📄 Report Structure

Title Page

- **Project Title**: Advanced Analysis of Sales / E-commerce Dataset
- **Intern Name**, **Internship Duration**
- **CODTECH Logo** (add at top-center)

1. Introduction

- Purpose: uncover trends & patterns using SQL.
- Dataset: e-commerce/sales sample (orders, customers, products).
- Tools: PostgreSQL (or MySQL), pgAdmin / DBeaver.
- Objectives: demonstrate Window Functions, CTEs, Subqueries.

2. Dataset Overview

Describe data tables:

- **orders**: order_id, customer_id, order_date, total_amount
- **order_items**: order_id, product_id, quantity, price
- **customers**: customer_id, name, region

🔍 Analysis Sections

A. Window Functions (3+ examples)

1. **Running Total of Daily Sales**

sql

CopyEdit

```
SELECT
  order_date,
  SUM(total_amount) OVER (ORDER BY order_date) AS running_total
FROM orders
ORDER BY order_date;
```

- Uses SUM(...) OVER (ORDER BY …) for cumulative total en.wikipedia.orggithub.comen.wikipedia.orgreddit.commedium.com+4crunchydata.com+4geeksforgeeks.org+4datacamp.comen.wikipedia.org
- *Interpretation*: shows how revenue has grown day by day.

2. **Per-Customer Ranking by Lifetime Spend**

sql

CopyEdit

```
SELECT
  customer_id,
  SUM(total_amount) AS total_spent,
  RANK() OVER (ORDER BY SUM(total_amount) DESC) AS spend_rank
FROM orders
GROUP BY customer_id;
```

- Uses RANK() OVER (ORDER BY SUM(...)) to rank customers by spend
- *Insight*: identify top spenders.

3. **Month-over-Month Sales Growth**

sql

CopyEdit

```
WITH Monthly AS (
  SELECT
    date_trunc('month', order_date) AS month,
    SUM(total_amount) AS total
  FROM orders
  GROUP BY month
)
SELECT
  month,
  total,
  LAG(total,1) OVER (ORDER BY month) AS prev_total,
```

ROUND((total - LAG(total,1) OVER (ORDER BY month)) * 100.0 / LAG(total,1) OVER (ORDER BY month),2) AS pct_change

FROM Monthly;

- Combines CTE with LAG() for MoM change crunchydata.com+4crunchydata.com+4ai2sql.io+4
- *Outcome*: calculates monthly growth %.

B. CTE-Based Queries (2+ examples)

1. **Sales per Customer & Regional Average**

sql

CopyEdit

WITH CustSales AS (

SELECT customer_id, SUM(total_amount) AS total_spent

FROM orders

GROUP BY customer_id

)

SELECT

cs.customer_id,

cs.total_spent,

AVG(cs.total_spent) OVER () AS avg_spent,

cs.total_spent - AVG(cs.total_spent) OVER () AS diff_from_avg

FROM CustSales cs;

- CTE encapsulates aggregated data; window function calculates average crunchydata.com+2crunchydata.com+2ai2sql.io+2datacamp.com+8en.wikipedia.org+8datalemur.com+8towardsai.net+13medium.com+13crunchydata.com+13
- *Insight*: compares individual to average.

2. **Top Products by Quantity and Revenue**

sql

CopyEdit

```
WITH ProdStats AS (
  SELECT
    oi.product_id,
    SUM(oi.quantity) AS total_qty,
    SUM(oi.quantity * oi.price) AS total_revenue
  FROM order_items oi
  GROUP BY oi.product_id
)
SELECT
  product_id,
  total_qty,
  total_revenue,
  RANK() OVER (ORDER BY total_revenue DESC) AS rev_rank
FROM ProdStats
WHERE total_qty > 100;
```

- CTE simplifies raw stats; filter & rank via window towardsai.net+3learnsql.com+3datacamp.com+3ai2sql.io+1medium.com+1
- *Findings*: high-volume, high-revenue products.

C. Subqueries (2+ examples)

1. **Customers Above-Avg Spend**

sql

CopyEdit

```
SELECT
  customer_id,
  SUM(total_amount) AS total_spent
FROM orders
GROUP BY customer_id
```

```
HAVING SUM(total_amount) > (
  SELECT AVG(total_amount * cnt) FROM (
    SELECT customer_id, SUM(total_amount) * COUNT(*) AS total_amount
    FROM orders
    GROUP BY customer_id
  ) AS subs
);
```

- Subquery computes average to filter above-average customers [en.wikipedia.org+1reddit.com+1github.com+11towardsai.net+11datalemur.com+11]
- *Result*: list of customers spending above peer average.

2. **Correlated Subquery: Above-Department Avg Sales**

```
sql
CopyEdit
SELECT
  s.salesperson_id,
  s.total_sales
FROM (
  SELECT salesperson_id, SUM(sale_amount) AS total_sales
  FROM sales
  GROUP BY salesperson_id
) s
WHERE s.total_sales > (
  SELECT AVG(total_sales) FROM (
    SELECT salesperson_id, SUM(sale_amount) AS total_sales
    FROM sales
    GROUP BY salesperson_id
  ) AS t
);
```

- Correlated logic inside WHERE clause bigtechinterviews.com+11en.wikipedia.org+11medium.com+11medium.com+12chayansraj.medium.com+12en.wikipedia.org+12
- *Insight*: top-performing sales reps.

📊 Output Tables & Screenshots

Include screenshots (or small tables) of query results for each section.

✅ Key Findings (Bullet Summary)

- Daily sales growth shown via running totals.
- Top customers & deviations from average identified.
- Certain products outperform in both quantity and revenue.
- Month-over-month revenue trends detected (growth or decline).
- A subset of customers spends significantly above peer average.
