```sql
--Lets look at the table structure:

SELECT COLUMN_NAME, DATA_TYPE, IS_NULLABLE, COLUMN_DEFAULT
FROM INFORMATION_SCHEMA.COLUMNS
WHERE TABLE_CATALOG = 'loan_data'
AND TABLE_SCHEMA = 'dbo'
AND TABLE_NAME = 'loandata';

--Lets get a preliminary understanding of the distribution and central tendencies of our data by running some summary statistics.
SELECT
    AVG(fico) AS average_fico_score
FROM
    Dbo.loandata;

SELECT
    MIN(dti) AS min_dti_ratio,
    MAX(dti) AS max_dti_ratio
FROM
    Dbo.loandata;

SELECT
    AVG(int_rate) AS average_interest_rate
FROM
    Dbo.loandata;

SELECT
    AVG(revol_bal) AS average_loan_amount,
    MIN(revol_bal) AS min_loan_amount,
    MAX(revol_bal) AS max_loan_amount
FROM
    Dbo.loandata;

SELECT
    SUM(CASE WHEN not_fully_paid = 0 THEN 1 ELSE 0 END) * 100.0 / COUNT(*) AS percentage_fully_paid,
    SUM(CASE WHEN not_fully_paid = 1 THEN 1 ELSE 0 END) * 100.0 / COUNT(*) AS percentage_not_fully_paid
FROM
    Dbo.loandata;


--Lets look into the most common reasons borrowers are seeking loans.
SELECT purpose, COUNT(*) as total_loans
FROM dbo.loandata
GROUP BY purpose
ORDER BY total_loans DESC;

--Let's look at the range of FICO scores, by grouping them into categories (e.g., 650-699, 700-749) to see how average interest rates vary across these ranges.
SELECT
    CASE
```

```
        WHEN fico >= 650 AND fico < 700 THEN '650-699'
        WHEN fico >= 700 AND fico < 750 THEN '700-749'
        WHEN fico >= 750 AND fico < 800 THEN '750-799'
        ELSE '800+'
    END AS ficorange,
    AVG(int_rate) as average_interest_rate
FROM dbo.loandata
GROUP BY
    CASE
        WHEN fico >= 650 AND fico < 700 THEN '650-699'
        WHEN fico >= 700 AND fico < 750 THEN '700-749'
        WHEN fico >= 750 AND fico < 800 THEN '750-799'
        ELSE '800+'
    END
ORDER BY ficorange;

--This query calculates the number of loans not fully paid and the total number of
  loans for each purpose, along with the proportion of loans not fully paid, offering
  insights into which loan purposes have higher default rates.
SELECT purpose,
       SUM(CASE WHEN not_fully_paid = 1 THEN 1 ELSE 0 END) as not_fully_paid,
       COUNT(*) as total_loans,
       SUM(CASE WHEN not_fully_paid = 1 THEN 1 ELSE 0 END) / COUNT(*) as
         proportion_not_paid
FROM dbo.loandata
GROUP BY purpose
ORDER BY proportion_not_paid DESC;

--This query provides a high-level view of loan repayment across the entire dataset,
  showing how many loans were fully paid versus those that were not.
SELECT
    not_fully_paid,
    COUNT(*) as total_loans
FROM dbo.loandata
GROUP BY not_fully_paid;


--I am interested to see the relationship between fico score and performance.

SELECT
    AVG(fico) AS avg_fico_score_fully_paid,
    COUNT(*) AS count_fully_paid
FROM
    dbo.loandata
WHERE
    not_fully_paid = 0;

SELECT
    AVG(fico) AS avg_fico_score_not_fully_paid,
    COUNT(*) AS count_not_fully_paid
FROM
    dbo.loandata
```

```sql
WHERE
    not_fully_paid = 1;

--I wanted to see the relationship between DTI and loan repayment:
SELECT
    not_fully_paid,
    AVG(dti) AS average_dti
FROM
    dbo.loandata
GROUP BY
    not_fully_paid;

--Now, let analyze interest rate across different fico ranges.
SELECT
    CASE
        WHEN fico >= 650 AND fico < 700 THEN '650-699'
        WHEN fico >= 700 AND fico < 750 THEN '700-749'
        WHEN fico >= 750 AND fico < 800 THEN '750-799'
        ELSE '800+'
    END AS fico_range,
    AVG(int_rate) AS average_interest_rate
FROM
    dbo.loandata
GROUP BY
    CASE
        WHEN fico >= 650 AND fico < 700 THEN '650-699'
        WHEN fico >= 700 AND fico < 750 THEN '700-749'
        WHEN fico >= 750 AND fico < 800 THEN '750-799'
        ELSE '800+'
    END
ORDER BY
    CASE
        WHEN fico >= 650 AND fico < 700 THEN '650-699'
        WHEN fico >= 700 AND fico < 750 THEN '700-749'
        WHEN fico >= 750 AND fico < 800 THEN '750-799'
        ELSE '800+'
    END;

--Comparing Loan Purposes and Their Impact on Repayment Rates.
SELECT
    purpose,
    SUM(CASE WHEN not_fully_paid = 0 THEN 1 ELSE 0 END) * 100.0 / COUNT(*) AS
      percent_fully_paid,
    SUM(CASE WHEN not_fully_paid = 1 THEN 1 ELSE 0 END) * 100.0 / COUNT(*) AS
      percent_not_fully_paid
FROM
    dbo.loandata
GROUP BY
    purpose
ORDER BY
    percent_not_fully_paid DESC;
```

```sql
--Let's confirm hypothesis
SELECT
    purpose,
    COUNT(*) AS total_loans,
    SUM(CASE WHEN not_fully_paid = 0 THEN 1 ELSE 0 END) AS fully_paid,
    SUM(CASE WHEN not_fully_paid = 1 THEN 1 ELSE 0 END) AS not_fully_paid,
    SUM(CASE WHEN not_fully_paid = 0 THEN 1 ELSE 0 END) * 100.0 / COUNT(*) AS
      percent_fully_paid,
    SUM(CASE WHEN not_fully_paid = 1 THEN 1 ELSE 0 END) * 100.0 / COUNT(*) AS
      percent_not_fully_paid
FROM
    dbo.loandata
GROUP BY
    purpose
ORDER BY
    percent_not_fully_paid DESC;

SELECT
    not_fully_paid,
    AVG(dti) AS average_dti,
    MIN(dti) AS min_dti,
    MAX(dti) AS max_dti
FROM
    dbo.loandata
GROUP BY
    not_fully_paid;
```