console_1.sql

```
USE mmai_db

SELECT *
    FROM assignment01.bakery_sales;


-- A1-Q1
  WITH MonthlySales AS (
    SELECT
        YEAR(sale_date) AS Year,
        MONTH(sale_date) AS Month,
        article AS item_name,
        SUM(quantity) AS total_quantity,
        SUM(quantity * unit_price) AS total_revenue,
        ROW_NUMBER() OVER (PARTITION BY YEAR(sale_date), MONTH(sale_date) ORDER BY SUM(quantity) DESC) AS rank
    FROM
        assignment01.bakery_sales
    GROUP BY
        YEAR(sale_date),
        MONTH(sale_date),
        article
)
SELECT
    Year,
    Month,
    item_name,
    total_quantity,
    total_revenue
FROM
    MonthlySales
WHERE
    rank <= 3
ORDER BY
    Year,
    Month,
    rank;

-----A1-Q2
SELECT
    ticket_number,
    COUNT(DISTINCT article) AS unique_articles
FROM
    assignment01.bakery_sales
WHERE
    YEAR(sale_date) = 2021
    AND MONTH(sale_date) = 12
GROUP BY
    ticket_number
HAVING
    COUNT(DISTINCT article) >= 5;









---Q1
SELECT bs.article,
       MIN(bs.unit_price) AS min_price,
       MAX(bs.unit_price) AS max_price
```

```
FROM assignment01.bakery_sales AS bs
GROUP BY bs.article;

WITH min_max_prices AS (
    SELECT
        MIN(bs.unit_price) AS min_price,
        MAX(bs.unit_price) AS max_price
    FROM
        assignment01.bakery_sales AS bs
    WHERE
        bs.unit_price > 0
)
SELECT
    bs.article,
    bs.unit_price
FROM
    assignment01.bakery_sales AS bs,
    min_max_prices
WHERE
    bs.unit_price = min_max_prices.min_price
    OR bs.unit_price = min_max_prices.max_price;
-------Q2
WITH sales_ranking AS (
    SELECT
        bs.article,
        SUM(bs.quantity) AS total_quantity,
        RANK() OVER (ORDER BY SUM(bs.quantity) DESC) AS sales_rank
    FROM
        assignment01.bakery_sales AS bs
    GROUP BY
        bs.article
)
SELECT
    article
FROM
    sales_ranking
WHERE
    sales_rank = 2;

--------------------Q3
WITH MonthlySales AS (
    SELECT
        article,
        YEAR(sale_datetime) AS sale_year,
        MONTH(sale_datetime) AS sale_month,
        SUM(quantity) AS total_quantity_sold
    FROM assignment01.bakery_sales
    WHERE YEAR(sale_datetime) = 2022
    GROUP BY article, YEAR(sale_datetime), MONTH(sale_datetime)
),
RankedSales AS (
    SELECT
        article,
        sale_year,
        sale_month,
        total_quantity_sold,
        RANK() OVER (PARTITION BY sale_year, sale_month ORDER BY total_quantity_sold DESC) AS sales_rank
    FROM MonthlySales
)
SELECT
    sale_year,
    sale_month,
    article,
    total_quantity_sold
FROM RankedSales
WHERE sales_rank <= 3
ORDER BY sale_year, sale_month, sales_rank;

------Q4
SELECT
```

```
    ticket_number,
    COUNT(article) AS number_of_articles
FROM assignment01.bakery_sales
WHERE
    YEAR(sale_datetime) = 2022
    AND MONTH(sale_datetime) = 8
GROUP BY
    ticket_number
HAVING
    COUNT(article) >= 5
ORDER BY
    number_of_articles DESC;
------- Q5
SELECT
    AVG(daily_sales) AS average_sales_per_day
FROM (
    SELECT
        CAST(sale_datetime AS DATE) AS sale_date,
        SUM(quantity * unit_price) AS daily_sales
    FROM assignment01.bakery_sales
    WHERE
        YEAR(sale_datetime) = 2022
        AND MONTH(sale_datetime) = 8
    GROUP BY
        CAST(sale_datetime AS DATE)
) AS DailySales;

------Q6
SELECT
    DATENAME(WEEKDAY, sale_datetime) AS day_of_week,
    SUM(quantity * unit_price) AS total_sales
FROM assignment01.bakery_sales
GROUP BY
    DATENAME(WEEKDAY, sale_datetime),
    DATEPART(WEEKDAY, sale_datetime)
ORDER BY
    total_sales DESC;
-----Q7

SELECT
    DATEPART(hour, sale_datetime) AS hour_of_day,
    SUM(quantity) AS total_sales
FROM assignment01.bakery_sales
WHERE article = 'Traditional Baguette'
GROUP BY DATEPART(hour, sale_datetime)
ORDER BY total_sales DESC;

-----Q8
WITH MonthlySales AS (
    SELECT
        article,
        YEAR(sale_datetime) AS sale_year,
        MONTH(sale_datetime) AS sale_month,
        SUM(quantity) AS total_quantity_sold
    FROM assignment01.bakery_sales
    GROUP BY article, YEAR(sale_datetime), MONTH(sale_datetime)
),
RankedSales AS (
    SELECT
        article,
        sale_year,
        sale_month,
        total_quantity_sold,
        RANK() OVER (PARTITION BY sale_year, sale_month ORDER BY total_quantity_sold ASC) AS sales_rank
    FROM MonthlySales
)
SELECT
    sale_year,
    sale_month,
    article,
```

```
    total_quantity_sold
FROM RankedSales
WHERE sales_rank = 1
ORDER BY sale_year, sale_month;

-----9
WITH TotalSales AS (
    SELECT
        article,
        SUM(quantity * unit_price) AS total_sales
    FROM assignment01.bakery_sales
    WHERE sale_datetime BETWEEN '2022-01-01' AND '2022-01-31'
    GROUP BY article
),
GrandTotal AS (
    SELECT
        SUM(total_sales) AS grand_total_sales
    FROM TotalSales
)
SELECT
    t.article,
    t.total_sales,
    (t.total_sales / g.grand_total_sales) * 100 AS sales_percentage
FROM TotalSales t
CROSS JOIN GrandTotal g
ORDER BY sales_percentage DESC;



----_Q10
WITH MonthlyItemSales AS (
    SELECT
        YEAR(sale_datetime) AS sale_year,
        MONTH(sale_datetime) AS sale_month,
        article,
        SUM(quantity) AS total_quantity
    FROM assignment01.bakery_sales
    WHERE YEAR(sale_datetime) = 2022
    GROUP BY YEAR(sale_datetime), MONTH(sale_datetime), article
),
MonthlyTotalSales AS (
    SELECT
        sale_year,
        sale_month,
        SUM(total_quantity) AS monthly_total_quantity
    FROM MonthlyItemSales
    GROUP BY sale_year, sale_month
)
SELECT
    m.sale_year,
    m.sale_month,
    m.article,
    m.total_quantity AS banette_quantity,
    t.monthly_total_quantity,
    (m.total_quantity * 1.0 / t.monthly_total_quantity) * 100 AS order_rate_percentage
FROM
    MonthlyItemSales m
JOIN
    MonthlyTotalSales t
ON
    m.sale_year = t.sale_year AND m.sale_month = t.sale_month
WHERE
    m.article = 'Banette'
ORDER BY
    m.sale_year, m.sale_month;
```