Using SQL Express and Basic Queries

1. Setting Up the Database and Importing Data

Follow these steps to set up the database and import the data using SQL Server Management Studio (SSMS):

1. From desktop, launch SQL Server Management Studio.
2. Choose **Connect**, accepting the default Windows Authentication.
3. Right-click **Databases** and choose **New Database...** Call it `data`.
4. Right-click the `data` database and choose **Tasks**, then **Import Flat File**.
5. Click **Next** on the wizard start screen.
6. For the input file, browse to `Desktop/dpm/product.csv` and call the table `productdata`.
7. Click **Next**. Preview Data and choose **NEXT**.
8. Choose **NEXT** on the Modify Columns page.
9. Click **FINISH** on the summary page, then **CLOSE**.
10. Expand **Tables** in the `data` database.
11. Navigate to `dbo.productdata`, right-click, and **SELECT TOP 100 rows**.

Review the data and discuss.

2. Basic Query: View All Data

Choose **New Query** from the Menu Ribbon.

This query shows all rows and all columns. Click on **Message** to see the row count returned.

```
SELECT *
FROM productdata;
```

3. Identify Columns Without Returning Data

Use the following command to identify the columns and their data types without returning the actual data.

```
sp_help productdata
```

4. Selecting Only the Columns You Need

Why this matters

Good analysts:

- Reduce noise
- Focus on the question they're answering

Use this query to select specific columns:

```
SELECT
    Date,
    User_ID,
    Plan_Type,
    Session_Duration_Min,
    Revenue
FROM productdata;
```

5. Filtering Data With WHERE

Example: Look at free users only

This query uses the `WHERE` clause to filter the data for only 'Free' users.

```
SELECT *
FROM productdata
WHERE Plan_Type = 'Free';
```

What this shows

- Behaviour of users before monetisation.
- Ideal for analysing conversion opportunities.

Example: Long sessions (high engagement)

```
SELECT *
FROM productdata
WHERE Session_Duration_Min >= 30;
```

What this shows

- Highly engaged users
- Potential power users or advocates

Product question answered:
“What does strong engagement actually look like in our data?”

6. Counting Users and Sessions

Total Sessions:

```
SELECT COUNT(*) AS Total_Sessions
FROM productdata;
```

Count Unique Users:

```
SELECT COUNT(DISTINCT User_ID) AS Unique_Users
FROM productdata;
```

What this shows

- Overall product usage
- Difference between **traffic** and **users**

Key DPM insight:
A growing session count does not always mean a growing user base.

7. Group By To Consolidate Data

Users by Plan Type

```
SELECT
    Plan_Type,
    COUNT(DISTINCT User_ID) AS Users
FROM productdata
GROUP BY Plan_Type;
```

Average Session Duration by Plan Type

```
SELECT
    Plan_Type,
    AVG(Session_Duration_Min) AS Avg_Session_Minutes
FROM productdata
GROUP BY Plan_Type;
```

What this shows

- Engagement differences by pricing tier
- Whether paid users get more value

Product decision supported:
“Are premium users actually more engaged?”

8. Revenue Focus

Revenue By Plan

```
SELECT
  Plan_Type,
  SUM(Revenue) AS Total_Revenue
FROM productdata
GROUP BY Plan_Type;
```

What this shows

- Which plan funds the product
- Revenue concentration risk

Revenue By User

```
SELECT
  User_ID,
  SUM(Revenue) AS User_Revenue
FROM productdata
GROUP BY User_ID
```

```
HAVING SUM(Revenue) >0
ORDER BY User_Revenue DESC;
```

What this shows

- High-value users
- Candidates for retention or concierge support

Commercial thinking:
Not all users are equal — SQL helps you prove that.

9. Churn Risk Insights

Average churn risk by plan

```
SELECT
  Plan_Type,
  AVG(Churn_Risk_Score) AS Avg_Churn_Risk
FROM productdata
GROUP BY Plan_Type;
```

What this shows

- Which customer segments are most fragile
- Where retention work should focus

10. NTILE and Churn Risk

```
USE data;
GO

/*
Using the NTILE function we can group chrun risk , in this case.
NTILE(4) Is really a quartile where each quartile represents 25%
so the total rows. So 250 in each quartile
Quartile 1 will have the lowest risk and 4 the highest
*/

SELECT
  Plan_Type,
  Churn_Risk_Score,
  NTILE(4) OVER (PARTITION BY Plan_Type ORDER BY Churn_Risk_Score)
AS Churn_Risk_Quartile
FROM productdata;
```

11. Group By Region and Plan

```
SELECT
    Region,
    Plan_Type,
    COUNT(*) AS Sessions
FROM productdata
GROUP BY Region, Plan_Type
ORDER BY Region;
```

What this shows

- Geographic monetisation patterns
- Regional differences in plan adoption

12. SQL Pivot Tables

```
SELECT *
FROM
(
    SELECT Region, Plan_Type, Revenue
    FROM productdata
) AS SourceTable
PIVOT
(
    SUM(Revenue)
    FOR Plan_Type IN ([Free], [Pro], [Premium])
) AS PivotTable;
```

This is really powerful without having to drop out to Excel

13. Create View

Views are stored SQL Code. As `SELECT` statements become more complex then views can abstract the complexity

Create View RevenueRegion AS
```
SELECT *
FROM
(
    SELECT Region, Plan_Type, Revenue
    FROM productdata
) AS SourceTable
PIVOT
(
    SUM(Revenue)
    FOR Plan_Type IN ([Free], [Pro], [Premium])
) AS PivotTable;
```

Then

```
SELECT * FROM RevenueRegion
```