**A blue and white logo

AI-generated content may be incorrect.SQL PRETRAINING SESSION 3**

**V1: CUSTOMER RISK & VALUE SEGMENTATION  
FOUNDATIONS**

**Subtitle**: Advanced CASE Logic, Behavioral Profiling & Business Recommendations

**Context:** This pretraining document is part of the Skill AI Data Analyst Track.  
It focuses on advanced SQL techniques for customer scoring, segmentation, and portfolio risk classification in the EduFin loan crisis scenario. You’ll use complex CASE statements, nested aggregations, and multi-factor scoring models to transform raw demographic and loan data into actionable business intelligence.  
By the end, you’ll know how to identify high-value customers, flag high-risk profiles, and generate targeted business recommendations.

# TABLE OF CONTENTS

1. Session Overview
2. Learning Objectives
3. Part 1: Multi-Layered Risk Scoring
4. Part 2: Customer Segmentation with Business Rules
5. Part 3: Manual Pivot Operations
6. Part 4: Cohort Analysis Patterns
7. Part 5: Nested Aggregations & Ranking
8. Conclusion & Next Steps

# **1. SESSION OVERVIEW**

**Session Focus:**  
Advanced Business Analytics — “How do we score, segment, and monitor our customers for better decision-making?”

**Time Investment:**  
6–7 hours of focused learning

**Business Context:**  
EduFin’s leadership needs sharper insights into portfolio health, customer value, and operational risks.  
In this session, you’ll move beyond simple aggregation into **multi-dimensional analysis** — building weighted risk scores, segmenting customers using advanced rules, analyzing behavior over time, and validating data integrity.  
By combining business logic with SQL best practices, you’ll learn how to transform raw data into **actionable intelligence** for executives.

## **2. LEARNING OBJECTIVES**

You will master:

* ✅ Weighted scoring models using multiple business dimensions
* ✅ Complex CASE statements for multi-factor classification
* ✅ Advanced segmentation logic tied to income, job type, and loan history
* ✅ Manual cross-tabulation without the PIVOT function
* ✅ Cohort analysis for tracking customer behavior over time
* ✅ Nested aggregations and ranking within partitions
* ✅ Performance tuning techniques for large datasets
* ✅ Data quality validation patterns to ensure reliable analytics

# PART 1: MULTI-DIMENSIONAL ANALYSIS

## 1.1 Advanced GROUP BY Techniques

**Purpose:**  
To segment customers based on multiple attributes — such as **employment type** and **income bracket** — so that patterns can be detected for targeted interventions.

**Learning Goal:**  
You will learn how to:

* Apply **multi-level grouping**
* Use **CASE statements** inside GROUP BY
* Combine dimensions for **multi-attribute segmentation**

**-- Basic multi-dimensional grouping**

SELECT

c.employment\_type,

CASE

WHEN c.annual\_income < 500000 THEN 'Low Income'

WHEN c.annual\_income < 1200000 THEN 'Medium Income'

ELSE 'High Income'

END AS income\_segment,

COUNT(\*) AS customer\_count,

ROUND(AVG(c.annual\_income), 2) AS avg\_income,

MIN(l.application\_date) AS first\_customer,

MAX(l.application\_date) AS latest\_customer

FROM customers c

LEFT JOIN loans l

ON c.customer\_id = l.customer\_id

WHERE c.annual\_income IS NOT NULL

GROUP BY

c.employment\_type,

CASE

WHEN c.annual\_income < 500000 THEN 'Low Income'

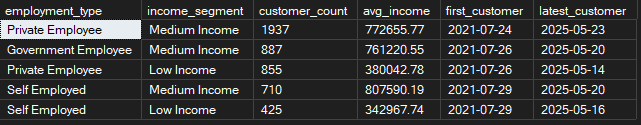
WHEN c.annual\_income < 1200000 THEN 'Medium Income'

ELSE 'High Income'

END

ORDER BY customer\_count DESC;

Output:



**-- Advanced multi-table segmentation**

SELECT TOP 5

c.employment\_type,

CASE

WHEN c.annual\_income < 500000 THEN 'Low Income'

WHEN c.annual\_income < 1200000 THEN 'Medium Income'

ELSE 'High Income'

END AS income\_segment,

dc.tier\_classification AS city\_tier,

COUNT(DISTINCT c.customer\_id) AS unique\_customers,

COUNT(l.loan\_id) AS total\_loans,

ROUND(AVG(l.loan\_amount), 2) AS avg\_loan\_amount,

SUM(l.loan\_amount) AS total\_portfolio

FROM customers c

JOIN dim\_city dc

ON c.city\_id = dc.city\_id

LEFT JOIN loans l

ON c.customer\_id = l.customer\_id

AND l.disbursement\_date IS NOT NULL

GROUP BY

c.employment\_type,

CASE

WHEN c.annual\_income < 500000 THEN 'Low Income'

WHEN c.annual\_income < 1200000 THEN 'Medium Income'

ELSE 'High Income'

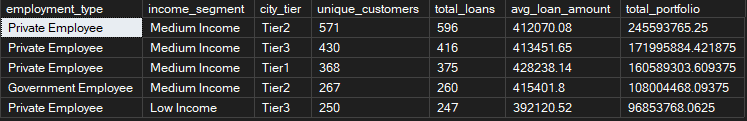
END,

dc.tier\_classification

HAVING COUNT(DISTINCT c.customer\_id) >= 5

ORDER BY total\_portfolio DESC;

Output:



# PART 2: STATISTICAL ANALYSIS FUNCTIONS

## 2.1 Percentiles and Distribution Analysis

**Purpose:**  
To understand the distribution of customer and loan characteristics across different segments, identify outliers, and detect performance patterns for targeted decision-making.

**Learning Goal:**  
You will learn how to:

* Use percentile functions (PERCENTILE\_CONT) to compute quartiles and medians
* Combine window functions with PARTITION BY for segmented statistics
* Compare distributions across employment types and income brackets
* Detect potential outliers using min, max, and standard deviation metrics

**-- Income distribution analysis by employment type**

SELECT DISTINCT

employment\_type,

COUNT(\*) OVER (PARTITION BY employment\_type) AS customer\_count,

MIN(annual\_income) OVER (PARTITION BY employment\_type) AS min\_income,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY annual\_income)

OVER (PARTITION BY employment\_type) AS income\_25th\_percentile,

PERCENTILE\_CONT(0.5) WITHIN GROUP (ORDER BY annual\_income)

OVER (PARTITION BY employment\_type) AS median\_income,

ROUND(AVG(annual\_income) OVER (PARTITION BY employment\_type), 2) AS mean\_income,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY annual\_income)

OVER (PARTITION BY employment\_type) AS income\_75th\_percentile,

MAX(annual\_income) OVER (PARTITION BY employment\_type) AS max\_income,

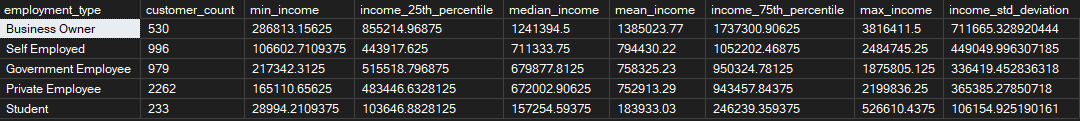
STDEV(annual\_income) OVER (PARTITION BY employment\_type) AS income\_std\_deviation

FROM customers

WHERE annual\_income IS NOT NULL

ORDER BY median\_income DESC;

**Output:**



**-- Loan amount distribution by customer characteristics**

SELECT DISTINCT TOP 5

c.employment\_type,

CASE

WHEN c.annual\_income < 500000 THEN 'Low Income'

WHEN c.annual\_income < 1200000 THEN 'Medium Income'

ELSE 'High Income'

END AS income\_segment,

COUNT(l.loan\_id) OVER (PARTITION BY c.employment\_type,

CASE

WHEN c.annual\_income < 500000 THEN 'Low Income'

WHEN c.annual\_income < 1200000 THEN 'Medium Income'

ELSE 'High Income'

END) AS loan\_count,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY l.loan\_amount)

OVER (PARTITION BY c.employment\_type,

CASE

WHEN c.annual\_income < 500000 THEN 'Low Income'

WHEN c.annual\_income < 1200000 THEN 'Medium Income'

ELSE 'High Income'

END) AS loan\_25th\_percentile,

PERCENTILE\_CONT(0.5) WITHIN GROUP (ORDER BY l.loan\_amount)

OVER (PARTITION BY c.employment\_type,

CASE

WHEN c.annual\_income < 500000 THEN 'Low Income'

WHEN c.annual\_income < 1200000 THEN 'Medium Income'

ELSE 'High Income'

END) AS median\_loan,

AVG(l.loan\_amount) OVER (PARTITION BY c.employment\_type,

CASE

WHEN c.annual\_income < 500000 THEN 'Low Income'

WHEN c.annual\_income < 1200000 THEN 'Medium Income'

ELSE 'High Income'

END) AS mean\_loan,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY l.loan\_amount)

OVER (PARTITION BY c.employment\_type,

CASE

WHEN c.annual\_income < 500000 THEN 'Low Income'

WHEN c.annual\_income < 1200000 THEN 'Medium Income'

ELSE 'High Income'

END) AS loan\_75th\_percentile,

MAX(l.loan\_amount) OVER (PARTITION BY c.employment\_type,

CASE

WHEN c.annual\_income < 500000 THEN 'Low Income'

WHEN c.annual\_income < 1200000 THEN 'Medium Income'

ELSE 'High Income'

END) AS max\_loan

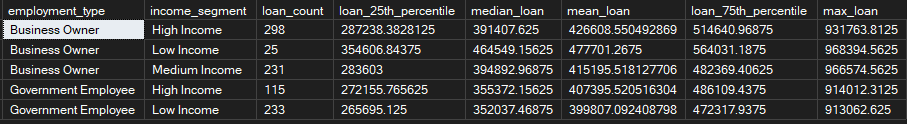
FROM customers c

JOIN loans l

ON c.customer\_id = l.customer\_id

WHERE l.disbursement\_date IS NOT NULL;

Output:



## 2.2 Variance and Risk Analysis

**Purpose:**To measure portfolio risk using statistical variance, classifying customer segments into stability tiers for better risk management.

**Learning Goal:**You will learn how to:

* Calculate standard deviation (STDEV) to assess portfolio volatility
* Create conditional classifications (High/Medium/Low variance)
* Group and segment data to isolate risky customer categories
* Apply HAVING clauses to ensure statistical relevance in analysis

**-- Customer portfolio risk analysis**

SELECT TOP 5

c.employment\_type,

CASE

WHEN c.annual\_income < 500000 THEN 'Low Income'

WHEN c.annual\_income < 1200000 THEN 'Medium Income'

ELSE 'High Income'

END AS income\_segment,

COUNT(DISTINCT c.customer\_id) AS customers,

COUNT(l.loan\_id) AS total\_loans,

ROUND(AVG(l.loan\_amount), 2) AS avg\_loan\_amount,

STDEV(l.loan\_amount) AS loan\_amount\_stddev,

MIN(l.loan\_amount) AS min\_loan,

MAX(l.loan\_amount) AS max\_loan,

CASE

WHEN STDEV(l.loan\_amount) > AVG(l.loan\_amount) \* 0.5

THEN 'HIGH VARIANCE - Unpredictable Segment'

WHEN STDEV(l.loan\_amount) > AVG(l.loan\_amount) \* 0.25

THEN 'MEDIUM VARIANCE - Moderate Risk'

ELSE 'LOW VARIANCE - Stable Segment'

END AS risk\_classification

FROM customers c

JOIN loans l

ON c.customer\_id = l.customer\_id

WHERE l.disbursement\_date IS NOT NULL

GROUP BY

c.employment\_type,

CASE

WHEN c.annual\_income < 500000 THEN 'Low Income'

WHEN c.annual\_income < 1200000 THEN 'Medium Income'

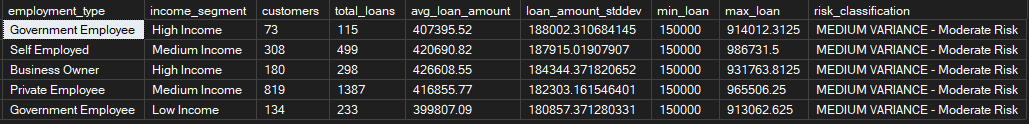
ELSE 'High Income'

END

HAVING COUNT(l.loan\_id) >= 15

ORDER BY loan\_amount\_stddev DESC;

Output:



# PART 3: ADVANCED CASE STATEMENTS FOR CUSTOMER SCORING

## 3.1 Multi-Layered Risk Scoring

**Purpose:**  
To create comprehensive customer risk scores by combining multiple factors such as income, employment type, geographic classification, and tenure, enabling a more accurate assessment of creditworthiness.

**Learning Goal:**  
You will learn how to:

* Use multiple CASE statements to assign scores to different attributes
* Apply weighted scoring for composite risk assessment
* Incorporate aggregated values (e.g., tenure from earliest loan date) into scoring
* Classify customers into qualitative categories based on total scores

**-- Complex customer risk scoring**

SELECT TOP 5

c.customer\_id,

c.full\_name,

c.employment\_type,

c.annual\_income,

-- Income score (30% weight)

CASE

WHEN c.annual\_income >= 1500000 THEN 30

WHEN c.annual\_income >= 800000 THEN 25

WHEN c.annual\_income >= 400000 THEN 20

WHEN c.annual\_income >= 200000 THEN 15

ELSE 10

END AS income\_score,

-- Employment score (25% weight)

CASE

WHEN c.employment\_type = 'Government' THEN 25

WHEN c.employment\_type = 'Private - MNC' THEN 23

WHEN c.employment\_type = 'Private - Local' THEN 20

WHEN c.employment\_type = 'Self Employed' THEN 15

ELSE 10

END AS employment\_score,

-- Geographic score (20% weight)

CASE

WHEN dc.tier\_classification = 'Tier1' THEN 15

WHEN dc.tier\_classification = 'Tier2' THEN 12

ELSE 8

END AS geographic\_score,

-- Tenure score (10% weight) - using loan disbursement date as proxy

CASE

WHEN DATEDIFF(YEAR, MIN(l.disbursement\_date), GETDATE()) >= 3 THEN 10

WHEN DATEDIFF(YEAR, MIN(l.disbursement\_date), GETDATE()) >= 1 THEN 8

ELSE 5

END AS tenure\_score

FROM customers c

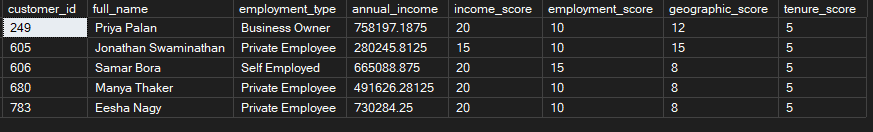
INNER JOIN dim\_city dc ON c.city\_id = dc.city\_id

LEFT JOIN loans l ON c.customer\_id = l.customer\_id

WHERE c.annual\_income IS NOT NULL

GROUP BY c.customer\_id, c.full\_name, c.employment\_type, c.annual\_income, dc.tier\_classification;

**Output:**

****

**-- Complete risk assessment with final scoring**

WITH customer\_scores AS (

SELECT

c.customer\_id,

c.full\_name,

CASE

WHEN c.annual\_income >= 1500000 THEN 30

WHEN c.annual\_income >= 800000 THEN 25

WHEN c.annual\_income >= 400000 THEN 20

WHEN c.annual\_income >= 200000 THEN 15

ELSE 10

END AS income\_score,

CASE

WHEN c.employment\_type = 'Government' THEN 25

WHEN c.employment\_type = 'Private - MNC' THEN 23

WHEN c.employment\_type = 'Private - Local' THEN 20

WHEN c.employment\_type = 'Self Employed' THEN 15

ELSE 10

END AS employment\_score,

CASE

WHEN dc.tier\_classification = 'Tier1' THEN 15

WHEN dc.tier\_classification = 'Tier2' THEN 12

ELSE 8

END AS geographic\_score,

CASE

WHEN DATEDIFF(YEAR, MIN(l.disbursement\_date), GETDATE()) >= 3 THEN 10

WHEN DATEDIFF(YEAR, MIN(l.disbursement\_date), GETDATE()) >= 1 THEN 8

ELSE 5

END AS tenure\_score

FROM customers c

INNER JOIN dim\_city dc ON c.city\_id = dc.city\_id

LEFT JOIN loans l ON c.customer\_id = l.customer\_id

WHERE c.annual\_income IS NOT NULL

GROUP BY c.customer\_id, c.full\_name, c.annual\_income, c.employment\_type, dc.tier\_classification

)

SELECT TOP 5

customer\_id,

full\_name,

(income\_score + employment\_score + geographic\_score + tenure\_score) AS total\_score,

CASE

WHEN (income\_score + employment\_score + geographic\_score + tenure\_score) >= 75

THEN 'EXCELLENT - Prime Customer'

WHEN (income\_score + employment\_score + geographic\_score + tenure\_score) >= 60

THEN 'GOOD - Standard Approval'

WHEN (income\_score + employment\_score + geographic\_score + tenure\_score) >= 45

THEN 'FAIR - Enhanced Due Diligence'

ELSE 'POOR - High Risk Customer'

END AS risk\_category

FROM customer\_scores

ORDER BY total\_score DESC;

Output:

A black screen with white text

AI-generated content may be incorrect.

## 3.2 Segmentation with Business Rules

**Purpose:**  
To define meaningful customer profiles based on income and employment combinations, enabling targeted strategies for acquisition, retention, and risk control.

**Learning Goal:**  
You will learn how to:

* Combine conditional logic with business-defined thresholds to create customer profiles
* Aggregate loan and default data for each segment
* Calculate portfolio metrics such as average income, loan count, portfolio size, and default rate
* Rank and compare segments based on performance indicators

**-- Advanced customer segmentation with business logic**

SELECT

customer\_profile,

COUNT(\*) AS customer\_count,

AVG(annual\_income) AS avg\_income,

COUNT(total\_loans) AS customers\_with\_loans,

AVG(total\_loan\_amount) AS avg\_customer\_portfolio,

COUNT(CASE WHEN has\_default = 1 THEN 1 END) AS customers\_with\_defaults,

ROUND(

COUNT(CASE WHEN has\_default = 1 THEN 1 END) \* 100 / COUNT(\*),

2

) AS default\_rate\_percent

FROM (

SELECT

c.customer\_id,

c.annual\_income,

COUNT(l.loan\_id) AS total\_loans,

SUM(l.loan\_amount) AS total\_loan\_amount,

MAX(CASE WHEN l.loan\_status = 'Defaulted' THEN 1 ELSE 0 END) AS has\_default,

CASE

WHEN c.annual\_income >= 1200000 AND c.employment\_type = 'Government'

THEN 'Government High Earner'

WHEN c.annual\_income >= 1200000 AND c.employment\_type LIKE 'Private%'

THEN 'Private Sector Executive'

WHEN c.annual\_income >= 1200000 AND c.employment\_type = 'Self Employed'

THEN 'Successful Entrepreneur'

WHEN c.annual\_income BETWEEN 500000 AND 1199999 AND c.employment\_type = 'Government'

THEN 'Government Mid-Level'

WHEN c.annual\_income BETWEEN 500000 AND 1199999 AND c.employment\_type LIKE 'Private%'

THEN 'Private Sector Professional'

WHEN c.annual\_income BETWEEN 500000 AND 1199999 AND c.employment\_type = 'Self Employed'

THEN 'Growing Business Owner'

WHEN c.annual\_income BETWEEN 200000 AND 499999

THEN 'Middle Income Earner'

ELSE 'Entry Level Professional'

END AS customer\_profile

FROM customers c

LEFT JOIN loans l ON c.customer\_id = l.customer\_id AND l.disbursement\_date IS NOT NULL

WHERE c.annual\_income IS NOT NULL

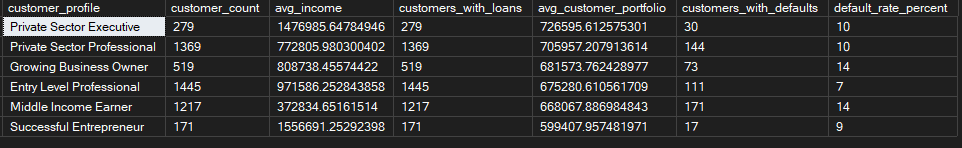
GROUP BY c.customer\_id, c.annual\_income, c.employment\_type

) customer\_analysis

GROUP BY customer\_profile

ORDER BY avg\_customer\_portfolio DESC;

Output:



# PART 4: CROSS-TABULATION AND PIVOT ANALYSIS

## 4.1 Manual Pivot Operations

**Purpose:**  
To create cross-tabulation reports that summarize key metrics (such as income distribution by employment type) in a pivot-style format for decision-makers.

**Learning Goal:**  
You will learn how to:

* Use conditional aggregation (COUNT with CASE WHEN) to produce pivot-like summaries
* Group by categorical dimensions to display multiple metrics in one row
* Calculate percentage contributions within groups
* Rank categories based on total counts or proportions

**-- Employment type vs. Income bracket cross-tabulation**

SELECT

c.employment\_type,

COUNT(CASE WHEN c.annual\_income >= 1500000 THEN 1 END) AS High\_Income\_Count,

COUNT(CASE WHEN c.annual\_income BETWEEN 1000000 AND 1499999 THEN 1 END) AS Upper\_Middle\_Count,

COUNT(CASE WHEN c.annual\_income BETWEEN 600000 AND 999999 THEN 1 END) AS Middle\_Count,

COUNT(CASE WHEN c.annual\_income < 600000 THEN 1 END) AS Low\_Income\_Count,

COUNT(\*) AS Total\_Customers,

ROUND(COUNT(CASE WHEN c.annual\_income >= 1500000 THEN 1 END) \* 100 / COUNT(\*), 1) AS High\_Income\_Percent

FROM customers c

WHERE c.annual\_income IS NOT NULL

GROUP BY c.employment\_type

ORDER BY Total\_Customers DESC;

Output:

A black screen with white text

AI-generated content may be incorrect.

## 4.2 Cohort Analysis Patterns

**Purpose:**  
To track and compare customer performance over time by grouping them into cohorts based on their first activity, helping identify behavioral trends and risk patterns.

**Learning Goal:**  
You will learn how to:

* Use date functions to assign customers to cohorts by year and quarter
* Aggregate cohort-level loan and default statistics
* Calculate cohort default rates and assess risk categories
* Apply filtering to focus on statistically significant cohorts

**-- Customer first-loan cohort analysis**

WITH registration\_cohorts AS (

SELECT

c.customer\_id,

c.full\_name,

c.employment\_type,

c.annual\_income,

MIN(l.disbursement\_date) AS first\_activity\_date,

YEAR(MIN(l.disbursement\_date)) AS registration\_year,

CONCAT('Q', CEILING(MONTH(MIN(l.disbursement\_date))/3.0), '-', YEAR(MIN(l.disbursement\_date))) AS registration\_quarter

FROM customers c

LEFT JOIN loans l ON c.customer\_id = l.customer\_id

WHERE c.annual\_income IS NOT NULL

GROUP BY c.customer\_id, c.full\_name, c.employment\_type, c.annual\_income

),

cohort\_loan\_behavior AS (

SELECT

rc.registration\_quarter,

rc.employment\_type,

COUNT(DISTINCT rc.customer\_id) AS cohort\_customers,

COUNT(l.loan\_id) AS total\_loans,

AVG(l.loan\_amount) AS avg\_loan\_amount,

COUNT(CASE WHEN l.loan\_status = 'Defaulted' THEN 1 END) AS defaults,

ROUND(

COUNT(CASE WHEN l.loan\_status = 'Defaulted' THEN 1 END) \* 100/ NULLIF(COUNT(l.loan\_id), 0),

2

) AS cohort\_default\_rate

FROM registration\_cohorts rc

LEFT JOIN loans l ON rc.customer\_id = l.customer\_id AND l.disbursement\_date IS NOT NULL

GROUP BY rc.registration\_quarter, rc.employment\_type

)

SELECT TOP 5

registration\_quarter,

employment\_type,

cohort\_customers,

total\_loans,

ROUND(avg\_loan\_amount, 0) AS avg\_loan\_formatted,

defaults,

COALESCE(cohort\_default\_rate, 0) AS default\_rate,

CASE

WHEN COALESCE(cohort\_default\_rate, 0) > 15 THEN 'HIGH RISK COHORT'

WHEN COALESCE(cohort\_default\_rate, 0) > 8 THEN 'MEDIUM RISK COHORT'

WHEN total\_loans > 0 THEN 'LOW RISK COHORT'

ELSE 'NO LOAN ACTIVITY'

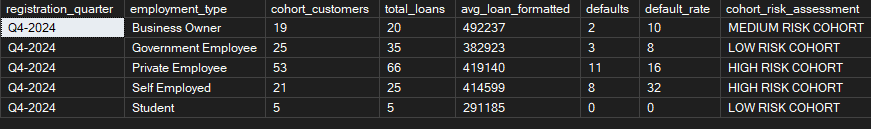
END AS cohort\_risk\_assessment

FROM cohort\_loan\_behavior

WHERE cohort\_customers >= 5

ORDER BY registration\_quarter DESC, employment\_type;

Output:



# PART 5: ADVANCED AGGREGATION PATTERNS

## 5.1 Nested Aggregations

**Purpose:**  
To generate deeper business insights by aggregating already aggregated results, allowing for advanced portfolio and segment-level analysis.

**Learning Goal:**  
You will learn how to:

* Create multi-step aggregations using Common Table Expressions (CTEs)
* Summarize individual customer metrics into segment-level statistics
* Calculate segment penetration, high-value customer counts, and default rates
* Apply classification logic to group segments by performance and risk

**-- Customer portfolio concentration analysis**

WITH customer\_portfolios AS (

SELECT

c.customer\_id,

c.employment\_type,

c.annual\_income,

COUNT(l.loan\_id) AS loan\_count,

SUM(l.loan\_amount) AS total\_borrowed,

AVG(l.loan\_amount) AS avg\_loan\_size,

MAX(CASE WHEN l.loan\_status = 'Defaulted' THEN 1 ELSE 0 END) AS has\_default

FROM customers c

LEFT JOIN loans l

ON c.customer\_id = l.customer\_id

AND l.disbursement\_date IS NOT NULL

WHERE c.annual\_income IS NOT NULL

GROUP BY c.customer\_id, c.employment\_type, c.annual\_income

),

portfolio\_segments AS (

SELECT

employment\_type,

COUNT(\*) AS customers\_in\_segment,

COUNT(CASE WHEN loan\_count > 0 THEN 1 END) AS customers\_with\_loans,

AVG(COALESCE(total\_borrowed, 0)) AS avg\_customer\_portfolio,

AVG(COALESCE(loan\_count, 0)) AS avg\_loans\_per\_customer,

COUNT(CASE WHEN has\_default = 1 THEN 1 END) AS customers\_with\_defaults,

COUNT(CASE WHEN total\_borrowed > 2000000 THEN 1 END) AS high\_value\_customers,

COUNT(CASE WHEN loan\_count >= 3 THEN 1 END) AS multi\_loan\_customers

FROM customer\_portfolios

GROUP BY employment\_type

)

SELECT

employment\_type,

customers\_in\_segment,

customers\_with\_loans,

ROUND(avg\_customer\_portfolio, 0) AS avg\_portfolio\_formatted,

ROUND(avg\_loans\_per\_customer, 2) AS avg\_loans\_per\_customer,

customers\_with\_defaults,

ROUND(customers\_with\_defaults \* 100 / NULLIF(customers\_with\_loans, 0), 2) AS segment\_default\_rate,

high\_value\_customers,

multi\_loan\_customers,

ROUND(customers\_with\_loans \* 100/ customers\_in\_segment, 1) AS loan\_penetration\_rate,

CASE

WHEN customers\_with\_defaults \* 100.0 / NULLIF(customers\_with\_loans, 0) > 20

THEN 'HIGH RISK SEGMENT'

WHEN avg\_customer\_portfolio > 1500000 AND customers\_with\_defaults \* 100.0 / NULLIF(customers\_with\_loans, 0) < 5

THEN 'PREMIUM SEGMENT'

WHEN customers\_with\_loans \* 100.0 / customers\_in\_segment > 80

THEN 'HIGH ENGAGEMENT SEGMENT'

ELSE 'STANDARD SEGMENT'

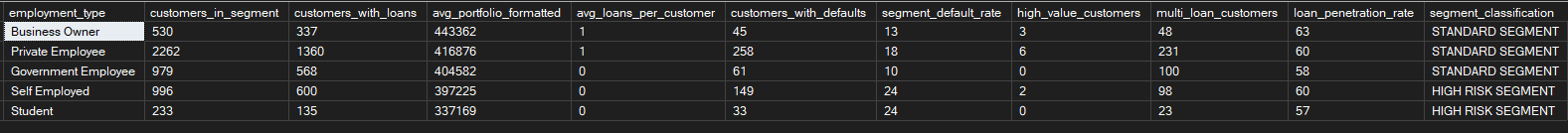
END AS segment\_classification

FROM portfolio\_segments

WHERE customers\_in\_segment >= 10

ORDER BY avg\_customer\_portfolio DESC;

Output:



## 5.2 Ranking and Percentile Analysis

**Purpose:**  
To identify top and bottom performers within defined groups, supporting prioritization for engagement, rewards, or risk control.

**Learning Goal:**  
You will learn how to:

* Rank customers within categories using ROW\_NUMBER and NTILE
* Calculate deciles for relative positioning
* Combine performance metrics into composite scoring
* Apply classification rules to label customers by value or risk level

**-- Customer ranking within employment type**

WITH customer\_metrics AS (

SELECT

c.customer\_id,

c.full\_name,

c.employment\_type,

c.annual\_income,

dc.tier\_classification,

COUNT(l.loan\_id) AS loan\_count,

SUM(l.loan\_amount) AS total\_portfolio,

COUNT(CASE WHEN l.loan\_status = 'Defaulted' THEN 1 END) AS default\_count,

CASE

WHEN COUNT(l.loan\_id) = 0 THEN 0

ELSE SUM(l.loan\_amount) \* (1 - COUNT(CASE WHEN l.loan\_status = 'Defaulted' THEN 1 END) \* 1.0 / COUNT(l.loan\_id))

END AS customer\_value\_score

FROM customers c

INNER JOIN dim\_city dc ON c.city\_id = dc.city\_id

LEFT JOIN loans l

ON c.customer\_id = l.customer\_id

AND l.disbursement\_date IS NOT NULL

WHERE c.annual\_income IS NOT NULL

GROUP BY c.customer\_id, c.full\_name, c.employment\_type, c.annual\_income, dc.tier\_classification

)

SELECT

customer\_id,

full\_name,

employment\_type,

ROUND(annual\_income, 0) AS income\_formatted,

tier\_classification,

loan\_count,

ROUND(total\_portfolio, 0) AS portfolio\_formatted,

default\_count,

ROUND(customer\_value\_score, 0) AS value\_score,

ROW\_NUMBER() OVER (PARTITION BY employment\_type ORDER BY customer\_value\_score DESC) AS rank\_in\_employment\_type,

NTILE(10) OVER (PARTITION BY employment\_type ORDER BY customer\_value\_score DESC) AS segment\_decile,

ROW\_NUMBER() OVER (ORDER BY customer\_value\_score DESC) AS overall\_rank,

CASE

WHEN customer\_value\_score > 5000000 THEN 'VIP CUSTOMER'

WHEN customer\_value\_score > 2000000 THEN 'HIGH VALUE CUSTOMER'

WHEN customer\_value\_score > 500000 THEN 'STANDARD CUSTOMER'

WHEN loan\_count > 0 THEN 'DEVELOPING CUSTOMER'

ELSE 'PROSPECT'

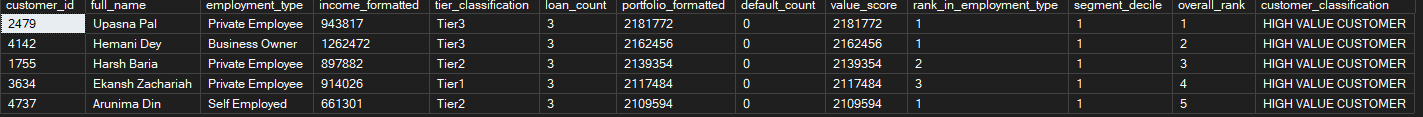
END AS customer\_classification

FROM customer\_metrics

WHERE customer\_value\_score > 0

ORDER BY customer\_value\_score DESC;

Output:



© 2025 Skill AI Path. All Rights Reserved.  
Licensed for educational and internal business use.  
Attribution required when sharing. Commercial redistribution prohibited.

Contact Us - [tech@skillaipath.com](mailto:tech@skillaipath.com) | [skillaipath.com](http://skillaipath.com/)