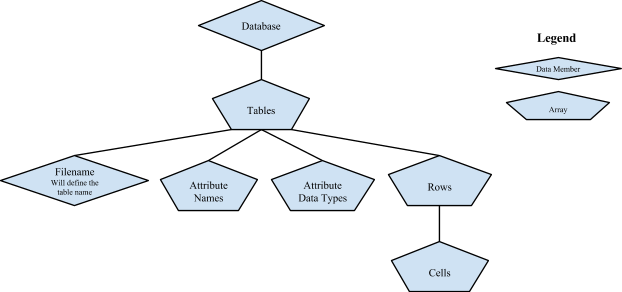
Relational Database Management System (RDBMS)

Design documents for Phases I and II

1. Project Objective

To create a database and an accompanying management system. The database will be then used to implement an application that uses the database system. We will be creating a car catalog application to demonstrate all the features the database has to offer and how it is managed through the use of our data definition language (DDL) and data manipulation language (DML).

In order to run the car catalog, our database must be able to store car data entries and create relation tables based on specific attributes. Specific cars in the database will also be related to user profiles through the use of search filters.The application will allow users to search and compare cars by these data attributes and hopefully find one they like. The database and its management system can later be used to develop many applications that require efficient data storage and manipulation.

2. High-level Design Overview

The entire database will be represented by an array of “table” objects. Each table will consist of a filename, which will define the name of the table, an array of attribute names (e.g., “firstName,” “lastName,” “email”), and an array of attribute data types (e.g., int, char, varchar). The table will also contain an array of rows, and each row will contain an array of cells, which will contain the data we are storing in the database.

3. Low-level Design Overview

**Usage:**

**Object: Car Table**

The car table will be used for storing all information about the individual cars, including members such as make, model, year, mileage, car ID and more. All relations will be applied to these tables by the input.

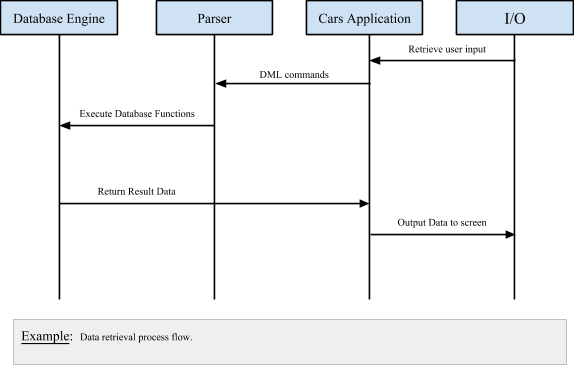
**Object: User Table**

The user table will be an object that is used for storing all information about the users who access the database and are linked to the cars. It will include members such as name, ID, and car ID that it is linked to. It is used to find which cars are related to which people.

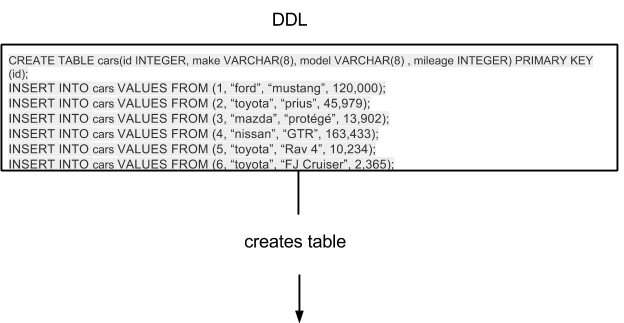
**Configuration:**

Different objects in the database will require different configurations based on their sizes. For instance, a car table will require more columns than the person table because it has more attributes.

**Interaction:**



**Model:**



|  |  |  |  |
| --- | --- | --- | --- |
| Id | make | model | mileage |
| 1 | ford | mustang | 121,000 |
| 2 | toyota | prius | 45,979 |
| 3 | mazda | protégé | 13,902 |
| 4 | nissan | GTR | 163,433 |
| 5 | toyota | Rav 4 | 10,234 |
| 6 | toyota | FJ Cruiser | 2,365 |

4. Benefits & Issues

**Benefits:**

* A more quick and concise implementation of a database
* Because of our linear storage technique, the database is easier to troubleshoot and maintain
* Predefined attribute enumerations insures data type safety
* Our car application allows flexible entry of user’s car data
* Since all data is output to text files, the information is very portable

**Risks:**

* Performance risks because searching through ASCII text is less efficient than a binary tree.
* No network capabilities
* Memory is not encrypted so it is prone to security risks

**Assumptions:**

* That the user knows the correct data to be entered when queried
* That there will be sufficient memory for all data that needs to be stored