A picture containing graphics, font, symbol, red

Description automatically generatedDatabase Technical Documentation.

**Database Design & Development**

Made by:

**Nasrullah Rami Hamad**

Supervised by:

**Dr. Aisha Al-Saadi**

**Eng. Jude Barakat**

1: Introduction.

This technical document contains detailed information about Nigel’s Optometry Clinic database, the document is primarily designed to give clear explanations and descriptions of every element inside the medical clinic, such as the doctors and employees and their information. The labs inside the clinic and the optics shops are contracted with the clinic. The document will also include the views and procedures made inside the database. With each user and their privileges inside the database, whether they can insert or edit or remove values from tables, or they can view some values.

The document will also include how security is maintained for the database by the privileges the users get and how the interface works and what it shows. It will include steps on how the testing process works and validations.

It will show an evaluation of the work and future improvements that can be made to increase the performance of the database and to handle any unexpected expansions in the future.

2: The Physical Schema.

A screenshot of a computer

Description automatically generated with medium confidence

This is an SVG image containing the physical schema, which includes all the tables and their attributes, and the relations between them. The primary keys are denoted by a golden key, and the foreign keys are denoted by a blue hashtag, and the table names are at the top.

3: Database Development

3.1: Database Overview

Tables:

|  |  |  |
| --- | --- | --- |
| **Table** | **Name** | **Description** |
|  | manager | The manager table stores the manager’s SSN, Date of birth, location (country, city, street), name (First and last name), years of charge, and their phone number. The table is notated with a primary key called Manager\_SSN and it is connected with the employee table creating a foreign key called Employee\_SSN |
|  | employee | The employee table stores information about the employees such as, their SSN, location (country, city, street), their name (first and last name), their phone number, and the manager they were hired by. The employee table is connected to the lab, doctor, patient, and optics shop table. Creating foreign keys denoted by their primary keys. And the primary key of this table is denoted by Employee\_SSN |
|  | Optics shop | The optics shop table stores information about the shops such as their ID, their location (street, country, city) their name, their phone, and their stock quantity and the SSN of the employee that handles them. and the constraints are primary key, and it is denoted by Shop\_ID, and it has a foreign key called Employee\_SSN since it is connected to it. |
|  | Doctor | The doctor table stores information about the doctors, such as their ID, their name (First and last name) and their phone and the employee that hires them. Its primary key is denoted by Doctor\_ID, and it has a foreign key called Employee\_SSN since it is connected to it |
|  | lab | The lab table stores information about the labs, such as their ID, the floor they’re located inside the clinic, and the room number they are placed in, and the SSN of the employee that manages them. Its denoted by Lab\_ID as a primary key, and it has a foreign key called Employee\_SSN since it is connected to the employee table. |
|  | Patient | The patient table stores information about the patients such as their ID, their name (First and last name), their phone number, and the ID of the lab they get treated in, the SSN of the employee who manages their appointments, the ID of the doctor they are treated by, and the ID of the shop they get recommended to. The constraints it holds are a primary key denoted by Patient\_ID, and foreign keys (Doctor\_ID, shop\_ID, Employee\_SSN, Lab\_ID) since the table is connected to the four tables. |
|  | Lab schedule | This table stores information about the lab schedules, such as the ID of the lab, the reserved time, and the status whether the lab is active or not active. The table relates to the Lab table because it’s a multivalued attribute. And the constraint in the table is primary key denoted by Lab\_ID |
|  | Appointments | This table stores information about the appointments, such as the ID of the appointment, the appointment date, and the doctor and patient IDs. The constraints are a primary key called Appointment\_ID, and two foreign keys called doctor\_ID and Patient\_ID. |

Views Table:

|  |  |  |
| --- | --- | --- |
| **View** | **Name** | **Description** |
|  | Doc list | This view lets the manager know the count of the patients that are performed on by the doctors |
|  | Lab patient | This view lets the lab manager know the count of patients that have entered the labs |
|  | Optic patient | This view lets the manager know the count of patients that have entered each shop |
|  | Stock emergency | This view lets the employee who’s responsible for shops know which shops have stock quantity less than 1000 so he can request more stock for the future. |

Procedures Table:

|  |  |  |
| --- | --- | --- |
| **Procedure** | **Name** | **Description** |
|  | Emp\_man\_shop | This procedure lets the manager know information about each optics shop handled by specific employees by entering the Manager’s SSN. |
|  | Man\_emp\_lab | This procedure lets the manager know information about each lab handled by specific employees inside the clinic by entering the Manager’s SSN. |
|  | Pat\_doc\_app | This procedure lets the doctors know information about their patients and their appointments by entering their ID. |
|  | Pat\_lab\_sched | This procedure lets the patient know what labs are active and when are they active with their location inside the clinic by entering the Lab\_ID they acquire from the receptionist. |

3.2: Security:

|  |  |  |  |
| --- | --- | --- | --- |
| **User name** | **Privilege Command** | **Description** | **Screenshot** |
| Manager | GRANT INSERT, UPDATE, DELETE ON employee.\* to manager; | This command grants the manager the privilege to add, edit, and remove employee accounts and information |  |
| GRATE INSERT UPDATE, DELETE ON optics\_shop.\* TO manager; | This command grants the manager the privilege to add, edit, and remove optics shop information. |  |
| Employee | GRANT INSERT, DELETE, UPDATE ON doctor.\* TO employee; | This command grants the employees privileges to add, edit, and remove doctor information |  |
| GRANT INSERT, DELETE, UPDATE ON appointments.\* to employee; | This command grants the employees privileges to add, edit, and remove appointment information | A picture containing text, screenshot, font  Description automatically generated |
| GRANT INSERT, DELETE, UPDATE ON patient.\* to employee; | This command grants the employees privileges to add, edit, and remove patient information and accounts |  |
| GRANT UPDATE, SELECT ON lab\_schedule.\* to employee; | This command grants the employees the privileges to edit and view lab schedules and their time. | A screenshot of a computer program  Description automatically generated with low confidence |
| GRANT SELECT ON employee.\* to employee; | This command grants the employees the privilege to view their information. |  |
| Doctor | GRANT UPDATE, SELECT ON appointments.\* to doctor; | This command grants the doctor the privilege to view and edit information from the lab schedule | A picture containing text, screenshot, font  Description automatically generated |
| GRANT SELECT ON patient.\* to doctor | This command grants the doctors the privilege to view patient information inside the database |  |
| GRANT SELECT ON lab\_schedule.\* to doctor; | This command grants the doctors the privilege to view lab schedules | A screenshot of a computer program  Description automatically generated with low confidence |
| Patient | GRANT SELECT ON patient.\* to patient; | This command grants the patients the privilege to view their information inside the database |  |
| GRANT SELECT ON appointments.\* to patient; | This command grants the patients the privilege to view their appointments; | A screenshot of a computer  Description automatically generated with low confidence |
| GRANT EXECUTE ON PROCEDURE pat\_doc\_app TO patient; | This command grants the patients the privilege to execute the procedure which lets them view the doctor with their appointment information | A screenshot of a computer  Description automatically generated with low confidence |

3.3: User Interface.

3.3.1: Flowchart and Data Movement Diagrams:

The flowchart included how the admin enters the database, which tables he has access to, and what privileges he was granted in each table, with the conditions of whether he wants to perform the privileges he was granted or not.

A diagram of a flowchart

Description automatically generated with low confidence

The DFD (Data Flow Diagram) focuses on each user and its tables, and privileges. Since there are four users in the database, there must be four DFDs each focusing on one user. Starting with the employee’s DFD:

A picture containing diagram, text, plan, technical drawing

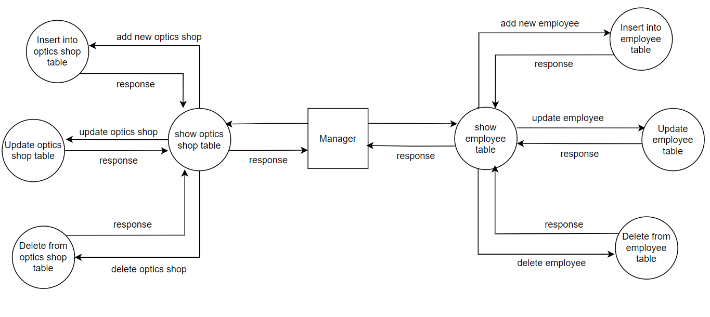
Description automatically generated

Doctor user’s DFD:

A diagram of a doctor

Description automatically generated with medium confidence

Manager User’s DFD:



Patient User’s DFD:

A diagram of a patient appointment table

Description automatically generated with medium confidence

3.3.2: Interfaces Development

|  |  |  |  |
| --- | --- | --- | --- |
| **Page ID** | **Title** | **Description** | **Screenshot** |
|  | Homepage | The homepage is the main page for the user, it includes all the tables the user can access and if they press on a table, it leads them to its data. There are five tables inside the homepage for the employee. |  |
|  | Patient | The patient button leads the employee to patient data, the employee has privileges on this table such as inserting more patients and deleting patients that got treated |  |
|  | Doctor | The doctor button leads the employee to the doctor’s data, the employee was granted privileges on this table such as inserting doctor information and deleting the doctors that left the clinic |  |
|  | Appointments | The appointments button leads to appointment information for both the doctor and patient. The employees were granted privileges on this table such as inserting more rows and appointments, they can also edit the existing appointments, and delete the appointments that already finished |  |

4: Maintenance

4.1: Database Recovery & Backups:

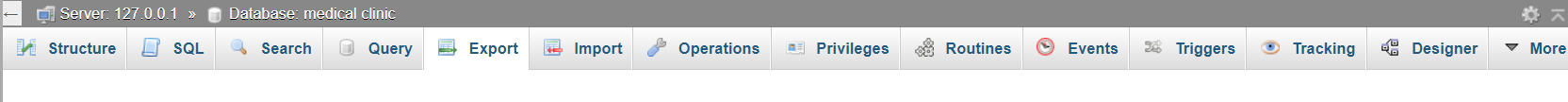
Database backups are essential because they prevent any data loss from hardware malfunction. Backups can either be time-regulated such as bimonthly backups or daily backups, or action-regulated are based on a certain action occurring inside the clinic, which might lead to emergency backups of data and the database.

Database recovery inside Xampp can be made by two main functions (Import & Export) since we can import the database and export it back with the same data, without it being lost.

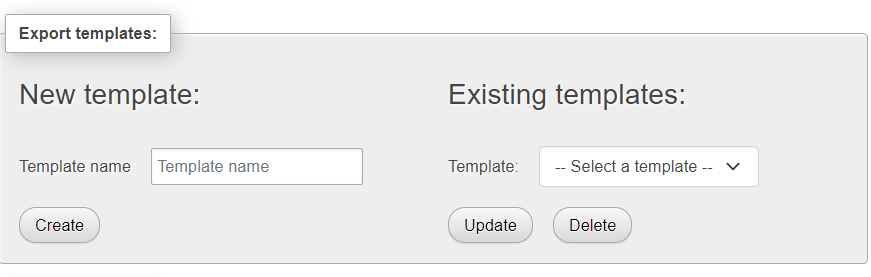
The process of exporting the database to the device storage and then importing it back again when needed to prevent loss of data and to recover the database efficiently and easily can be made in the following steps:

1. Exporting:

* Exporting the database can be done by pressing the ‘Export’ button that’s located in the toolbar of the database:



* We can export a new template or update an existing template or delete it.

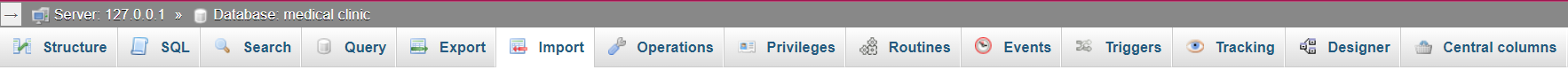


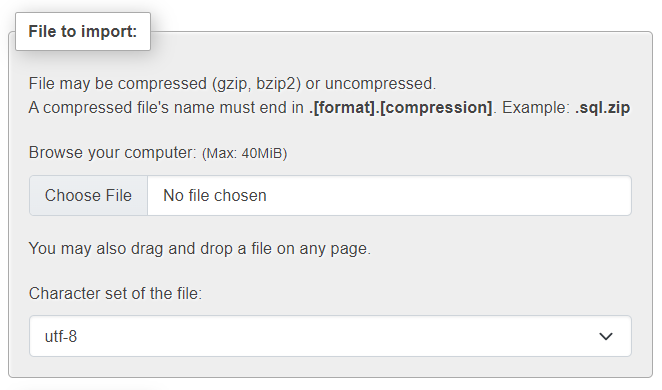
* We can then choose a format for the file and press ‘Export’ and finish the exporting process by selecting a location on the device that will save the exported file.



1. Importing:

* The importing process starts in the same way by pressing the ‘import’ button from the toolbar of the database above:



* After that we can choose the file that we want to import by choosing the file from the device:
* After selecting the file that needs to be imported, we can now select several options that can improve the file’s importing process. The format of the file must be the same format that the file was exported in. Then to finish the importing process we just need to press ‘Import’ to import the file successfully.

A screenshot of a computer

Description automatically generated with medium confidence

4.2: Database Maintenance:

Maintenance is a very crucial step in every database because it keeps the database running with no errors or slowness when saving data, a set of methods that can be followed to ensure the efficiency and longevity of the database is called maintenance.

Database backups and recovery processes increase the performance of database maintenance, it is necessary to organize and optimize the databases with the supervision of a person of expertise since they know about the characteristics of the database and its structure and details.

Database maintenance has many benefits, for instance:

1. It keeps companies updated on technological improvement since it is quickly enhancing, and companies usually fail to be in touch with the developmental field. Maintenance helps the companies be efficient and fast when moving within updates.
2. Easier data recovery, since maintenance can help with providing a quicker backup process therefore, it can lead to easier and faster data recovery. Without losing the data and corrupting it.
3. Maintaining the database can rid the database of useless and duplicated data with no meaning since they can result in slower performance, and it could require additional operational costs.

Here are some ways that can be used to maintain the databases on a higher scale:

* Storing the data in a single file: That way it can be easier to manage the structure and easier to control and access. So, when it comes to updating the database, a single file will only be accessed.
* Using informing titles: Clear, meaningful titles and definitions can help the experts understand the database more, and they can access what they need in a very efficient and easier way.
* Regular updates must be performed inside the database so it doesn’t lose its efficiency and so it stays up to date with whatever is going on in the developmental zone.
* Maintenance plans are one of the most important ways to keep the database maintained since they can provide a strategy that works on every aspect of the database while ensuring risk prevention and hardware failures. So, the database works at the highest peak in performance and efficiency.

5: Testing

5.1: Data validation:

|  |  |  |  |
| --- | --- | --- | --- |
| **Number** | **Type** | **Description** | **screenshot** |
|  | All cases of PK | I tried to insert a row without a primary key, but SQL wasn’t allowing it to be inserted with a null primary key since one of its features is not allowing null values as primary keys. |  |
| Since the primary key is a unique attribute, no duplicated entries can be inserted.  I tried inserting an already existing primary key inside this row. This is the result. | A picture containing text, screenshot, font, number  Description automatically generated |
|  | All cases of FK | The inserted foreign keys must be in between the range of the primary keys.  I tried inserting a foreign key that is out-of-bounds of the primary key range here |  |
| The update cascades allow the user to update existing rows, so they meet the requirement of the user.  We want to update the patient ID 92 to 95. The foreign key references by the primary key will also be changed to 95. |  |
| The delete cascade works by deleting the primary key of a table that references other foreign keys in other tables. Deleting the primary key will result in the deletion of the foreign keys in other tables.  We want to delete the patient that has the ID 95, which will delete its referenced foreign key from the appointments table. |  |
|  | Unique | The unique constraint ensures that the values of a column don’t have the same value and aren’t duplicated.  I implemented the doctor number as a unique attribute; therefore, it shouldn’t occur more than one time in the table. |  |
|  | Not Null | The not-null constraint ensures that no data is inserted as NULL, every cell should have a value.  Here I implemented the not null constraint on the fname attribute, and I tried inserting a null value in it. |  |
|  | Default | The default constraint sets up a default value if a certain condition was met.  Here I implemented a default constraint on the years of charge column of the manager table and inserted a without the years of charge value, the default value is set to 0. |  |
|  | Check | The check constraint checks if the inserted value is in the range of the condition provided inside the check statement.  I implemented a check constraint that will not allow stock quantities less than 0 inside the optics shop table. |  |

5.2: Output validation

|  |  |  |  |
| --- | --- | --- | --- |
| **Number** | **Query Description** | **Screenshot (query + result)** | **Result validation** |
|  | This procedure call lets the doctors see their patients’ names, and phone numbers, along with their appointment date and time. By entering their doctor\_ID  In this example, I called the procedure and added a doctor ID, then it showed all the patients that are appointed to this doctor. Along with their information. |  |  |
|  | This query inserts a new instance into the doctor table. The primary key must be unique, so it works as intended. |  |  |
|  | This query will update the previously inserted row, with a new phone number, since the employee changed his phone. |  |  |
|  | This query the view that shows shops with stock quantity less than 1000 so employees can request more stock |  |  |

5.3: Security Validation:

|  |  |  |  |
| --- | --- | --- | --- |
| **Number** | **User Name** | **Description of privilege/no privilege** | **Screenshot (query + result)** |
|  | Employee | The employee wasn’t given the privilege to delete employee information from the table. (No Privilege) |  |
|  | Doctor | The doctor can insert new rows in the patient table, that contains the patient’s information (Privilege)  The doctor logged in and can insert a new instance inside the patient table. |  |
|  | Doctor | The doctor doesn’t have the privilege to able to delete appointments (No Privilege) |  |
|  | Manager | The manager was granted the privilege to update optic shops information (Privilege)  The manager will try to update the name of the optics shop to another. |  |

5.4: GUI Validations:

|  |  |  |
| --- | --- | --- |
| **Number** | **Description** | **screenshot** |
|  | The employees were granted the privilege to insert new instances inside the doctor’s table.  As the picture shows, the employee can insert a new instance. And the instance is added to the myphpadmin too. |  |
|  | The employee can update the instances of the doctor table. We will update the first and last name of a doctor since it was mistakenly inserted. |  |
|  | The employee can delete patient information, so we will delete the patient with id = 85.  These images will show the instance that was deleted and then the table with it deleted inside the php my admin and the PHP generator. |  |
|  | Employees have been given the privilege to insert new appointments inside the appointment table.  The images show that the employee can insert new instances inside the appointment table, with the new appointment appearing in the sql. |  |

5.5: Assess whether meaningful data has been extracted:

The (1:M) relations done inside the database were meaningful. For instance, the relation between the manager table and the employee table was (1:M) because each manager can hire multiple employees. At the same time, multiple employees are hired by one manager. For example, the manager with the SSN 2000632245 has multiple employees that can be shown in the image under. Which means they were hired by him.

A screenshot of a computer

Description automatically generated with low confidence

A screenshot of a computer

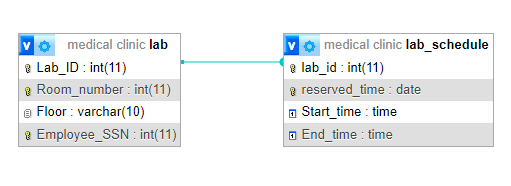
Description automatically generated with low confidence

A multivalued attribute called Lab\_schedule was created from the Lab table since schedules can contain the reserved date with the start and end time. Therefore, a multivalued relation was created. For example, Here’s the lab with ID = 11111, and its 5-day schedule including the start and end time of its schedule.

A screenshot of a computer

Description automatically generated with low confidenceA screenshot of a computer

Description automatically generated with low confidence

The procedures and views implemented were also thought of and were meaningful since using them will save time and work for the employees and doctors.

For instance, the procedure pat\_doc\_app looks to get information about patients and doctor appointments. The doctor inserts their ID, and they get provided with information regarding the patients the doctors have and their appointments. The procedure man\_emp\_lab provides the manager the labs that are inside the clinic when inserting the employee ID that’s managing them.

The views implemented such as the ‘Stock emergency’ view can provide the employees of the shops that have insufficient stock. Employees then will have to email the shops to alert them that they need more stock and they would prepare more stock.

According to the notes above, it can be said that the data inserted inside the database was thought of according to a specific scenario and was not meaningless. The procedures and views served their purpose which is making access to certain items inside the database easier for the employees and faster.

5.6: Effectiveness of testing the database:

Testing is a very essential part of database designing because with it we make sure that everything inside the database is working according to plan. There were many factors and types of testing that were done inside the database to ensure that it was working smoothly for every user that is privileged to access it. The testing of the database was split into four parts:

1. Data validation: this step was to make sure that the primary and foreign keys worked as intended with high efficiency, many constraints were also tested so they meet the expectations of the users. Such as, how foreign keys should be inside the range of the primary keys they were referenced from, and that primary keys needed to be unique and not a null value. We tested four constraints such as check, default, unique, and not null for some columns and found that they worked perfectly.
2. Output validation: this type was to test how outputs from different queries looked like and if they worked as intended. I used multiple queries such as calling a procedure that provided the result it was designed to provide. And I tried inserting an instance in a table to check if inserting works smoothly. Lastly, I checked a view by using the select statement followed by the view name, and it gave results specific to its design.
3. Security validation: This testing type tests the privileges that were granted to the four users; the manager, employee, doctor, and patient. Each of these users was granted access to the database and was also granted some privileges according to the level of authority they are in. I started testing the users with the privileges that they do not have, to make sure that the system is preventing them from accessing certain locations. And the results were as intended. Then I tested some of their privileges such as viewing and inserting rows inside tables or updating and deleting them. And they provided high efficiency and performance.
4. GUI validation: after implementing the PHP generator as the GUI system, it showed the same users that the SQL had, with their privileges intact, the point of the GUI validation was to test whether each command done on the GUI would be also done inside the SQL, so I tested doing some commands with the administrator such as deleting instances, updating and adding some instances too, in multiple tables. Every command done showed its result inside the GUI system, and the same result appeared inside the SQL system.

These are some of the ways testing was used inside the database and judging from the types, testing provided positive results with good efficiency and high performance, and testing will have to be a regular step in the future since it will ensure the longevity and maintenance of the database without loss and slowness of the system and the database in general.

6: Evaluation of database solution:

6.1 effectiveness of the database solution based on user and system requirements:

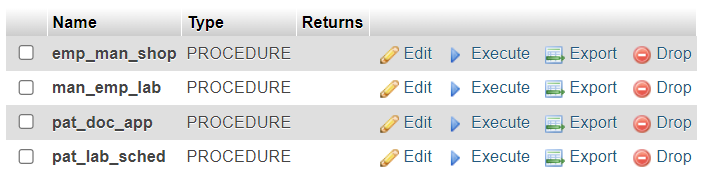
The users that were granted access to the database had their privileges from the user and system requirements file that was provided way earlier, It can be said that this part was achieved accordingly, and they should be able to continue their work in an efficient and stable manner.

The medical clinic database itself had multiple changes from the physical design that was just the plan of its implementation, since there were some changes to how tables worked, and how some attributes were put. The structure of the database at this moment is as the following:

A screenshot of a computer

Description automatically generated with medium confidence

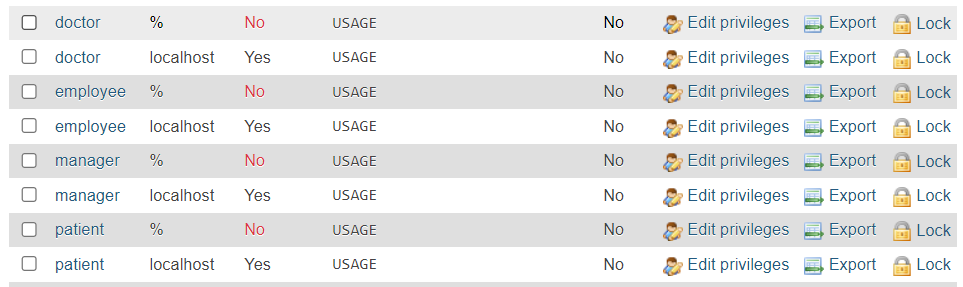
This structure contains all the tables, with the views and the possible options. Each table has a set of columns that include a primary key and often foreign keys, with data types and constraints. The procedures and views created inside the database to increase the efficiency and enhance the flexibility and simplicity of the database were the following:

A screenshot of a computer

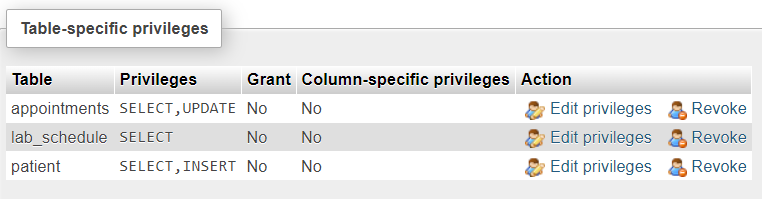
Description automatically generated with low confidence

The images above contain all the views and procedures created in a meaningful way, where each procedure takes a value and provides information from two different tables using inner joins. The views were also created using joins between tables, and views work without entering a value into it.

The users created inside the database were managers, employees, doctors, and patients. As the following:



They were created with passwords inside the database so they can access it, while being granted some privileges depending on where they work or what they do. For example, the doctor was granted access to some tables:



Judging by the image above, the doctor was granted the privilege to view three tables, and update the appointments table, and insert into the patient’s table. He cannot grant other users with privileges, and he cannot view other tables.

6.2: Improvement suggestions:

As mentioned before, the medical clinic is eventually going to expand, perhaps into two medical clinics or the same clinic might expand its structure and get more attractions and patients and doctors. To deal with the unexpected and unusual expansion and change of the usual, there will be multiple plans or suggestions that will be studied and then implemented in the future, to prevent any loss of credibility and reputation.

Creating new tables and expanding the database: new tables must be added in the future to save more expected data in the future. Such as department which could include how many departments are inside the clinics, opening schedule if the clinic stops being a 24/7 clinic, staff since the clinic will eventually recruit more, insurance companies because the clinic will need to contract with some later the line, and more.

To be able to continue the work in the clinic in a more efficient way, the creation of the department table should be studied instantly, to plan for expansion. The table will include the department ID as the primary key, and columns such as its location and the workdays. It can also contain the SSN of the manager that is managing the department as a foreign key referencing the Manager SSN from the manager table. By doing that, time and cost will be saved for the future.

Some IT management employees should be hired to maintain the database’s work and the website’s performance and stability for the users. They should be hired based on their experience with maintaining expandable databases and their trustworthiness, so the data stays integrated safely with no breaches and attacks happening soon.

6.3: Evaluation based on the improvements:

Since the clinic might expand, the old database might start getting full and slow, which will lead to lower performance and less efficiency when saving new instances, this can lead to loss of data and not loading correct instances and views not working accordingly. The database will have to be moved onto a new server with better storage that can handle the massive amounts of data provided daily. This can be doable once the IT management is hired, and their work is acknowledged as experienced. So, moving the database wouldn’t be a problem.

And as mentioned before, expanding will lead to less coverage for the area of the medical clinic, inside and outside. So, there must be some security protocols that protect the clinic and its data. Otherwise, the clinic will be vulnerable to attackers, and the database will be easier to breach and hack. The implementation of firewalls to protect the website and its database is a necessity and some security protocols like two-factor authentication for the users to protect the medical clinic’s database from unauthorized access. This suggestion can be done once the clinic itself is stable and the database is working efficiently. So, security tools will be implemented according to the database.

Adding a new table to the database called department should be doable at this moment. Since the database was just created, it would be easier and simpler to create it now. While having the primary key denoted as department ID and adding some columns such as its name and location, and the SSN of the manager that’s managing the department.