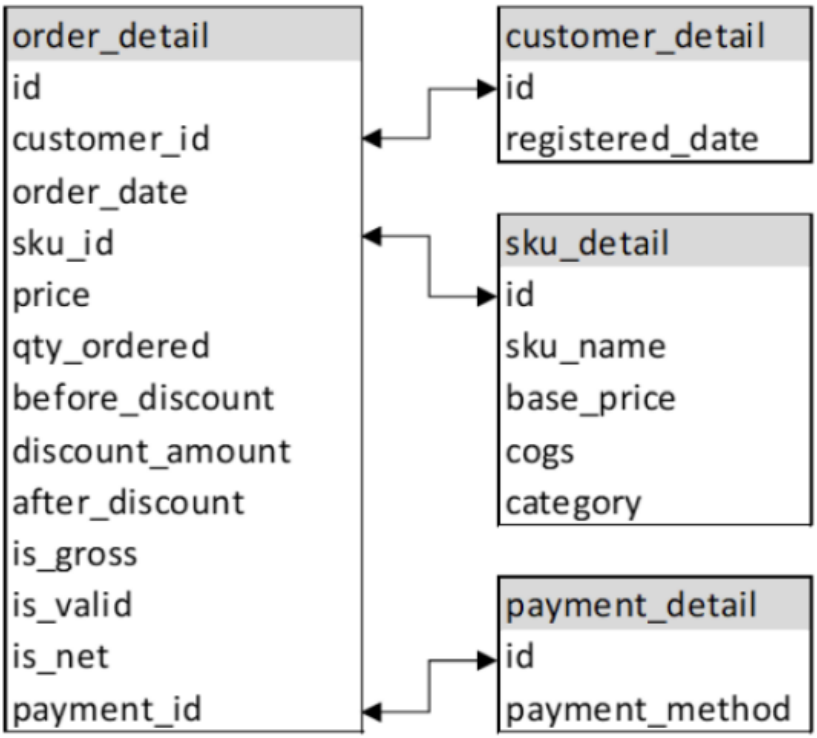
SQL QUERIES

There are following four tables (schema) you should use to answer the questions. The tables are described as follows:

1. order_detail
2. sku_detail
3. customer_detail
4. payment_detail

SCHEMA



CREATE TABLES

```sql
CREATE TABLE customer_detail (
    id VARCHAR(10) PRIMARY KEY,         -- Unique customer ID
    registered_date DATE        -- Customer registration date
);

CREATE TABLE sku_detail (
    id VARCHAR(10) PRIMARY KEY,         -- Unique SKU ID
    sku_name VARCHAR(50),       -- Name of the product
    base_price DECIMAL(10, 2),  -- Price stated on the price tag
    cogs DECIMAL(10, 2),        -- Cost of goods sold
    category VARCHAR(50)      -- Product category
);

CREATE TABLE payment_detail (
    id VARCHAR(10) PRIMARY KEY,         -- Unique payment method ID
    payment_method VARCHAR(20)  -- Payment method (e.g., cash, card)
);
```

```sql
CREATE TABLE order_detail (
    id VARCHAR(10) PRIMARY KEY,                     -- Unique order ID
    customer_id VARCHAR(10),                        -- Foreign key for customer ID
    order_date DATE,                                -- Order date
    sku_id VARCHAR(10),                             -- Foreign key for SKU ID
    price DECIMAL(10, 2),                           -- Price of the item
    qty_ordered INT,                                -- Quantity ordered
    before_discount DECIMAL(10, 2),                 -- Total price before discount
    discount_amount DECIMAL(10, 2),                 -- Discount amount
    after_discount DECIMAL(10, 2),                  -- Total price after discount
    is_gross BOOLEAN,                               -- Gross transaction indicator
    is_valid BOOLEAN,                               -- Valid payment indicator
    is_net BOOLEAN,                                 -- Net transaction indicator
    payment_id VARCHAR(10),                         -- Foreign key for payment method
    FOREIGN KEY (customer_id) REFERENCES customer_detail(id),
    FOREIGN KEY (sku_id) REFERENCES sku_detail(id),
    FOREIGN KEY (payment_id) REFERENCES payment_detail(id)
);
```

INSERT VALUES

```sql
INSERT INTO customer_detail (id, registered_date) VALUES
('C-001', '2020/01/15'),
('C-002', '2020/03/20'),
('C-003', '2020/05/10'),
('C-004', '2020/06/25'),
('C-005', '2020/07/30'),
('C-006', '2020/09/05'),
('C-007', '2020/10/10'),
('C-008', '2020/11/15'),
('C-009', '2021/01/20'),
('C-010', '2021/03/05'),
('C-011', '2021/04/10'),
('C-012', '2021/06/15'),
('C-013', '2021/07/20'),
('C-014', '2021/09/05'),
('C-015', '2021/10/15'),
('C-016', '2021/12/01'),
('C-017', '2022/01/10'),
('C-018', '2022/03/25'),
('C-019', '2022/05/15'),
('C-020', '2022/06/20');
```

```sql
INSERT INTO sku_detail (id, sku_name, base_price, cogs, category) VALUES
('SK-001', 'Samsung Galaxy S21', 799.99, 600.00, 'Electronics'),
('SK-002', 'Apple iPhone 13', 999.99, 750.00, 'Electronics'),
('SK-003', 'Sony WH-1000XM4', 349.99, 250.00, 'Electronics'),
('SK-004', 'Huawei P50 Pro', 899.99, 670.00, 'Electronics'),
('SK-005', 'Lenovo ThinkPad X1', 1399.99, 1000.00, 'Electronics'),
('SK-006', 'Google Pixel 6', 599.99, 450.00, 'Electronics'),
('SK-007', 'Mens Casual Shirt', 29.99, 15.00, 'Clothing'),
('SK-008', 'Womens Summer Dress', 49.99, 25.00, 'Clothing'),
('SK-009', 'Leather Jacket', 199.99, 120.00, 'Clothing'),
('SK-010', 'Non-Stick Cookware Set', 89.99, 60.00, 'Home & Kitchen'),
('SK-011', 'Electric Kettle', 39.99, 25.00, 'Home & Kitchen'),
('SK-012', 'Air Fryer', 99.99, 70.00, 'Home & Kitchen'),
('SK-013', 'Organic Olive Oil', 19.99, 15.00, 'Grocery'),
('SK-014', 'Pack of Almonds', 12.99, 8.00, 'Grocery'),
('SK-015', 'Canned Tuna', 2.99, 1.50, 'Grocery'),
('SK-016', 'Fitness Tracker', 59.99, 40.00, 'Electronics'),
('SK-017', 'Bluetooth Speaker', 89.99, 50.00, 'Electronics'),
('SK-018', 'Wireless Charging Pad', 29.99, 15.00, 'Electronics');
```

```sql
INSERT INTO order_detail (id, customer_id, order_date, sku_id, price, qty_ordered, before_discount, discount_amount,
after_discount, is_gross, is_valid, is_net, payment_id) VALUES
('A101', 'C-009', '2021-01-15', 'SK-001', 799.99, 2, 1599.98, 159.99, 1439.99, TRUE, FALSE, FALSE, 'PY-002'),
('A102', 'C-017', '2021-02-05', 'SK-002', 999.99, 1, 999.99, 99.99, 900.00, FALSE, TRUE, TRUE, 'PY-001'),
('A103', 'C-006', '2021-03-01', 'SK-003', 349.99, 1, 349.99, 34.99, 315.00, TRUE, TRUE, FALSE, 'PY-005'),
('A104', 'C-018', '2021-03-15', 'SK-004', 899.99, 1, 899.99, 89.99, 810.00, TRUE, FALSE, FALSE, 'PY-006'),
('A105', 'C-012', '2021-04-10', 'SK-005', 1399.99, 1, 1399.99, 139.99, 1260.00, FALSE, TRUE, TRUE, 'PY-003'),
('A106', 'C-013', '2021-05-05', 'SK-006', 599.99, 3, 1799.97, 179.99, 1619.98, TRUE, FALSE, FALSE, 'PY-004'),
('A107', 'C-019', '2021-05-20', 'SK-007', 29.99, 10, 299.90, 29.99, 269.91, FALSE, TRUE, TRUE, 'PY-001'),
('A108', 'C-008', '2021-06-10', 'SK-008', 49.99, 2, 99.98, 9.99, 89.99, TRUE, FALSE, FALSE, 'PY-007'),
('A109', 'C-011', '2021-07-05', 'SK-009', 199.99, 1, 199.99, 19.99, 180.00, TRUE, TRUE, FALSE, 'PY-005'),
('A110', 'C-010', '2021-08-15', 'SK-010', 89.99, 5, 449.95, 44.99, 404.96, TRUE, FALSE, FALSE, 'PY-003'),
('A111', 'C-001', '2021-09-01', 'SK-011', 39.99, 2, 79.98, 7.99, 71.99, TRUE, TRUE, FALSE, 'PY-002'),
('A112', 'C-020', '2021-10-20', 'SK-012', 99.99, 1, 99.99, 9.99, 90.00, FALSE, TRUE, TRUE, 'PY-001'),
('A113', 'C-004', '2021-11-25', 'SK-013', 19.99, 10, 199.90, 19.99, 179.91, TRUE, FALSE, FALSE, 'PY-003'),
('A114', 'C-014', '2021-12-10', 'SK-014', 12.99, 15, 194.85, 19.49, 175.36, TRUE, FALSE, FALSE, 'PY-001'),
('A115', 'C-016', '2022-01-10', 'SK-015', 2.99, 20, 59.80, 5.98, 53.82, FALSE, TRUE, TRUE, 'PY-002'),
('A116', 'C-005', '2022-02-05', 'SK-016', 59.99, 3, 179.97, 17.99, 161.98, TRUE, FALSE, FALSE, 'PY-006'),
('A117', 'C-015', '2022-03-15', 'SK-017', 89.99, 2, 179.98, 17.99, 161.99, FALSE, TRUE, TRUE, 'PY-005'),
('A118', 'C-003', '2022-03-25', 'SK-018', 29.99, 10, 299.90, 29.99, 269.91, FALSE, TRUE, TRUE, 'PY-004'),
('A119', 'C-007', '2022-04-10', 'SK-018', 29.99, 1, 89.99, 8.99, 81.00, TRUE, FALSE, FALSE, 'PY-003'),
('A120', 'C-012', '2022-05-05', 'SK-015', 2.99, 2, 99.98, 9.99, 89.99, TRUE, TRUE, FALSE, 'PY-002'),
('A121', 'C-020', '2022-06-01', 'SK-001', 799.99, 1, 799.99, 79.99, 720.00, FALSE, TRUE, TRUE, 'PY-003'),
('A122', 'C-019', '2022-06-15', 'SK-002', 999.99, 1, 999.99, 99.99, 900.00, TRUE, FALSE, FALSE, 'PY-006'),
('A123', 'C-014', '2022-07-10', 'SK-003', 349.99, 3, 1049.97, 104.99, 944.98, FALSE, TRUE, TRUE, 'PY-001'),
('A124', 'C-018', '2022-08-05', 'SK-004', 899.99, 1, 899.99, 89.99, 810.00, TRUE, FALSE, FALSE, 'PY-004'),
('A125', 'C-002', '2022-08-15', 'SK-005', 1399.99, 1, 1399.99, 139.99, 1260.00, FALSE, TRUE, TRUE, 'PY-005'),
('A126', 'C-008', '2022-09-01', 'SK-006', 599.99, 1, 599.99, 59.99, 540.00, TRUE, FALSE, FALSE, 'PY-003'),
('A127', 'C-011', '2022-09-20', 'SK-007', 29.99, 10, 299.90, 29.99, 269.91, FALSE, TRUE, TRUE, 'PY-002'),
('A128', 'C-010', '2022-10-05', 'SK-008', 49.99, 2, 99.98, 9.99, 89.99, TRUE, FALSE, FALSE, 'PY-006'),
('A129', 'C-005', '2022-10-25', 'SK-009', 199.99, 1, 199.99, 19.99, 180.00, FALSE, TRUE, TRUE, 'PY-001'),
```

```sql
INSERT INTO payment_detail
(id, payment_method) VALUES
('PY-001', 'Credit Card'),
('PY-002', 'Debit Card'),
('PY-003', 'Bank Transfer'),
('PY-004', 'PayPal'),
('PY-005', 'Apple Pay'),
('PY-006', 'Google Pay'),
('PY-007', 'Cash'),
('PY-008', 'Gift Card');
select * from payment_detail;
```

TABLES

```
select * from customer_detail LIMIT 10;
```

	id [PK] character varying (10)	registered_date date
1	C-001	2020-01-15
2	C-002	2020-03-20
3	C-003	2020-05-10
4	C-004	2020-06-25
5	C-005	2020-07-30
6	C-006	2020-09-05
7	C-007	2020-10-10
8	C-008	2020-11-15
9	C-009	2021-01-20
10	C-010	2021-03-05

```
select * from sku_detail LIMIT 10;
```

	id [PK] character varying (10)	sku_name character varying (50)	base_price numeric (10,2)	cogs numeric (10,2)	category character varying (50)
1	SK-001	Samsung Galaxy S21	799.99	600.00	Electronics
2	SK-002	Apple iPhone 13	999.99	750.00	Electronics
3	SK-003	Sony WH-1000XM4	349.99	250.00	Electronics
4	SK-004	Huawei P50 Pro	899.99	670.00	Electronics
5	SK-005	Lenovo ThinkPad X1	1399.99	1000.00	Electronics
6	SK-006	Google Pixel 6	599.99	450.00	Electronics
7	SK-007	Mens Casual Shirt	29.99	15.00	Clothing
8	SK-008	Womens Summer Dress	49.99	25.00	Clothing
9	SK-009	Leather Jacket	199.99	120.00	Clothing
10	SK-010	Non-Stick Cookware Set	89.99	60.00	Home & Kitchen

```
select * from payment_detail LIMIT 10;
```

	id [PK] character varying (10)	payment_method character varying (20)
1	PY-001	Credit Card
2	PY-002	Debit Card
3	PY-003	Bank Transfer
4	PY-004	PayPal
5	PY-005	Apple Pay
6	PY-006	Google Pay
7	PY-007	Cash
8	PY-008	Gift Card

```sql
select * from order_detail LIMIT 10;
```

	id [PK] character varying (10)	customer_id character varying (10)	order_date date	sku_id character varying (10)	price numeric (10,2)	qty_ordered integer	before_discount numeric (10,2)	discount_amount numeric (10,2)	after_discount numeric (10,2)	is_gross boolean	is_valid boolean	is_net boolean	payment_id character varying (10)
1	A101	C-009	2021-01-15	SK-001	799.99	2	1599.98	159.99	1439.99	true	false	false	PY-002
2	A102	C-017	2021-02-05	SK-002	999.99	1	999.99	99.99	900.00	false	true	true	PY-001
3	A103	C-006	2021-03-01	SK-003	349.99	1	349.99	34.99	315.00	true	true	false	PY-005
4	A104	C-018	2021-03-15	SK-004	899.99	1	899.99	89.99	810.00	true	false	false	PY-006
5	A105	C-012	2021-04-10	SK-005	1399.99	1	1399.99	139.99	1260.00	false	true	true	PY-003
6	A106	C-013	2021-05-05	SK-006	599.99	3	1799.97	179.99	1619.98	true	false	false	PY-004
7	A107	C-019	2021-05-20	SK-007	29.99	10	299.90	29.99	269.91	false	true	true	PY-001
8	A108	C-008	2021-06-10	SK-008	49.99	2	99.98	9.99	89.99	true	false	false	PY-007
9	A109	C-011	2021-07-05	SK-009	199.99	1	199.99	19.99	180.00	true	true	false	PY-005
10	A110	C-010	2021-08-15	SK-010	89.99	5	449.95	44.99	404.96	true	false	false	PY-003

ANALYSIS

Question 1:
During the transactions that occurred in 2021, in which month did the total transaction value
(after_discount) reach its highest? Use is_valid = 1 to filter transaction data.

***For Boolean Value, TRUE used instead of 1.

```sql
SELECT
    EXTRACT(MONTH FROM order_date) AS month,
    EXTRACT(YEAR FROM order_date) AS year_text,
    TO_CHAR(order_date, 'Month') AS month_text,
    SUM(after_discount) AS total_after_discount
FROM order_detail
WHERE EXTRACT(YEAR FROM order_date) = 2021
      AND is_valid = true
GROUP BY EXTRACT(MONTH FROM order_date), EXTRACT(YEAR FROM order_date), TO_CHAR(order_date, 'Month')
ORDER BY total_after_discount DESC;
```

	month numeric	year_text numeric	month_text text	total_after_discount numeric
1	4	2021	April	1260.00
2	2	2021	February	900.00
3	3	2021	March	315.00
4	5	2021	May	269.91
5	7	2021	July	180.00
6	10	2021	October	90.00
7	9	2021	September	71.99

HIGHEST VALUE

```sql
SELECT
    EXTRACT(MONTH FROM order_date) AS month,
    EXTRACT(YEAR FROM order_date) AS year_text,
    TO_CHAR(order_date, 'Month') AS month_text,
    SUM(after_discount) AS total_after_discount
FROM order_detail
WHERE EXTRACT(YEAR FROM order_date) = 2021
      AND is_valid = true
GROUP BY EXTRACT(MONTH FROM order_date), EXTRACT(YEAR FROM order_date), TO_CHAR(order_date, 'Month')
ORDER BY total_after_discount DESC
LIMIT 1;
```

	month numeric	year_text numeric	month_text text	total_after_discount numeric
1	4	2021	April	1260.00

Question 2: During transactions in the year 2022, which category generated the highest transaction value? Use is_valid = 1 to filter transaction data.

***For Boolean Value, TRUE used instead of 1.

```
SELECT
    sd.category,
    SUM(od.after_discount) AS total_transaction_value
FROM
    order_detail od
JOIN
    sku_detail sd ON od.sku_id = sd.id
WHERE
    od.is_valid = TRUE
    AND EXTRACT(YEAR FROM od.order_date) = 2022
GROUP BY
    sd.category
ORDER BY
    total_transaction_value DESC;
```

	category character varying (50)	total_transaction_value numeric
1	Electronics	3356.88
2	Clothing	449.91
3	Grocery	143.81
4	Home & Kitchen	71.99

HIGHEST TRANSACTION VALUE

```
SELECT
    sd.category,
    SUM(od.after_discount) AS total_transaction_value
FROM
    order_detail od
JOIN
    sku_detail sd ON od.sku_id = sd.id
WHERE
    od.is_valid = TRUE
    AND EXTRACT(YEAR FROM od.order_date) = 2022
GROUP BY
    sd.category
ORDER BY
    total_transaction_value DESC
LIMIT 1;
```

	category character varying (50)	total_transaction_value numeric
1	Electronics	3356.88

Question 3: Compare the transaction values of each category in the years 2021 and 2022. Mention which categories experienced an increase and which categories experienced a decrease in transaction values from 2021 to 2022. Use is_valid = 1 to filter transaction data. Source table : order_detail, sku_detail.

***For Boolean Value, TRUE used instead of 1.

```sql
SELECT
    year_2021.category,
    year_2021.total_sales AS transaction_2021,
    year_2022.total_sales AS transaction_2022,
    year_2022.total_sales - year_2021.total_sales AS difference,
    CASE
        WHEN year_2022.total_sales > year_2021.total_sales THEN 'INCREASE'
        WHEN year_2022.total_sales < year_2021.total_sales THEN 'DECREASE'
        ELSE 'NO CHANGE'
    END AS remark
FROM
    (SELECT
        sd.category,
        SUM(od.after_discount) AS total_sales
     FROM
        order_detail od
     JOIN
        sku_detail sd ON od.sku_id = sd.id
     WHERE
        od.is_valid = TRUE
        AND EXTRACT(YEAR FROM od.order_date) = 2021
     GROUP BY
        sd.category
    ) year_2021
JOIN
    (SELECT
        sd.category,
        SUM(od.after_discount) AS total_sales
     FROM
        order_detail od
     JOIN
        sku_detail sd ON od.sku_id = sd.id
     WHERE
        od.is_valid = TRUE
        AND EXTRACT(YEAR FROM od.order_date) = 2022
     GROUP BY
        sd.category
    ) year_2022
ON year_2021.category = year_2022.category;
```

	category character varying (50)	transaction_2021 numeric	transaction_2022 numeric	difference numeric	remark text
1	Clothing	449.91	449.91	0.00	NO CHANGE
2	Electronics	2475.00	3356.88	881.88	INCREASE
3	Home & Kitchen	161.99	71.99	-90.00	DECREASE

Question 4: Display the top 5 most popular payment methods used during 2022 (based on total unique orders). Use is_valid = 1 to filter transaction data.

***For Boolean Value, TRUE used instead of 1.

```sql
SELECT
    pd.payment_method,
    COUNT(DISTINCT od.id) AS total_unique_orders
FROM
    order_detail od
JOIN
    payment_detail pd ON od.payment_id = pd.id
WHERE
    od.is_valid = TRUE
    AND EXTRACT(YEAR FROM od.order_date) = 2022
GROUP BY
    pd.payment_method
ORDER BY
    total_unique_orders DESC
LIMIT 5;
```

	payment_method character varying (20)	total_unique_orders bigint
1	Debit Card	4
2	Apple Pay	2
3	Credit Card	2
4	Bank Transfer	1
5	PayPal	1

Question 5. Sort these 5 products based on their transaction values.
1. Samsung,
2. Apple,
3. Sony,
4. Huawei, 5
. Lenovo
Use is_valid = 1 to filter transaction data.

***For Boolean Value, TRUE used instead of 1.

```sql
SELECT
    CASE
        WHEN LOWER(sd.sku_name) LIKE '%samsung%' THEN 'Samsung'
        WHEN LOWER(sd.sku_name) LIKE '%apple%' THEN 'Apple'
        WHEN LOWER(sd.sku_name) LIKE '%sony%' THEN 'Sony'
        WHEN LOWER(sd.sku_name) LIKE '%huawei%' THEN 'Huawei'
        WHEN LOWER(sd.sku_name) LIKE '%lenovo%' THEN 'Lenovo'
    END AS product_brand,
    SUM(od.after_discount) AS total_transaction_value
FROM
    order_detail od
JOIN
    sku_detail sd ON od.sku_id = sd.id
WHERE
    od.is_valid = TRUE
    AND (LOWER(sd.sku_name) LIKE '%samsung%'
        OR LOWER(sd.sku_name) LIKE '%apple%'
        OR LOWER(sd.sku_name) LIKE '%sony%'
        OR LOWER(sd.sku_name) LIKE '%huawei%'
        OR LOWER(sd.sku_name) LIKE '%lenovo%')
GROUP BY
    product_brand
ORDER BY
    total_transaction_value DESC;
```

	product_brand text	total_transaction_value numeric
1	Lenovo	2520.00
2	Sony	1259.98
3	Apple	900.00
4	Samsung	720.00