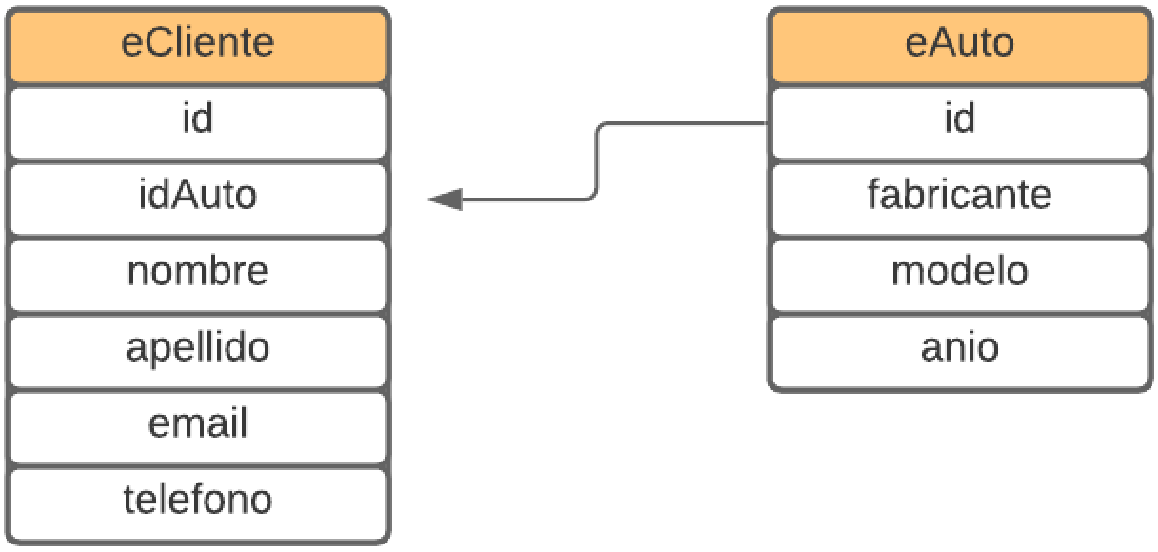
# Recuperatorio Segundo Parcial
# Laboratorio 1

Nombre y Apellido: Mariano Forte
Curso: 1F
DNI: 36785993
N.º Legajo: 110518
E-mail: marianoforte92@gmail.com

## Diagrama Entidad-Relación de las Estructuras:



## Enunciados:

1. Cargar ambas listas desde sus respectivos archivos csv.
2. Listar los datos relacionados. No se deben visualizar IDs.
3. Utilizar la función ll_filter para realizar dos filtros distintos.
4. Utilizar la función ll_count para realizar dos cálculos (contador/acumulador)
5. Guardar en un archivo csv los datos de alguno de los dos filtros del punto 3.

# Prototipos de funciones del núcleo del programa:

```
/** \brief Loads data from a text file
 *
 * \param path char* Path where to find the file
 * \param myList LinkedList* Pointer to the linkedlist
 * \param option int The option of which list to load
 * \return int  [-1] if path or pointer to list are NULL
               [0] if elements already exists in the list and the users cancel the load
                OR amount of elements successfully loaded
 *
 */
int forte_controller_loadFromText(char* path , LinkedList* myList, int option);
```

--------------------------------------------------------------------------------------------------------------

```
/** \brief Shows all clients with their cars, no IDs
 *
 * \param myClients LinkedList* pointer to client's list
 * \param myCars LinkedList* pointer to car's list
 * \return int [-1] if pointer to list is NULL
              [0] if the list is empty
              [1] if the print was successfull
 *
 */
int forte_controller_ListClientsWithCars(LinkedList* myClients, LinkedList* myCars);
```

--------------------------------------------------------------------------------------------------------------

```
/** \brief Searchs the clients list for the client that matches the passed id
 *
 * \param myList LinkedList* pointer to client list
 * \param targetId int searched ID
 * \return int  [-2] if pointer to employee list is NULL or the list is empty
               [-1] if passed id is not found
               Index of employee if ok
 *
 */
int forte_controller_searchClientById(LinkedList* myList, int targetId);
```

--------------------------------------------------------------------------------------------------------------

```
/** \brief Searchs the cars list for the car that matches the passed id
 *
 * \param myList LinkedList* pointer to car list
 * \param targetId int searched ID
 * \return int  [-2] if pointer to employee list is NULL or the list is empty
               [-1] if passed id is not found
               Index of employee if ok
 *
 */
int forte_controller_searchCarById(LinkedList* myList, int targetId);
```

--------------------------------------------------------------------------------------------------------------

```
/** \brief Filters a list by a criteria
 *
 * \param clients LinkedList* The list to filter
 * \param cars LinkedList* The list to filter
 * \param criterio void* The criteria to filter
 * \return int [-1] if the list is NULL
             [0] if the filtering was successfull
 *
 */
int forte_controller_FilterClientsByCar(LinkedList* clients, LinkedList* cars, int criterio);
```

----------------------------------------------------------------------------------------------------------------

```
/** \brief Filters a list by a criteria
 *
 * \param cars LinkedList* The list to filter
 * \param criterio void* The criteria to filter
 * \return int [-1] if the list is NULL
             [0] if the filtering was successfull
 *
 */
int forte_controller_FilterCarsByYear(LinkedList* cars, int criterio);
```

----------------------------------------------------------------------------------------------------------------

```
/** \brief Counts the number of appearences of a single element on the listing
 *
 * \param this LinkedList* The list where to count
 * \param criteria char* The criteria to filter
 * \return int [-1] if the list is NULL
             [0] If none element was found
             The amount of elements if the count was successfull
 *
 */
int forte_controller_CountClientsByLastName(LinkedList* this, char* criteria);
```

----------------------------------------------------------------------------------------------------------------

```
/** \brief Counts the number of appearences of a single element on the listing
 *
 * \param this LinkedList* The list where to count
 * \param cars LinkedList* The list where to count
 * \param criteria char* The criteria to filter
 * \return int [-1] if the list is NULL
             [0] If none element was found
             The amount of elements if the count was successfull
 *
 */
int forte_controller_CountCarsByBrand(LinkedList* this, LinkedList* cars, char* destination);
```

----------------------------------------------------------------------------------------------------------------

```
/** \brief Filters a list by a criteria
 *
 * \param this void* The list to filter
 * \param criteria void* The criteria to filter
 * \return int [-1] if the list or criteria are NULL
            [0] if the filtering was successfull
 *
 */
int cliente_filterByCarId(void* this, void* criteria);
```

-----------------------------------------------------------------------------------------------------------------------------

```
/** \brief Counts the number of appearences of a single element on the listing
 *
 * \param pElement void* The element to count
 * \return int [-1] if the list or criteria are NULL
            [0] if the filtering was successfull
 *
 */
int cliente_countClientsByLastName(void* pElement);
```

-----------------------------------------------------------------------------------------------------------------------------

```
/** \brief Filters a list by a criteria
 *
 * \param this void* The list to filter
 * \param criteria void* The criteria to filter
 * \return int [-1] if the list or criteria are NULL
            [0] if the filtering was successfull
 *
 */
int auto_filterByBrand(void* this, void* criteria);
```

-----------------------------------------------------------------------------------------------------------------------------

```
/** \brief Filters a list by a criteria
 *
 * \param this void* The list to filter
 * \param criteria void* The criteria to filter
 * \return int [-1] if the list or criteria are NULL
            [0] if the filtering was successfull
 *
 */
int auto_filterByModel(void* this, void* criteria);
```

-----------------------------------------------------------------------------------------------------------------------------

```
/** \brief Filters a list by a criteria
 *
 * \param this void* The list to filter
 * \param criteria void* The criteria to filter
 * \return int [-1] if the list or criteria are NULL
            [0] if the filtering was successfull
 *
 */
int auto_filterByYear(void* this, void* criteria);
```

```
----------------------------------------------------------------------------------------------------------------------------------

/** \brief Counts the number of appearences of a single element on the listing
 *
 * \param pElement void* The element to count
 * \return int [-1] if the list or criteria are NULL
             [0] if the filtering was successfull
 *
 */
int auto_countCarsByBrand(void* pElement);
```

## <u>Enlace a Google Drive con el video explicativo</u>:

https://drive.google.com/file/d/1uUAIG0W9SvwOlWBqBV-xr-M3I6Qz9hh4/view?
usp=sharing