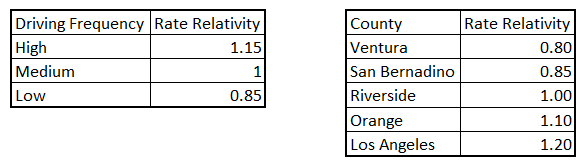
**SQL Workshop**

**Goal:** To calculate 2018 loss ratio. We will assume that claims information includes loss adjustment expenses. The following formulas will be useful:

Proportion of policy period elapsed, in red, is the hard part. You are given the tables policy.csv and claim.csv. Your task is to familiarize yourself with SQL and to solve the problem above. The annual base rate is **$2,200**, and rate relativities for driving frequency and county are below. For your convenience, driving frequency is given in discrete bins instead of as an annual number of miles driven. Additionally, you may assume that for the entire duration a policy is inforce, driving frequency and county do not change. (This can be easily verified.)



**Contents**

|  |  |
| --- | --- |
| **Part I** | **Part II** |
| 1. Introduction to databases and SQL    1. What is a database?    2. What is a relational database?    3. What is the relational model?    4. Why do I care?    5. Introduction to the Access environment 2. Table Creation: Importing from CSVs 3. Selecting Data    1. Selecting columns    2. Only selecting certain rows    3. Selecting distinct rows    4. Ordering the data 4. Variables 5. Aggregating the data    1. AVG, COUNT, and SUM    2. Where: What doesn't work    3. Having: What works | 1. Table Creation and Inserting Data: Using SQL    1. Creation    2. Insertion    3. Updating    4. Deleting    5. Dropping    6. Creating a table from a query 2. VBA    1. Basic Syntax: Creating Table    2. Running multiple lines at once: Populating Table 3. Joins    1. Inner    2. Left    3. Multiple Joins 4. Subqueries 5. A difficult example with SWITCH: Calculating Earned Premiums and Loss Ratio    1. Calculating Exposures    2. Joining on Written Premium    3. Calculating Written Premium    4. Calculating Loss Ratio: A different type of subquery |

1. Table Creation: Importing from CSVs
   1. Import policy.csv through External Data 🡪 New Data Source 🡪 From File 🡪 Text File. Since PolicyNumber contains duplicates, let Access create primary key.
   2. Import claim.csv through the same process but set ClaimNumber to the primary key.
2. Selecting Data: Create 🡪 Query Design. Go to SQL View.
   1. Selecting a column
      * SELECT PolicyNumber FROM Policy;
   2. Selecting more than one column
      * SELECT PolicyNumber, PolicyStartDate FROM Policy;
   3. Selecting all columns
      * SELECT \* FROM Policy;
   4. Only selecting certain rows
      * SELECT \* FROM Policy WHERE DrivingFrequency = "Low"
      * SELECT \* FROM Policy WHERE DrivingFrequency = "High" AND County = "Los Angeles"
      * SELECT \* FROM Policy WHERE DrivingFrequency = "High" OR County = "Los Angeles"
      * SELECT PolicyNumber, PolicyStartDate, DrivingFrequency FROM Policy WHERE County IN ("Los Angeles", "Riverside")
   5. Selecting distinct rows
      * SELECT DISTINCT PolicyNumber, DrivingFrequency, County FROM Policy WHERE PolicyNumber = "P100000"
      * SELECT DISTINCT PolicyNumber, County FROM Policy WHERE County NOT IN ("Ventura", "Orange")
   6. Ordering the data
      * SELECT \* FROM Claim ORDER BY PolicyNumber;
      * SELECT \* FROM Claim ORDER BY LossDate DESC;
3. Variables
   * + SELECT \* FROM policy WHERE YEAR(PolicyStartDate) = myyear;
4. Aggregating the data
   1. AVG
      * SELECT AVG(ClaimAmount) FROM Claim;
      * SELECT AVG(ClaimAmount) AS Severity FROM Claim;
   2. COUNT
      * SELECT PolicyNumber, COUNT(ClaimNumber) AS ClaimCount FROM Claim GROUP BY PolicyNumber;
   3. SUM
      * SELECT YEAR(LossDate) AS LossYear, SUM(ClaimAmount) AS TotalLoss FROM Claim GROUP BY YEAR(LossDate) ORDER BY YEAR(LossDate);
   4. Where: What doesn't work
      * SELECT PolicyNumber, SUM(ClaimAmount) AS TotalClaims FROM Claim WHERE TotalClaims < 1000 GROUP BY PolicyNumber;
      * SELECT PolicyNumber, SUM(ClaimAmount) AS TotalClaims FROM Claim WHERE SUM(ClaimAmount) < 1000 GROUP BY PolicyNumber;
   5. Having: What works
      * SELECT PolicyNumber, SUM(ClaimAmount) AS TotalClaims FROM Claim GROUP BY PolicyNumber HAVING SUM(ClaimAmount) < 1000;
5. Table Creation and Inserting Data: Using SQL
   1. Creation
      * CREATE TABLE DrivingFrequency (DrivingFrequency CHAR, Relativity DOUBLE);
      * CREATE TABLE CreditScore(CreditScore INT, Relativity DOUBLE);
   2. Insertion
      * INSERT INTO DrivingFrequency (DrivingFrequency, Relativity) VALUES ("Low", 0.85);
      * INSERT INTO DrivingFrequency (DrivingFrequency, Relativity) VALUES ("Medium", 1);
      * INSERT INTO DrivingFrequency (DrivingFrequency, Relativity) VALUES ("High", 1.1);
      * INSERT INTO DrivingFrequency (DrivingFrequency, Relativity) VALUES ("Really High", 1.3);
   3. Updating
      * UPDATE DrivingFrequency SET Relativity = 1.15 WHERE DrivingFrequency = "High";
   4. Deleting
      * DELETE FROM DrivingFrequency WHERE DrivingFrequency NOT IN ("Low", "Medium", "High")
   5. Dropping
      * DROP TABLE CreditScore;
   6. Creating a table from a query
      * SELECT DISTINCT PolicyNumber, DrivingFrequency, County INTO UniquePolicies FROM Policy;
6. VBA: Database tools 🡪 Visual Basic. Insert Macro and write a Subroutine.
   1. Basic Syntax: Creating Table
      * Sub CreateCountyTable()

Dim sql As String

sql = "CREATE TABLE County (County CHAR, Relativity DOUBLE);"

DoCmd.RunSQL sql

End Sub

* 1. Running multiple lines at once: Populating Table
     + Sub PopulateCountyTable()

DoCmd.RunSQL "INSERT INTO County (County, Relativity) VALUES (""Ventura"", 0.8);"

DoCmd.RunSQL "INSERT INTO County (County, Relativity) VALUES (""San Bernadino"", 0.85);"

DoCmd.RunSQL "INSERT INTO County (County, Relativity) VALUES (""Riverside"", 1);"

DoCmd.RunSQL "INSERT INTO County (County, Relativity) VALUES (""Orange"", 1.1);"

DoCmd.RunSQL "INSERT INTO County (County, Relativity) VALUES (""Los Angeles"", 1.2);"

End Sub

1. Joins
   1. Inner
      * SELECT Claim.\*, UniquePolicies.county

FROM Claim INNER JOIN UniquePolicies

ON Claim.PolicyNumber = UniquePolicies.PolicyNumber;

* + - SELECT p.county, SUM(c.ClaimAmount) AS TotalClaims

FROM Claim AS c INNER JOIN UniquePolicies AS p

ON c.PolicyNumber = p.PolicyNumber

GROUP BY p.county;

* 1. Left
     + SELECT p.PolicyNumber, p.DrivingFrequency, p.County, SUM(c.ClaimAmount) AS TotalClaims

FROM Policy as p LEFT JOIN Claim as c

ON p.PolicyNumber = c.PolicyNumber

GROUP BY p.PolicyNumber, p.DrivingFrequency, p.County;

* + - SELECT p.PolicyNumber, p.DrivingFrequency, p.County, SUM(NZ(c.ClaimAmount, 0)) AS TotalClaims

FROM Policy as p LEFT JOIN Claim as c

ON p.PolicyNumber = c.PolicyNumber

GROUP BY p.PolicyNumber, p.DrivingFrequency, p.County;

* 1. Multiple Joins
     + SELECT DISTINCT p.PolicyNumber, 2200 AS BaseRate, df.Relativity AS DrivingFreqRel, cty.Relativity as CountyRel

FROM ((Policy AS p

LEFT JOIN DrivingFrequency AS df ON p.DrivingFrequency = df.DrivingFrequency)

LEFT JOIN County as cty ON p.County = cty.County);

1. Subqueries
   * + SELECT PolicyNumber, BaseRate \* DrivingFreqRel \* CountyRel AS WrittenPrem FROM

(SELECT DISTINCT p.PolicyNumber, 2200 AS BaseRate, df.Relativity AS DrivingFreqRel, cty.Relativity as CountyRel

FROM ((Policy AS p

LEFT JOIN DrivingFrequency AS df ON p.DrivingFrequency = df.DrivingFrequency)

LEFT JOIN County as cty ON p.County = cty.County));

* + - SELECT PolicyNumber, BaseRate \* DrivingFreqRel \* CountyRel AS WrittenPrem INTO WrittenPremium FROM

(SELECT DISTINCT p.PolicyNumber, 2200 AS BaseRate, df.Relativity AS DrivingFreqRel, cty.Relativity as CountyRel

FROM ((Policy AS p

LEFT JOIN DrivingFrequency AS df ON p.DrivingFrequency = df.DrivingFrequency)

LEFT JOIN County as cty ON p.County = cty.County));

1. A difficult example with SWITCH: Calculating Earned Premiums and Loss Ratio
   1. Calculating Exposures
      * SELECT year, PolicyNumber, PolicyStartDate,

SWITCH(PolicyStartDate <= DateSerial(year - 1, 1, 1), 0,

PolicyStartDate <= DateSerial(year - 1, 12, 31),

DateDiff("d", DateSerial(year - 1, 1, 1), PolicyStartDate),

PolicyStartDate <= DateSerial(year, 12, 31),

DateDiff("d", PolicyStartDate, DateSerial(year + 1, 1, 1)),

PolicyStartDate > DateSerial(year, 12, 31), 0) AS exposures

FROM policy

* 1. Joining on Written Premium
     + SELECT p.PolicyNumber, p.PolicyStartDate, p.year AS RunYear, p.exposures/365 AS exposures, wp.WrittenPrem

FROM

(SELECT year, PolicyNumber, PolicyStartDate,

SWITCH(PolicyStartDate <= DateSerial(year - 1, 1, 1), 0,

PolicyStartDate <= DateSerial(year - 1, 12, 31),

DateDiff("d", DateSerial(year - 1, 1, 1), PolicyStartDate),

PolicyStartDate <= DateSerial(year, 12, 31),

DateDiff("d", PolicyStartDate, DateSerial(year + 1, 1, 1)),

PolicyStartDate > DateSerial(year, 12, 31), 0) AS exposures

FROM policy) AS p

LEFT JOIN WrittenPremium AS wp ON p.PolicyNumber = wp.PolicyNumber

WHERE p.exposures > 0;

* 1. Calculating Written Premium
     + SELECT "12-31-" & year AS RunDate, SUM(exposures \* WrittenPrem) AS EarnedPrem FROM

(SELECT p.PolicyNumber, p.PolicyStartDate, p.year AS RunYear, p.exposures/365 AS exposures, wp.WrittenPrem

FROM

(SELECT year, PolicyNumber, PolicyStartDate,

SWITCH(PolicyStartDate <= DateSerial(year - 1, 1, 1), 0,

PolicyStartDate <= DateSerial(year - 1, 12, 31),

DateDiff("d", DateSerial(year - 1, 1, 1), PolicyStartDate),

PolicyStartDate <= DateSerial(year, 12, 31),

DateDiff("d", PolicyStartDate, DateSerial(year + 1, 1, 1)),

PolicyStartDate > DateSerial(year, 12, 31), 0) AS exposures

FROM policy) AS p

LEFT JOIN WrittenPremium AS wp ON p.PolicyNumber = wp.PolicyNumber

WHERE p.exposures > 0);

* 1. Calculating Loss Ratio: A different type of subquery
* SELECT "12-31-" & year AS RunDate, (SELECT Sum(ClaimAmount) FROM Claim WHERE Year(ReportDate) = year)/SUM(exposures \* WrittenPrem) AS LossRatio FROM

(SELECT p.PolicyNumber, p.PolicyStartDate, p.year AS RunYear, p.exposures/365 AS exposures, wp.WrittenPrem

FROM

(SELECT year, PolicyNumber, PolicyStartDate,

SWITCH(PolicyStartDate <= DateSerial(year - 1, 1, 1), 0,

PolicyStartDate <= DateSerial(year - 1, 12, 31),

DateDiff("d", DateSerial(year - 1, 1, 1), PolicyStartDate),

PolicyStartDate <= DateSerial(year, 12, 31),

DateDiff("d", PolicyStartDate, DateSerial(year + 1, 1, 1)),

PolicyStartDate > DateSerial(year, 12, 31), 0) AS exposures

FROM policy) AS p

LEFT JOIN WrittenPremium AS wp ON p.PolicyNumber = wp.PolicyNumber

WHERE p.exposures > 0);