**A blue and white logo

AI-generated content may be incorrect.SQL PRETRAINING SESSION 4**

**V1: INSTITUTION PARTNERSHIP ANALYSIS FOUNDATIONS**

**Subtitle**: Advanced Window Functions, Competitive Positioning & Partnership Strategy in Lending Analytics

**Registered: Krishnav Tech**

**Initiative: Skill AI**

**Context:** This pretraining document is part of the **Skill AI Data Analyst Track**.  
It covers advanced SQL window function techniques, competitive positioning analysis, and institutional performance benchmarking for the EduFin loan crisis.  
You’ll write SQL queries that leverage ranking, percentile, quartile, rolling calculations, and benchmarking to evaluate lending partners.  
By the end, you’ll be able to identify high-performing institutions, detect risk-prone partners, and recommend strategic partnership decisions.

**Organization:** Krishnav Tech | Skill AI Path

# TABLE OF CONTENTS

1. Session Overview
2. Learning Objectives
3. Part 1: Advanced Ranking Functions
4. Part 2: Percentiles & Performance Segmentation
5. Part 3: Rolling & Cumulative Metrics
6. Part 4: Competitive Positioning & Benchmarking
7. Skill Check & Assessment
8. Conclusion & Next Steps

# 1. SESSION OVERVIEW

**Session Focus:**  
Institution Partnership Analysis — “Who are our best and worst lending partners?”

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

**Business Context:**  
EduFin’s leadership wants to improve loan portfolio performance by partnering only with high-quality institutions.  
In this session, you’ll use advanced SQL window functions to rank institutions, calculate rolling performance metrics, segment them by risk and volume, and perform detailed competitive benchmarking.  
By the end, you’ll know exactly which institutions are market leaders, which are high-risk, and which offer the best growth opportunities.

## **2. LEARNING OBJECTIVES**

You will master:

* ✅ RANK(), DENSE\_RANK(), and NTILE() for competitive positioning
* ✅ Rolling and cumulative aggregations with ROWS and UNBOUNDED PRECEDING
* ✅ Percentile and quartile analysis for risk segmentation
* ✅ Multi-metric benchmarking using partitioned window functions
* ✅ Combining portfolio size, default rates, and efficiency scores into composite rankings
* ✅ Translating analytics into actionable partnership recommendations

# PART 1: WINDOW FUNCTION FUNDAMENTALS

## 1.1 Basic Ranking Functions

**Purpose:**  
To rank institutions based on their loan volume and portfolio size, helping identify top and bottom performers.

**Learning Objective:**  
Master the use of ROW\_NUMBER, RANK, and DENSE\_RANK functions for ranking, and apply PARTITION BY for tier-based rankings.

**-- Basic ranking of institutions by loan volume**

SELECT

institution\_name,

institution\_type,

COUNT(loan\_id) as total\_loans,

SUM(loan\_amount) as total\_portfolio,

AVG(loan\_amount) as avg\_loan\_size,

-- Different ranking methods

ROW\_NUMBER() OVER (ORDER BY SUM(loan\_amount) DESC) as portfolio\_rank,

RANK() OVER (ORDER BY SUM(loan\_amount) DESC) as portfolio\_rank\_with\_ties,

DENSE\_RANK() OVER (ORDER BY SUM(loan\_amount) DESC) as dense\_portfolio\_rank

FROM institutions i

INNER JOIN loans l ON i.institution\_id = l.institution\_id

WHERE l.disbursement\_date IS NOT NULL

GROUP BY institution\_name, institution\_type

ORDER BY total\_portfolio DESC;

**Output:**



**-- Ranking within tiers (partitioned ranking)**

SELECT

institution\_name,

institution\_type,

COUNT(loan\_id) as loan\_count,

FORMAT(SUM(loan\_amount), 'C0', 'en-IN') as portfolio\_value,

-- Ranking within each tier

RANK() OVER (PARTITION BY institution\_type ORDER BY SUM(loan\_amount) DESC) as rank\_within\_tier,

-- Overall ranking

RANK() OVER (ORDER BY SUM(loan\_amount) DESC) as overall\_rank,

-- Percentile ranking

PERCENT\_RANK() OVER (ORDER BY SUM(loan\_amount)) as portfolio\_percentile

FROM institutions i

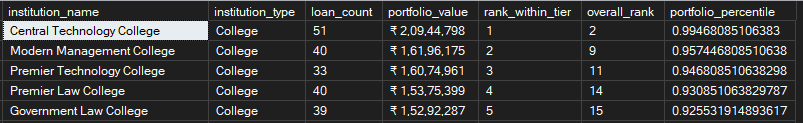
INNER JOIN loans l ON i.institution\_id = l.institution\_id

WHERE l.disbursement\_date IS NOT NULL

GROUP BY institution\_name, institution\_type

ORDER BY institution\_type, rank\_within\_tier;

**Output:**

****

**Practice Exercise 1.1:**  
Rank customers within each employment\_type by total loan portfolio using ROW\_NUMBER, RANK, and DENSE\_RANK.

## 1.2 Quartile and Distribution Analysis

**Purpose:**  
Classify institutions into quartiles and deciles to analyze performance distribution.

**Learning Objective:**  
Use NTILE to assign quartiles/deciles and classify performance based on portfolio and risk.

**-- Quartile analysis of institutions**

SELECT

institution\_name,

institution\_type,

total\_loans,

FORMAT(total\_portfolio, 'C0', 'en-IN') AS portfolio\_formatted,

default\_rate,

NTILE(4) OVER (ORDER BY total\_portfolio) AS portfolio\_quartile,

NTILE(4) OVER (ORDER BY default\_rate DESC) AS risk\_quartile,

NTILE(10) OVER (ORDER BY total\_portfolio) AS portfolio\_decile,

CASE

WHEN NTILE(4) OVER (ORDER BY total\_portfolio) = 4 AND NTILE(4) OVER (ORDER BY default\_rate DESC) <= 2

THEN 'STAR PERFORMER'

WHEN NTILE(4) OVER (ORDER BY total\_portfolio) = 4 AND NTILE(4) OVER (ORDER BY default\_rate DESC) > 2

THEN 'HIGH VOLUME, HIGH RISK'

WHEN NTILE(4) OVER (ORDER BY total\_portfolio) >= 3 AND NTILE(4) OVER (ORDER BY default\_rate DESC) <= 2

THEN 'SOLID PERFORMER'

WHEN NTILE(4) OVER (ORDER BY default\_rate DESC) = 4

THEN 'PROBLEM PARTNER'

ELSE 'AVERAGE PERFORMER'

END AS performance\_category

FROM (

SELECT TOP 5

i.institution\_name,

i.institution\_type,

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

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

ROUND(COUNT(CASE WHEN l.loan\_status = 'Defaulted' THEN 1 END) \* 100.0 / COUNT(l.loan\_id), 2) AS default\_rate

FROM institutions i

INNER JOIN loans l ON i.institution\_id = l.institution\_id

WHERE l.disbursement\_date IS NOT NULL

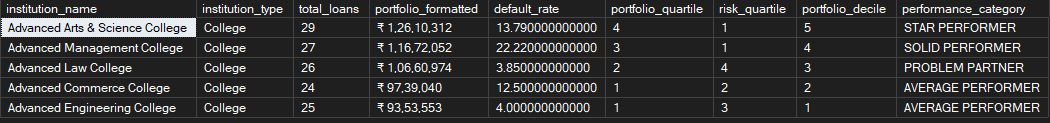
GROUP BY i.institution\_name, i.institution\_type

HAVING COUNT(l.loan\_id) >= 10

) AS institution\_metrics

ORDER BY total\_portfolio DESC;

**Output:**



# PART 2: COMPARATIVE ANALYSIS FUNCTIONS

## 2.1 FIRST\_VALUE and LAST\_VALUE Functions

**Purpose:**  
Compare each institution’s portfolio and risk with the best and worst performers, both overall and within the same type.

**Learning Objective:**  
Use FIRST\_VALUE and LAST\_VALUE for benchmarking and calculate performance gaps.

**-- Overall best/worst performance comparison**

SELECT

institution\_name,

institution\_type,

total\_loans,

FORMAT(total\_portfolio, 'C0', 'en-IN') AS portfolio\_value,

default\_rate,

FIRST\_VALUE(institution\_name) OVER (ORDER BY total\_portfolio DESC) AS top\_performer\_by\_volume,

FIRST\_VALUE(total\_portfolio) OVER (ORDER BY total\_portfolio DESC) AS top\_portfolio\_value,

FIRST\_VALUE(institution\_name) OVER (ORDER BY default\_rate ASC) AS best\_quality\_institution,

FIRST\_VALUE(default\_rate) OVER (ORDER BY default\_rate ASC) AS best\_default\_rate,

LAST\_VALUE(institution\_name) OVER (ORDER BY total\_portfolio DESC ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) AS worst\_performer\_by\_volume,

LAST\_VALUE(default\_rate) OVER (ORDER BY default\_rate ASC ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) AS worst\_default\_rate,

total\_portfolio - FIRST\_VALUE(total\_portfolio) OVER (ORDER BY total\_portfolio DESC) AS gap\_from\_leader,

default\_rate - FIRST\_VALUE(default\_rate) OVER (ORDER BY default\_rate ASC) AS quality\_gap\_from\_best

FROM (

SELECT

i.institution\_name,

i.institution\_type,

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

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

ROUND(COUNT(CASE WHEN l.loan\_status = 'Defaulted' THEN 1 END) \* 100 / COUNT(l.loan\_id), 2) AS default\_rate

FROM institutions i

INNER JOIN loans l ON i.institution\_id = l.institution\_id

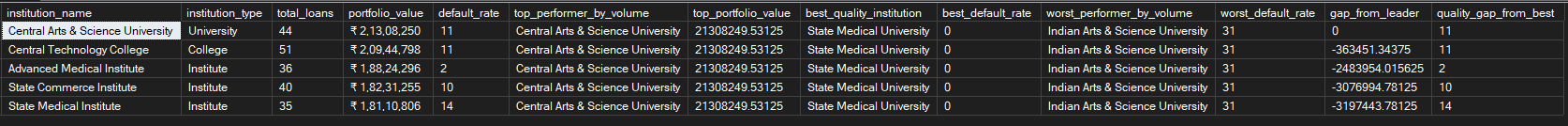
WHERE l.disbursement\_date IS NOT NULL

GROUP BY i.institution\_name, i.institution\_type

HAVING COUNT(l.loan\_id) >= 5

) AS institution\_performance

ORDER BY total\_portfolio DESC;

**Output:**

**-- Tier-specific benchmarking**

SELECT TOP 5

institution\_name,

institution\_type,

total\_portfolio,

default\_rate,

FIRST\_VALUE(institution\_name) OVER (

PARTITION BY institution\_type

ORDER BY total\_portfolio DESC

) AS type\_leader\_by\_volume,

FIRST\_VALUE(total\_portfolio) OVER (

PARTITION BY institution\_type

ORDER BY total\_portfolio DESC

) AS type\_best\_portfolio,

FIRST\_VALUE(institution\_name) OVER (

PARTITION BY institution\_type

ORDER BY default\_rate ASC

) AS type\_quality\_leader,

total\_portfolio - FIRST\_VALUE(total\_portfolio) OVER (

PARTITION BY institution\_type

ORDER BY total\_portfolio DESC

) AS gap\_from\_type\_leader

FROM (

SELECT

i.institution\_name,

i.institution\_type,

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

ROUND(COUNT(CASE WHEN l.loan\_status = 'Defaulted' THEN 1 END) \* 100 / COUNT(l.loan\_id), 2) AS default\_rate

FROM institutions i

INNER JOIN loans l ON i.institution\_id = l.institution\_id

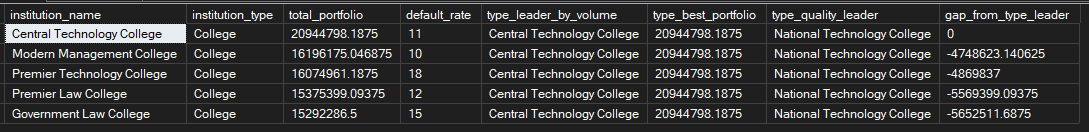
WHERE l.disbursement\_date IS NOT NULL

GROUP BY i.institution\_name, i.institution\_type

) AS type\_analysis

ORDER BY institution\_type, total\_portfolio DESC;

**Output:**

****

**Practice Exercise 2.1:**  
Benchmark customers in the same employment\_type against the top and bottom performers by portfolio and default rate.

## 2.2 LAG and LEAD for Trend Analysis

**Purpose:**  
Identify monthly changes in institution performance and classify trends as growing, declining, or stable.

**Learning Objective:**  
Use LAG and LEAD to compare performance across time periods.

**-- Monthly performance trends**

WITH monthly\_performance AS (

SELECT

i.institution\_name,

YEAR(l.disbursement\_date) AS loan\_year,

MONTH(l.disbursement\_date) AS loan\_month,

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

SUM(l.loan\_amount) AS monthly\_volume,

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

FROM institutions i

INNER JOIN loans l ON i.institution\_id = l.institution\_id

WHERE l.disbursement\_date IS NOT NULL

AND l.disbursement\_date >= '2023-01-01'

GROUP BY i.institution\_name, YEAR(l.disbursement\_date), MONTH(l.disbursement\_date)

)

SELECT TOP 5

institution\_name,

loan\_year,

loan\_month,

monthly\_loans,

FORMAT(monthly\_volume, 'C0', 'en-IN') AS volume\_formatted,

monthly\_defaults,

LAG(monthly\_loans, 1) OVER (

PARTITION BY institution\_name

ORDER BY loan\_year, loan\_month

) AS previous\_month\_loans,

LAG(monthly\_volume, 1) OVER (

PARTITION BY institution\_name

ORDER BY loan\_year, loan\_month

) AS previous\_month\_volume,

monthly\_loans - LAG(monthly\_loans, 1) OVER (

PARTITION BY institution\_name

ORDER BY loan\_year, loan\_month

) AS loan\_growth,

ROUND(

(monthly\_volume - LAG(monthly\_volume, 1) OVER (

PARTITION BY institution\_name

ORDER BY loan\_year, loan\_month

)) \* 100.0 / NULLIF(LAG(monthly\_volume, 1) OVER (

PARTITION BY institution\_name

ORDER BY loan\_year, loan\_month

), 0),

2

) AS volume\_growth\_percent,

LEAD(monthly\_loans, 1) OVER (

PARTITION BY institution\_name

ORDER BY loan\_year, loan\_month

) AS next\_month\_loans,

CASE

WHEN monthly\_loans > LAG(monthly\_loans, 1) OVER (

PARTITION BY institution\_name

ORDER BY loan\_year, loan\_month

) THEN 'GROWING'

WHEN monthly\_loans < LAG(monthly\_loans, 1) OVER (

PARTITION BY institution\_name

ORDER BY loan\_year, loan\_month

) THEN 'DECLINING'

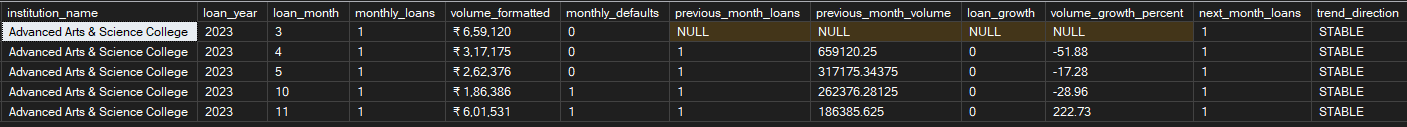
ELSE 'STABLE'

END AS trend\_direction

FROM monthly\_performance

ORDER BY institution\_name, loan\_year, loan\_month;

**Output:**

****

**Practice Exercise 2.2:**  
Analyze quarterly customer acquisition by institution using LAG and LEAD to show growth/decline trends.

# PART 3: ADVANCED WINDOW FRAMES

## 3.1 Custom Window Frames

**Purpose:**  
Track moving averages, cumulative totals, and volatility for institutional loan performance over time.

**Learning Objective:**  
Use ROWS PRECEDING and UNBOUNDED PRECEDING to perform rolling and cumulative analytics.

**-- Rolling performance analysis for institutions**

WITH monthly\_metrics AS (

SELECT

i.institution\_name,

YEAR(l.disbursement\_date) AS year,

MONTH(l.disbursement\_date) AS month,

COUNT(l.loan\_id) AS loans,

SUM(l.loan\_amount) AS volume,

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

FROM institutions i

INNER JOIN loans l ON i.institution\_id = l.institution\_id

WHERE l.disbursement\_date IS NOT NULL

GROUP BY i.institution\_name, YEAR(l.disbursement\_date), MONTH(l.disbursement\_date)

)

SELECT TOP 5

institution\_name,

year,

month,

loans,

FORMAT(volume, 'C0', 'en-IN') AS monthly\_volume,

-- 3-month rolling average

AVG(loans) OVER (

PARTITION BY institution\_name

ORDER BY year, month

ROWS 2 PRECEDING

) AS rolling\_3m\_avg\_loans,

AVG(volume) OVER (

PARTITION BY institution\_name

ORDER BY year, month

ROWS 2 PRECEDING

) AS rolling\_3m\_avg\_volume,

-- 6-month rolling totals

SUM(loans) OVER (

PARTITION BY institution\_name

ORDER BY year, month

ROWS 5 PRECEDING

) AS rolling\_6m\_total\_loans,

SUM(volume) OVER (

PARTITION BY institution\_name

ORDER BY year, month

ROWS 5 PRECEDING

) AS rolling\_6m\_total\_volume,

-- Year-to-date cumulative

SUM(loans) OVER (

PARTITION BY institution\_name, year

ORDER BY month

ROWS UNBOUNDED PRECEDING

) AS ytd\_loans,

SUM(volume) OVER (

PARTITION BY institution\_name, year

ORDER BY month

ROWS UNBOUNDED PRECEDING

) AS ytd\_volume,

-- Loan volatility over last 6 months

STDEV(loans) OVER (

PARTITION BY institution\_name

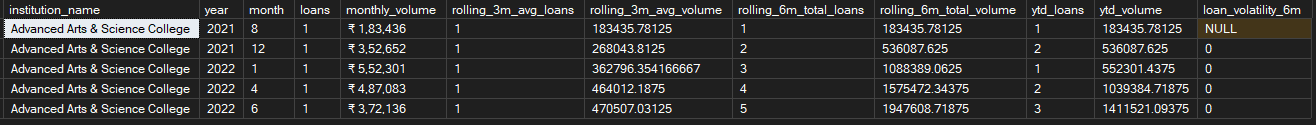
ORDER BY year, month

ROWS 5 PRECEDING

) AS loan\_volatility\_6m

FROM monthly\_metrics

ORDER BY institution\_name, year, month;

**Output:**

**Practice Exercise 3.1:**  
Create a rolling 12-month analysis of customer registrations by city with moving averages and cumulative totals.

## 3.2 Range-Based Window Frames

**Purpose:**  
Analyze institutions’ loan performance within dynamic loan amount ranges.

**Learning Objective:**  
Use RANGE BETWEEN to compare metrics within specific value ranges.

WITH loan\_data AS (

SELECT

l.loan\_id,

c.full\_name,

i.institution\_name,

i.institution\_id,

l.loan\_amount,

l.loan\_status

FROM loans l

INNER JOIN customers c ON l.customer\_id = c.customer\_id

INNER JOIN institutions i ON l.institution\_id = i.institution\_id

WHERE l.disbursement\_date IS NOT NULL

AND l.loan\_amount >= 100000

)

SELECT

base.loan\_id,

base.full\_name,

base.institution\_name,

base.loan\_amount,

base.loan\_status,

DENSE\_RANK() OVER (ORDER BY base.loan\_amount) AS similar\_amount\_rank,

COUNT(\*) AS similar\_sized\_loans,

AVG(CASE WHEN compare.loan\_status = 'Defaulted' THEN 1.0 ELSE 0.0 END) AS similar\_amount\_default\_rate,

COUNT(CASE WHEN compare.institution\_id = base.institution\_id THEN 1 END) AS institution\_loans\_in\_range

FROM loan\_data base

JOIN loan\_data compare

ON compare.loan\_amount BETWEEN base.loan\_amount - 500000

AND base.loan\_amount + 500000

GROUP BY

base.loan\_id,

base.full\_name,

base.institution\_name,

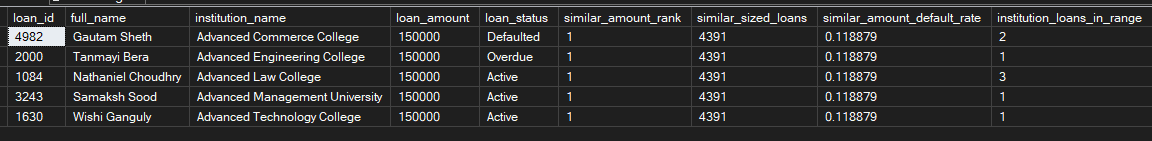
base.institution\_id,

base.loan\_amount,

base.loan\_status

ORDER BY base.loan\_amount, base.institution\_name;

Output:



**Practice Exercise 3.2:**  
Analyze default rate trends for loans within ±200K ranges.

# PART 4: COMPETITIVE POSITIONING ANALYSIS

## 4.1 Market Share and Positioning

**Purpose:**  
Calculate market share, rank institutions competitively, and assign them to performance quadrants.

**Learning Objective:**  
Use NTILE, ranking functions, and aggregations for competitive positioning.

-- Comprehensive competitive positioning analysis

WITH institution\_metrics AS (

SELECT

i.institution\_id,

i.institution\_name,

i.institution\_type,

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

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

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

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

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

COUNT(CASE WHEN l.loan\_status IN ('Active', 'Closed') THEN 1 END) AS performing\_loans

FROM institutions i

INNER JOIN loans l ON i.institution\_id = l.institution\_id

INNER JOIN customers c ON l.customer\_id = c.customer\_id

WHERE l.disbursement\_date IS NOT NULL

GROUP BY i.institution\_id, i.institution\_name, i.institution\_type

),

market\_totals AS (

SELECT

SUM(total\_loans) AS market\_total\_loans,

SUM(total\_portfolio) AS market\_total\_portfolio,

SUM(unique\_customers) AS market\_total\_customers

FROM institution\_metrics

)

SELECT TOP 5

im.institution\_name,

im.institution\_type,

im.total\_loans,

FORMAT(im.total\_portfolio, 'C0', 'en-IN') AS portfolio\_formatted,

im.unique\_customers,

FORMAT(im.avg\_loan\_size, 'C0', 'en-IN') AS avg\_loan\_formatted,

ROUND(im.defaults \* 100 / im.total\_loans, 2) AS default\_rate,

-- Market share

ROUND(im.total\_loans \* 100 / mt.market\_total\_loans, 2) AS loan\_market\_share,

ROUND(im.total\_portfolio \* 100 / mt.market\_total\_portfolio, 2) AS portfolio\_market\_share,

ROUND(im.unique\_customers \* 100 / mt.market\_total\_customers, 2) AS customer\_market\_share,

-- Rankings

RANK() OVER (ORDER BY im.total\_portfolio DESC) AS portfolio\_rank,

RANK() OVER (ORDER BY im.defaults \* 100.0 / im.total\_loans ASC) AS quality\_rank,

RANK() OVER (ORDER BY im.avg\_loan\_size DESC) AS avg\_loan\_size\_rank,

-- Tier rankings

RANK() OVER (PARTITION BY im.institution\_type ORDER BY im.total\_portfolio DESC) AS type\_portfolio\_rank,

-- Performance quadrant

CASE

WHEN NTILE(2) OVER (ORDER BY im.total\_portfolio) = 2

AND NTILE(2) OVER (ORDER BY im.defaults \* 100 / im.total\_loans ASC) = 2

THEN 'MARKET LEADER'

WHEN NTILE(2) OVER (ORDER BY im.total\_portfolio) = 2

AND NTILE(2) OVER (ORDER BY im.defaults \* 100 / im.total\_loans ASC) = 1

THEN 'HIGH VOLUME, HIGH RISK'

WHEN NTILE(2) OVER (ORDER BY im.total\_portfolio) = 1

AND NTILE(2) OVER (ORDER BY im.defaults \* 100/ im.total\_loans ASC) = 2

THEN 'NICHE QUALITY PLAYER'

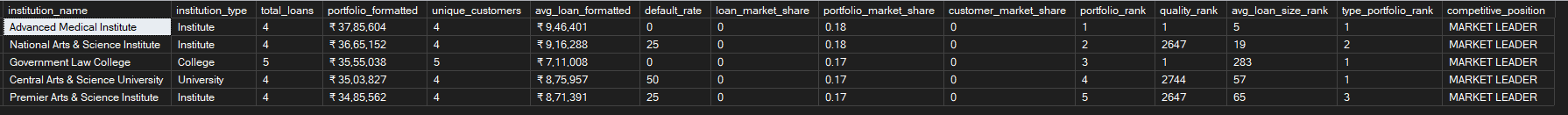
ELSE 'UNDERPERFORMER'

END AS competitive\_position

FROM institution\_metrics im

CROSS JOIN market\_totals mt

ORDER BY im.total\_portfolio DESC;

**Output:**

**Practice Exercise 4.1:**  
Do the same market share and positioning analysis for customer employment segments.

**Enterprise Revenue Forecasting Blueprint**  
© 2024 Skill AI Path. All Rights Reserved.  
Created by: Viresh Gendle | tech@skillaipath.com | skillaipath.com

Licensed for educational and internal business use.  
Attribution required when sharing. Commercial redistribution prohibited.