Problem Statement 1

Just like every other day at Autochek, The overall goal starts by getting our dealers to list their cars, displaying these cars to the prospective customers on our marketplace and then providing affordable car loans to make life easy, you as our data engineer suddenly gets a data request around one of our most delicate dataset, Apparently, the business leaders would like to see a summarized table generated from data of the **customer (borrower_table)**, the **loans they currently have(loans table)**, the dates they have been **scheduled to repay (payment_schedule)**, **how frequent they are paying back (loan_payment)**, lastly a table that shows, history of times customers have **missed their payments (missed_payment)**, these data lives in source areas in the data warehouse.

You are now tasked with creating dimensional models that will allow Data and, BI Analytics to efficiently query, and cost effectively query the data for Financial Reporting for the business leaders.

Once the design is complete, create a ELT Pipeline to load the data into source area and transform it into the required formats

NOTE: You are free to use any school of thought for Data warehouse design. You can also use PostgreSQL, MariaDB, BigQuery or any other SQL Syntax Database/Warehouse you are comfortable with.

Data and Schema Explanation

(1) Borrower_table (One borrower, one record) - https://docs.google.com/spreadsheets/d/1mfm4NUfv4wOJfMdjOIA5k99N79JFJDJeTBhzUflYtzs/edit?usp=sharing
- Borrower_id (pk)
- State
- City
- zip code
- created_at

(2) Loan_table (One borrower, one or multiple loans) - https://drive.google.com/file/d/1Lid1RDMGNpXWv2PyU0Alsv-sSBnkGMuS/view?usp=sharing
- Borrower_id (fk)
- Loan_id(pk)
- Date_of_release
- Term
- InterestRate
- LoanAmount
- Downpayment
- Payment_frequency

- Maturity_date
- created_at

(3) Payment_Schedule (One loan, one or multiple payment schedules) - https://docs.google.com/spreadsheets/d/1LawsJQtLGpO6AQZFDpgSUEn8A_TVFD9I/edit?usp=sharing&ouid=112035781102155860914&rtpof=true&sd=true
- loan_id(fk)
- schedule_id(pk)
- Expected_payment_date
- Expected_payment_amount
- created_at

(4) Loan_payment (One loan, one or multiple payments) - https://drive.google.com/file/d/1qdV_WcVmvo7VsyxJxfU8RXpdOgKZ7V3A/view?usp=sharing
- loan_id(fk)
- payment_id(pk)
- Amount_paid
- Date_paid
- created_at

Expected steps:

1. Go through the above schemas, try to understand the relationship between the schemas
2. Design and develop dimensional designs that will be used for calculating ***Step 4***.
3. Implement an ELT Pipeline that will be used for loading the data into the source area and transforming it into required Dimensional models.
4. Using the dimensional tables above, write queries to Calculate PAR Days - Par Days means the number of days the loan was not paid in full. Eg If the loan repayment was due on the 10th Feb 2022 and payment was made on the 15th Feb 2022 the par days would be 5 days ***(NOTE: For each day a customer missed a payment the amount_at_risk is the total amount of money we are expecting from the customer as at that time, for instance, if the customer owes for 5000 from month 1 and 10000 for current month the amount_at_risk will be the total amount 5000 + 10000 = 15000)***
5. You can use any Python based tool to achieve this
6. Implement a monitoring and alerting system that can help identify and alert on errors that happen when the pipeline is running.
7. You can optionally Implement a Deployment Workflow to handle auto deployments when certain checks are met.
8. Write proper documentation for your implementation.
9. Push this to a github repo alongside the solution for statement 2, detailing your solution.

Problem Statement 2

As a data engineer at Autochek you will be required to gather data from different sources, and one of the techniques that you will use is Website Scraping. On this Problem statement you will be required to write a crawler that will be used to scrape car prices from a website of your choice. The crawler that you will be implementing, should be intelligent enough to bypass common scraping patterns like domain blacklisting, ip blocking, rate limiting, and Dynamic JavaScript page contents. In this implementation, you should write a pipeline to schedule and run the spider on a daily basis and load the data into a warehouse in an incremental manner.

Expected steps:

10. Choose your favorite website to scrape for car prices
11. Analyze html structure and DOM elements to understand it better.
12. Write a spyder to scrape the data from the website.
13. Write a script to schedule and automate the scraping process on a daily basis.
14. Add monitoring and Alerting for success, errors and failures.
15. Write proper documentation for your implementation.
16. Push your code to a github page, detailing your script and how to run it.

NOTE: When scraping, scrape responsibly and make sure you do not infringe any policies for the website you are scraping.