LinkedIn

Project on subtitles generation
Portfolio to reduce the effort for HR to navigate
Novipro, maven sample
SQL Projects - Case Study based - ER diagram,
Problem stmnt - optimized solution,
Carbon for ppt
8weekschallenge website - 15 days
daniss dinner case study
Project portfolio websites - LinkedTree, github, google sites, wakes
3 projects per tool
Naukri and LinkedIn - 100% optimized
Like everyone knows what to do to become a DA but in actual what things are required?
Did i take a wrong step by shifting my focus from Developer to DA
should i apply for entry level jobs or experienced ones for DA->
Any unique tip for increasing ATS score
Whenever i apply for any DA job a new skill/ technology comes up for which i have no idea
Online meet up as i don't live in Hyderabad

Pain point - people running in metro for work
Ghost Writers
Tech-MBA
Check https://www.linkedin.com/sales/ssi -> should be greater than 73%
algorithm of linkedin
LinkedIn wants to prioritize long posts
but short posts are for consistency
Avoiding too many links
Meaningful comments -> more than 12 words
Try using Selfie

danny's diner

```
/* --------------------
   Case Study Questions
   --------------------*/



-- 6. Which item was purchased first by the customer after they became a member?
-- 7. Which item was purchased just before the customer became a member?
-- 8. What is the total items and amount spent for each member before they became a member?
```

-- 9. If each $1 spent equates to 10 points and sushi has a 2x points multiplier - how many points would each customer have?
-- 10. In the first week after a customer joins the program (including their join date) they earn 2x points on all items, not just sushi - how many points do customer A and B have at the end of January?

```
-- 1. What is the total amount each customer spent at the restaurant?
SELECT s.customer_id, SUM(m.price) as Total_amount
FROM dannys_diner.sales s
INNER JOIN dannys_diner.menu m on s.product_id = m.product_id
GROUP BY s.customer_id;

-- 2. How many days has each customer visited the restaurant?
SELECT customer_id, COUNT(order_date) as customer_visited
FROM dannys_diner.sales
GROUP BY customer_id
ORDER BY customer_visited DESC;

-- 3. What was the first item from the menu purchased by each customer?
SELECT
s.customer_id,
m.product_name,
s.order_date
FROM
dannys_diner.sales s
JOIN
dannys_diner.menu m ON s.product_id = m.product_id
WHERE
s.order_date = (
SELECT MIN(order_date)
FROM dannys_diner.sales
WHERE customer_id = s.customer_id
);

-- 4. What is the most purchased item on the menu and how many times was it purchased
by all customers?
SELECT m.product_name, COUNT(m.product_id) as Most_purchased
FROM dannys_diner.sales s
INNER JOIN dannys_diner.menu m on m.product_id = s.product_id
GROUP BY m.product_name
ORDER BY Most_purchased DESC LIMIT 1;

--5. Which item was the most popular for each customer?
WITH most_popular AS (
SELECT
sales.customer_id,
menu.product_name,
```

```
COUNT(menu.product_id) AS order_count,
DENSE_RANK() OVER(
PARTITION BY sales.customer_id
ORDER BY COUNT(sales.customer_id) DESC) AS rank
FROM dannys_diner.menu
JOIN dannys_diner.sales
ON menu.product_id = sales.product_id
GROUP BY sales.customer_id, menu.product_name
)
SELECT
customer_id,
product_name,
order_count
FROM most_popular
WHERE rank = 1;


--6. Which item was purchased first by the customer after they became a member?


WITH joined_as_member AS (
SELECT
members.customer_id,
sales.product_id,
ROW_NUMBER() OVER(
PARTITION BY members.customer_id
ORDER BY sales.order_date) AS row_num
FROM dannys_diner.members
JOIN dannys_diner.sales
ON members.customer_id = sales.customer_id
AND sales.order_date > members.join_date
)
SELECT
customer_id,
product_name
FROM joined_as_member
JOIN dannys_diner.menu
ON joined_as_member.product_id = menu.product_id
WHERE row_num = 1
ORDER BY customer_id ASC;
```

```
--7. Which item was purchased just before the customer became a member?


WITH purchased_prior_member AS (
SELECT
members.customer_id,
sales.product_id,
ROW_NUMBER() OVER(
PARTITION BY members.customer_id
ORDER BY sales.order_date DESC) AS rank
FROM dannys_diner.members
JOIN dannys_diner.sales
ON members.customer_id = sales.customer_id
AND sales.order_date < members.join_date
)
SELECT
p_member.customer_id,
menu.product_name
FROM purchased_prior_member AS p_member
JOIN dannys_diner.menu
ON p_member.product_id = menu.product_id
WHERE rank = 1
ORDER BY p_member.customer_id ASC;




--8. What is the total items and amount spent for each member before they became a
member?
SELECT s.customer_id, COUNT(s.product_id) as total_items, SUM(m.price) AS total_sales
FROM dannys_diner.menu m
INNER JOIN dannys_diner.sales s on s.product_id = m.product_id
INNER JOIN dannys_diner.members me on me.customer_id = s.customer_id
WHERE s.order_date < me.join_date
GROUP BY s.customer_id
ORDER BY s.customer_id;




--
--9. If each $1 spent equates to 10 points and sushi has a 2x points multiplier — how
many -points would each customer have?
WITH Point_multi AS (
SELECT product_id,
```

```
CASE WHEN
product_id = 1 then price * 20
ELSE price* 10
END AS points
FROM dannys_diner.menu
)
SELECT s.customer_id, SUM(p.points) as Total_points
FROM dannys_diner.sales s
JOIN Point_multi p
ON s.product_id = p.product_id
GROUP BY s.customer_id
ORDER BY s.customer_id;

--10. In the first week after a customer joins the program (including their join
date) they earn 2x points on all items, not just sushi — how many points do customer
A and B have at the end of January?
WITH customer_points AS (
SELECT
s.customer_id,
s.order_date,
m.product_name,
m.price,
CASE
WHEN s.order_date BETWEEN me.join_date AND me.join_date + INTERVAL '6 day' THEN 2 *
m.price
ELSE m.price
END AS points
FROM dannys_diner.sales s JOIN dannys_diner.menu m ON s.product_id = m.product_id
JOIN dannys_diner.members me ON s.customer_id = me.customer_id
WHERE s.order_date BETWEEN '2021-01-01' AND '2021-01-31'
)
SELECT
customer_id,
SUM(points) AS total_points
FROM customer_points
GROUP BY customer_id;
```

--

I’m thrilled to share that I’ve just published an in-depth article diving into a fascinating SQL case study on Danny's Diner 🍣🍜🍹 from
Danny Ma
, the creator of the
hashtag
#8WeekSQLChallenge
. In this study, I explored customer behaviors, spending patterns, and the effectiveness of a customer loyalty program. 📊

🚀 I recently tackled a fascinating SQL case study from Data With Danny that focused on analyzing customer interactions and sales data at a fictional restaurant. 🍜 Here’s what I learned and practiced: ✅ Total Customer Spend: Calculated the total amount spent by each customer by joining sales and menu data, using SUM to aggregate prices. ✅ Visit Frequency: Determined how many days each customer visited the restaurant, utilizing COUNT on distinct order_date values. ✅ First Purchase Identification: Found the first item purchased by each customer by comparing order_date values. ✅ Most Purchased Item: Identified the most popular item by counting sales of each product and ordering the results. ✅ Customer Preferences: Used RANK to determine each customer’s favorite item based on purchase frequency. ✅ Post-Membership Purchases: Analyzed the first item bought by customers after joining the membership program. ✅ Pre-Membership Purchases: Investigated the last item purchased before customers joined the membership. ✅ Pre-Membership Spending: Summed up total spending and items purchased by customers before becoming members. ✅ Customer Points Calculation: Created a points system where each $1 spent equaled 10 points, with a special multiplier for sushi. ✅ Promotional Points: Calculated bonus points for new members during their first week, applying a 2x multiplier to all purchases.Throughout these analyses, I utilized advanced SQL techniques including window functions (like RANK), common table expressions (CTEs), and conditional logic with CASE statements. These methods provided a comprehensive understanding of customer behavior and spending patterns.