A picture containing text

Description automatically generated

**SESSION 2020/2021 SEMESTER 1**

**SECD2523**

**DATABASE**

**Alternative Assessment**

**NAME**

Hamzeh Wahid Bajbouj

**MATRIC NO**

A19EC4009

**LECTURER’S NAME**

Dr.Ahmad Fadhil Bin Yusof

**SECTION**

5

**1-Introduction:**

**Overview of current system:**

The current database system of the Homeless Coalition exists on the county’s mainframe. That is, all the data that the organization needs to print and update reports, which consists of the location, availability and condition of low-income households, homeless and low-income people names and personal information, as well as the information of houses and property owners. This causes many problems and difficulties for the employees of the organization to finish their work in the appropriate time.

**Mission statement:**

The mission statements here are paraphrased and shortened from the previous one with more clarity.

The purpose of the Homeless Coalition databases is to maintain the data that they collect, locally instead of storing them in second party providers, the data also is used to generate reports that is requested by the Homeless Coalition clients who work with the low-income and homeless population.

the mission statements can be summarized into these points:

1. The data that the organization needs is stored in second party organization which is in the county’s mainframes, not locally.
2. Generating and printing out new reports by using the data take days and sometimes months, since it’s not stored locally.
3. Importing and updating the data into the database take a lot of time.

**Mission objectives:**

1- To create a database system to store data locally.

2- To Provide a method for faster handling and upgrading of data.

3- To reduce the time required to import and export data.

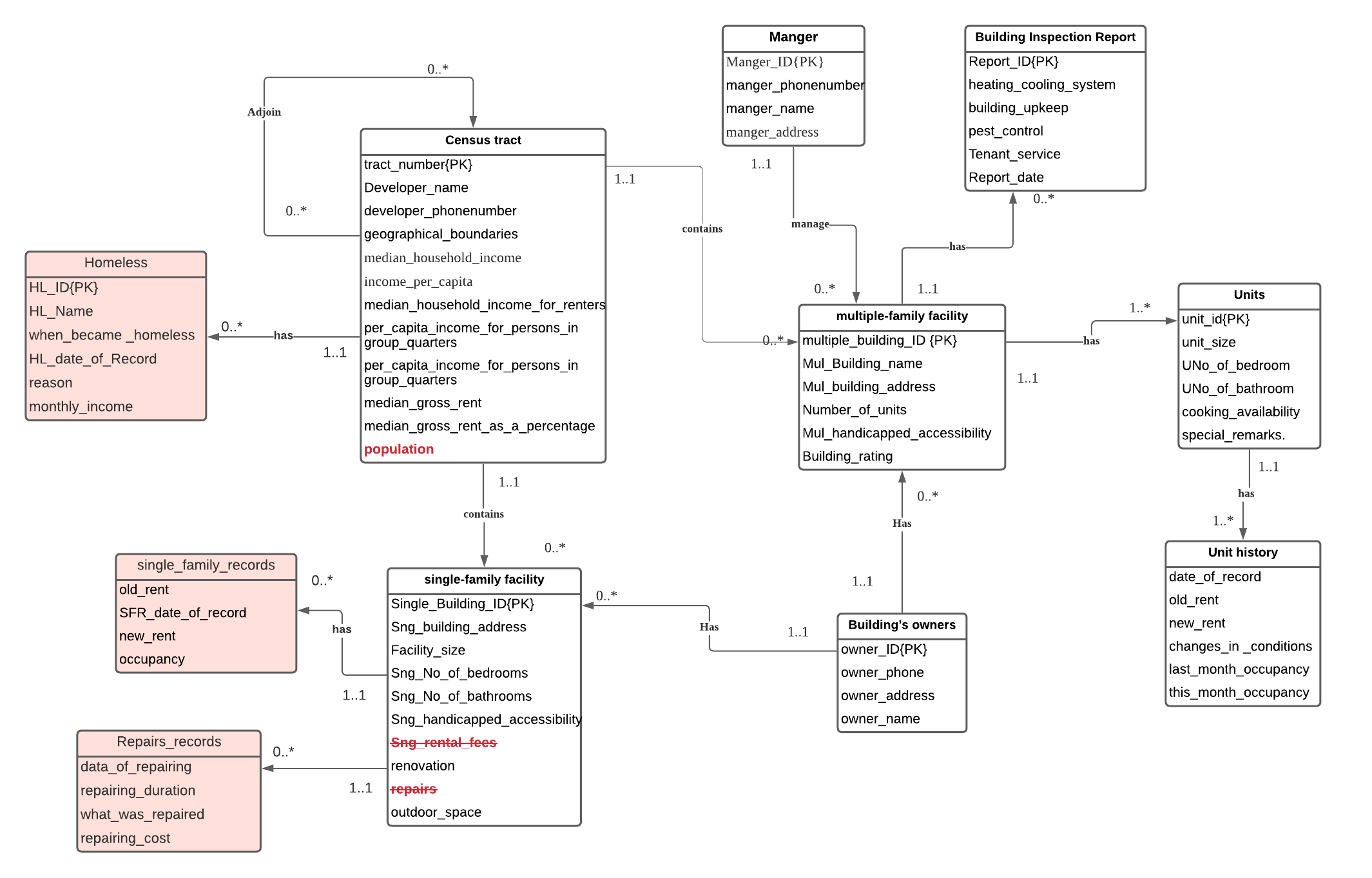
**2- Summary of database planning and design:**

In our project, I found out that there are some improvements can be done on the database design that can provide more functionally and feature to the system, these improvements can be done by adding a new entity that will store the data regarding to homeless people in each tract, a new attribute also will be introduced to census tract table to store the population number in each tract, the new table will provide us more informative reports for the clients, since it will contain the homeless name, their reason of that, since how long they been living in the street and also where are their current location all these information are already collected by Homeless Coalition.

I also found out some weaknesses points in the database design, for the single-family facility table there should be another table to track the history of single-family facilities in order to provide more details regrading to each facility as we’ve done the unit table. The new table will contain almost the same attributes that unit history table has.

A third table also will be added to the database design which will also be related to single-family facility table, this new table will store information about each repair operation that has been done on the facility, it will show what kind of repairs it’s also will record the date of the repair with some additional data.

**Part (A): Updated ERD (conceptual design)**



**Updated Data Dictionary for ERD**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entity Name** | **Attributes** | **Description** | | **Data type & length** |
| **Manger** | **Manger\_name**  Manger\_Fname  Manger\_Lname | First name of the manger  Last name of the manger | | 15 variable characters  15 variable characters |
| **manger\_phone\_number** | Manger’s phone number | | 9 variable numbers |
| **Manger\_ID** | Uniquely identifier for the mangers | | 5 variable numbers |
| **manger\_address**  manger\_ZIP  manger\_city  manger\_street | The ZIP codes  The city name.  The street name. | | 3 variable numbers  10 variable characters  20 variable characters |
| **Building Inspection Report** | **Report\_ID** | Uniquely identifier for the reports | | 6 variable numbers |
| **Tenant\_service** | the availability of Tenant Services | | 1 character (Y – N) |
| **Heating\_cooling system** | the availability H/C system | | 1 character (Y – N) |
| **building upkeep** | Any maintenance since last report has been done | | 1 character (Y – N) |
| **pest control** | the availability of pest control services | | 1 character (Y – N) |
| **Report\_date** | The date of the report | | Date |
| **Building’s owners** | **owner\_name**  Owner\_Fname  Owner\_Lname | First name of the Owner  Last name of the Owner | | 15 variable characters  15 variable characters |
| **owner\_phone** | Owner’s phone number | | 9 variable numbers |
| **owner\_address**  Owner\_ZIP  Owner\_city  Owner\_street | The ZIP codes  The city name.  The street name. | | 3 variable numbers  10 variable characters  20 variable characters |
| **owner\_ID** | Uniquely identifier for the owners | | 5 variable numbers |
| **Units** | **Unit\_id** | Uniquely identifier for the units | | 10 variable numbers |
| **unit\_size** | The size of the unit in meter squared | | 4 variable numbers |
| **UNo\_of\_bedroom** | Number of bedrooms within the unit | | 2 variable numbers (0-99) |
| **UNo\_of\_bathroom** | Number of bathrooms within the unit | | 1 variable number (0-9) |
| **cooking\_availability** | the availability of cooking facilities | | 1 character (Y – N) |
| **special\_remarks** | Any special remarks | | 500 variable characters |
| **Unit history** | **date\_of\_record** | Will contain the date of each record, when it was recorded | | date |
| **Old\_rent** | The old rent of the unit | | 10 variable numbers |
| **New\_rent** | The new rent of the unit | | 10 variable numbers |
| **changes\_in\_conditions** | Any changes in the condition, new conditions, modified conditions | | 1000 variable characters |
| **Last\_month\_occupancy** | The unit occupancy for last month | | 1 character (O – V)  O: occupied  V: vacant |
| **This\_month\_occupancy** | The unit occupancy for current month | | 1 character (O – V)  O: occupied  V: vacant |
| **Single-family facility** | **Single\_Building\_ID** | Uniquely identifier for the single-family buildings | | 6 variable numbers |
| **Sng\_building\_address**  Sng\_ZIP  Sng \_city  Sng\_street | The ZIP codes  The city name.  The street name. | | 3 variable numbers  10 variable characters  20 variable characters |
| **facility\_size** | The size of the single-family building in meter squared | | 4 variable numbers |
| **Sng\_No\_of\_bedrooms** | Number of bedrooms within the single-family building | | 2 variable numbers (0-99) |
| **Sng\_No\_of\_bathrooms** | Number of bathrooms within the single-family building | | 1 variable number (0-9) |
| **Sng\_handicapped\_accessibility** | supporting of Handicapped accessibility | | 1 character (Y – N) |
| **~~Sng\_rental\_fees~~ (removed)** | ~~The rent fees for the single-family building~~ | | ~~5 variable numbers~~ |
| **renovation** | Has it been renovated | | 1 character (Y – N) |
| **~~Repairs~~ (removed)** | ~~Number of repairs times~~ | | ~~2 variable numbers~~ |
| **outdoor\_space** | Are there outdoor spaces (e.g. yard)? | | 1 character (Y – N) |
| **multiple-family facility** | **multiple\_building\_ID** | Uniquely identifier for the multiple-family buildings | | 6 variable numbers |
| **Mul\_Building\_name** | The name of the building | | 25 variable characters |
| **Mul\_building\_address**  Mul\_ZIP  Mul\_city  Mul\_street | The ZIP codes  The city name.  The street name. | | 3 variable numbers  10 variable characters  20 variable characters |
| **Number\_of\_units** | Number of units within each building | | 3 variable numbers |
| **Mul\_handicapped\_accessibility** | supporting of Handicapped accessibility | | 1 character (Y – N) |
| **Building\_rating** | The rating of the building | | 1 variable number |
| **Census tract** | **tract\_number** | Uniquely identifier for each tract | | 2 variable numbers |
| **Developer\_name**  Developer\_Fname  Developer\_Lname | The developer first name and last name for this tract | | 15 variable characters  15 variable characters |
| **developer\_phonenumber** | The developer phone number | | 9 variable numbers |
| **geographical\_boundaries** | The geographical boundaries for each tract, bordered with other tracts | | 4 variable numbers |
| **median\_household\_income** | The median income for the households in the tract | | 8 variable numbers |
| **income\_per\_capita** | The income per capita for each tract | | 8 variable numbers |
| **median\_household\_income\_for\_renters** | The median income for the household’s renters in the tract | | 8 variable numbers |
| **per\_capita\_income\_for\_persons\_in\_group\_quarters** | The per capita income for each person in group quarters | | 8 variable numbers |
| **median\_gross\_rent** | Median gross rent in the tract | | 8 variable numbers |
| **median\_gross\_rent\_as\_a\_percentage** | Median gross rent as a percentage in the tract | | 4 variable numbers |
| **Population** | Will contain the population within each tract | | 10 variable numbers |
| **Homeless** | **HL\_ID** | Uniquely identifier for each homeless record | | 10 variable numbers |
| **HL\_Name**  **HL\_Fname**  **HL\_Lname** | Homeless first and last name | | 15 variable characters  15 variable characters |
| **when\_became \_homeless** | What was the date that he/she when became homeless | | Date |
| **HL\_date\_of\_Record** | The date when recording this information | | Date |
| **reason** | the reason for their homelessness | | 1000 variable characters |
| **monthly\_income** | How much they earned per month | | 10 10 variable numbers |
| **single\_family\_records** | **old\_rent** | The old rent of the facility | | 10 variable numbers |
| **SFR\_date\_of\_record** | Will contain the date of each record, when it was recorded | | Date |
| **new\_rent** | The new rent of the facility | | 10 variable numbers |
| **occupancy** | The unit occupancy for current month | | 1 character (O – V)  O: occupied  V: vacant |
| **Repairs\_records** | **data\_of\_repairing** | | When was the repairing | Date |
| **repairing\_duration** | | How many days did it take | 3 variable numbers |
| **what\_was\_repaired** | | What are the things that was repaired and fixed | 1000 variable characters |
| **repairing\_cost** | | How much did the repairing cost | 10 variable numbers |

**System’s functional requirement:**

**Data requirements:**

**Census tract:**

the data are provided by Census Bureau includes a lot of details about census tracts in the county, in matter of fact the data held here contains the information about 50 census tracts even there is more than 300 census tracts in the county, the reason behind that they have the highest percentage of population living below poverty level. The data generally contains details about tract number within the county, its geographical boundaries, population in each census tract, it also gives an overview of the population in terms of their income, it as well describes the median gross rent in addition to many details regarding to households’ incomes …etc.

**single-family facility:**

the describes the single-family facilities in terms of the building address (its location in which census tract), the size of the facility, number of bedrooms, number of bathrooms, accessibility by handicapped, rental fees, renovations and repairs as well as if there is outdoor living space (e.g fenced backyard …etc) we also want to know the building owner.

**multiple-family facility:**

this kind of facilities can contain many units (apartments for instance) and owned by a person or maybe a private company, the data stored regarding to each facility are, the building name, the building address, number of units, manger or owner phone number, manger name, accessibility by handicapped, rating of the building.

**Units:**

In each **multiple-family facility** there will be many units these units are different from each other even if they are in the same building, each unit is distinguished by a unique number, and they are described by the size, number of bedrooms, rental fees, number of bathrooms, availability for cooking facility as well if there any special remarks, keep in mind that the units that have 0 bedrooms are efficiency units.

**Building’s owners:**

The facilities are owned by people usually and sometimes by private companies, the data held here are the owner name, owner’s phone number as well as his/ her address.

**Unit history:**

Each unit in the multiple-family facility will have a history which will contains details about the units such as old rental fees, the new rental fees, change the condition status, and some details regarding to occupancy for the unit for the previous months and current month.

If the facility is single-family there will be history for it as well.

**Building Inspection Report:**

The data here will used to rate the building in terms of the services that provide such as tenant service, heating/ cooling system, building upkeep, pest control.

**Manager:**

The data held here are the manager name as well as his phone number and his address.

**Homeless:**

All the homeless important information will be stored and held by this table such as their name, what was their reasons for homelessness and their monthly income if they have and many other data.

**single\_family\_records:**

the data will be held here will be the history for each single-family building which will hold the old rent, new rent, the occupancy, date of record.

**Repairs\_records:**

For each single-family facility the data for each repairing that has been done if any will be stored, it will contains the date of repairing, how long did it take , and the cost as well as what was fixed in the building.

**Transaction requirements:**

**Data entry (Census tract):**

* Enter details about Census tract geographical boundaries.
* Enter a new tract information.

**Data update/deletion (Census tract):**

* Update median gross rent.
* Update income per capita
* Delete a tract data from the records.

**Data quires (Census tract):**

* List all the tract number along with their geographical boundaries.
* List all the income per capita for all the tract in ascending order.

**Data entry (facilities):**

* enter a new facility address.
* Enter the building name.
* Enter the number of units.

**Data update/deletion (facilities):**

* Update the owner name.
* Update the rental fees.
* Update the building name.

**Data quires (facilities):**

* List all the facilities located in a specific tract.
* List all the units in specific building.
* View all the units by the number of bedrooms.
* Show the average rent fees for a specific building.

**Data entry (Building’s owners):**

* Enter the owner information.

**Data update/deletion (Building’s owners):**

* Update the owner phone number.
* Update the owner address.

**Data quires (Building’s owners):**

* List all the owners name along the with the building they own.

**Data entry (Unit history).**

* Enter a new data about a new building facility

**Data update/deletion (Unit history).**

* Update the rental fees.
* Delete the old rental fee.
* Update month occupancy.

**Data quires (Unit history).**

* List all the units in ascending order by their cheapest rental fees.
* View the current month occupancy for a specific unit.

**Data entry (Building Inspection Report):**

Enter the building inspection data.

**Data update/deletion (Building Inspection Report).**

Update the pest control rate.

Update the heating and cool system service rate

. **Data quires (Building Inspection Report):**

List the first 10 ranked building along with the owner phone number as well his name.

Show the average rate for each single building in a specific tract.

**Data entry (Homeless):**

* Enter details about homeless people
* Enter their location \*tract

**Data update/deletion (Homeless):**

* Delete a homeless person data from the records.

**Data quires (Homeless):**

* List all homeless with their monthly income
* Show the average homelessness duration for all homelessness

**Data entry (single\_family\_records):**

* Enter a new record for a single-family building.

**Data update/deletion (single\_family\_records):**

* Update record date.
* Update the new rent.

**Data quires (single\_family\_records):**

* List all the records based on the new rent from highest rent to lowest rent

**Data entry (Repairs\_records):**

* Enter new repairing record.
* Insert a the cost for the repairing

.

**Data update/deletion (Repairs\_records):**

* Update the repairing cost.
* Update repairing date
* Update the repairing duration

**Data quires (Repairs\_records):**

* List all the building with their repairing records
* List only the repairing records that has been done on the wall.

**More complex Transaction requirements:**

* list all the homeless names along with the duration they spent it in the homeless order in ascending order based on the duration.
* show me the total number of homeless people within each tract.
* list me all the single-family building latest rentals along with the homeless who his/her monthly income higher than the latest rental and then show the difference if he/she rented the building.
* compare between how many times the single-family buildings were occupied and when they were vacant.
* Display only the buildings that have been repaired and in the same time show if there is a changed happened to the rental as well as show what has been repaired.
* show the percentage of homeless people to the population in their tract.
* Display the highest repairing cost beside the highest single family building among all the years.

**Functional requirement:**

**Maintain the Data:**

Maintaining data is a beneficial way to keep data secure and available all the time. In the HIS new system, one of the main features is data maintain. Employees of the organization would have the chance to store the data in a particular order and keep it in an available location. It will also allow them to make copies and back-ups in case of its loss.

**Analyzing Data:**

The new system will have a data analyzing function; this function will help the employees analyze by allowing them to export the data to specialized applications which will help in the testing and analysis process.

**Data Access:**

The system has fast and safe access to the data and does not need a third party to provide the organization with the required data. This function will lead to quicker and more reliable data manipulation.

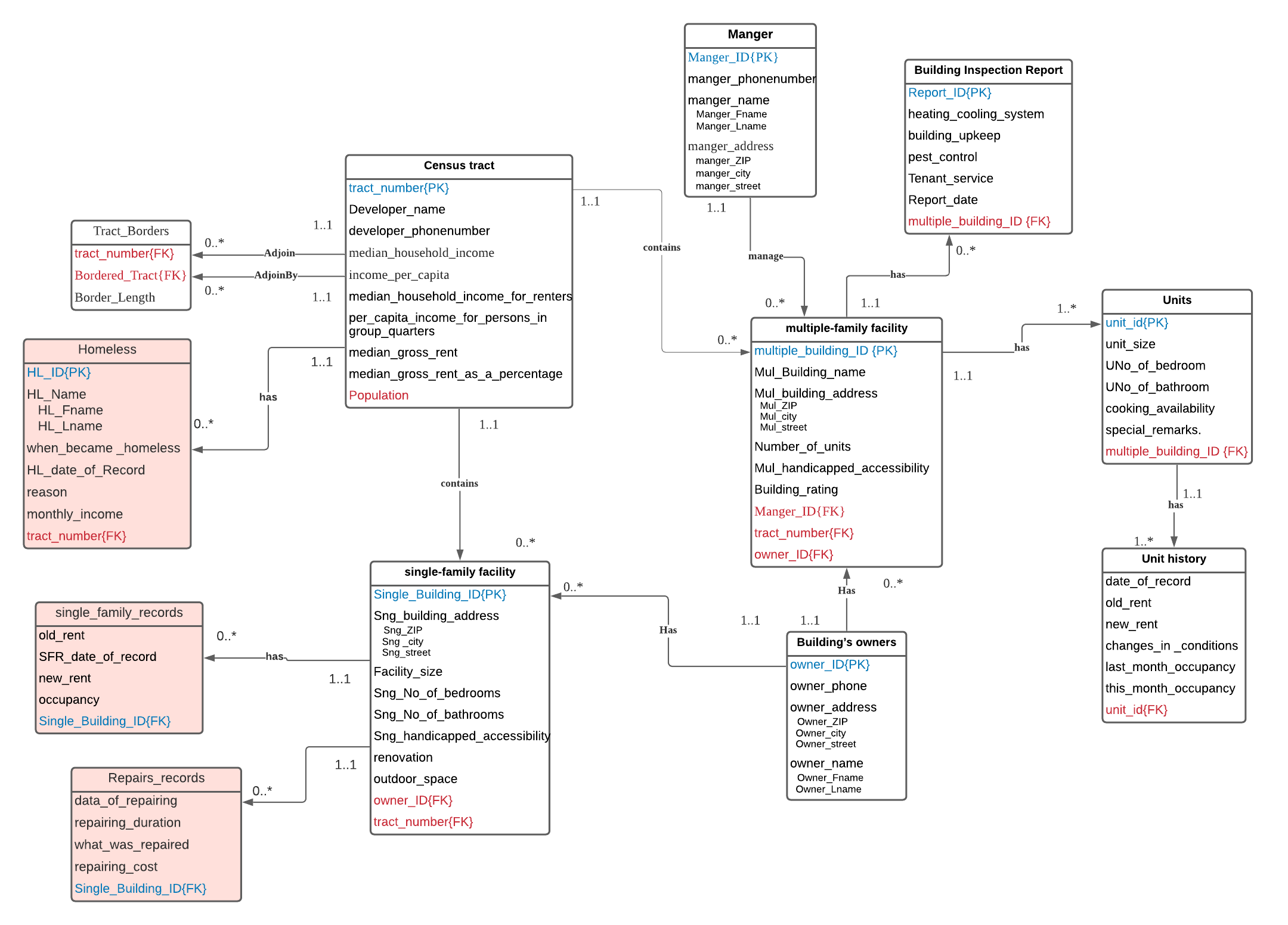
**Updating the Data:**

The employees of Homeless Coalition will have the ability update the data stored in the tables.

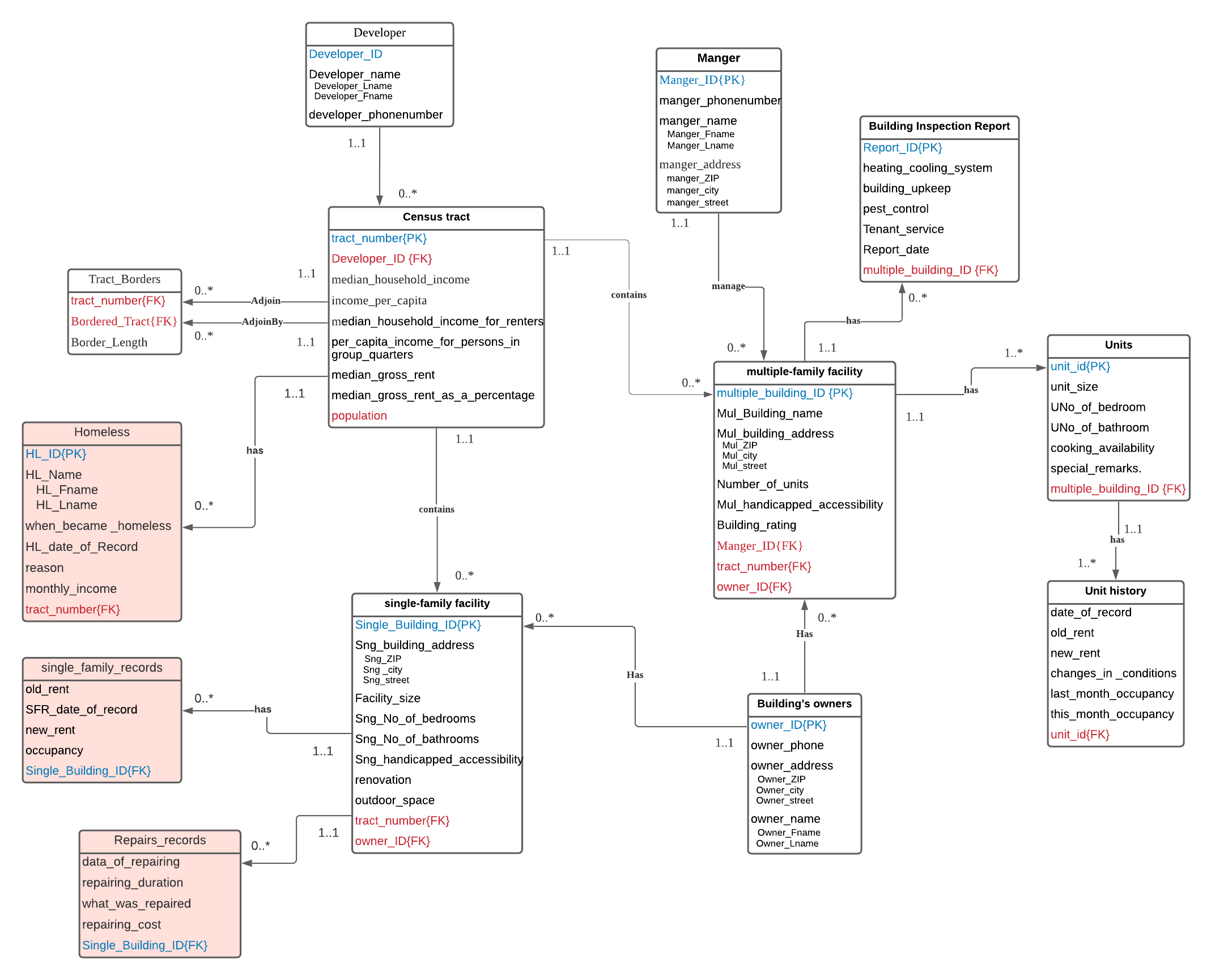
**Generating reports:**

The new database system will provide the ability to generate reports from the data stored within it, the reports can be generated from multiple tables and not limited to one table only.

**Part (B): Updated Logical ERD**



**Updated Logical ERD After Normalization:**



**Normalized Relational database schemas:**

**unit**(unit\_id, unit\_size, UNo\_of\_bedroom, UNo\_of\_bathroom, cooking\_availability, special\_remarks, multiple\_building\_ID).

**FK: multiple\_building\_ID, reference to multiple\_building\_ID on multipleFamilyFacility**

**unitHistory**(date\_of\_record, old\_rent, new\_rent, changes\_in \_conditions, last\_month\_occupancy, this\_month\_occupancy, unit\_id).

**FK: unit\_id, reference to unit\_id io unit**

**BuildingInspectionReport**(Report\_ID, heating\_cooling\_system, building\_upkeep, pest\_control, Tenant\_service, Report\_date, multiple\_building\_ID)

**FK: multiple\_building\_ID reference to multiple\_building\_ID on multipleFamilyFacility**

**Manger**(Manger\_ID, manger\_phonenumber, Manger\_Fname, Manger\_Lname,manger\_ZIP, manger\_city,manger\_street)

**BuildingOwner**(owner\_ID, owner\_phone, Owner\_ZIP, Owner\_city, Owner\_street, Owner\_Fname, Owner\_Lname)

**multipleFamilyFacility**(multiple\_building\_ID,Mul\_Building\_name, Mul\_ZIP, Mul\_city, Mul\_street, Number\_of\_units, Mul\_handicapped\_accessibility, Building\_rating, Manger\_ID, tract\_number, owner\_ID)

**FK:Manger\_ID, reference to Manger\_ID on Manger.**

**FK:tract\_number, reference to tract\_number on CensusTract.**

**FK:owner\_ID reference to owner\_ID on BuildingOwner.**

**singleFamilyFacility**(Single\_Building\_ID, Sng\_ZIP, Sng \_city, Sng\_street, Facility\_size, Sng\_No\_of\_bedrooms, Sng\_No\_of\_bathrooms, Sng\_handicapped\_accessibility, renovation,, outdoor\_space,owner\_ID, tract\_number)

**FK: owner\_ID, reference to owner\_ID on BuildingOwner.**

**FK:tract\_number, reference to tract\_number on CensusTract.**

**Developer**(Developer\_ID, Developer\_Fname, Developer\_Lname, developer\_phonenumber)

**CensusTract**(tract\_number, Developer\_ID, median\_household\_income, income\_per\_capita, median\_household\_income\_for\_renters, per\_capita\_income\_for\_persons\_in group\_quarters, median\_gross\_rent, median\_gross\_rent\_as\_a\_percentage, population)

**FK:Developer\_ID, reference to Developer\_ID on Developer.**

**Tract\_Borders**(tract\_number, Bordered\_Tract, Border\_Length)

**FK:tract\_number, reference to tract\_number on CensusTract.**

**FK:Bordered\_Tract,reference to tract\_number on CensusTract.**

**The new updates:**

**Homeless**(HL\_ID, HL\_Fname, HL\_Lname, when\_became \_homeless, HL\_date\_of\_Record, reason, monthly\_income,tract\_number)

**FK:tract\_number, reference to tract\_number on Censustract.**

**single\_family\_records**(old\_rent, SFR\_date\_of\_record, new\_rent, occupancy, Single\_Building\_ID)

**FK:Single\_Building\_ID, reference to Single\_Building\_ID on single-family facility.**

**Repairs\_records**(data\_of\_repairing, repairing\_duration, what\_was\_repaired, repairing\_cost, Single\_Building\_ID)

**FK:Single\_Building\_ID, reference to Single\_Building\_ID on single-family facility.**

**Updated Data Dictionary for Normalized Logical Design**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entity Name** | **Attributes** | **Description** | | **Data type & length** |
| **Manger** | **Manger\_name**  Manger\_Fname  Manger\_Lname | First name of the manger  Last name of the manger | | 15 variable characters  15 variable characters |
| **manger\_phone\_number** | Manger’s phone number | | 9 variable numbers |
| **Manger\_ID** | Uniquely identifier for the mangers | | 5 variable numbers |
| **manger\_address**  manger\_ZIP  manger\_city  manger\_street | The ZIP codes  The city name.  The street name. | | 3 variable numbers  10 variable characters  20 variable characters |
| **Building Inspection Report** | **Report\_ID** | Uniquely identifier for the reports | | 6 variable numbers |
| **Tenant\_service** | the availability of Tenant Services | | 1 character (Y – N) |
| **Heating\_cooling system** | the availability H/C system | | 1 character (Y – N) |
| **building upkeep** | Any maintenance since last report has been done | | 1 character (Y – N) |
| **pest control** | the availability of pest control services | | 1 character (Y – N) |
| **Report\_date** | The date of the report | | Date |
| **Building’s owners** | **owner\_name**  Owner\_Fname  Owner\_Lname | First name of the Owner  Last name of the Owner | | 15 variable characters  15 variable characters |
| **owner\_phone** | Owner’s phone number | | 9 variable numbers |
| **owner\_address**  Owner\_ZIP  Owner\_city  Owner\_street | The ZIP codes  The city name.  The street name. | | 3 variable numbers  10 variable characters  20 variable characters |
| **owner\_ID** | Uniquely identifier for the owners | | 5 variable numbers |
| **Units** | **Unit\_id** | Uniquely identifier for the units | | 10 variable numbers |
| **unit\_size** | The size of the unit in meter squared | | 4 variable numbers |
| **UNo\_of\_bedroom** | Number of bedrooms within the unit | | 2 variable numbers (0-99) |
| **UNo\_of\_bathroom** | Number of bathrooms within the unit | | 1 variable number (0-9) |
| **cooking\_availability** | the availability of cooking facilities | | 1 character (Y – N) |
| **special\_remarks** | Any special remarks | | 500 variable characters |
| **Unit history** | **date\_of\_record** | Will contain the date of each record, when it was recorded | | date |
| **Old\_rent** | The old rent of the unit | | 10 variable numbers |
| **New\_rent** | The new rent of the unit | | 10 variable numbers |
| **changes\_in\_conditions** | Any changes in the condition, new conditions, modified conditions | | 1000 variable characters |
| **Last\_month\_occupancy** | The unit occupancy for last month | | 1 character (O – V)  O: occupied  V: vacant |
| **This\_month\_occupancy** | The unit occupancy for current month | | 1 character (O – V)  O: occupied  V: vacant |
| **Single-family facility** | **Single\_Building\_ID** | Uniquely identifier for the single-family buildings | | 6 variable numbers |
| **Sng\_building\_address**  Sng\_ZIP  Sng \_city  Sng\_street | The ZIP codes  The city name.  The street name. | | 3 variable numbers  10 variable characters  20 variable characters |
| **facility\_size** | The size of the single-family building in meter squared | | 4 variable numbers |
| **Sng\_No\_of\_bedrooms** | Number of bedrooms within the single-family building | | 2 variable numbers (0-99) |
| **Sng\_No\_of\_bathrooms** | Number of bathrooms within the single-family building | | 1 variable number (0-9) |
| **Sng\_handicapped\_accessibility** | supporting of Handicapped accessibility | | 1 character (Y – N) |
| **~~Sng\_rental\_fees~~ (removed)** | ~~The rent fees for the single-family building~~ | | ~~5 variable numbers~~ |
| **renovation** | Has it been renovated | | 1 character (Y – N) |
| **~~Repairs~~ (removed)** | ~~Number of repairs times~~ | | ~~2 variable numbers~~ |
| **outdoor\_space** | Are there outdoor spaces (e.g. yard)? | | 1 character (Y – N) |
| **multiple-family facility** | **multiple\_building\_ID** | Uniquely identifier for the multiple-family buildings | | 6 variable numbers |
| **Mul\_Building\_name** | The name of the building | | 25 variable characters |
| **Mul\_building\_address**  Mul\_ZIP  Mul\_city  Mul\_street | The ZIP codes  The city name.  The street name. | | 3 variable numbers  10 variable characters  20 variable characters |
| **Number\_of\_units** | Number of units within each building | | 3 variable numbers |
| **Mul\_handicapped\_accessibility** | supporting of Handicapped accessibility | | 1 character (Y – N) |
| **Building\_rating** | The rating of the building | | 1 variable number |
| **Census tract** | **tract\_number** | Uniquely identifier for each tract | | 2 variable numbers |
| **Developer\_name**  Developer\_Fname  Developer\_Lname | The developer first name and last name for this tract | | 15 variable characters  15 variable characters |
| **developer\_phonenumber** | The developer phone number | | 9 variable numbers |
| **geographical\_boundaries** | The geographical boundaries for each tract, bordered with other tracts | | 4 variable numbers |
| **median\_household\_income** | The median income for the households in the tract | | 8 variable numbers |
| **income\_per\_capita** | The income per capita for each tract | | 8 variable numbers |
| **median\_household\_income\_for\_renters** | The median income for the household’s renters in the tract | | 8 variable numbers |
| **per\_capita\_income\_for\_persons\_in\_group\_quarters** | The per capita income for each person in group quarters | | 8 variable numbers |
| **median\_gross\_rent** | Median gross rent in the tract | | 8 variable numbers |
| **median\_gross\_rent\_as\_a\_percentage** | Median gross rent as a percentage in the tract | | 4 variable numbers |
| **Population** | Will contain the population within each tract | | 10 variable numbers |
| **Homeless** | **HL\_ID** | Uniquely identifier for each homeless record | | 10 variable numbers |
| **HL\_Name**  **HL\_Fname**  **HL\_Lname** | Homeless first and last name | | 15 variable characters  15 variable characters |
| **when\_became \_homeless** | What was the date that he/she when became homeless | | Date |
| **HL\_date\_of\_Record** | The date when recording this information | | Date |
| **reason** | the reason for their homelessness | | 1000 variable characters |
| **monthly\_income** | How much they earned per month | | 10 10 variable numbers |
| **single\_family\_records** | **old\_rent** | The old rent of the facility | | 10 variable numbers |
| **SFR\_date\_of\_record** | Will contain the date of each record, when it was recorded | | Date |
| **new\_rent** | The new rent of the facility | | 10 variable numbers |
| **occupancy** | The unit occupancy for current month | | 1 character (O – V)  O: occupied  V: vacant |
| **Repairs\_records** | **data\_of\_repairing** | | When was the repairing | Date |
| **repairing\_duration** | | How many days did it take | 3 variable numbers |
| **what\_was\_repaired** | | What are the things that was repaired and fixed | 1000 variable characters |
| **repairing\_cost** | | How much did the repairing cost | 10 variable numbers |
| **Developer** | **Developer\_name**  Developer\_Fname  Developer\_Lname | | The developer first name and last name for this tract | 15 variable characters  15 variable characters |
| **Developer\_ID** | | Uniquely identifier for each developer | 5 variable numbers |
| **developer\_phonenumber** | | The developer phone number | 9 variable numbers |
| **Tract\_Borders** | **Border\_Length** | | The length or the borders in meters | 10 variables numbers |

**3- Summary SQL implementation:**

**Relational database schemas:**

**unit**(unit\_id, unit\_size, UNo\_of\_bedroom, UNo\_of\_bathroom, cooking\_availability, special\_remarks, multiple\_building\_ID).

**FK: multiple\_building\_ID, reference to multiple\_building\_ID on multipleFamilyFacility**

**unitHistory**(date\_of\_record, old\_rent, new\_rent, changes\_in \_conditions, last\_month\_occupancy, this\_month\_occupancy, unit\_id).

**FK: unit\_id, reference to unit\_id io unit**

**BuildingInspectionReport**(Report\_ID, heating\_cooling\_system, building\_upkeep, pest\_control, Tenant\_service, Report\_date, multiple\_building\_ID)

**FK: multiple\_building\_ID reference to multiple\_building\_ID on multipleFamilyFacility**

**Manger**(Manger\_ID, manger\_phonenumber, Manger\_Fname, Manger\_Lname,manger\_ZIP, manger\_city,manger\_street)

**BuildingOwner**(owner\_ID, owner\_phone, Owner\_ZIP, Owner\_city, Owner\_street, Owner\_Fname, Owner\_Lname)

**multipleFamilyFacility**(multiple\_building\_ID,Mul\_Building\_name, Mul\_ZIP, Mul\_city, Mul\_street, Number\_of\_units, Mul\_handicapped\_accessibility, Building\_rating, Manger\_ID, tract\_number, owner\_ID)

**FK:Manger\_ID, reference to Manger\_ID on Manger.**

**FK:tract\_number, reference to tract\_number on CensusTract.**

**FK:owner\_ID reference to owner\_ID on BuildingOwner.**

**singleFamilyFacility**(Single\_Building\_ID, Sng\_ZIP, Sng \_city, Sng\_street, Facility\_size, Sng\_No\_of\_bedrooms, Sng\_No\_of\_bathrooms, Sng\_handicapped\_accessibility, renovation, outdoor\_space,owner\_ID, tract\_number)

**FK: owner\_ID, reference to owner\_ID on BuildingOwner.**

**FK:tract\_number, reference to tract\_number on CensusTract.**

**Developer**(Developer\_ID, Developer\_Fname, Developer\_Lname, developer\_phonenumber)

**CensusTract**(tract\_number, Developer\_ID, median\_household\_income, income\_per\_capita, median\_household\_income\_for\_renters, per\_capita\_income\_for\_persons\_in group\_quarters, median\_gross\_rent, median\_gross\_rent\_as\_a\_percentage, population)

**FK:Developer\_ID, reference to Developer\_ID on Developer.**

**Tract\_Borders**(tract\_number, Bordered\_Tract, Border\_Length)

**FK:tract\_number, reference to tract\_number on CensusTract.**

**FK:Bordered\_Tract,reference to tract\_number on CensusTract.**

**The new updates:**

**Homeless**(HL\_ID, HL\_Fname, HL\_Lname, when\_became \_homeless, HL\_date\_of\_Record, reason, monthly\_income,tract\_number)

**FK:tract\_number, reference to tract\_number on Censustract.**

**single\_family\_records**(old\_rent, SFR\_date\_of\_record, new\_rent, occupancy, Single\_Building\_ID)

**FK:Single\_Building\_ID, reference to Single\_Building\_ID on single-family facility.**

**Repairs\_records**(data\_of\_repairing, repairing\_duration, what\_was\_repaired, repairing\_cost, Single\_Building\_ID)

**FK:Single\_Building\_ID, reference to Single\_Building\_ID on single-family facility**

**Set of queries:**

\*Please check the SQL files or the txt files that are uploaded with the report.

**/\*1-list all the homeless names along with the duration they spent it in the homeless order in ascending order based on the duration:\*/**

SELECT HL\_ID AS 'ID' , CONCAT\_WS(' ', HL\_Fname,HL\_Lname) AS 'Homeless\'s Name', DATEDIFF(HL\_date\_of\_Record ,when\_became\_homeless) AS 'Duration','Days'

FROM homeless

ORDER BY DATEDIFF(HL\_date\_of\_Record ,when\_became\_homeless) ASC;

**/\*2-show me the total number of homeless people within each tract:\*/**

SELECT 'There are ', Count(HL\_ID) AS HomelessNumber,' homeless people in tract :',censustract.tract\_number AS 'Tract Number'

FROM censustract

RIGHT JOIN homeless ON homeless.tract\_number= censustract.tract\_number

GROUP BY homeless.tract\_number, censustract.tract\_number;

**/\*3-list me all the single-family building latest rentals along with the homeless who his/her monthly income higher than the latest rental and then show the difference if he/she rented the building:\*/**

SELECT CONCAT\_WS(' ', HL\_Fname,HL\_Lname) AS 'Homeless\'s Name',monthly\_income AS 'Monthly Income',new\_rent AS 'Latest Rental ',SFR\_date\_of\_record AS 'Date of Rental Record','How much Left if he/she rented :', (monthly\_income - new\_rent) AS 'Left'

FROM ((censustract RIGHT JOIN Homeless ON homeless.tract\_number= censustract.tract\_number)

INNER JOIN singlefamilyfacility ON censustract.tract\_number = singlefamilyfacility.tract\_number )

INNER JOIN single\_family\_records ON singlefamilyfacility.Single\_Building\_ID = single\_family\_records.Single\_Building\_ID

WHERE monthly\_income>= new\_rent;

**/\*4-compare between how many times the single-family buildings were occupied and when they were vacant :\*/**

SELECT DISTINCT (SELECT Count(\*) FROM single\_family\_records WHERE occupancy ='O') AS Occupied , (SELECT Count(\*) FROM single\_family\_records WHERE occupancy ='V') AS Vacant, Count(\*) AS 'Total Number of Records'

FROM single\_family\_records;

**/\*5-Display only the buildings that have been repaired and in the same time show if there is a changed happened to the rental as well as show what has been repaired:\*/**

SELECT singlefamilyfacility.Single\_Building\_ID AS 'Building ID', old\_rent AS 'OLD Rental',new\_rent AS 'New Rental',(new\_rent -old\_rent) AS 'Difference in Rental',SFR\_date\_of\_record AS 'Date of rental record', data\_of\_repairing AS 'Date of repairing',repairing\_cost AS 'Repairing Cost',what\_was\_repaired AS Repaired

FROM (singlefamilyfacility INNER JOIN single\_family\_records ON singlefamilyfacility.Single\_Building\_ID = single\_family\_records.Single\_Building\_ID)

INNER JOIN repairs\_records ON singlefamilyfacility.Single\_Building\_ID = repairs\_records.Single\_Building\_ID

WHERE month(SFR\_date\_of\_record) = month(data\_of\_repairing) ;

**/\*6- show the percentage of homeless people to the population in their tract: \*/**

SELECT censustract.tract\_number AS 'Tract Number',Count(HL\_ID) AS 'Homeless Number', population,(Count(HL\_ID) / population) \*100 AS 'Homeless Percentage','%'

FROM censustract

RIGHT JOIN homeless ON homeless.tract\_number= censustract.tract\_number

GROUP BY homeless.tract\_number, censustract.tract\_number;

**/\*7-Display the highest repairing cost beside the highest single family building among all the years :\*/**

SELECT RE\_REC.Single\_Building\_ID AS 'Building ID',MAX(repairing\_cost) AS 'Highest Repairing Cost',what\_was\_repaired AS 'Repaired' ,single\_family\_records.Single\_Building\_ID AS 'Building ID', GREATEST(MAX(new\_rent),MAX(old\_rent)) AS 'Highest Rental'

FROM (singlefamilyfacility INNER JOIN single\_family\_records ON singlefamilyfacility.Single\_Building\_ID = single\_family\_records.Single\_Building\_ID)

INNER JOIN repairs\_records RE\_REC ON singlefamilyfacility.Single\_Building\_ID = RE\_REC.Single\_Building\_ID;

**/\*8-List all homeless with their monthly income\*/**

SELECT CONCAT\_WS(' ', HL\_Fname,HL\_Lname) AS 'Homeless\'s Name',monthly\_income AS 'Monthly Income'

FROM homeless;

**/\*9-Show the average homelessness duration for all homelessness\*/**

SELECT AVG(DATEDIFF(HL\_date\_of\_Record ,when\_became\_homeless)) AS 'Average Homelessness Duration For all Homelessness','Days'

FROM homeless;

**/\*10-List all the records based on the new rent from highest rent to lowest rent\*/**

SELECT Single\_Building\_ID AS 'Building ID',new\_rent AS 'Last Updated Rent', SFR\_date\_of\_record AS 'Date of Record'

FROM single\_family\_records

ORDER BY new\_rent DESC;

**/\*11-List only the repairing records that has been done on the wall.\*/**

SELECT Single\_Building\_ID, what\_was\_repaired

FROM repairs\_records

WHERE what\_was\_repaired LIKE '%wall%';

**Conclusion:**

After finishing the improvement on the final project, we can conclude that there is no doubts that when dealing with databases, we actually are not limited to the number of queries or the type of queries or the number of entities within our databases as long as these entities are located within in our system boundary and there is a relation existed between the new entity and the rest of the entities, the database can be improved more and more by adding new entities like adding the buildings renters, what they work, if they are single or married and if they have children all these entities can be add to our database to store more specific details about the targeted people.

Along with the final database project and with this Alternative Assessment, it’s now obvious that there is no doubt about the importance of the databases and how they are located on the bottom of the developing process where later we will retrieve the data stored within the tables by the programming (Back end) and then process it and view them to the user, among a lot of queries we have done in this Alternative Assessment, I can basically categorize into two main parts: simple queries and complicated queries.

For the complicated queries I requested the data from two tables, and in some queries I retrieved the data from three tables by combining different JOIN types like : LEFT JOIN ,RIGHT JOIN , INNER JOIN, combining them together to create queries for complex requests was the difficult part since I had to relate it with the set theory and imaging the tables as they are in Venn Diagram all this thing in order to make sure to get the correct and expected results, I also made these complicated queries more by using aggregate function within our queries.

**Reflection:**

Database subject was little bit challenging at the beginning of the course, especially after we knew that the final project is combined with the system analysis and design course, the first time we got our case study it was little bit scary especially after seeing a lot of pages that we had to analyze them, our case study was about coalition that work and clearinghouse information, where we had to solve a current issue that the coalition is facing now, at the beginning it was confusing and I understood the case study wrongly, but after reviewing it more than one it became more clearer and more understandable.

I was lucky enough to work with such a great team who was well organized, my team was consist only of three member, Rafah, Al-Muhannad and I, we managed to organize and distribute the work equally among us where we always had an online meeting to discuss about next steps in the project. No doubt these meeting helped us a lot in completing the project since unfortunately duo to the current situation it was difficult for us to meet face to face. We generally distributed the works based on our understanding and knowledge, sometimes my mates don’t know how to do a specific part so the one who knows who to do it correctly, he will be responsible on it, by this way we were able to complete the entire project. I can’t deny that the long meeting discussions has affected my view about the project, and how to do my tasks.

Before I took this subject, I had prior experience in dealing with SQL language, but unfortunately my focus was only on creating queries, but how the databases is originally created I wasn’t fully understand this, but while the weeks flies and while we are learning new concepts and how to design a database from scratch all what we have learned are just theoretical things from what I can view, but when I started working on the project phases I saw how these theoretical things can be applied on the case study, Starting from the most challenging part that will decided how the entire database will become which is figuring out the correct boundaries for our database and what are the major user views, among all what we been through this part was the most difficult for me, because upon this part the rest will be either correct or the opposite, the second challenging part was creating the ERD but it was very interesting for me despite the importance and how the relationships between the tables should be correct so it can easy for us to retrieve the data, I had to redesign the ERD twice until I made sure that’s all the relationships are correct.

All the parts that I just talked about it was what I most lacking, which is designing and planning the databases, since I had zero experience in that, but when we reached implementation parts from creation the tables by using DDL and inserting the data, this part was much easier although it took from us more time, but it wasn’t that challenging.

After we went through all these challenging, I found out that a good planning and understanding for the case study will help you a lot to overcome these challenges, not mentioning also the good choosing of the tools that will assist you in the designing process providing a variety of options to manage the database planning. It’s important to know that when you start to overcome these issues when dealing with the system boundaries it’s important to split the system boundaries into three main things who will interact with it? is it apart from a larger system? and finally defining the things that are matched directly to the system that you are developing after we did this, we were able to find out the correct system boundaries and start with rest of the project.

Another important thing that defining and finding the correct entities and the accurate relationships between them will spare a lot of your time especially when it comes to the normalization part, since we were able to find the correct entities, we honestly did not suffer a lot in the normalization part neither in the implementation part.

By the end of this long journey no doubt that’s I learned a lot of things related to the databases planning and the importance of databases and how much they are important when it comes to back end developing since we have to deal directly with the databases and retriever and store the data from and to the database, no doubt what we been through had built for us at least what we can call a solid understand of the databases and how they are created, after reaching this point I can say that the database design is not such an easy job, is step by step where each step is based on the one before it, building a solid and strong foundations for the database will help me a lot later on when dealing we back end programming.