

**Database**

**Section (3)**

**Assignment Title**

Voting System Database Designing and Development

**Submitted to**

Dr. Ashraf Al-Smadi

**Submitted by**

Raghad Jaber 23110155

1. Database Requirement
2. Scenario

NASA & ROSCOSMOS (The American and Russian Space Agencies) are collaborating for a mission called The Dragon. 7 Astronauts are chosen by an internal election by all staff in both space agencies. Each elected **Astronaut** has a unique ID, Name, multiple Phone Numbers, Degree, Age, and Gender the astronaut can belong to many **Previous** **Missions** that has a unique ID, Name, Date, and Destination but can’t have multiple astronauts from the same previous mission chosen. Any previous missions of any astronaut can be associated with any **Agency** that has a unique ID, Name, Address, Capacity, one Phone Number. Moreover, Astronauts can only work in one of the agencies but can have multiple astronauts from the same agency. Note that each **Voter** that has a unique ID, Name, proficiency, Phone Number, Email, and Age must be a member in one of the agencies and all agencies can have multiple Voters. All voters Can vote for any astronauts from any agency.

1. Data Requirements

* This program was created to take care of the internal voting for choosing 7 Astronauts to go on a mission. Each astronaut has AstroID, AstroName, PhoneNumbers, Age, Gender and Degree.
* Each astronaut can be in many previous missions but can’t have multiple astronauts from the same previous mission chosen. Each previous mission has MissionID, MissionName, MissionDate, and Destination.
* An astronaut can be chosen by all voters from any agency. An agency has AgencyID, AgencyName, Address, Capacity, and PhoneNumber.
* Astronauts can belong to any agency.
* Note that each **Voter** that has a unique ID, Name, proficiency, Phone Number, Email, and Age must be a member in one of the agencies and all agencies can have multiple Voters.

1. User and System Requirements
   1. CEO (Agency Administrator) of the Agencies:

The CEOs are privileged to:

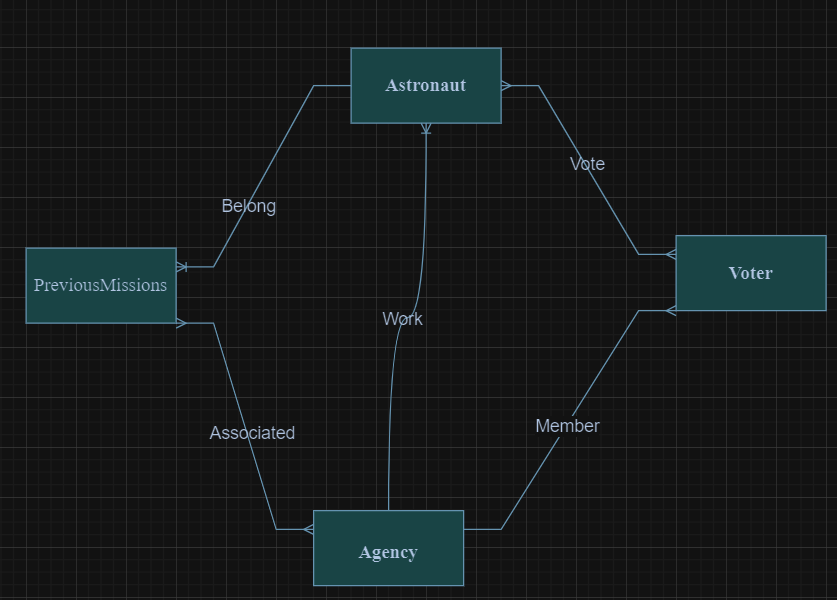
* + 1. Access all data for all entities
    2. Can edit any data of any entity
    3. Can add a new row of any entity
    4. Can add new data to any row in any entity
    5. Can delete any data in any row in any entity
  1. Voters

Voters are privileged to:

1. View their data
2. Vote for astronauts
3. edit their data
4. Add new data about themselves
   1. Astronauts

Astronauts are privileged to:

1. View their data
2. Know how many votes they gained
3. edit their data
4. view their previous mission’s data
5. Add new data about themselves
6. Database Design
7. Conceptual Design



A conceptual design is the process of identifying the essential data elements, relationships, and constraints in a data model. The conceptual design stage is the first step in the designing process, which precedes the other steps which are the logical and physical design. The objective of building the conceptual model is to Identify the entities and their attributes, Define the relationships, and establish the constraints. In my design I have 4 entities which are Astronaut, Previous Missions, Agency, and Voter. And had a brief explanation of the relationships of these entities for instance, in an agency multiple astronauts can work in it. Moreover, any previous mission can be associated with any agency etc.

1. Schema and Mapping

Entities:

1. **Astronaut** (AstroID, AstroName, PhoneNumbers, Degree, Age, Gender)
2. **PreviousMissions** (MissionID, MissionName, MissionDate, Destination)
3. **Agency** (AgencyID, AgencyName, Address, Capacity, PhoneNumber)
4. **Voter** (VoterID, VoterName, Proficiency, PhoneNumber, Age, Gender, Email)

Relationship:

1. **Astronaut** & **PreviousMissions** (1:M) (Belong)
2. **PreviousMissions & Agency** (M: N) (Associated)
3. **Agency** & **Voter** (1:M) (Member)
4. **Voter** & **Astronaut** (M: N) (Vote)
5. **Agency & Astronaut** (1:M) (Work)

**Mapping:**

1. Derived

Not Available in this design.

1. Multi-value
   1. **Astronaut** (AstroID, AstroName, Degree, Age, Gender)
   2. **Astronaut**\_**PhoneNumber (**AstroID, PhoneNumbers**)**
   3. **PreviousMissions** (MissionID, MissionName, MissionDate, Destination)
   4. **Agency** (AgencyID, AgencyName, Address, Capacity, PhoneNumber)
   5. **Voter** (VoterID, VoterName, Proficiency, PhoneNumber, Email, Age, Gender)
2. Composite
   1. **Astronaut** (AstroID, AstroFname, AstroLname, Degree, Age, Gender)
   2. **Astronaut**\_**PhoneNumber (**AstroID, PhoneNumber**)**
   3. **PreviousMissions** (MissionID, MissionName, MissionDate, Planet, Xcoordinate, Ycoordinate)
   4. **Agency** (AgencyID, AgencyName, Country, City, StreetName, BuildingNumber, Capacity, PhoneNumber)
   5. **Voter** (VoterID, VoterFname, VoterLname, Proficiency, PhoneNumber, Email, Age, Gender)
3. 1:1 Relation

Not Available in this design.

1. 1:M Relation
   1. **Astronaut** (AstroID, AstroFname, AstroLname, Degree, Age, Gender, AgencyID)
   2. **Astronaut**\_**PhoneNumber (**AstroID, PhoneNumber**)**
   3. **PreviousMissions** (MissionID, MissionName, MissionDate, Planet, Xcoordinate, Ycoordinate, AstroID)
   4. **Agency** (AgencyID, AgencyName, Country, City, StreetName, BuildingNumber, Capacity, PhoneNumber)
   5. **Voter** (VoterID, VoterFname, VoterLname, Proficiency, PhoneNumber, Email, Age, Gender, AgencyID)
2. M: N Relation
   1. **Astronaut** (AstroID, AstroFname, AstroLname, Degree, Age, Gender, AgencyID)
   2. **Astronaut**\_**PhoneNumber (**AstroID, PhoneNumber**)**
   3. **PreviousMissions** (MissionID, MissionName, MissionDate, Planet, Xcoordinate, Ycoordinate, AstroID)
   4. **Agency** (AgencyID, AgencyName, Country, City, StreetName, BuildingNumber, Capacity, PhoneNumber)
   5. **Voter** (VoterID, VoterFname, VoterLname, Proficiency, AgencyID, PhoneNumber, Email, Age, Gender)
   6. **AgencyPreMissions (**AgencyID, MissionID, MissionDescription**)**
   7. **Selected** (VoterID, AstroID)

*Finalized Entities:*

1. **Astronaut** (AstroID, AstroFname, AstroLname, Degree, AgencyID, Age, Gender)
2. **Astronaut**\_**PhoneNumber (**AstroID, PhoneNumber**)**
3. **PreviousMissions** (MissionID, MissionName, MissionDate, Planet, Xcoordinate, Ycoordinate, AstroID)
4. **Agency** (AgencyID, AgencyName, Country, City, StreetName, BuildingNumber, Capacity, PhoneNumber)
5. **Voter** (VoterID, VoterFname, VoterLname, Proficiency, PhoneNumber, Email, Age, Gender, AgencyID)
6. **AgencyPreMissions (**AgencyID, MissionID, MissionDescription**)**
7. **Selected** (VoterID, AstroID)

**Weak & Strong Entities:**

1. **Astronaut** (AstroID, AstroFname, AstroLname, Degree, Age, Gender, AgencyID) [Strong]
2. **Astronaut**\_**PhoneNumber (**AstroID, PhoneNumber**)** [Weak]
3. **PreviousMissions** (MissionID, MissionName, MissionDate, Planet, Xcoordinate, Ycoordinate, AstroID) [Strong]
4. **Agency** (AgencyID, AgencyName, Country, City, StreetName, BuildingNumber, Capacity, PhoneNumber) [Strong]
5. **Voter** (VoterID, VoterFname, VoterLname, Gender, Proficiency, Email, PhoneNumber, Age, AgencyID) [Strong]
6. **AgencyPreMissions (**AgencyID, MissionID, MissionDescription**)** **[Weak]**
7. **Selected** (VoterID, AstroID) [Weak]
8. Normalization
9. 1st NF

|  |  |  |  |
| --- | --- | --- | --- |
| **Relations** | **Attributes** | **Violation Description** | **Solution – Relations** |
| The relations schema | The attribute name | Describe why it is not in the 1st NF (the violation) | Show the schema for each affected relation. |
| **Astronaut** (AstroID,  PhoneNumbers,  AstroName, Degree, Age, Gender) | AstroName | Multi value attributes are attributes that may have multiple data so here in this case we can’t tell whether the user has 1 or more numbers this will leads to creating more columns to add more numbers for some of the astronauts and that’s not efficient for the memory and cost. due to this creating a new entity for those minority is the best suited solution for this case. | **Astronaut** (AstroID, AstroFname, AstroLname, Degree, Age, Gender)  **Astronaut**\_**PhoneNumber (**AstroID, PhoneNumber) |
| **voter** (VoterID, VoterName, proficiency, PhoneNumber, Age, Gender) |  | There is no primary key that uniquely identifies the entity. And this will affect the process of extracting data, editing and updating it. | **voter** (VoterID, VoterName, proficiency, PhoneNumber, Age, Gender) |

1. 2nd NF

|  |  |  |  |
| --- | --- | --- | --- |
| **Relations** | **FDs** | **Violation Description** | **Solution – Relations** |
| The relations schema | Show the functional dependencies causing the violation | Describe why it is not in the 2nd NF (the violation) | Show the schema for each affected relation. |
| **Voter Astronaut (**AstroID,  VoterID,  AstroFName, AstroLName, AstroPhoneNumbers,  AstroDegree,  AstroAge, AstroGender,  VoterFName, VoterLName, Voterproficiency, VoterPhoneNumber, VoterAge,  VoterEmail,  Gender**)** | **Partial FD1:**  AstroID  AstroFName, AstroLName, AstroPhoneNumbers,  AstroDegree,  AstroAge, AstroGender  **Partial FD2:**  VoterID  VoterFName, VoterLName, Voterproficiency, VoterPhoneNumber,  VoterEmail,  VoterAge | In this case we have partial dependencies since those non-prime attributes (AstroName, AstroPhoneNumbers,  AstroDegree,  AstroAge, AstroGender)  Depend on the PK AstroID. And the other non-prime (VoterFName, VoterLName, Voterproficiency, VoterPhoneNumber, VoterEmail,  VoterAge) Depend on the PK VoterID. This will cause trouble when dealing with data whether by updating, deleting or extracting it. So, by breaking the table into more manageable pieces we will have a better environment for data. | **Astronaut** (AstroID  AstroFName, AstroLName,  Degree,  Age, Gender)  **Voter** (VoterID  VoterFName, VoterLName, Proficiency, PhoneNumber,  Email,  Age, Gender)  **Selected** (VoterID, AstroID) |
| **Agency** (AgencyID, MissionID, AgencyName, Country, City, StreetName, BuildingNumber, Capacity, PhoneNumber, MissionName, MissionDate, Planet, Xcoordinate, Ycoordinate) | **Partial FD1:**  AgencyID  AgencyName, Country, City, StreetName, BuildingNumber, Capacity, PhoneNumber  **Partial FD2:**  MissionID MissionName, MissionDate, Planet, Xcoordinate, Ycoordinate | In this case we have partial dependencies since those non-prime attributes (AgencyName, Country, City, StreetName, BuildingNumber, Capacity, PhoneNumber)  Depend on the PK AgencyID. And the other non-prime (MissionName, MissionDate, Planet, Xcoordinate, Ycoordinate) Depend on the PK MissionID. This will cause trouble when dealing with data whether by updating, deleting or extracting it. So, by breaking the table into more manageable pieces we will have a better environment for data. | **PreviousMissions** (MissionID, MissionName, MissionDate, Planet, Xcoordinate, Ycoordinate)  **Agency** (AgencyID, AgencyName, Country, City, StreetName, BuildingNumber, Capacity, PhoneNumber)  **AgencyPreMissions (**AgencyID, MissionID, MissionDescription**)** |

1. 3rd NF

|  |  |  |  |
| --- | --- | --- | --- |
| **Relations** | **FDs** | **Violation Description** | **Solution – Relations** |
| The relations schema | Show the functional dependencies causing the violation | Describe why it is not in the 3rd NF (the violation) | Show the schema for each affected relation. |
| Astronaut (AstroID, AstroFname, AstroLname,  Degree,  Age, Gender,  AgencyID  AgencyName,  Country, City, StreetName, BuildingNumber, Capacity, PhoneNumber) | **TD1:**  AstroID  AstroFname, AstroLname,  Degree,  Age, Gender, AgencyID  **TD2:**  AgencyID  AgencyName, Country, City, StreetName, BuildingNumber, Capacity, PhoneNumber  **TD3:**  AstroID  AgencyName, Country, City, StreetName, BuildingNumber, Capacity, PhoneNumber | The PK AstroID determines AstroFname, AstroLname,  Degree,  Age, Gender,  AgencyID AgencyName,  Country, City, StreetName, BuildingNumber, Capacity, PhoneNumber. And AgencyID also determines  AgencyName,  Country, City, StreetName, BuildingNumber, Capacity, PhoneNumber. in this case AgencyID is not a candidate key so no attributes should depend on it. Due to this, we have transitive dependency which makes the program harder in editing, updating, and extracting. | **Astronaut** (AstroID, AstroFname, AstroLname, Degree, Age, Gender, AgencyID)  **Agency** (AgencyID, AgencyName, Country, City, StreetName, BuildingNumber, Capacity, PhoneNumber) |
| PreviousMissions (MissionID, MissionName, MissionDate, Planet, Xcoordinate, Ycoordinate,  AstroID,  AstroFName, AstroLName, AstroPhoneNumbers,  AstroDegree,  AstroAge, AstroGender) | **TD1:**  MissionID  MissionName, MissionDate, Planet, Xcoordinate, Ycoordinate, AstroID  TD2:  AstroID  AstroFname, AstroLname,  Degree,  Age, Gender  TD3:  MissionID  AstroFname, AstroLname,  Degree,  Age, Gender | The PK MissionID determines MissionName, MissionDate, Planet, Xcoordinate, Ycoordinate, AstroID, AstroFName, AstroLName, AstroPhoneNumbers,  AstroDegree,  AstroAge, AstroGender. And AstroID also determines  AstroFName, AstroLName, AstroPhoneNumbers,  AstroDegree,  AstroAge, AstroGender. in this case AstroID is not a candidate key so no attributes should depend on it. Due to this, we have transitive dependency which makes the program harder in editing, updating, and extracting. | **PreviousMissions** (MissionID, MissionName, MissionDate, Planet, Xcoordinate, Ycoordinate, AstroID)  **Astronaut** (AstroID, AstroFname, AstroLname, Degree, Age, Gender) |

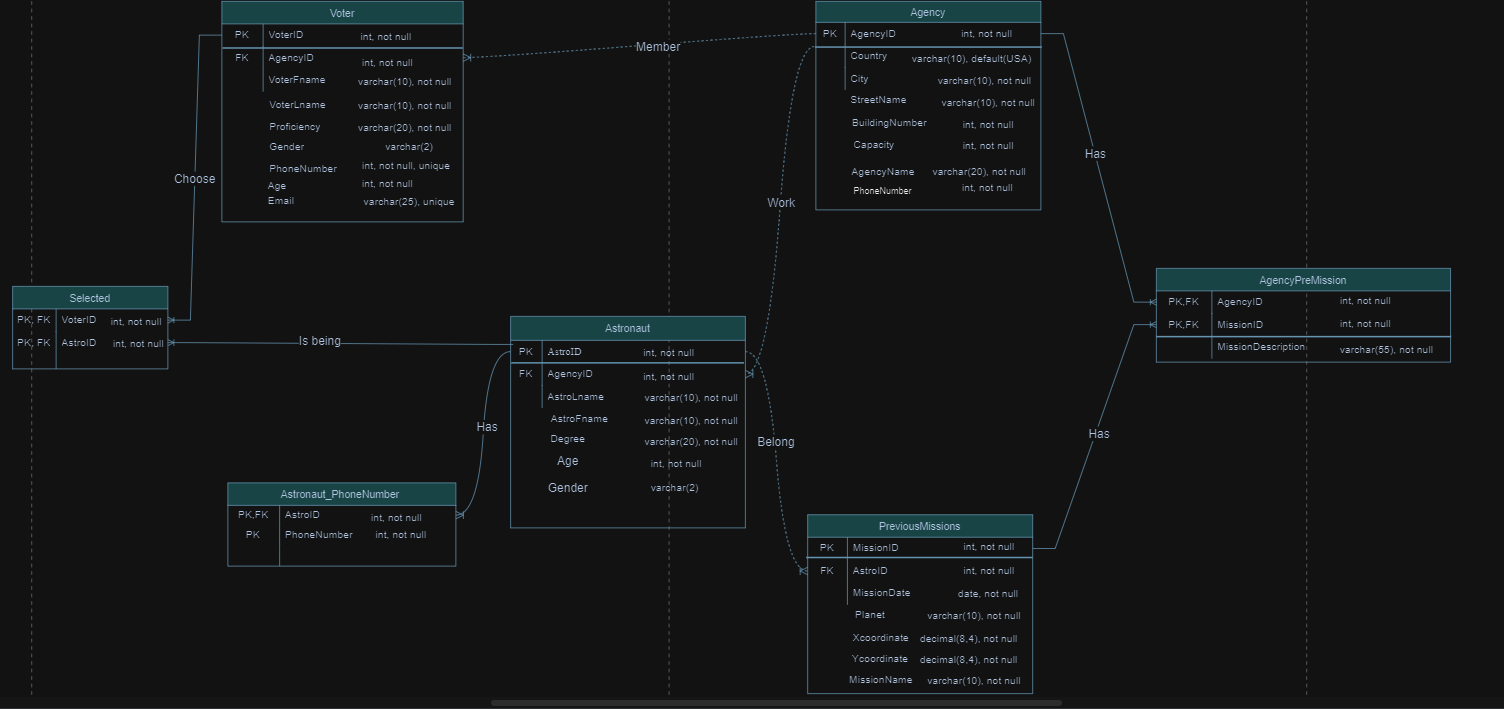
1. Logical Design



A logical data model establishes the structure of data elements and the relationships among them. It takes the elements of conceptual data modeling a step further by adding more information to them such as attributes. The components of this design are Entities, Relationships, and Attributes. The main objective of the logical design is to establish relational databases by converting entity-relationship diagrams into relational tables.

Describe and perform the data normalization process. In my logical design I represented all the entities with its attributes and the primary and foreign keys and their relations weather they are weak or strong relations.

1. Physical Design



A physical model is the structure of a database, defining how data is being stored and organized. It provides a detailed view of the database, specifying tables, columns, data types, relationships, and constraints. It is like a framework to guide developers in creating the actual database. The goal of creating a physical model is to produce a visual representation of its structure, Reduce incomplete implementations. in my physical design I illustrated all entities with their attributes, constraint and keys also relations are also illustrated with specific relations.

1. Effectiveness of the Design

To see how effective our database is we should see its reflection on our user and system requirements. Since the Astronauts and Voters are privileged to update/ add / display data so to have a secure and easy path to do that the database is normalized, and the attributes are correctly mapped. so, by the 1st NF the data are correctly divided. for instance, composite attributes are divided to multiple individual attributes (e.g. AstroName -> AstroFname, AstroLname) this will not only make the process of editing/ updating/ adding data easier for Astronauts, and voters but also eliminating redundancy and maintaining a unique representation of each attribute in the table. This reduces the risk of data inconsistencies. and, by the help of the 1st NF multivalued attributes are also separated to a new entity

(e.g. Astronaut\_PhoneNumber (AstroID, PhoneNumber)) this will help by saving space in storge since not all astronauts have multiple phone numbers so by this, we will have an extra place to add phone numbers for only the astronauts who have an extra one not adding a whole new column just for one new number for one astronaut. Also, by authorizing one attribute in each entity to be the PK, by this we will have an easy way to identify whatever entities we want without mentioning all attributes in the entities. In conclusion, the first Normal Form sets the essential foundation for a well-structured database. By ensuring atomic values, unique attribute names, and order independence it enhances data integrity, simplifies data management, and enables flexibility and scalability.

For the 2nd normal form by creating clean entities that the non-prime attributes depend on one PK so no Partial relations that make the tables big and helps with organizing your database for instance if we didn’t normalize the entity **Voter Astronaut** with the attributes AstroID, VoterID, AstroFName, AstroLName, AstroPhoneNumbers, AstroDegree, AstroAge, AstroGender, VoterFName, VoterLName, Voterproficiency, VoterPhoneNumber, VoterAge, Gender, and Email**.** And seperatetd the entity into two entities Voters and Astronauts and each entity has its attributes and creating a third entity resulted from the M: N relation this would make the database less efficient and harder for the users to deal with. Because why would the Astronaut need from the Voters data and vice versa. That would also make the database more secure since each user will enter the table that they need. Also, the third table Selected with the attributes Astro and Voter IDs would help in the voting process since the voter can access the table and vote for the Astronaut they want in a smooth way. So, the 2nd normal form made the data easier to store and retrieve data, which also corresponds to the user and system requirements.

3rd NF also played a huge role in making the database design efficient by dealing with transitive dependencies. so, if we did not normalize the transitive relation in this case for example PreviousMissions (MissionID, MissionName, MissionDate, Planet, Xcoordinate, Ycoordinate,

AstroID, AstroFName, AstroLName, AstroPhoneNumbers, AstroHeight, AstroDegree,

AstroAge, AstroGender) we will have trouble when dealing with data since each entity will contain big amount of data that aren’t related to each other so by separating the attributes to two different entities Astronaut and Previous Missions we will have more efficient way to deal with the data as what the users need editing/viewing/inserting and adding data so it will improve the overall data integrity and organize the information.

Also, all the Normalization Forms do indeed align with the CEO (Agency Administrator) requirements since they will be the user who will deal with the most data and tables because they will have full access to all tables. and by the normalization and mapping we will provide the most convenient design to make dealing with that much data easier and more approachable.

To sum up, having atomic attributes by mapping the multi-valued and composite attributes and defining a PK for each entity and getting rid of the transitive and partial dependencies by the 2nd and 3rd NFs did increase the overall efficiency and not only meets with the user and systems requirements but also exceed them. Since the design is capable of being used more in the future since it serves an easy environment for adding new data and attributes for further details and adding new users to the database.

Technical Documentation

1. Introduction

This document is to whom it may concern for developing the database and its overall tables and attributes to help with the process of the election of the Dragon mission. The documentation is to make it easy to save the database and to last the longest time possible and to update or edit whenever needed. We will go throw the physical schema, Database Development, security, Maintenance, Testing, and Evaluation of the Database and improvements.

1. A screenshot of a computer

   Description automatically generatedPhysical Schema
2. Database Development
3. Database Overview

|  |  |  |
| --- | --- | --- |
| **Table** | **Name** | **Description** |
|  | Astronaut | This Table is for storing the data of every individual astronaut such as Their ID, First Name, Last Name, Gender, Degree, Age, And the AgencyID that they work in. This table keeps the important information that may concern the voter or CEO or Astronaut themselves. |
|  | Agency | This Table is for storing the data of the Agencies which are the AgencyID, AgencyName, Country, City, StreetName, BuildingNumber, Capacity and one main Phone Number. This table is important for the Voters and Astronauts to reach out for the agency they want. By Address or Phone Number. |
|  | PreviousMissions | This Table is for storing the data of the Previous Missions which are the MissionID, MissionName, MissionDate, Planet, Xcoordinate, Ycoordinate, and AstroID for the astronauts that were in the mission. This table is important for storing the main important data for the Missions that they were successfully completed by any agency. |
|  | Voter | This Table is for storing the data of the Voters which are staff in any of the agencies. their data involves around their ID, First Name, Last Name, Proficiency, Gender, one main PhoneNumber, Age, And the AgencyID that they work in. This table is important to make sure that the Voters are staff in one of the agencies. And to have their main data they need to be able to vote. |

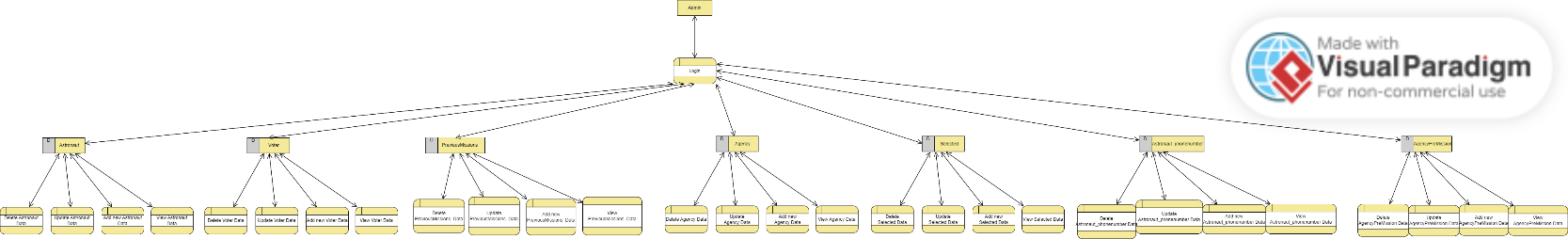
|  |  |  |
| --- | --- | --- |
| **View** | **Name** | **Description** |
|  | Astro\_experience | This view is made to display the Mission Name, Mission Description, Agency Name, Astro First Name, and Astro Last Name. This view will help the voters into choosing the right candidates. |
|  | Agency\_Mission\_Data | This view is made to display the Agency Name, Total Missions, and The Average of the Astronauts Ages who went to the Mission. This view will help the agencies in keeping track of their missions and accomplishments. By calculating the number of missions, they have done and the age average of their Astronauts. |

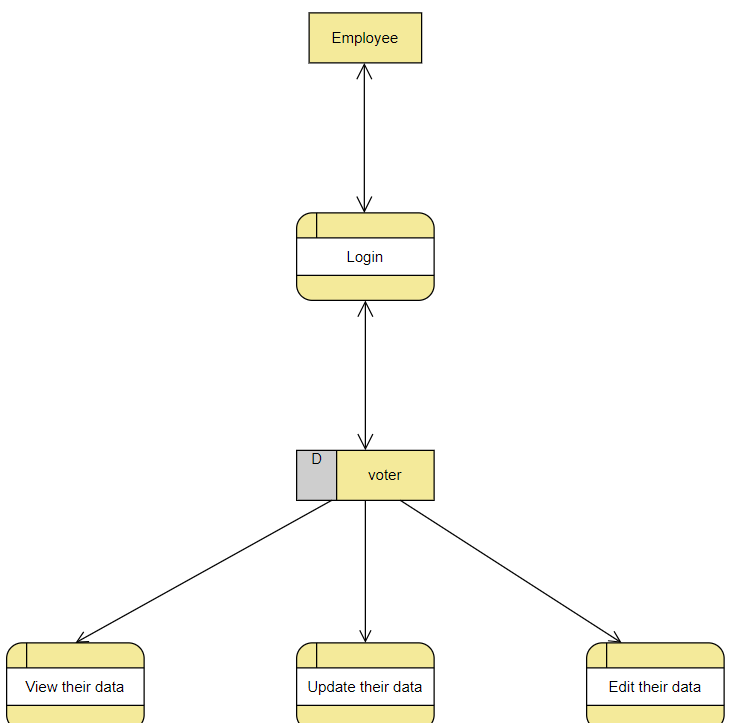
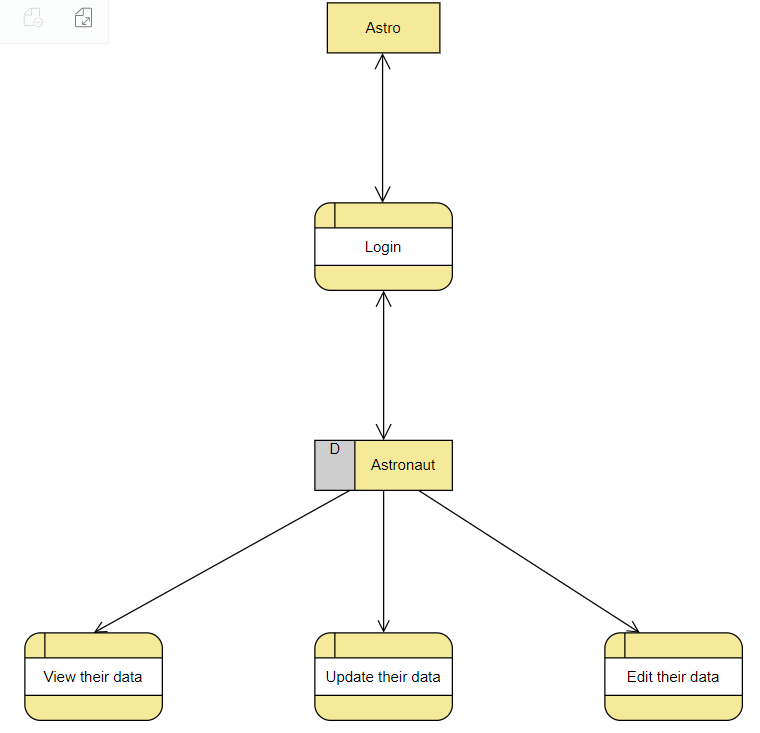
|  |  |  |
| --- | --- | --- |
| **Procedure** | **Name** | **Description** |
|  | HireAstronaut | This Stored Procedure is to help the Agencies CEO when hiring a new astronaut to the agency. it stores all their data such as their First Name, Last Name, Gender, Degree, Age, Agency ID that they in, and unique ID. |
|  | EditingMissionDescription | This Stored Procedure is to help the Agencies staff when editing and updating the Description of their Previous Missions. to make the process of adding or editing new data easier since space agencies are continually discovering and working on new data and discoveries. Therefore, having this procedure is a must. And to make it even more practical a new table is created to show the Agencies name and Mission name and the Description since dealing with IDs and numbers may be difficult. |

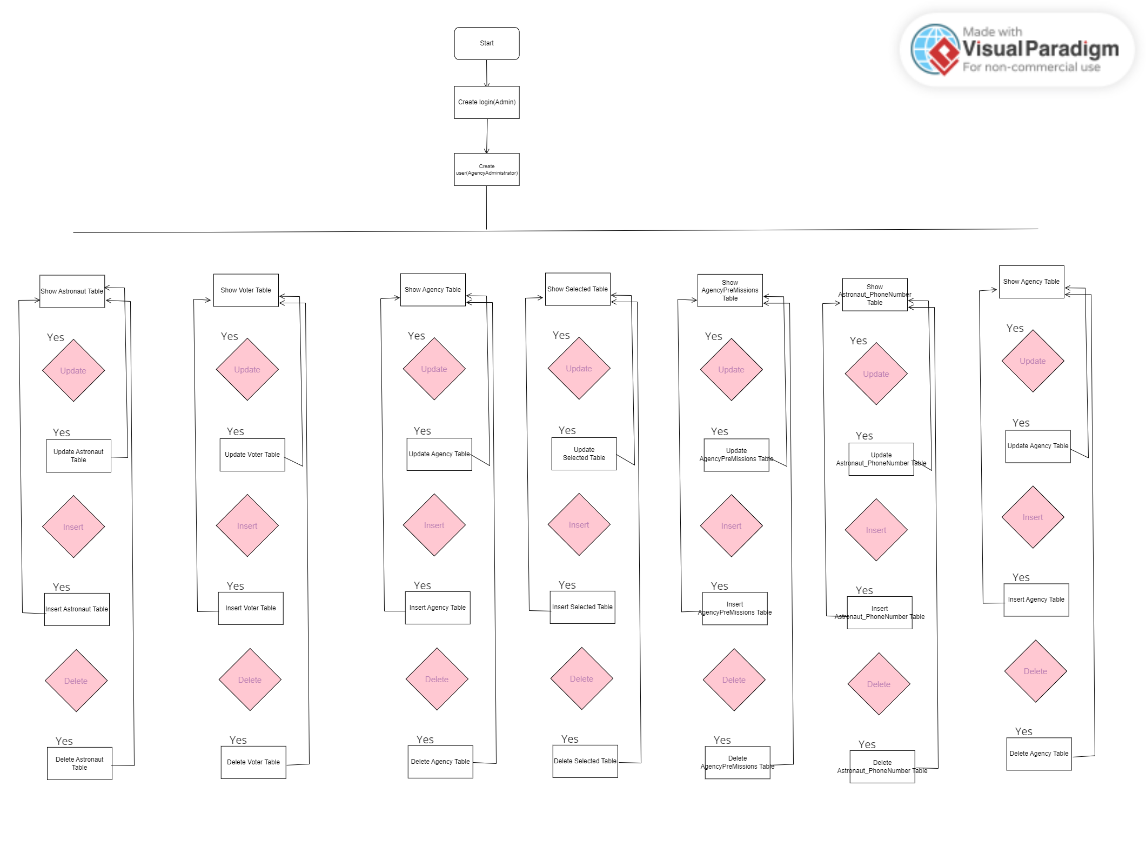
1. Security

|  |  |  |  |
| --- | --- | --- | --- |
| **User name** | **Privilege Command** | **Description** | **Screenshot** |
| **AgencyAdministrator** | GRANT ALL ON PreviousMissions TO AgencyAdministrator; | This command allows the AgencyAdministrator to view, insert, update, and delete the data in the PreviousMissions entity |  |
| GRANT ALL ON Voter TO AgencyAdministrator; | This command allows the AgencyAdministrator to view, insert, update, and delete the data in the Voter entity |  |
| GRANT ALL ON Agency TO AgencyAdministrator; | This command allows the AgencyAdministrator to view, insert, update, and delete the data in the Agency entity |  |
| GRANT ALL ON AgencyPreMissions TO AgencyAdministrator; | This command allows the AgencyAdministrator to view, insert, update, and delete the data in the AgencyPreMissions entity |  |
| GRANT ALL ON Astronaut\_PhoneNumber TO AgencyAdministrator; | This command allows the AgencyAdministrator to view, insert, update, and delete the data in the Astronaut\_PhoneNumber entity |  |
| GRANT ALL ON Selected TO AgencyAdministrator; | This command allows the AgencyAdministrator to view, insert, update, and delete the data in the Selected entity |  |
| GRANT ALL ON Astronaut TO AgencyAdministrator; | This command allows the AgencyAdministrator to view, insert, update, and delete the data in the Astronaut entity |  |
| GRANT ALL ON Astro\_experience TO AgencyAdministrator; | This command allows the AgencyAdministrator to view, insert, update, and delete the data in the Astro\_experience view |  |
| GRANT ALL ON Agency\_Mission\_Data TO AgencyAdministrator; | This command allows the AgencyAdministrator to view, insert, update, and delete the data in the Agency\_Mission\_Data view |  |
| GRANT EXECUTE ON OBJECT::EditingMissionDescription TO AgencyAdministrator; | This command allows the AgencyAdministrator to excute the procedure EditingMissionDescription |  |
| GRANT EXECUTE ON OBJECT::HireAstronaut TO AgencyAdministrator; | This command allows the AgencyAdministrator to excute the procedure HireAstronaut |  |
| Employee | GRANT SELECT, INSERT, UPDATE ON Voter TO Employee; | This command allows the Employee to view, insert, update the data in the Voter entity |  |
| Astro | GRANT SELECT, INSERT, UPDATE ON Astronaut TO Astro; | This command allows the Astro to view, insert, update the data in the Astronaut entity |  |
| Scientist | GRANT ALL ON PreviousMissions TO Scientist; | This command allows the Scientist to view, insert, update, and delete the data in the PreviousMissions entity |  |
|  | GRANT ALL ON Agency\_Mission\_Data TO Scientist; | This command allows the Scientist to view, insert, update, and delete the data in the Agency\_Mission\_Data view |  |
|  | GRANT ALL ON AgencyPreMissions TO Scientist; | This command allows the Scientist to view, insert, update, and delete the data in the AgencyPreMissions entity |  |
|  | GRANT EXECUTE ON OBJECT::EditingMissionDescription TO Scientist; | This command allows the Scientist to excute the procedure EditingMissionDescription |  |
| PublicRelation | GRANT SELECT, INSERT, UPDATE ON Agency TO PublicRelation; | This command allows the PublicRelation to view, insert, update the data in the Agency entity |  |
|  | GRANT SELECT, INSERT, UPDATE ON Astronaut TO PublicRelation; | This command allows the PublicRelation to view, insert, update the data in the Astronaut entity |  |
|  | GRANT SELECT, INSERT, UPDATE ON Voter TO PublicRelation; | This command allows the PublicRelation to view, insert, update the data in the Voter entity |  |
|  | GRANT SELECT, INSERT, UPDATE ON Astro\_experience TO PublicRelation; | This command allows the PublicRelation to view, insert, update, and delete the data in the Astro\_experience entity view |  |

1. User Interface
2. Flowchart and Data Movement Diagrams

**DFD**:



**Activity Chart:**

1. Interfaces Development

|  |  |  |  |
| --- | --- | --- | --- |
| **Page ID** | **Title** | **Description** | **Screenshot** |
|  | Home Page | This page allows the user to see the tables that they are accessible to view. and to select the tables they wish to view. The user can select multiple tables at one time. |  |
|  | Astronaut | This page displays the tables attributes and the accessible actions that the user is privileged to do whether it’s viewing, updating, deleting, or adding. |  |
|  | Add record | This page allows the user to add a new row of data to the entity. To do so add your data to the boxes then press save to add the row. |  |
|  | Edit | This page displays the data for a specific row and allows the user to edit it. |  |
|  | delete | This page allows the user to delete a row with its data when you press delete. |  |

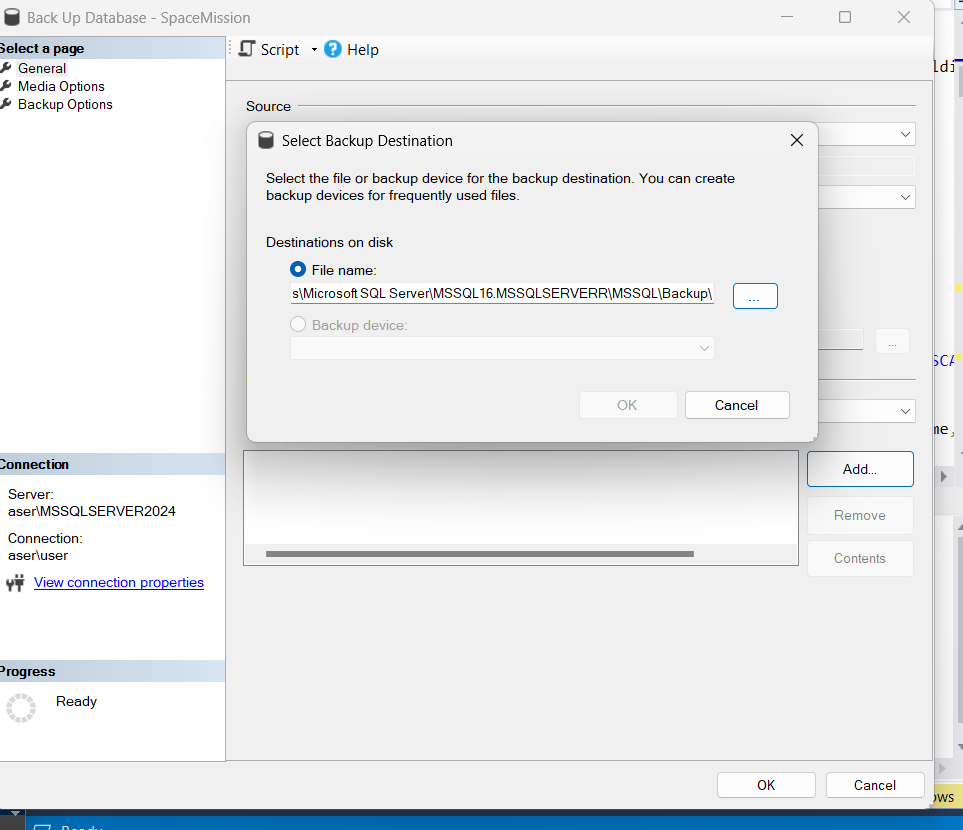
1. Maintenance
2. Database recovery & backups

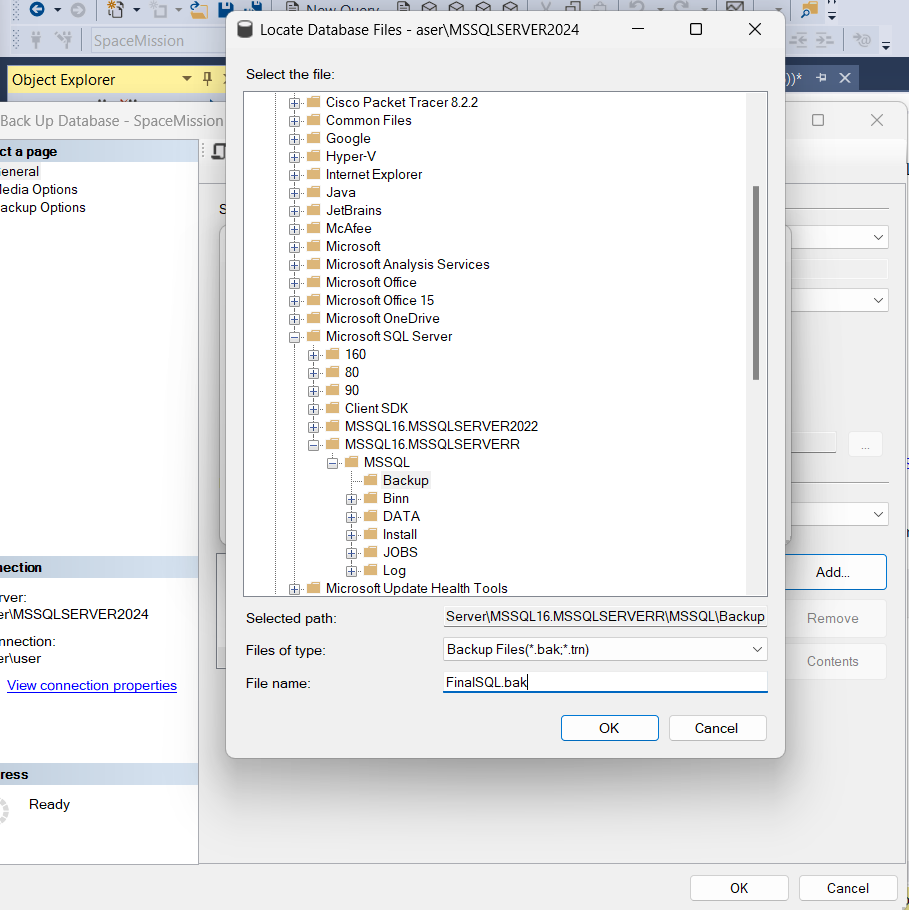
Recovery and Backups are important for the database to keep it safe from any inconvenience or damage. Because losing it would cause a huge problem in the business so here is how to protect it:

**Steps For Database Backup:**

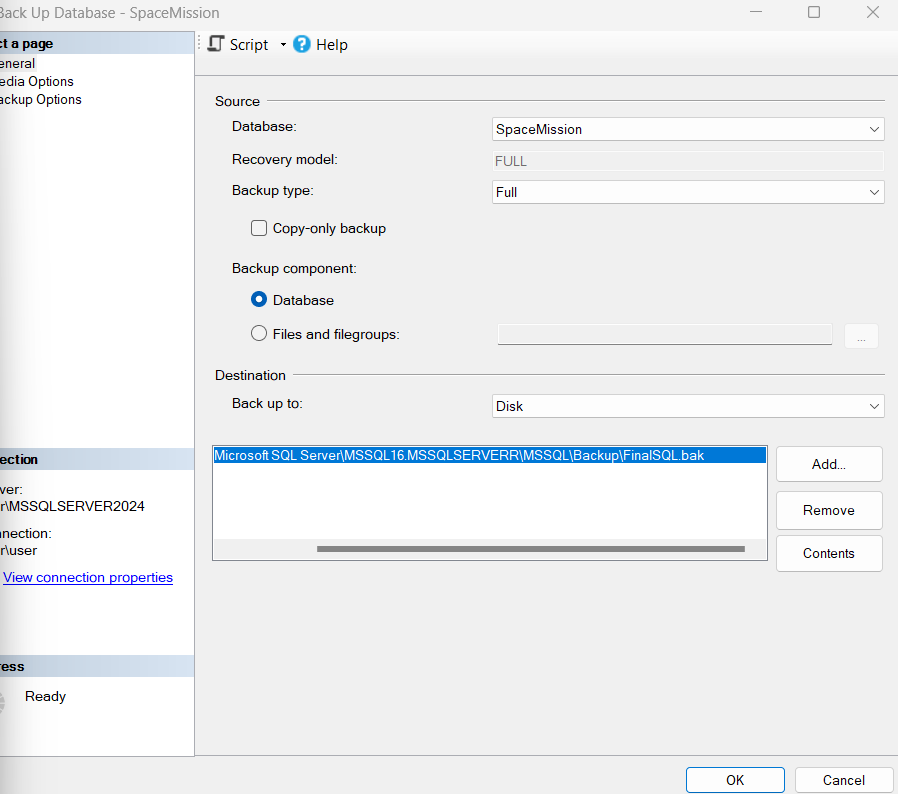


* 1. First click the right click on the database file then go to Tasks then click on backup.

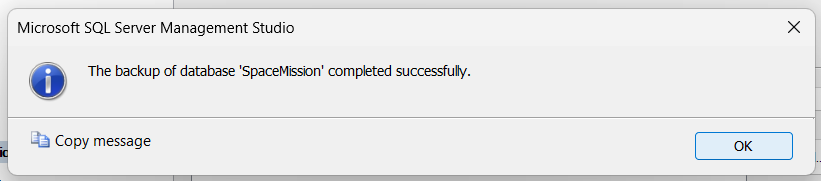
2.The Backup Database Page will appear. And we will have the file path with .bak destination we remove it (By clicking on Remove). Then click on add then the Select Backup Destination Page will appear. Then click on the […] button



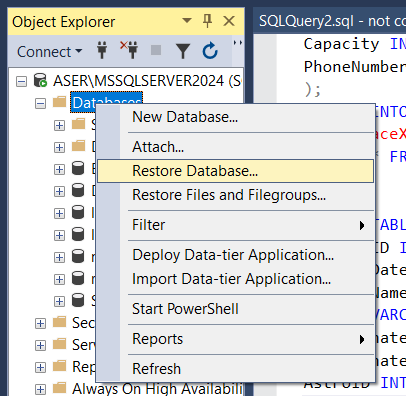
1. Then the locate Database Files will appear. chose the file that you want to store your backup database in (I chose the default windows backup file). after choosing a file chose a name for your file (I named it FinalSQL) with the file extension [.bak]. Then click OK.



1. Now you will be able to see the new file path where the backup file is going to get saved. finally click OK.



**Steps For Database Recovery:**



* + 1. click right click on the Database file

A screenshot of a computer

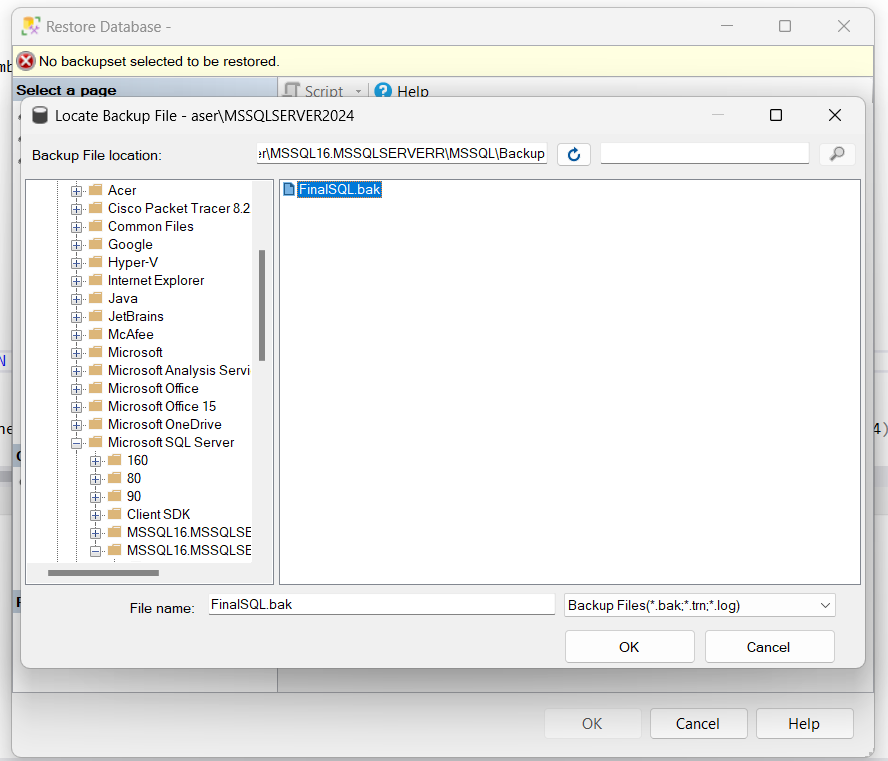
Description automatically generated

* + 1. Press device and then click on the […].

A screenshot of a computer

Description automatically generated

* + 1. Click on Add.



* + 1. Then the backup file will immediately appear. Then click on the file you want to restore then press OK.

1. Database maintenance in general

Database maintenance is the critical task of securing the right functions and optimizations of a database. It includes performing routine checkups, fixing problems, and adding the agency’s administrative tasks to ensure data integrity and its performance. Overall, a well-maintained database results to the overall stability of the Space agency data infrastructure.

The agency’s database maintenance is important for the smooth functioning performance of any database. It also helps in identifying and resolving troubles such as data corruption, performance downfall, and security vulnerabilities. By regularly monitoring and managing the database this will increase reliability. Our goal is to have faster query and a more system efficiency.

So, **the importance of Regular Database Maintenance emphasizes in:**

* 1. Ensuring data integrity

Added on the point about involving a series of tasks that are performed on a regular basis to keep the database running smoothly. These tasks include identifying and fixing any issues or errors, backing up data to prevent losing them, and making sure that the database is secure from unauthorized access. This will improve system reliability and ensure the accuracy of the data.

* 1. Optimizing database performance

It involves identifying and resolving any performance bottlenecks, polishing queries, and improving performance. This helps to prevent data corruption, minimize executing time, and enhancing the user’s experience.

* 1. Preventing data loss

by SQL query execution plan.

However, maintaining a database can be difficult due to its complexity and the big amount of data that companies handle. Here are the best techniques to keep the database running at its top performance:

* 1. Regular Backups

By creating backups at scheduled times, you can protect your data against accidental loss, or system failures. Implementing automated backup solutions.

* 1. Performance Monitoring

By this it will identify bottlenecks, inefficiencies, or potential and partial issues.

* 1. Index Maintenance

This helps in improving the quality of the query execution process and data retrieval. By regularly reviewing indexes to limit and element redundant or not useful indexes and ensure they align with the database domain of work.

* 1. Security Audits

to search for vulnerabilities, unauthorized access, or potential threats. To handle these threats whenever they accrue immediately.

1. Testing
2. Data Validation

|  |  |  |  |
| --- | --- | --- | --- |
| **Number** | **Type** | **Description** | **screenshot** |
| 1 | All cases of PK | PK should be unique so here I tried duplicating the AstroID (The PK) |  |
| PK should not be null. so here is the result when you enter a null value in a PK column |  |
| 2 | All cases of FK | Here in the Astronaut phone number table the FK is set to on update cascade so when I update the PK in the main (referenced) table it should immediately change in the other table |  |
| Here in the Astronaut phone number table the FK is set to on delete cascade so when I delete the PK in the main (referenced) table it should immediately be deleted in the other table |  |
| Since the PK in the AgencyPreMissions table is a foreign key from the referenced Agency entity. So, the FK should be in the domain of the main Agency entity (there should be an AgencyID=4) |  |
| Here when I tried to delete the AgencyID from the Agency entity the request was declined since the Agency ID is an FK in the Selected entity and had the constraint ON DELETE NO ACTION. |  |
| 3 | Unique | Here I tried to put the same email to two different users but that’s not accessible. since each user should have a unique email |  |
| 4 | Default | When you don’t add an attribute for the default attribute it will automatically assign the default value USA |  |
| 5 | Not null | Here I tried to put the value for a not null attribute to null. |  |

1. Output Validation

|  |  |  |  |
| --- | --- | --- | --- |
| **Number** | **Query Description** | **Screenshot (query + result)** | **Result validation** |
| 1 | This Query shows the Astronauts experience by showing the Astronauts First and last name and the agency’s name they have went with. And mission description and name. this will help the voter when choosing the right astronaut for the mission. |  | Here we can see that the data’s view corresponds with the tables. |
| 2 | This Query shows the missions data which are the total missions and the average age of the astronauts who went on the mission for each agency. |  | Here we can see the astronauts’ ages and when we calculate their ages, we will have the same results we had on the view. and we indeed have 4 missions on the previous missions table 3 for Nasa and 1 for Roscosmos. |
| 3 | This query makes the process of hiring (adding) an astronaut easier as it adds an astronaut with its data to the astronaut’s table. |  | Here we can see that the astronaut has been successfully added to the astronaut table. |
| 4 | The query helps the user to edit or add new data to the mission’s description because it is a crucial point since space agencies always have new data from the new discoveries that they constantly do. so, it makes the process easier. |  | Here we can see that the Mission description of the MissionID 1 has been changed successfully |

1. Security Validation

**Note**: you need to test the given and not given privileges.

|  |  |  |  |
| --- | --- | --- | --- |
| **Numb**  **er** | **User Name** | **Description of privilege/no privilege** | **Screenshot (query + result)** |
|  |  |  |  |
| 1 | Agency Administrator | For the Agency Administrator I gave them full privilege of all tables, views and procedures. The privileges are select, insert, update, delete, and alter for the tables. And view, edit, delete and execute for the procedures and views. |  |
| 2 | Employee | For the Employee I gave them the privilege to select(view), insert, and update their data on the voter table. They also have access to the view Astro experience to help them choose their candidates. And they also have the privilege to select(view), insert, and update their data on the Selected table. |  |
| 3 | Astro | For Astro I gave them the privilege to select(view), insert, and update their data on the Astronaut table. and all privileges for their phone number entity. |  |
| 4 | Scientist | For the Scientist I gave them full privilege for Previous missions and agency Mission Data entities. and full privilege for the Agency mission data view. And the privilege to execute the Editing mission description procedure. |  |
| 5 | Public Relation | For the Public Relation user, I gave them the privilege to select(view), insert, and update the data on the voter, Astronaut, Astronaut Phone numbers, and Agency table. They also have access to the view Astro experience. |  |

1. GUI Validation

|  |  |  |
| --- | --- | --- |
| **Number** | **Description** | **screenshot** |
| 1 | Here are the tables that the Research Scientist user can view. |  |
| 2 | This page shows the user Research Scientist the privilege they have on these tables and show them the attributes on each table |  |
| 3 | Here the Research Scientist can view the data on each row specifically and can edit them. |  |
| 4 | Here the Research Scientist can delete a specific row. |  |
| 5 | Here the Research Scientist user can add a new row with new data.  And we can see that all the data we deleted, edited, and updated are shown on the SQL table. |  |

1. Assess whether meaningful data has been extracted

This Database reflects the election process that we have as it covers all data and functions that we have. It represents the reflection by entities (tables) and relations to help the users with what they need. As we have procedures and views that makes the user extract meaningful data.

Multi valued attributes gave an effective way of expressing them. As each Astronaut may have multiple phone numbers so storing them in an independent table with the astronauts ID helped with organising the data and saving more space in the memory. If we have an extra number for an Astronaut, the table will be obligated to add a whole new column just for one extra number for just one astronaut and that’s inefficient.

Also, relationships between entities helped in dependence and extracting meaningful data. For example, a Voter is a worker in one of the Agencies, so it doesn’t make sense when there isn’t a relation between them so by having a 1:M relation we were able to connect both entities by adding the Agency’s ID that the Voter works in in the Voter’s entity. And this 1:M relation also helped with connecting other entities such as, Astronaut and Agency, Previous missions and Astronaut.

Adding more on the helpful relationships between entities since some entities are both equally connected (M: N) this may cause complexity when dealing with data. And by creating a new table to simplify this relation we will have an efficient way to represent this relation resulting with a redundant database and avoiding duplicated data. For example, the previous mission’s entity relates to the agency’s entity with a M: N relation and this relation resulted in having a new entity Called Agency previous mission which has both IDs for both entities and a common attribute which is the Mission’s description.

After creating all the tables needed with their attributes, we need to think about what we will need to add to make this database system more convenient. So, by creating procedures like Hire Astronaut helps the Agencies Administrators with hiring a new astronaut to the agency. It stores all their data (First Name, Last Name, Gender, Degree, Age, Agency ID that they in, and unique ID). Also, the procedure Editing Mission Description helps the Research Scientists when editing and updating the Description of their Previous Missions. And to make the process of adding or editing new data easier since space agencies are continually discovering and working on new data and discoveries. Therefore, having this procedure is a must. And to make it even more practical a new table is created to show the Agencies name and Mission name and the Description since dealing with IDs and numbers may be difficult.

Views also assets with creating the database system by creating Astro\_experience view the Voter can display the Mission Name, Mission Description, Agency Name, Astro First Name, and Astro Last Name. This view is made to help them with choosing the right 7 astronauts for the mission since the effective decision is constructed by considering the experiences of each astronaut by choosing a crew of 7 with diverse backgrounds and perspectives. So, it will help the voters into choosing the right candidates. The other view is Agency\_Mission\_Data it was made to display the Agency Name, Total Missions, and The Average of the Astronauts Ages who went to the Mission. This view will help the agencies in keeping track of their missions and accomplishments. By calculating the number of missions, they have done and the age average of their Astronauts.

1. Assess the effectiveness of testing

Testing the data, output, security, and GUI helps with creating a safe environment for storing data and ensures efficiency of the system. By testing we will discover the bugs and fix them before really using the database system.

First, we tested the data validation by checking the PKs, FKs, constraints, and data type. By writing wrong commands and documenting the output. For instance, we changed the value of the FK AstroID in the Astronaut phone number entity to see if it’s going to automatically change on its main table since we have the command (ON DELETE ON UPDATE CASCADE) to make sure that the command is working. We also checked the PK to see if we can have duplicate IDs, but the system will reject that. And checked the constraints by setting NULL values to a NOT NULL attributes and again the system rejected. This data validation process allowed us to ensure reliability on the system.

Second, we tested the output validation by writing command to see the results of these commands such as SELECT (for viewing tables), updating data, adding rows, deleting data, executing procedures, viewing the views, updating, adding and deleting their data. this helped in ensuring that the desired output aligns with the business requirements and caught any bugs that may affect the database system and ensuring that all data are correctly inserted in the database and finally tables are flexible when manipulation (deleting or updating).

Third, we tested the security validation by writing commands to see if the user has the right privileges on the tables, procedures, and views. or by having the privilege of manipulating them.

Finally, we tested the GUI validation by connecting the database to the hyper link and login-ing to the users account and checking the tables they can view and manipulate and checking their passwords.

Overall, testing is a very important step after creating the database system to look for any bugs and bottlenecks and to handle them before inserting real data to ensure the safest environment.

1. Evaluation of database solution
2. Effectiveness of the database solution based on user and system requirement

This database system was created to serve the process of the Dragon mission election and to see how effective this database is we should align it with the system and user requirements.

As mentioned before the CEO (Agency administrator) of both Agencies since they are the head of the election process and to make sure that everything is going by the plan

Their need from the system is to have full access to all entities and have the access to delete/edit/update and add new data for each entity in any column and that was completely succeeded. the privileges were given to them and to make it easier the GUI was created to make dealing with the entities and data easier. Moreover, creating views were also effective for the administrator requirements such as Astronaut experience and Agency\_Mission\_Data a view that shows how many missions were done by each agency and the average ages of the astronauts who went to that mission this view made the process of viewing data by the administrator and collecting data easier and more reliable. The other view is the Astronaut experience that shows the Mission’s Name, Mission Description, Agency Name, and Astronaut name. This view also helps to eliminate and choose the right 7 astronauts for the mission since they are responsible for making that decision. Procedures were also created to make things easier on the administrator such as Hire astronaut procedure that helps them with adding new astronaut with their full data to the astronaut entity. Another procedure is created, the edit mission description procedure it helps the administrator when editing and updating the Description of their Previous Missions. to keep up with the research scientists who are continually discovering and working on new data and discoveries. And since dealing with IDs and numbers may be difficult a new table is created to show the Agencies name, Mission name and the edited Description for more approachable way of tracking. Overall, the database system is efficiently aligned with the CEO (administrator) requirements in an easy reliable way.

For the other users like the Astronaut what they need from the database system is to view/edit/update and add new data to the Astronaut entity and their phone numbers entity. The privileges were given to them and to make it easier the GUI was created to make dealing with the entities and data easier. Also creating a separate entity for their phone numbers made the process of dealing with that kind of data easier since some astronauts may have multiple numbers. Astronauts can’t delete any of the data for a more secure database and to avoid any data loss. After creating the database, the Astronauts can’t have the privilege to see their previous mission data to have that they need to have access on the previous mission entity and that’s not possible because it will cause Privacy Violation on the Previous mission entity. so, a new user is created the public relations users who are responsible in delivering that information to the Astronauts without any Privacy Violation.

For the Employee users their work on the database is on the Voter entity so they are privileged to view/edit/update and add new data to that entity also the GUI was created to make dealing with entities and data more efficient. Also, they have access on the Astro\_experience view to help them in the decision-making process on who from the astronauts fits more in the Dragon mission so they can see by this view the Mission’s Name, Description, Agency name, and the Astronaut’s name this view helped them to see some of the data in some entities without any Privacy Violation. They also have access to view/edit/update and add new data on the Selected entity to make the process of choosing their candidates easy and efficient. Overall, the voters’ requirements are fully covered and aligned with what they want and need with the help of the GUI and views to make it easy and efficient.

For the Public Relation users their purpose is to reach out to other users and deliver and receive data from the Voters, Astronauts and the Agencies such as reaching out for the astronaut to tell them how many votes they got or to continually check on their data and update them whenever needed. So, they have access to those entities and the privileges to view/edit/update and add new data on the Voter, Astronaut, Astronaut\_PhoneNumber, Agency, and Selected entities. This user will increase the overall efficiency of the election process and deliver trustable data with the help of the GUI that allows them to reach the data they want efficiently. They also have access to the Astro\_experience view that helps them with their job too.

For the Research Scientist user their purpose is to take care of the agencies’ missions’ data and to keep them updated and safe. so, they have access on the Previous missions and AgencyPreMissions entities. They can delete/view/edit/update and add new data to those entities. And they also have the privilege to execute the procedure EditingMissionDescription. It helps with editing and updating the Description of their Previous Missions. since space agencies are continually discovering new data and discoveries. To make it even more practical a new table is created to show the Agencies name and Mission name and the Description since dealing with IDs and numbers may be difficult.

To sum up, this database design aligns with all our users’ requirements and more with the help of the GUI to make dealing with entities easier, views that are easy to understand and not only align with the requirements but also exceeds them. for example, using the group by in the Agency\_mission\_data ensures that the calculations of the values of each agency is calculated individually. And by testing all the queries and the GUI we are sure that this system works perfectly with what we wanted to be created to. And aligns with the requirements.

1. Suggested improvements

This database system has already covered the user and system requirements in an efficient and reliable way. However, we can improve the database in some areas to make it more well planned.

Such as diminishing the load on the agency’s administrators by assigning delegates on each space agency and divide the load on two users. Also creating a procedure to count the numbers of votes for each Astronaut or a procedure to ensure that each voter did indeed vote for 7 Astronauts not more or less.

Moreover, creating more views for the users to make the data that they view cleaner and smoother since sometimes showing the whole table won’t give the user what they want to see.

Also listening to the users and what they really need on a database system will help with aligning the database with their requirements. Going to the Agencies and learning about the business to check that your database really aligns with what the system had covered and that the attributes are specifically specified.

Security is also a very sensitive and important point when it comes to databases so using security tools and hiring reverse engineers to check on the database security will help in securing it.

1. Evaluation based on improvements needed

By illustrating some tips to improve the database system we need to evaluate the overall implementation to see how effective these tips are.

starting with dividing the load into two users rather than making the Agency’s Administrator the only one who is responsible on the database. This will increase the efficiency of dealing with the tables and their columns since the user will be able to focus on a certain domain rather than focusing on a very large and complex number of tables and data. This won’t only make the process of viewing, inserting, updating, and deleting the data easier but makes it smoother and easier for the user too. But still the admin can have all the access on the tables to see and document the overall system.

And by creating a procedure to count the numbers of votes for each Astronaut we will have a more reliable way of counting the votes by only executing the procedure rather than counting the votes from the entity Selected. This will make us avoid counting from a complex table and dealing with a big amount of data at once. Another procedure is to count that each Voter voted exactly 7 times it will help aligning with the election requirements since the Voter can only vote for 7 astronauts not more or less. and by this procedure we will have an effective way to do it avoiding dealing with complex entity and data.

Also creating more views for the users will help them reach certain data that may concern them. For example, the Voter and the Astronaut don’t have access to see the Agency’s entity. But some of them may need some data from this entity such as their address (Country, City, StreetName, BuildingNumber) or the Agency’s phone number to reach the Agency. So, creating a view for that will help with creating a better overall experience for the users.

Elaborating more on the point of creating a better overall experience for the users. we will have to listen to them and what they really need on their database system to know if there are certain attributes needed in the entities for example having an email for the Agency will be helpful for the users. And learning about the business will help the developer in imaging more specific database entities and their columns and more specific attributes that the user may be concerned about. such as more specific data about the previous mission’s entity like the cost of the mission, the score of it, and the duration of the mission these attributes are important to some users.

As mentioned, security is a severely important topic when constructing databases to create a safe environment for the data to be stored away from attacks, hackers, and from being stolen. Therefore, by implementing security tools such as MSSQL DataMask, SCUBA, AppDetectivePro, and Nmap. These tools will help prevent any bugs, unauthorized access, and vulnerabilities and this will increase reliability

1. User Documentation

1.1 System Overview

This Database system was created to help the space agencies administrators, Voters, Astronauts, Research Scientists and Public relations workers with the election process and its tasks for the Dragon mission for choosing 7 Astronauts. As I created the best protected environment to save whatever data that we will have during this process without losing any data. And I also created specific logins for each user with the data they will want and need to view, edit, update, and delete. I also included functions to make the tasks needed in this process easier and more approachable. This system is user friendly and easy to maintain to by any user with any background.

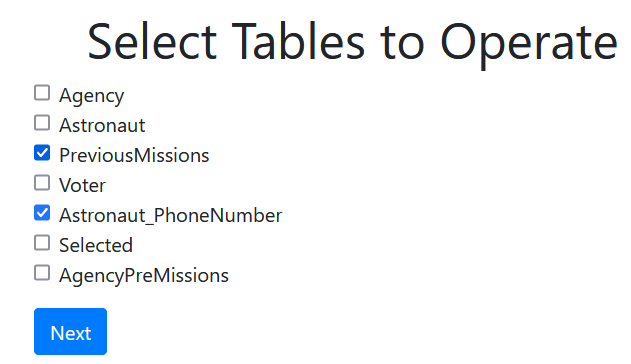
1.2 Using the system

Here is a guide for each user that will have access to the database on how to use it and what they can do with the data.

1. The Agency’s Administrator:

A screenshot of a computer

Description automatically generated

* + 1. First enter the servers name, Database name (Space Mission), your username and your password. Then press connect.
    2. Then this page will appear with all the tables that you will need. It also allows you to select the tables that you want to view. So, select the tables that you want and click Next.

A screenshot of a computer

Description automatically generated

3. Then this page will appear. It will display the tables that you selected to view. By pressing on the table’s name, you will be able to see all the data stored on each column.

A screenshot of a computer

Description automatically generated4. By clicking on the Edit button, the User will be able to edit the data on any row make sure to write the right data type on each column (e.g. write an Integer number for IDs, Phone Numbers etc. and Write letters in the Names and write decimal when needed). When you are done press save to save any new updates.

1. A screenshot of a computer

   Description automatically generatedBy clicking on Add Record you can add a new row with new data. make sure to write the right data type on each column (e.g. write an Integer number for IDs, Phone Numbers etc. and Write letters in the Names and write decimal when needed). And write a unique ID for the new row.

When you are done press save to save any new updates.

A screenshot of a computer

Description automatically generated6.By clicking on delete and then OK you can delete a row from the table. This step Is very crucial because if you delete it, you won’t be able to get the data back and that may cause a data to lose. Note that some users on some tables can’t delete the row since they don’t have the privileged to but the delete button will still be there but when they click OK the row won’t be deleted.

1. Voter:

A screenshot of a computer

Description automatically generated

* + 1. First enter the servers name, Database name (Space Mission), your username and your password. Then press connect.

A black text on a white background

Description automatically generated2.Then this page will appear with all the tables that you will need. It also allows you to select the tables that you want to view. So, select the tables that you want and click Next.

A screenshot of a computer

Description automatically generated

3. Then this page will appear. It will display the tables that you selected to view. By pressing on the table’s name, you will be able to see all the data stored on each column.

A screenshot of a computer

Description automatically generated

4. By clicking on the Edit button, the User will be able to edit the data on any row make sure to write the right data type on each column (e.g. write an Integer number for IDs, Phone Numbers etc. and Write letters in the Names and write decimal when needed). When you are done press save to save any new updates.

A screenshot of a phone number

Description automatically generatedA screenshot of a computer

Description automatically generated5. By clicking on Add Record you can add a new row with new data. make sure to write the right data type on each column (e.g. write an Integer number for IDs, Phone Numbers etc. and Write letters in the Names and write decimal when needed). And write a unique ID for the new row. When you are done press save to save any new updates.

6.By clicking on delete and then OK you can delete a row from the table. This step Is very crucial because if you delete it, you won’t be able to get the data back and that may cause a data to lose. Note that some users on some tables can’t delete the row since they don’t have the privileged to but the delete button will still be there but when they click OK the row won’t be deleted.

1. Astronaut

A screenshot of a computer

Description automatically generated

* + 1. First enter the servers name, Database name (Space Mission), your username and your password. Then press connect.

A close up of a text

Description automatically generated

2.Then this page will appear with all the tables that you will need. It also allows you to select the tables that you want to view. So, select the tables that you want and click Next.

A screenshot of a computer

Description automatically generated

3. Then this page will appear. It will display the tables that you selected to view. By pressing on the table’s name, you will be able to see all the data stored on each column.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

4. By clicking on the Edit button, the User will be able to edit the data on any row make sure to write the right data type on each column (e.g. write an Integer number for IDs, Phone Numbers etc. and Write letters in the Names and write decimal when needed). When you are done press save to save any new updates.

A screenshot of a computer

Description automatically generated

5. By clicking on Add Record you can add a new row with new data. make sure to write the right data type on each column (e.g. write an Integer number for IDs, Phone Numbers etc. and Write letters in the Names and write decimal when needed). And write a unique ID for the new row. When you are done press save to save any new updates.

A screenshot of a computer

Description automatically generated

6.By clicking on delete and then OK you can delete a row from the table. This step Is very crucial because if you delete it, you won’t be able to get the data back and that may cause a data to lose. Note that some users on some tables can’t delete the row since they don’t have the privileged to but the delete button will still be there but when they click OK the row won’t be deleted.

1. A screenshot of a computer

   Description automatically generatedResearch scientist
   * 1. First enter the servers name, Database name (Space Mission), your username and your password. Then press connect.

A close up of a text

Description automatically generated2.Then this page will appear with all the tables that you will need. It also allows you to select the tables that you want to view. So, select the tables that you want and click Next.

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

3. Then this page will appear. It will display the tables that you selected to view. By pressing on the table’s name, you will be able to see all the data stored on each column.

4. By clicking on the Edit button, the User will be able to edit the data on any row make sure to write the right data type on each column (e.g. write an Integer number for IDs, Phone Numbers etc. and Write letters in the Names and write decimal when needed). When you are done press save to save any new updates.

A screenshot of a computer

Description automatically generated

5. By clicking on Add Record you can add a new row with new data. make sure to write the right data type on each column (e.g. write an Integer number for IDs, Phone Numbers etc. and Write letters in the Names and write decimal when needed). And write a unique ID for the new row. When you are done press save to save any new updates.A screenshot of a computer

Description automatically generated

6.By clicking on delete and then OK you can delete a row from the table. This step Is very crucial because if you delete it, you won’t be able to get the data back and that may cause a data to lose. Note that some users on some tables can’t delete the row since they don’t have the privileged to but the delete button will still be there but when they click OK the row won’t be deleted.

E. Public Relations:

A screenshot of a computer

Description automatically generated

* + 1. First enter the servers name, Database name (Space Mission), your username and your password. Then press connect.

A white background with black text

Description automatically generated 2.Then this page will appear with all the tables that you will need. It also allows you to select the tables that you want to view. So, select the tables that you want and click Next.

A screenshot of a computer

Description automatically generated

3. Then this page will appear. It will display the tables that you selected to view. By pressing on the table’s name, you will be able to see all the data stored on each column.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

4. By clicking on the Edit button, the User will be able to edit the data on any row make sure to write the right data type on each column (e.g. write an Integer number for IDs, Phone Numbers etc. and Write letters in the Names and write decimal when needed). When you are done press save to save any new updates.

A screenshot of a computer

Description automatically generated

5. By clicking on Add Record you can add a new row with new data. make sure to write the right data type on each column (e.g. write an Integer number for IDs, Phone Numbers etc. and Write letters in the Names and write decimal when needed). And write a unique ID for the new row. When you are done press save to save any new updates.

A screenshot of a computer

Description automatically generated

6.By clicking on delete and then OK you can delete a row from the table. This step Is very crucial because if you delete it, you won’t be able to get the data back and that may cause a data to lose. Note that some users on some tables can’t delete the row since they don’t have the privileged to but the delete button will still be there but when they click OK the row won’t be deleted

1.3 Frequently asked questions

Here are some frequently asked questions from the users:

* + - 1. Why Can’t I delete data from the tables that I have access to?

Because some users are not allowed to do that since they don’t have the privilege to.

* + 1. What should I do if I forgot my password?

Kindly reach out to the email on the Business card and ask the developer to fix your issue.

1. I deleted the database by accident. What should I do? and is all the data gone?

No, the data is not gone we have the database backed up and ready to be recovered whenever needed, please contact the developer to recover it.

1. When I edit the attributes on the tables sometimes the database rejects it. What are the issues?

This is due to not adding the right datatype in the attributes. Make sure to write the right data type.

1.4 Contact information





References

NASA. (2020). Astronaut Requirements - NASA. [online] Available at: <https://www.nasa.gov/humans-in-space/astronauts/astronaut-requirements/>.

www.tutorialspoint.com. (n.d.). Conceptual Database Design. [online] Available at: <https://www.tutorialspoint.com/conceptual-database-design>.

CHAPTER 7: LOGICAL DATABASE DESIGN (2019). Fundamentals of Database Management Systems, Second Edition. [online] O’Reilly | Safari. Available at: <https://www.oreilly.com/library/view/fundamentals-of-database/9781118213575/15_chap07.html>.

TIBCO. (2024). What is a Logical Data Model? | TIBCO. [online] Available at: <https://www.tibco.com/glossary/what-is-a-logical-data-model>.

‌Ticong, L. (2024). What is a Physical Data Model? | Definition and Examples. [online] Datamation. Available at: <https://www.datamation.com/big-data/physical-data-models/>.

‌ Ticong, L. (2024). What is a Physical Data Model? | Definition and Examples. [online] Datamation. Available at: <https://www.datamation.com/big-data/physical-data-models/>.

‌Itexus.com. (2024). Custom Software Developers. Experienced in Fintech, Insurtech etc. [online] Available at: https://itexus.com/glossary/1nf-first-normal-form/#gref [Accessed 2 Sep. 2024].

‌@Vinchin. (2024). Backup Maintenance: How to Ensure The Usability of Data Backup | Vinchin Backup. [online] Available at: https://www.vinchin.com/disaster-recovery/backup-maintenance.html#:~:text=Backup%20maintenance%20refers%20to%20the [Accessed 2 Sep. 2024].

‌Youtu.be. (2024). - YouTube. [online] Available at: https://youtu.be/CTqEjguqqiI?si=\_hxOO9IrMrJlfSEw [Accessed 2 Sep. 2024].

‌https://www.sciencedirect.com/topics/computer-science/database-maintenance#:~:text=Database%20maintenance%20refers%20to%20the,ensure%20data%20integrity%20and%20performance.

OptimizDBA.com. (2023). The Importance of Regular Database Maintenance. [online] Available at: https://optimizdba.com/the-importance-of-regular-database-maintenance/.

‌www.infosecinstitute.com. (n.d.). Top 10 database security tools you should know | Infosec. [online] Available at: https://www.infosecinstitute.com/resources/general-security/top-10-database-security-tools-you-should-know/.

‌