

# **Dynamic Credit Utilization Forecasting -Predictive & Prescriptive Segmentation**

**Submitted by  
Group 3**

**SQL Query to build the table For analysis**

**Under the Guidance of Industry  
Mentor**

**Abhishek Mukherjee**

**And Faculty Mentor  
Sundar Raj Vijayanagar**

```

USE CATALOG workspace;
USE SCHEMA credit_card_project;

-- =====
-- CREDIT CARD ANALYTICS RUNBOOK (Databricks SQL)
-- Schema: workspace.credit_card_project
-- Purpose: End-to-end SQL pipeline for Segmentation → Predictive → Prescriptive
-- Author: Group 3 – Subham Kumar Sahana (PGDM 2024-2026)
-- =====

-- 0) CONTEXT: set Unity Catalog catalog & schema so all objects are created here
USE CATALOG workspace;
USE SCHEMA credit_card_project;

-- check the tables created
--SHOW TABLES;

SELECT * FROM cc_general LIMIT 5;
SELECT * FROM credit_card_demographics LIMIT 5;

--DROP TABLE IF EXISTS cc_general_bronze;

-- -----
-- Map your already-uploaded UI tables to our "bronze" names (as VIEWS)
-- You uploaded:
--   • cc_general          (credit behavior)
--   • credit_card_demographics (demographics)
-- We create skinny views with explicit columns (no SELECT *), so downstream
-- Silver/Gold steps remain identical and Spark can prune columns.
-- -----

-- =====
-- 1) BRONZE: Ingest raw credit behavior into Bronze table
-- =====
USE CATALOG workspace;
USE SCHEMA credit_card_project;

CREATE OR REPLACE VIEW cc_general_bronze AS
SELECT * FROM cc_general;

-- =====
-- 2) BRONZE: Ingest raw demographics into Bronze table
-- =====
USE CATALOG workspace;
USE SCHEMA credit_card_project;

CREATE OR REPLACE VIEW cc_demo_bronze AS
SELECT * FROM credit_card_demographics;

```

```

-- -----
-- SILVER: clean/standardize (explicit casts; still no SELECT *)
-- -----

USE CATALOG workspace;
USE SCHEMA credit_card_project;
CREATE OR REPLACE TABLE cc_general_silver AS
SELECT
    b.CUST_ID,
    CAST(b.BALANCE AS DOUBLE) AS BALANCE,
    CAST(b.BALANCE_FREQUENCY AS DOUBLE) AS BALANCE_FREQUENCY,
    CAST(b.PURCHASES AS DOUBLE) AS PURCHASES,
    CAST(b.ONEOFF_PURCHASES AS DOUBLE) AS ONEOFF_PURCHASES,
    CAST(b.INSTALLMENTS_PURCHASES AS DOUBLE) AS INSTALLMENTS_PURCHASES,
    CAST(b.CASH_ADVANCE AS DOUBLE) AS CASH_ADVANCE,
    CAST(b.PURCHASES_FREQUENCY AS DOUBLE) AS PURCHASES_FREQUENCY,
    CAST(b.ONEOFF_PURCHASES_FREQUENCY AS DOUBLE) AS ONEOFF_PURCHASES_FREQUENCY,
    CAST(b.PURCHASES_INSTALLMENTS_FREQUENCY AS DOUBLE) AS
PURCHASES_INSTALLMENTS_FREQUENCY,
    CAST(b.CASH_ADVANCE_FREQUENCY AS DOUBLE) AS CASH_ADVANCE_FREQUENCY,
    CAST(b.CASH_ADVANCE_TRX AS INT) AS CASH_ADVANCE_TRX,
    CAST(b.PURCHASES_TRX AS INT) AS PURCHASES_TRX,
    CAST(b.CREDIT_LIMIT AS DOUBLE) AS CREDIT_LIMIT,
    CAST(b.PAYMENTS AS DOUBLE) AS PAYMENTS,
    CAST(b.MINIMUM_PAYMENTS AS DOUBLE) AS MINIMUM_PAYMENTS,
    CAST(b.PRC_FULL_PAYMENT AS DOUBLE) AS PRC_FULL_PAYMENT,
    CAST(b.TENURE AS INT) AS TENURE
FROM cc_general_bronze b;

CREATE OR REPLACE TABLE cc_demo_silver AS
SELECT
    d.CUST_ID,
    CAST(d.AGE AS INT) AS AGE,
    d.GENDER,
    d.MARITAL_STATUS,
    d.OCCUPATION,
    CAST(d.ANNUAL_INCOME AS DOUBLE) AS ANNUAL_INCOME,
    d.CITY_TIER,
    d.EDUCATION,
    d.HOUSE_OWNERSHIP,
    CAST(d.DEPENDENTS AS INT) AS DEPENDENTS,
    d.MOBILE_PHONE,
    d.EMAIL_SUBSCRIBED,
    d.LOYALTY_CARD,
    d.REGION
FROM cc_demo_bronze d;

```

```

-- -----
-- GOLD: feature engineering used by Q7-Q20 (unchanged from earlier)
-- -----

USE CATALOG workspace;
USE SCHEMA credit_card_project;
CREATE OR REPLACE TABLE cc_features_gold AS
WITH joined AS (
    SELECT
        g.CUST_ID, g.BALANCE, g.PURCHASES, g.ONEOFF_PURCHASES, g.INSTALLMENTS_PURCHASES,
        g.CASH_ADVANCE, g.PURCHASES_TRX, g.PURCHASES_FREQUENCY, g.CREDIT_LIMIT,
        g.PAYMENTS, g.MINIMUM_PAYMENTS, g.PRC_FULL_PAYMENT, g.TENURE,
        d.AGE, d.GENDER, d.MARITAL_STATUS, d.OCCUPATION, d.ANNUAL_INCOME,
        d.CITY_TIER, d.EDUCATION, d.HOUSE_OWNERSHIP, d.DEPENDENTS, d.REGION
    FROM cc_general_silver g
    LEFT JOIN cc_demo_silver d
        ON g.CUST_ID = d.CUST_ID)
SELECT
    j.CUST_ID,
    j.BALANCE, j.PURCHASES, j.ONEOFF_PURCHASES, j.INSTALLMENTS_PURCHASES,
    j.CASH_ADVANCE, j.PURCHASES_TRX, j.PURCHASES_FREQUENCY, j.CREDIT_LIMIT,
    j.PAYMENTS, j.MINIMUM_PAYMENTS, j.PRC_FULL_PAYMENT, j.TENURE,
    j.AGE, j.GENDER, j.MARITAL_STATUS, j.OCCUPATION, j.ANNUAL_INCOME,
    j.CITY_TIER, j.EDUCATION, j.HOUSE_OWNERSHIP, j.DEPENDENTS, j.REGION,
    (j.BALANCE / NULLIF(j.CREDIT_LIMIT,0)) AS
UTILIZATION_RATIO,
    COALESCE(j.PRC_FULL_PAYMENT,0.0) AS
FULL_PAY_PCT,
    (COALESCE(j.PAYMENTS,0.0) - COALESCE(j.MINIMUM_PAYMENTS,0.0)) / NULLIF(j.PAYMENTS,0.0) AS
ABOVE_MIN_PAY_RATIO,
    COALESCE(j.CASH_ADVANCE,0.0) / NULLIF(j.PURCHASES,0.0) AS
CASH_ADV_DEPENDENCY,
    COALESCE(j.ONEOFF_PURCHASES,0.0) / (COALESCE(j.INSTALLMENTS_PURCHASES,0.0)+1e-6) AS
ONEOFF_TO_INSTALLMENT_RATIO
FROM joined j;

-- Quantile cutoffs for segmentation (used by Q7)
CREATE OR REPLACE VIEW cc_quantiles AS
SELECT
    percentile_approx(UTILIZATION_RATIO, 0.25) AS u_p25,
    percentile_approx(UTILIZATION_RATIO, 0.75) AS u_p75,
    percentile_approx(FULL_PAY_PCT, 0.25) AS fp_p25,
    percentile_approx(FULL_PAY_PCT, 0.75) AS fp_p75
FROM cc_features_gold;

```

```
-- -----  
-- Small sanity check  
-- -----
```

```
USE CATALOG workspace;  
USE SCHEMA credit_card_project;  
SELECT COUNT(*) FROM workspace.credit_card_project.cc_general_bronze;  
SELECT COUNT(*) FROM workspace.credit_card_project.cc_demo_bronze;  
SELECT COUNT(*) FROM workspace.credit_card_project.cc_general_silver;  
SELECT COUNT(*) FROM workspace.credit_card_project.cc_demo_silver;  
SELECT COUNT(*) FROM workspace.credit_card_project.cc_features_gold;
```

=====

Q1) DESCRIPTIVE SEGMENTATION VIEW – TRANSLATE RAW RATIOS INTO CLEAR, BUSINESS-READABLE CUSTOMER GROUPS

-----

Business Impact (longer):

Creates a shared language between analytics, marketing, risk, and product. Rather than debating raw ratios, teams can act on personas that map to policy and offers. Enables targeted rewards for safe customers, guardrails for risky ones, and better stakeholder buy-in because segmentation logic is transparent and auditable.

=====

===== \*/

```
CREATE OR REPLACE VIEW workspace.credit_card_project.cc_segments AS
SELECT
  f.CUST_ID,
  f.UTILIZATION_RATIO,
  f.FULL_PAY_PCT,
  f.CREDIT_LIMIT,
  f.BALANCE,
  CASE
    WHEN f.UTILIZATION_RATIO <= q.u_p25 AND f.FULL_PAY_PCT >= q.fp_p75 THEN 'Safe High
Spenders'
    WHEN f.UTILIZATION_RATIO BETWEEN q.u_p25 AND q.u_p75 AND f.FULL_PAY_PCT >= q.fp_p25
THEN 'Moderate Users'
    WHEN f.UTILIZATION_RATIO > q.u_p75 AND f.FULL_PAY_PCT < q.fp_p25 THEN 'Over-Leveraged'
    ELSE 'Revolvers'
  END AS SEGMENT
FROM workspace.credit_card_project.cc_features_gold f
CROSS JOIN workspace.credit_card_project.cc_quantiles q;

SELECT * FROM workspace.credit_card_project.cc_segments LIMIT 10;
```

=====

Q2) CENTRAL STRESS-TEST PARAMETERS – ONE EDITABLE PLACE TO CONTROL “WHAT-IF” SCENARIOS

-----

Business Impact (longer):

Risk, Finance, and Leadership frequently ask “what if rates go up?” or “what if customers repay less?”. By separating parameters, you can answer those questions in minutes, not days, improving governance, speed of decision-making, and confidence in your analytics because the assumptions are explicit and documented.

=====

```
CREATE OR REPLACE VIEW workspace.credit_card_project.stress_params AS
SELECT
  200 AS interest_hike_bps,    -- +2.00% APR
  0.15 AS repayment_drop_pct; -- 15% drop in full-pay behavior
```

=====

### Q3) PREDICTIVE UTILIZATION & RISK UNDER STRESS – ESTIMATE FUTURE LEVERAGE/REPAYMENT PRESSURE

-----

#### Business Impact (longer):

Shifts the portfolio view from static to forward-looking. Identifies customers who will become stretched under macro changes before delinquency happens. Supports proactive credit actions (tighten, watchlist) and informs capital planning by quantifying exposure sensitivity to rates/repayment shocks.

=====

```
CREATE OR REPLACE VIEW workspace.credit_card_project.cc_util_forecast AS
WITH p AS (SELECT * FROM workspace.credit_card_project.stress_params),
calc AS (
  SELECT
    s.CUST_ID,
    s.SEGMENT,
    s.CREDIT_LIMIT,
    s.BALANCE,
    s.UTILIZATION_RATIO,
    s.FULL_PAY_PCT
  FROM workspace.credit_card_project.cc_segments s
)
SELECT
  c.CUST_ID,
  c.SEGMENT,
  c.CREDIT_LIMIT,
  c.BALANCE,
  (c.UTILIZATION_RATIO + 0.10 * (1 - COALESCE(c.FULL_PAY_PCT,0))) AS
util_pred_base,
  GREATEST(0.0, c.FULL_PAY_PCT * (1 - p.repayment_drop_pct)) AS
full_pay_stress,
  (c.BALANCE * (1 + p.interest_hike_bps / 10000.0)) AS
balance_stress,
  LEAST(5.0,
    (c.BALANCE * (1 + p.interest_hike_bps / 10000.0)) / NULLIF(c.CREDIT_LIMIT,1)
    + 0.20 * (1 - GREATEST(0.0, c.FULL_PAY_PCT * (1 - p.repayment_drop_pct)))
  ) AS util_pred_stress,
  LEAST(1.0,
    0.5 * ((c.BALANCE * (1 + p.interest_hike_bps / 10000.0)) /
  NULLIF(c.CREDIT_LIMIT,1))
    + 0.4 * (1 - GREATEST(0.0, c.FULL_PAY_PCT * (1 - p.repayment_drop_pct)))
  ) AS risk_score
FROM calc c CROSS JOIN p;
```

=====

#### Q4) PRESCRIPTIVE POLICY ENGINE – TRANSLATE RISK METRICS INTO CLEAR LIMIT/APR ACTIONS

-----

##### Description:

Maps predicted risk and utilization to tangible decisions Operations can execute: increase limit (by 10-20%), reduce APR for the safest customers, or tighten policy for the riskiest. Rules are simple and auditable.

Projection is kept skinny for downstream consumption.

##### Business Impact (longer):

Moves from “insight” to “action.” Safe customers get more headroom and better pricing, which increases spend and loyalty; high-risk customers are controlled to reduce loss. Decisions are explainable to Compliance and Regulators because thresholds are transparent and consistent.

=====

```
CREATE OR REPLACE VIEW workspace.credit_card_project.cc_prescriptions AS
SELECT
  u.CUST_ID,
  u.SEGMENT,
  u.util_pred_base,
  u.util_pred_stress,
  u.risk_score,
  CASE
    WHEN u.risk_score <= 0.25 AND u.util_pred_base BETWEEN 0.20 AND 0.60 THEN 0.20
    WHEN u.risk_score <= 0.40 AND u.util_pred_base BETWEEN 0.10 AND 0.50 THEN 0.10
    WHEN u.risk_score >= 0.70 OR u.util_pred_stress >= 0.90 THEN -0.10
    ELSE 0.00
  END AS recommended_limit_change_pct,
  CASE
    WHEN u.risk_score <= 0.25 THEN -100
    WHEN u.risk_score > 0.60 THEN +100
    ELSE 0
  END AS recommended_apr_bps
FROM workspace.credit_card_project.cc_util_forecast u;
```



=====

#### Q5) OPERATIONAL ACTION TABLE – SMALL, MATERIALIZED LIST READY FOR CRM/EXECUTION

-----

##### Description:

Materializes the prescriptive recommendations in a compact Delta table. This decouples analytics from operational systems so dashboards/exports can use a stable object without recomputing heavy logic.

##### Business Impact (longer):

Shortens time-to-value. Marketing or Credit Ops can immediately launch campaigns (limit upgrades, APR changes, watchlists) using an always-up-to-date table, improving execution speed and governance.

=====

===== \*/

```
CREATE OR REPLACE TABLE workspace.credit_card_project.cc_actions AS
SELECT
  p.CUST_ID,
  p.SEGMENT,
  p.risk_score,
  p.util_pred_base,
  p.util_pred_stress,
  p.recommended_limit_change_pct,
  p.recommended_apr_bps
FROM workspace.credit_card_project.cc_prescriptions p;
```

=====

#### Q6) CENTRAL KPI ECONOMICS – EXPLICIT, ADJUSTABLE PARAMETERS FOR P&L ESTIMATION

-----

##### Description:

A tiny TEMP VIEW that captures the economics used in P&L modeling: interchange\_rate (fee on spend), spend\_elasticity (how much extra spend we get per limit uplift), and loss\_given\_default (cost on stressed exposure). Centralizing these numbers keeps Finance and Analytics aligned.

##### Business Impact (longer):

Prevents “dueling spreadsheets.” Everyone uses the same assumptions, scenario tweaks are instant, and the P&L outputs become trusted across Finance, Product, and Risk committees.

=====

===== \*/

```
CREATE OR REPLACE VIEW workspace.credit_card_project.kpi_params AS
SELECT
  0.015 AS interchange_rate,    -- 1.5% fee on incremental spend
  0.25 AS spend_elasticity,     -- 25% of limit uplift turns into spend for safe segments
  0.03 AS loss_given_default;   -- 3% proxy on stressed exposure for risky segments
```

=====

## Q7) SEGMENT-LEVEL P&L – REVENUE UPLIFT MINUS RISK COST TO RANK WHERE TO INVEST OR CONTAIN

-----

### Description:

Computes per-customer uplift and risk cost in a CTE (joining actions with limit) and then aggregates by segment. Outputs counts, average risk, incremental revenue, expected risk cost, and net impact. The design minimizes shuffles by trimming columns before JOIN/GROUP BY.

### Business Impact (longer):

Puts money values on each segment so leadership can prioritize. Invest in segments with highest net impact(positive ROI); tighten or monitor segments with negative net impact. Enables budget allocation grounded in data.

=====

```
CREATE OR REPLACE VIEW workspace.credit_card_project.cc_business_impact AS
WITH p AS (SELECT * FROM workspace.credit_card_project.kpi_params),
per_cust AS (
  SELECT
    a.CUST_ID,
    a.SEGMENT,
    a.risk_score,
    a.util_pred_base,
    a.util_pred_stress,
    a.recommended_limit_change_pct,
    g.CREDIT_LIMIT,
    CASE
      WHEN a.recommended_limit_change_pct > 0 AND a.risk_score <= 0.40
      THEN (a.recommended_limit_change_pct * g.CREDIT_LIMIT) * (SELECT spend_elasticity
FROM p)
      ELSE 0.0
    END AS incr_spend,
    (a.util_pred_stress * g.CREDIT_LIMIT) AS stressed_exposure
  FROM workspace.credit_card_project.cc_actions a
  JOIN workspace.credit_card_project.cc_features_gold g
    ON a.CUST_ID = g.CUST_ID
)
SELECT
  pc.SEGMENT,
  COUNT(*) AS customers,
  ROUND(AVG(pc.risk_score),3) AS avg_risk,
  SUM(pc.incr_spend) AS incr_spend,
  SUM(pc.incr_spend * (SELECT interchange_rate FROM
workspace.credit_card_project.kpi_params)) AS incr_revenue_est,
  SUM(CASE WHEN pc.risk_score > 0.60
    THEN pc.stressed_exposure * (SELECT loss_given_default FROM
workspace.credit_card_project.kpi_params)
    ELSE 0 END) AS risk_cost_est,
```

```

SUM(pc.incr_spend * (SELECT interchange_rate FROM
workspace.credit_card_project.kpi_params))
- SUM(CASE WHEN pc.risk_score > 0.60
      THEN pc.stressed_exposure * (SELECT loss_given_default FROM
workspace.credit_card_project.kpi_params)
      ELSE 0 END) AS net_impact_est
FROM per_cust pc
GROUP BY pc.SEGMENT
ORDER BY net_impact_est DESC;

```

=====

Q8) UPGRADE SHORTLIST – SAFE CUSTOMERS WITH HEADROOM (TOP 200, ORDERED BY UTILIZATION)

-----

#### Description:

Creates an ops-ready shortlist of the safest customers who are recommended for limit upgrades and are most likely to respond (higher baseline utilization). We JOIN to a narrow demographic slice for context and order by util\_pred\_base to prioritize the highest potential first.

#### Business Impact (longer):

Hands Sales/CRM a qualified call list that drives near-term revenue while maintaining portfolio risk discipline. Campaigns become more efficient because outreach targets customers with both capacity and willingness to spend.

=====

```

CREATE OR REPLACE VIEW workspace.credit_card_project.target_limit_upgrades AS
SELECT

```

```

    a.CUST_ID,
    a.SEGMENT,
    a.risk_score,
    a.util_pred_base,
    a.recommended_limit_change_pct,
    g.AGE,
    g.ANNUAL_INCOME,
    g.CITY_TIER,
    g.EDUCATION
FROM workspace.credit_card_project.cc_actions a
JOIN (
    SELECT CUST_ID, AGE, ANNUAL_INCOME, CITY_TIER, EDUCATION
    FROM workspace.credit_card_project.cc_demo_silver
) g
ON a.CUST_ID = g.CUST_ID
WHERE a.recommended_limit_change_pct > 0
      AND a.risk_score <= 0.40
ORDER BY a.util_pred_base DESC
LIMIT 200;

```

=====

## Q9) RISK HEATMAP BY GEOGRAPHY – FIND WHERE RISK & STRESS CLUSTER (REGION × CITY TIER)

-----

### Description:

Joins a narrow demographic slice (region, city tier) with a narrow forecast slice (risk, stressed utilization), then aggregates by REGION × CITY\_TIER. Designed to aggregate after trimming to reduce memory.

### Business Impact (longer):

Reveals geographic hot-spots where additional education, collections support, or pricing changes may be needed. Enables regional portfolio steering, targeted playbooks, and better allocation of field resources.

=====

```
CREATE OR REPLACE VIEW workspace.credit_card_project.risk_heatmap_region_tier AS
```

```
SELECT
```

```
    dim.REGION,
```

```
    dim.CITY_TIER,
```

```
    COUNT(*)                                AS customers,
```

```
    ROUND(AVG(fore.risk_score), 3)          AS avg_risk,
```

```
    ROUND(AVG(fore.util_pred_stress), 3)    AS avg_stressed_util
```

```
FROM (
```

```
    SELECT CUST_ID, REGION, CITY_TIER
```

```
    FROM workspace.credit_card_project.cc_demo_silver
```

```
) dim
```

```
JOIN (
```

```
    SELECT CUST_ID, risk_score, util_pred_stress
```

```
    FROM workspace.credit_card_project.cc_util_forecast
```

```
) fore
```

```
    ON dim.CUST_ID = fore.CUST_ID
```

```
GROUP BY dim.REGION, dim.CITY_TIER
```

```
ORDER BY avg_risk DESC, customers DESC;
```

=====

## Q10) COHORT PROFITABILITY PROXY – LIFECYCLE VALUE BY TENURE BAND × SEGMENT

-----

### Description:

Groups customers by tenure bands (0-12, 13-24, 25-36, 37+) and segment to estimate a revenue proxy from purchases (1.5% interchange). Uses a CTE to compute bands once and then aggregates. Only the necessary columns are scanned.

### Business Impact (longer):

Shows which life-stage cohorts are most valuable within each segment. Lets Marketing prioritize onboarding and early-life retention campaigns where the ROI is highest, and informs lifecycle communication design.

=====

```
CREATE OR REPLACE VIEW workspace.credit_card_project.cohort_profitability AS
```

```
WITH base AS (
  SELECT
    f.CUST_ID,
    f.TENURE,
    f.PURCHASES,
    s.SEGMENT
  FROM workspace.credit_card_project.cc_features_gold f
  JOIN workspace.credit_card_project.cc_segments s
    ON f.CUST_ID = s.CUST_ID
),
banded AS (
  SELECT
    b.SEGMENT,
    CASE
      WHEN b.TENURE <= 12 THEN '0-12'
      WHEN b.TENURE <= 24 THEN '13-24'
      WHEN b.TENURE <= 36 THEN '25-36'
      ELSE '37+'
    END AS tenure_band,
    b.PURCHASES
  FROM base b
)
SELECT
  bb.SEGMENT,
  bb.tenure_band,
  COUNT(*) AS customers,
  ROUND(AVG(bb.PURCHASES),2) AS avg_purchases,
  ROUND(SUM(bb.PURCHASES) * 0.015, 2) AS revenue_proxy
FROM banded bb
GROUP BY bb.SEGMENT, bb.tenure_band
ORDER BY bb.SEGMENT, bb.tenure_band;
```

=====

## Q11) REPAYMENT DISCIPLINE VS UTILIZATION – INCOME BANDS TO SHAPE POLICY (APR/LIMITS)

-----

### Description:

Bins customers by ANNUAL\_INCOME and summarizes utilization and full-payment behavior per band. Early column trimming (CTE) reduces shuffle and memory. Helps detect aggressive credit use in certain income tiers.

### Business Impact (longer):

Guides fair and effective pricing/limit policies by income tier; highlights where to educate customers and where to promote premium products. Improves both customer outcomes and portfolio quality.

=====

```
CREATE OR REPLACE VIEW workspace.credit_card_project.income_util_repayment AS
```

```
WITH proj AS (
  SELECT
    f.CUST_ID,
    f.UTILIZATION_RATIO,
    f.PRC_FULL_PAYMENT,
    d.ANNUAL_INCOME
  FROM workspace.credit_card_project.cc_features_gold f
  JOIN workspace.credit_card_project.cc_demo_silver d
    ON f.CUST_ID = d.CUST_ID
),
banded AS (
  SELECT
    p.CUST_ID,
    p.UTILIZATION_RATIO,
    p.PRC_FULL_PAYMENT,
    CASE
      WHEN p.ANNUAL_INCOME < 500000 THEN '<5L'
      WHEN p.ANNUAL_INCOME < 1000000 THEN '5L-10L'
      WHEN p.ANNUAL_INCOME < 2000000 THEN '10L-20L'
      ELSE '20L+'
    END AS income_band
  FROM proj p
)
SELECT
  b.income_band,
  COUNT(*) AS customers,
  ROUND(AVG(b.UTILIZATION_RATIO),3) AS avg_utilization,
  ROUND(AVG(COALESCE(b.PRC_FULL_PAYMENT,0.0)),3) AS avg_full_pay_pct
FROM banded b
GROUP BY b.income_band
ORDER BY customers DESC;
```

=====

## Q12) CASH-ADVANCE DEPENDENCY HOT-SPOTS – REGION × CITY TIER FOR RISK & PRODUCT DESIGN

-----

### Description:

Finds geographies with high reliance on cash advances (a risk signal). We pre-select only the fields required from demographics and features, join, then aggregate to locate hot-spots.

### Business Impact (longer):

Informs where to offer safer short-term credit alternatives, targeted financial education, or additional controls. Improves NPS by replacing expensive behavior with better-fit products while reducing portfolio risk.

=====

```
CREATE OR REPLACE VIEW workspace.credit_card_project.cash_advance_hotspots AS
WITH dim AS (
    SELECT CUST_ID, REGION, CITY_TIER
    FROM workspace.credit_card_project.cc_demo_silver
),
fact AS (
    SELECT CUST_ID, CASH_ADV_DEPENDENCY
    FROM workspace.credit_card_project.cc_features_gold
)
SELECT
    d.REGION,
    d.CITY_TIER,
    COUNT(*) AS customers,
    ROUND(AVG(COALESCE(f.CASH_ADV_DEPENDENCY,0.0)),3) AS avg_cash_adv_depend
FROM dim d
JOIN fact f
    ON d.CUST_ID = f.CUST_ID
GROUP BY d.REGION, d.CITY_TIER
ORDER BY avg_cash_adv_depend DESC, customers DESC;
```

=====

### Q13) APR LADDER SUMMARY – PROPOSED APR DELTAS BY SEGMENT × INCOME BAND (FAIRNESS & STRATEGY)

-----

#### Description:

Combines prescriptive APR recommendations with income bands to show where pricing changes concentrate. Subqueries keep only necessary columns, then we GROUP BY to summarize counts and average APR deltas.

#### Business Impact (longer):

Demonstrates that pricing adjustments are targeted, explainable, and fair across income tiers. Supports compliance reviews and helps product teams tune the APR ladder to meet growth and risk objectives.

=====

```
CREATE OR REPLACE VIEW workspace.credit_card_project.apr_ladder_summary AS
```

```
WITH pres AS (
  SELECT CUST_ID, SEGMENT, recommended_apr_bps
  FROM workspace.credit_card_project.cc_prescriptions
),
inc AS (
  SELECT CUST_ID, ANNUAL_INCOME
  FROM workspace.credit_card_project.cc_demo_silver
),
banded AS (
  SELECT
    p.SEGMENT,
    CASE
      WHEN i.ANNUAL_INCOME < 500000 THEN '<5L'
      WHEN i.ANNUAL_INCOME < 1000000 THEN '5L-10L'
      WHEN i.ANNUAL_INCOME < 2000000 THEN '10L-20L'
      ELSE '20L+'
    END AS income_band,
    p.recommended_apr_bps
  FROM pres p
  JOIN inc i
    ON p.CUST_ID = i.CUST_ID
)
SELECT
  b.SEGMENT,
  b.income_band,
  COUNT(*) AS customers,
  SUM(CASE WHEN b.recommended_apr_bps < 0 THEN 1 ELSE 0 END) AS apr_cuts,
  SUM(CASE WHEN b.recommended_apr_bps > 0 THEN 1 ELSE 0 END) AS apr_hikes,
  ROUND(AVG(b.recommended_apr_bps),1) AS avg_apr_bps
FROM banded b
GROUP BY b.SEGMENT, b.income_band
ORDER BY b.SEGMENT, b.income_band;
```



=====

## Q14) ELASTICITY TARGETING – WHO WILL LIKELY RESPOND TO LIMIT UPGRADES (POOL SIZE & QUALITY)

-----

### Description:

Builds a compact view of customers with low modeled risk and mid-to-high baseline utilization who also have a recommended positive limit change. Uses a subquery to pre-trim columns, then groups by segment to size the opportunity and report average risk/utilization.

### Why important:

Directs Sales/CRM to the highest-ROI upgrade pool by identifying those with unmet spending demand and safe risk profiles, improving campaign efficiency.

### Business Impact (longer):

Delivers incremental revenue from safe customers who are likely to spend more when limits increase, while keeping portfolio risk controlled. Helps planning teams forecast uplift by segment and allocate campaign budget to the most responsive audiences.

===== \*/

```
CREATE OR REPLACE VIEW workspace.credit_card_project.upgrade_candidates AS
WITH safe_pool AS (
  SELECT
    a.CUST_ID,
    a.SEGMENT,
    a.risk_score,
    a.util_pred_base,
    a.recommended_limit_change_pct
  FROM workspace.credit_card_project.cc_actions a
  WHERE a.risk_score <= 0.40
    AND a.util_pred_base BETWEEN 0.20 AND 0.70
    AND a.recommended_limit_change_pct > 0
)
SELECT
  s.SEGMENT,
  COUNT(*) AS candidates,
  ROUND(AVG(s.util_pred_base),3) AS avg_current_util,
  ROUND(AVG(s.risk_score),3) AS avg_risk
FROM safe_pool s
GROUP BY s.SEGMENT
ORDER BY candidates DESC;
```