MS-SQL Coding Challenge

-Sarthak Niranjan Kulkarni (Maverick)

- [sarthakkul2311@gmail.com](mailto:sarthakkul2311@gmail.com) - (+91) 93256 02791

**Day 5 – 8/11/2024 (Friday)**

1. **Querying Data by Using Joins and Subqueries & subtotal: -**
2. **What is the total count of each type of burger ordered by customers, along with the total count of all burgers ordered?**

🡪

* JOIN: This query uses an INNER JOIN between the customer\_orders table (aliased as co) and the burger\_names table (aliased as b).
* Grouping: The GROUP BY b.burger\_name groups the results by each unique burger\_name.
* ROLLUP: with rollup adds an extra row at the end of the result set to show a subtotal (or grand total) across all groups.

A screenshot of a computer

Description automatically generated

1. **Which customers ordered both 'Meatlovers' and 'Vegetarian' burgers?**

🡪

* JOIN: This query uses an INNER JOIN between customer\_orders (aliased as co) and burger\_names (aliased as b).
* Filtering: The WHERE clause filters the records to include only orders for burgers named "Meatlovers" and "Vegetarian".
* Grouping: The GROUP BY customer\_id groups the results by each unique customer\_id.

A screen shot of a computer

Description automatically generated

1. **For each runner, what is the total number of deliveries they completed, and the total number of orders they couldn’t deliver due to cancellations?**

🡪

* SUM with CASE: The SUM function with CASE is used to count the completed and canceled deliveries separately.
* Grouping: The GROUP BY ro.runner\_id groups the results by each unique runner\_id.

**A screenshot of a computer

Description automatically generated**

1. **What is the total number of burgers ordered per customer?**

🡪

* COUNT: COUNT(co.burger\_id) counts the number of burgers ordered by each customer.
* Grouping: The GROUP BY co.customer\_id groups the results by each unique customer\_id.

A screenshot of a computer

Description automatically generated

1. **What is the most popular burger type, and how many times was it ordered?**

🡪

* JOIN: An INNER JOIN is used between customer\_orders (co) and burger\_names (b).
* COUNT: COUNT(co.order\_id) counts the number of times each burger type was ordered.
* Grouping: The GROUP BY b.burger\_name groups the results by each unique burger type.

A screenshot of a computer code

Description automatically generated

1. **Manipulate data by using sql commands using groupby and having clause: -**
2. **What is the total number of orders placed by each customer who has ordered more than 2 times?**

🡪

* COUNT: COUNT(order\_id) counts the number of orders placed by each customer.
* Grouping: The GROUP BY customer\_id groups the results by each unique customer\_id.
* HAVING: The HAVING clause filters the results to only include customers who have placed more than 2 orders.

A screenshot of a computer

Description automatically generated

1. **Find the customers who have ordered more than one unique type of burger.**

🡪

* COUNT: COUNT (DISTINCT burger\_id) is calculated to determine how many unique types of burgers that customer has ordered.
* HAVING: The HAVING clause is used after the GROUP BY to filter the groups. It ensures that only those customers who ordered more than one unique type of burger are included in the result.

A screenshot of a computer

Description automatically generated