```sql
/* Count of Transactions by Customer and Category */

SELECT Customers.customer_name, Merchants.category, COUNT(Transactions.transaction_id) AS total_transactions
FROM Customers
INNER JOIN Transactions ON Customers.customer_id = Transactions.customer_id
INNER JOIN Transaction_Details ON Transactions.transaction_id = Transaction_Details.transaction_id
INNER JOIN Merchants ON Transaction_Details.merchant_id = Merchants.merchant_id
GROUP BY Customers.customer_name, Merchants.category;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_name	category	total_transactions
John Doe	Electronics	1
Jane Smith	Grocery	1
Sam Brown	Electronics	1
Emily White	Grocery	1
Michael Green	Electronics	1
Aisha Khan	Grocery	1
Liu Wei	Electronics	1
Carlos Ruiz	Convenience	1
Anna Muller	Grocery	1
Yuki Tanaka	Electronics	1

```
104     /* Transaction Count by Payment Method */
105
106 •   SELECT payment_method, COUNT(transaction_id) AS total_transactio
107     FROM Transactions
108     GROUP BY payment_method;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

payment_method	total_transactions
Wallet	3
Credit Card	3
Debit Card	2
Net Banking	2

```sql
/* Used DENSE_RANK to rank customers based on their spending within each merchant category */

SELECT customer_name, category, total_spent, DENSE_RANK() OVER (PARTITION BY category ORDER BY total_spent DESC) AS category_rank
FROM (
    SELECT Customers.customer_name, Merchants.category, SUM(Transaction_Details.total_amount) AS total_spent
    FROM Customers
    INNER JOIN Transactions ON Customers.customer_id = Transactions.customer_id
    INNER JOIN Transaction_Details ON Transactions.transaction_id = Transaction_Details.transaction_id
    INNER JOIN Merchants ON Transaction_Details.merchant_id = Merchants.merchant_id
    GROUP BY Customers.customer_name, Merchants.category
) AS spending_by_category;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_name	payment_method	total_spent	payment_rank
Emily White	Credit Card	200.00	1
Anna Muller	Credit Card	150.00	2
Jane Smith	Credit Card	120.00	3
Liu Wei	Debit Card	250.00	1
Sam Brown	Debit Card	75.00	2
Aisha Khan	Net Banking	90.00	1
Yuki Tanaka	Net Banking	80.00	2
Carlos Ruiz	Wallet	60.00	1
John Doe	Wallet	50.00	2
Michael Green	Wallet	30.00	3

```sql
/* Find the Top 3 Highest Spending Customers for Each Merchant (Window Functions) */

SELECT customer_name, merchant_name, total_spent, ranking
FROM (
    SELECT Customers.customer_name, Merchants.merchant_name, SUM(Transaction_Details.total_amount) AS total_spent,
           RANK() OVER (PARTITION BY Merchants.merchant_name ORDER BY SUM(Transaction_Details.total_amount) DESC) AS ranking
    FROM Customers
    INNER JOIN Transactions ON Customers.customer_id = Transactions.customer_id
    INNER JOIN Transaction_Details ON Transactions.transaction_id = Transaction_Details.transaction_id
    INNER JOIN Merchants ON Transaction_Details.merchant_id = Merchants.merchant_id
    GROUP BY Customers.customer_name, Merchants.merchant_name
) AS ranked_spending_per_merchant
WHERE ranking <= 3;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_name	merchant_name	total_spent	ranking
Anna Muller	Aldi	150.00	1
John Doe	Amazon	50.00	1
Michael Green	Best Buy	30.00	1
Aisha Khan	Carrefour	90.00	1
Emily White	Coles	200.00	1
Sam Brown	Flipkart	75.00	1
Liu Wei	JD.com	250.00	1
Carlos Ruiz	OXXO	60.00	1
Yuki Tanaka	Rakuten	80.00	1
Jane Smith	Walmart	120.00	1

```sql
INSERT INTO Customers (customer_id, customer_name, email, city, country, signup_date)
VALUES
(1, 'John Doe', 'john@example.com', 'New York', 'USA', '2022-01-01'),
(2, 'Jane Smith', 'jane@example.com', 'London', 'UK', '2022-02-10'),
(3, 'Sam Brown', 'sam@example.com', 'Delhi', 'India', '2022-03-15'),
(4, 'Emily White', 'emily@example.com', 'Sydney', 'Australia', '2022-04-20'),
(5, 'Michael Green', 'michael@example.com', 'Toronto', 'Canada', '2022-05-25'),
(6, 'Aisha Khan', 'aisha@example.com', 'Dubai', 'UAE', '2022-06-30'),
(7, 'Liu Wei', 'liu@example.com', 'Beijing', 'China', '2022-07-05'),
(8, 'Carlos Ruiz', 'carlos@example.com', 'Mexico City', 'Mexico', '2022-08-10'),
(9, 'Anna Muller', 'anna@example.com', 'Berlin', 'Germany', '2022-09-15'),
(10, 'Yuki Tanaka', 'yuki@example.com', 'Tokyo', 'Japan', '2022-10-20');

INSERT INTO Merchants (merchant_id, merchant_name, category, city, country)
VALUES
(201, 'Amazon', 'Electronics', 'Seattle', 'USA'),
(202, 'Walmart', 'Grocery', 'London', 'UK'),
(203, 'Flipkart', 'Electronics', 'Bangalore', 'India'),
(204, 'Coles', 'Grocery', 'Sydney', 'Australia'),
(205, 'Best Buy', 'Electronics', 'Toronto', 'Canada'),
(206, 'Carrefour', 'Grocery', 'Dubai', 'UAE'),
(207, 'JD.com', 'Electronics', 'Beijing', 'China'),
(208, 'OXXO', 'Convenience', 'Mexico City', 'Mexico'),
(209, 'Aldi', 'Grocery', 'Berlin', 'Germany'),
(210, 'Rakuten', 'Electronics', 'Tokyo', 'Japan');
```

```sql
INSERT INTO Transactions (transaction_id, customer_id, transaction_date, amount, transaction_type, payment_method)
VALUES
(1001, 1, '2023-01-10', 50.00, 'Mobile Recharge', 'Wallet'),
(1002, 2, '2023-02-15', 120.00, 'Shopping', 'Credit Card'),
(1003, 3, '2023-03-05', 75.00, 'Bill Payment', 'Debit Card'),
(1004, 4, '2023-04-10', 200.00, 'Shopping', 'Credit Card'),
(1005, 5, '2023-05-12', 30.00, 'Mobile Recharge', 'Wallet'),
(1006, 6, '2023-06-18', 90.00, 'Bill Payment', 'Net Banking'),
(1007, 7, '2023-07-20', 250.00, 'Shopping', 'Debit Card'),
(1008, 8, '2023-08-25', 60.00, 'Mobile Recharge', 'Wallet'),
(1009, 9, '2023-09-05', 150.00, 'Shopping', 'Credit Card'),
(1010, 10, '2023-10-10', 80.00, 'Bill Payment', 'Net Banking');

INSERT INTO Transaction_Details (transaction_detail_id, transaction_id, merchant_id, total_amount)
VALUES
(3001, 1001, 201, 50.00),
(3002, 1002, 202, 120.00),
(3003, 1003, 203, 75.00),
(3004, 1004, 204, 200.00),
(3005, 1005, 205, 30.00),
(3006, 1006, 206, 90.00),
(3007, 1007, 207, 250.00),
(3008, 1008, 208, 60.00),
(3009, 1009, 209, 150.00),
(3010, 1010, 210, 80.00);
```

```sql
/* Query 1: Retrieve Customer Transaction Information using joins */

SELECT Customers.customer_name, Transactions.transaction_date, Transactions.amount, Transactions.payment_method
FROM Transactions
INNER JOIN Customers ON Transactions.customer_id = Customers.customer_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_name	transaction_date	amount	payment_method
John Doe	2023-01-10	50.00	Wallet
Jane Smith	2023-02-15	120.00	Credit Card
Sam Brown	2023-03-05	75.00	Debit Card
Emily White	2023-04-10	200.00	Credit Card
Michael Green	2023-05-12	30.00	Wallet
Aisha Khan	2023-06-18	90.00	Net Banking
Liu Wei	2023-07-20	250.00	Debit Card
Carlos Ruiz	2023-08-25	60.00	Wallet
Anna Muller	2023-09-05	150.00	Credit Card
Yuki Tanaka	2023-10-10	80.00	Net Banking

```sql
/* Merchant Wise Revenue */

SELECT Merchants.merchant_name, SUM(Transaction_Details.total_amount) AS total_revenue
FROM Merchants
INNER JOIN Transaction_Details ON Merchants.merchant_id = Transaction_Details.merchant_id
GROUP BY Merchants.merchant_name;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

merchant_name	total_revenue
Amazon	50.00
Walmart	120.00
Flipkart	75.00
Coles	200.00
Best Buy	30.00
Carrefour	90.00
JD.com	250.00
OXXO	60.00
Aldi	150.00
Rakuten	80.00

```sql
/*  Rank Customers by Spending per Payment Method */

SELECT customer_name, payment_method, total_spent, RANK() OVER (PARTITION BY payment_method ORDER BY total_spent DESC) AS payment_rank
FROM (
    SELECT Customers.customer_name, Transactions.payment_method, SUM(Transactions.amount) AS total_spent
    FROM Customers
    INNER JOIN Transactions ON Customers.customer_id = Transactions.customer_id
    GROUP BY Customers.customer_name, Transactions.payment_method
) AS customer_spending;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_name	payment_method	total_spent	payment_rank
Emily White	Credit Card	200.00	1
Anna Muller	Credit Card	150.00	2
Jane Smith	Credit Card	120.00	3
Liu Wei	Debit Card	250.00	1
Sam Brown	Debit Card	75.00	2
Aisha Khan	Net Banking	90.00	1
Yuki Tanaka	Net Banking	80.00	2
Carlos Ruiz	Wallet	60.00	1
John Doe	Wallet	50.00	2
Michael Green	Wallet	30.00	3

```sql
/* Rank Customers by Total Spending */

SELECT customer_name, total_spent, RANK() OVER (ORDER BY total_spent DESC) AS ranking
FROM (
    SELECT Customers.customer_name, SUM(Transactions.amount) AS total_spent
    FROM Customers
    INNER JOIN Transactions ON Customers.customer_id = Transactions.customer_id
    GROUP BY Customers.customer_name
) AS spending;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_name	total_spent	ranking
Liu Wei	250.00	1
Emily White	200.00	2
Anna Muller	150.00	3
Jane Smith	120.00	4
Aisha Khan	90.00	5
Yuki Tanaka	80.00	6
Sam Brown	75.00	7
Carlos Ruiz	60.00	8
John Doe	50.00	9
Michael Green	30.00	10

```
95 •  SELECT *
96    FROM Customers
97
98
99
```

esult Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell

customer_id	customer_name	email	city	country	signup_date
1	John Doe	john@example.com	New York	USA	2022-01-01
2	Jane Smith	jane@example.com	London	UK	2022-02-10
3	Sam Brown	sam@example.com	Delhi	India	2022-03-15
4	Emily White	emily@example.com	Sydney	Australia	2022-04-20
5	Michael Green	michael@example.com	Toronto	Canada	2022-05-25
6	Aisha Khan	aisha@example.com	Dubai	UAE	2022-06-30
7	Liu Wei	liu@example.com	Beijing	China	2022-07-05
8	Carlos Ruiz	carlos@example.com	Mexico City	Mexico	2022-08-10
9	Anna Muller	anna@example.com	Berlin	Germany	2022-09-15
10	Yuki Tanaka	yuki@example.com	Tokyo	Japan	2022-10-20
NULL	NULL	NULL	NULL	NULL	NULL

```
95 •  SELECT *
96    FROM Transactions;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

transaction_id	customer_id	transaction_date	amount	transaction_type	payment_method
1001	1	2023-01-10	50.00	Mobile Recharge	Wallet
1002	2	2023-02-15	120.00	Shopping	Credit Card
1003	3	2023-03-05	75.00	Bill Payment	Debit Card
1004	4	2023-04-10	200.00	Shopping	Credit Card
1005	5	2023-05-12	30.00	Mobile Recharge	Wallet
1006	6	2023-06-18	90.00	Bill Payment	Net Banking
1007	7	2023-07-20	250.00	Shopping	Debit Card
1008	8	2023-08-25	60.00	Mobile Recharge	Wallet
1009	9	2023-09-05	150.00	Shopping	Credit Card
1010	10	2023-10-10	80.00	Bill Payment	Net Banking
NULL	NULL	NULL	NULL	NULL	NULL

```
95 •    SELECT *
96      FROM Merchants;
97
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

merchant_id	merchant_name	category	city	country
201	Amazon	Electronics	Seattle	USA
202	Walmart	Grocery	London	UK
203	Flipkart	Electronics	Bangalore	India
204	Coles	Grocery	Sydney	Australia
205	Best Buy	Electronics	Toronto	Canada
206	Carrefour	Grocery	Dubai	UAE
207	JD.com	Electronics	Beijing	China
208	OXXO	Convenience	Mexico City	Mexico
209	Aldi	Grocery	Berlin	Germany
210	Rakuten	Electronics	Tokyo	Japan
NULL	NULL	NULL	NULL	NULL

```
95 •    SELECT *
96      FROM Transaction_Details;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

transaction_detail_id	transaction_id	merchant_id	total_amount
3001	1001	201	50.00
3002	1002	202	120.00
3003	1003	203	75.00
3004	1004	204	200.00
3005	1005	205	30.00
3006	1006	206	90.00
3007	1007	207	250.00
3008	1008	208	60.00
3009	1009	209	150.00
3010	1010	210	80.00
NULL	NULL	NULL	NULL

```sql
CREATE DATABASE IF NOT EXISTS PAYMENTS;
USE PAYMENTS;

CREATE TABLE Customers (
    customer_id INT PRIMARY KEY,
    customer_name VARCHAR(100),
    email VARCHAR(100),
    city VARCHAR(50),
    country VARCHAR(50),
    signup_date DATE
);

CREATE TABLE Transactions (
    transaction_id INT PRIMARY KEY,
    customer_id INT,
    transaction_date DATE,
    amount DECIMAL(10, 2),
    transaction_type VARCHAR(50),  -- e.g., 'Mobile Recharge', 'Bill Payment', 'Shopping'
    payment_method VARCHAR(50),    -- e.g., 'Credit Card', 'Debit Card', 'Wallet'
    FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
);
```

```sql
CREATE TABLE Merchants (
    merchant_id INT PRIMARY KEY,
    merchant_name VARCHAR(100),
    category VARCHAR(50),  -- e.g., 'Electronics', 'Grocery', 'Utilities'
    city VARCHAR(50),
    country VARCHAR(50)
);

CREATE TABLE Transaction_Details (
    transaction_detail_id INT PRIMARY KEY,
    transaction_id INT,
    merchant_id INT,
    total_amount DECIMAL(10, 2),
    FOREIGN KEY (transaction_id) REFERENCES Transactions(transaction_id),
    FOREIGN KEY (merchant_id) REFERENCES Merchants(merchant_id)
);
```

```sql
/* Total Spending per Customer */

SELECT Customers.customer_name, SUM(Transactions.amount) AS total_spent
FROM Customers
INNER JOIN Transactions ON Customers.customer_id = Transactions.customer_id
GROUP BY Customers.customer_name;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_name	total_spent
John Doe	50.00
Jane Smith	120.00
Sam Brown	75.00
Emily White	200.00
Michael Green	30.00
Aisha Khan	90.00
Liu Wei	250.00
Carlos Ruiz	60.00
Anna Muller	150.00
Yuki Tanaka	80.00

```sql
/* Customer's Monthly Spending Trends (Window Function) */

SELECT customer_name,
       DATE_FORMAT(transaction_date, '%Y-%m') AS month,
       SUM(amount) AS monthly_spent,
       SUM(SUM(amount)) OVER (PARTITION BY customer_name ORDER BY DATE_FORMAT(transaction_date, '%Y-%m')) AS cumulative_spent
FROM Customers
INNER JOIN Transactions ON Customers.customer_id = Transactions.customer_id
GROUP BY customer_name, DATE_FORMAT(transaction_date, '%Y-%m')
ORDER BY customer_name, month;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_name	month	monthly_spent	cumulative_spent
Aisha Khan	2023-06	90.00	90.00
Anna Muller	2023-09	150.00	150.00
Carlos Ruiz	2023-08	60.00	60.00
Emily White	2023-04	200.00	200.00
Jane Smith	2023-02	120.00	120.00
John Doe	2023-01	50.00	50.00
Liu Wei	2023-07	250.00	250.00
Michael Green	2023-05	30.00	30.00
Sam Brown	2023-03	75.00	75.00
Yuki Tanaka	2023-10	80.00	80.00