**Team F:**, Allison Escott, Bishoy Soliman, Franco Bazzini, Katelyn Tropeano, Katie Houskeeper, Vishnu Priya Dendukuri

MIS 3720 Final Project

# Hospital Database Final Report

## Business Problem

The business problem we worked to solve is the issue of a hospital admitting more patients than the number of beds available. Relevant aspects of the hospital we are looking into in order to develop our database are nurses, doctors, patients, rooms and departments. We developed a database that stores information regarding which rooms are currently occupied and which are still available. This will allow for an easily accessible database to allow for the correct number of patients to be admitted based on beds available.

## Database Design

To solve the business problem of a hospital admitting more patients than the number of beds available, our team developed a database in Microsoft Access.

Overall, developing a database in Microsoft Access can help solve the business problem of a hospital admitting more patients than the number of beds available by providing easy access to information about occupied and available rooms.

The main steps to develop the database included:

1. Determine the necessary tables for the database: For this particular problem, tables for patients, doctors, nurses, rooms, and departments would be needed.
2. Define the relationships between the tables: The relationships between the tables should be established to ensure data integrity and prevent data duplication. For instance, each patient will be assigned to a room, and each room will belong to a department.
3. Establish data entry forms: This will provide an easy way for users to input and update data in the database.
4. Create reports: These will be used to generate insights on the number of occupied and available beds, patients' medical history, etc.
5. Develop a central switchboard that includes sub-search boards and prepare a user documentation report that outlines how to navigate our switchboard for individuals who are unfamiliar with the complexities of Access.

Before creating the database in Access, however, we needed to start by developing a conceptual model, Entity-Relationship Diagram, logical model, and data dictionary.

A conceptual database model is a high-level representation of the overall structure and organization of a database. It describes the entities and their attributes. It does not include details about how the data is stored or how it will be accessed or each entity's foreign keys.

### Conceptual Model

Patients:

* PatientID - Identifier
* FirstName
* LastName
* DOB
* Gender
* DateAdmitted
* City
* State
* Country
* ZipCode
* InsuranceName

Nurse:

* NurseID - Identifier
* FirstName
* LastName
* YearsPracticing
* Availability

Doctor:

* DoctorID - Identifier
* YearsPracticing
* FirstName
* LastName
* Availability

Room:

* RoomID - Identifier
* NumberOfBeds
* NoBedsAvailable
* RoomType

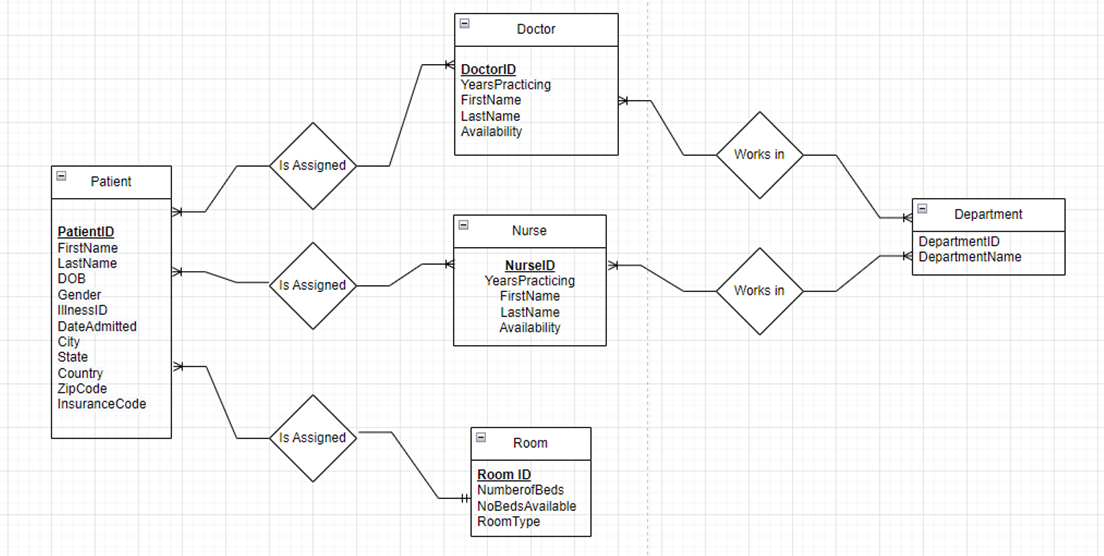
Department:

* DepartmentID - Identifier
* DepartmentName

### 

### Entity-Relationship Diagram

An Entity-Relationship Diagram (ERD) is a representation of the entities, attributes, and relationships involved in a database. ERDs are used in database design to illustrate the logical structure of a database, and they help to ensure that all the data required for a particular application is accurately captured.



### Logical Model

A logical model is a representation of a database that describes the data in terms of its structure, relationships, and constraints, without specifying how the data will be physically stored or implemented.

tbPatients (**PatientID**, FirstName, LastName, DOB, Gender*,* DateAdmitted, City, State, Country, ZipCode, InsuranceName, *DoctorID*, *NurseID*, *RoomID*)

tbNurse (**NurseID***, DepartmentID* , FirstName, LastName, YearsPracticing, Availability)

tbDoctor (**DoctorID***, DepartmentID .*YearsPracticing, FirstName, LastName, Availability)

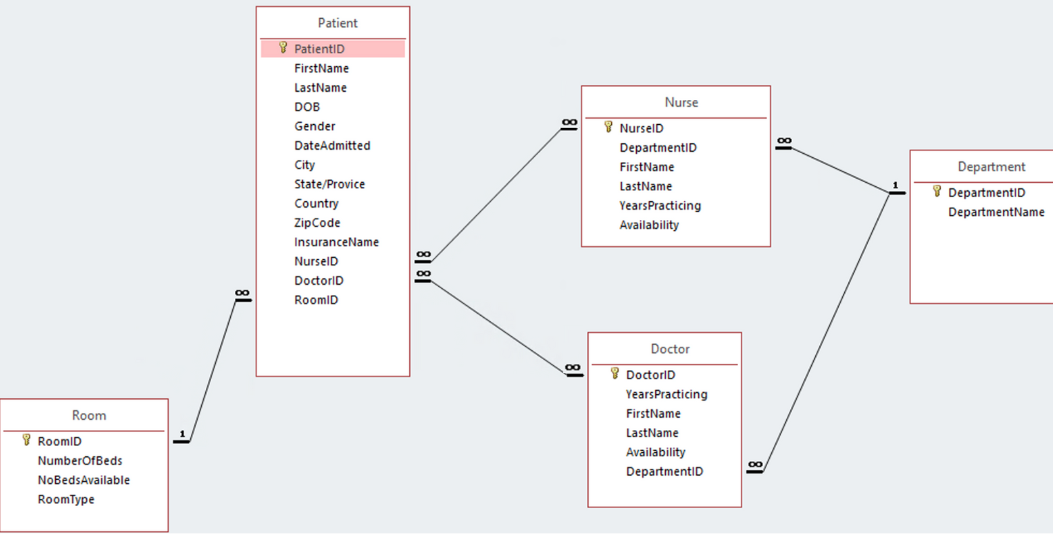
tbRoom (**RoomID**, NumberOfBeds, NoBedsAvailable, RoomType)

tbDepartment(**DepartmentID**, DepartmentName)

### Microsoft Access Database

After developing the conceptual and logical models and defining the entities, attributes, and relationships, we created an Access database that reflects the design. The Access database is a physical implementation of the logical model and includes tables, fields, relationships, and other components necessary to store and manage data.

To help visualize the database structure and relationships, we have attached the Access relationship diagram below:



In the database design, the following relationships were captured:

1. Room (1) - Patient (Many): This relationship assumes that each room can have many patients at the same time, but each patient can only be assigned to one room at a time.
2. Patient (Many) - Nurse (Many): This relationship assumes that each patient can be assigned to multiple nurses for care, and each nurse can be assigned to multiple patients.
3. Patient (Many) - Doctor (Many): This relationship assumes that each patient can be treated by multiple doctors for different medical conditions, and each doctor can treat multiple patients.
4. Doctor (Many) - Department (1): This relationship assumes that each doctor can be part of only one department in the hospital, but many doctors can belong to the same department.
5. Nurse (Many) - Department (1): This relationship assumes that each nurse can be part of only one department in the hospital, but many nurses can belong to the same department.

The assumptions behind each relationship are based on the typical operating procedures in a hospital environment. For example, it is reasonable to assume that each room can accommodate more than one patient at a time, as this is common practice in many hospitals. Similarly, it is common for patients to be assigned to multiple doctors and nurses for different treatments and care.

Furthermore, it is appropriate to assume that each doctor and nurse can belong to only one department at a time, as this helps to manage their workload and responsibilities effectively. Overall, these assumptions reflect the realities of hospital operations and provide a solid foundation for the database design.

### Data Dictionary

A data dictionary is a collection of data that provides information about the data in a database. It contains the attribute, data type, description, and purpose of the attribute being in the database.

We have described the purpose of each table at the beginning of each data dictionary table.

Data Dictionary Key:

**Bold: Primary Key**  *Italicize: Primary key*

**Patient:** The purpose of the patient table is to provide all of the information in relation to the patient that will be needed when they are admitted into a hospital.

| Column | Data Type | Description | Purpose |
| --- | --- | --- | --- |
| **PatientID** | Short Text | Primary key of the Patient Table | PatientID is in the table because it will be the key identifier of the patient that is in the hospital. Each patient will have their own unique ID. |
| FirstName | Short Text | First name of  the patient | FirstName is in the table because it will be used to record the patient's first name. |
| LastName | Short Text | Last name of the patient | LastName is in the table because it will be used to record the patient's last name. |
| DOB | Date/Time | DOB of the  patient | DOB is in the table because it will be used to record the patient's date of birth. |
| Gender | Short Text | Gender of the patient | Gender is in the table because it will be used to record the patient's gender. |
| DateAdmitted | Date/Time | Date admitted  into hospital | DateAdmitted is in the table because it will be used to record when a patient was admitted into the hospital. |
| City | Short Text | City of patient | City is in the table because it will be used to record the city that the patient is from. |
| State/Providence | Short Text | State of the  patient | State is in the table because it will be used to record the state that the patient is from. |
| Country | Short Text | Country of the patient | Country is in the table because it will be used to record the country that the patient is from. |
| ZipCode | Short Text | Zip code of the  patient | ZipCode is in the table because it will be used to record the zipcode of the patient. |
| InsuranceName | Short Text | Name of the insurance company | InsuranceName is in the table because it will be used to record the that the patient will be using to cover their stay. |
| *DoctorID* | Short Text | Foreign key referencing Doctor ID from the Doctor Table | DoctorID is in the table to record the doctor that will be working with the patient. |
| *NurseID* | Short Text | Foreign key referencing Nurse ID from the Nurse table | NurseID in the table to record the nurse that will be working with the patient. |
| *RoomID* | Short Text | Foreign key referencing Room ID from the Room table | RoomID in the table to record what room the patient is in. |

**Nurse:** The purpose of the nurse table is to provide all of the information in relation to the nurse that will be needed when they are working at a hospital.

| Column | Data Type | Description | Purpose |
| --- | --- | --- | --- |
| **NurseID** | Short Text | Primary key of the Nurse Table | NurseID is in the table because it will be the key identifier for each nurse that works the hospital. Each nurse will have their own unique ID. |
| *DepartmentID* | Short Text | Foreign Key referencing Department | DepartmentID is in the table because it will be used to identify the department that the nurse is in. |
| FirstName | Short Text | First name of the nurse | FirstName is in the table because it will be used to record a nurse's first name. |
| LastName | Short Text | Last name of the nurse | LastName is in the table because it will be used to record a nurse's last name. |
| YearsPracticing | Number | Number of years of experience | YearsPracticing is in the table because it will be used to record the amount of years each nurse has been working for. |
| Availability | Yes/No | Checking if the nurse is available or not | Availability is on the table because it will be able to tell the hospital if the nurse is available or busy. |

**Doctor:** The purpose of the doctor table is provide all of the information in relation to the doctor that will be needed when they are working at the hospital.

| **Column** | **Data Type** | **Description** | **Purpose** |
| --- | --- | --- | --- |
| **DoctorID** | Short Text | Primary key of the Doctor Table | DoctorID is in the table because it will be the key identifier for each doctor that works the hospital. Each doctor will have their own unique ID. |
| FirstName | Short Text | First name of the nurse | FirstName is in the table because it will be used to record a doctor’s first name. |
| LastName | Short Text | Last name of the nurse | LastName is in the table because it will be used to record a doctor’s last name. |
| YearsPracticing | Number | Number of years of experience | YearsPracticing is in the table because it will be used to record the amount of years each doctor has been working for. |
| Availability | Yes/No | Checking if the doctor is available or not | Availability is on the table because it will be able to tell the hospital if the doctor is available or busy. |
| *DepartmentID* | Short Text | Foreign Key referencing Department | DepartmentID is in the table because it will be used to identify the department that the doctor is in. |

**Room:** The purpose of the room table is to provide information about rooms in the hospital.

| Column | Data Type | Description | Purpose |
| --- | --- | --- | --- |
| **RoomID** | Short Text | Primary key of the Room Table | RoomID is in the table because it will be the way a room is identified. Each room will have its own unique ID |
| NumberOfBeds | Number | Number of Beds in the room | NumberOfBeds is in the table because it will provide the number of beds that are in each room. |
| NoBedsAvailable | Number | Number of beds available in the room | NoBedsAvailable is in the table because it will tell how many beds are available for use in the table, and how many beds are currently empty. |
| RoomType | Short Text | Description of room type | RoomType is in the table because it will be used to record the type of room that each room is. If the room is private or if it is shared. |

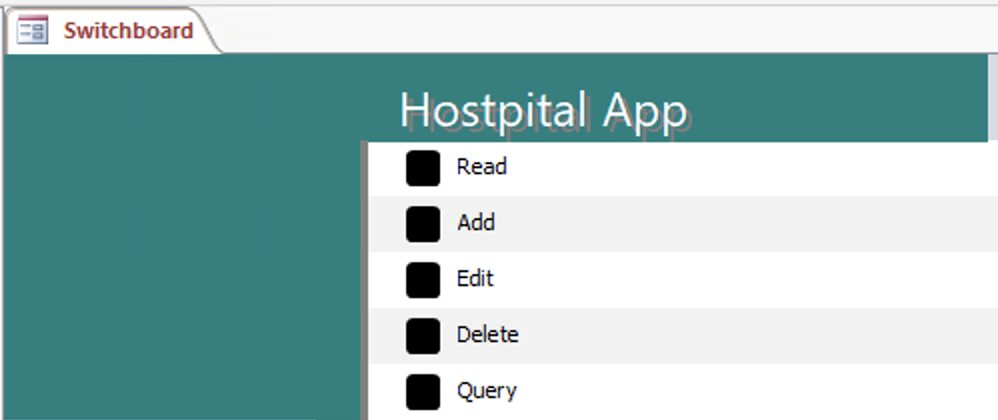
**Department:** The purpose of the department table is to provide information about the departments in the hospital.

| Column | Data Type | Description | Purpose |
| --- | --- | --- | --- |
| **DepartmentID** | Short Text | Primary key of the Department Table | DepartmentID is in the table because it will be the way a department is identified. Each department will have its own unique ID |
| DepartmentName | Short Text | Name of the department | DepartmentName is in the table because it will be used to identify the name of each department. |

### Switchboard

In addition to creating the database, we also designed an Access switchboard. The purpose of the switchboard is to provide a user-friendly interface for individuals who may not be familiar with using Access, allowing them to easily navigate and utilize the database.

Our main switchboard comprises five sub-switchboards, each equipped with a back button that allows users to easily navigate back to the main switchboard. For detailed instructions on how to use our switchboard, please refer to our User Documentation report.

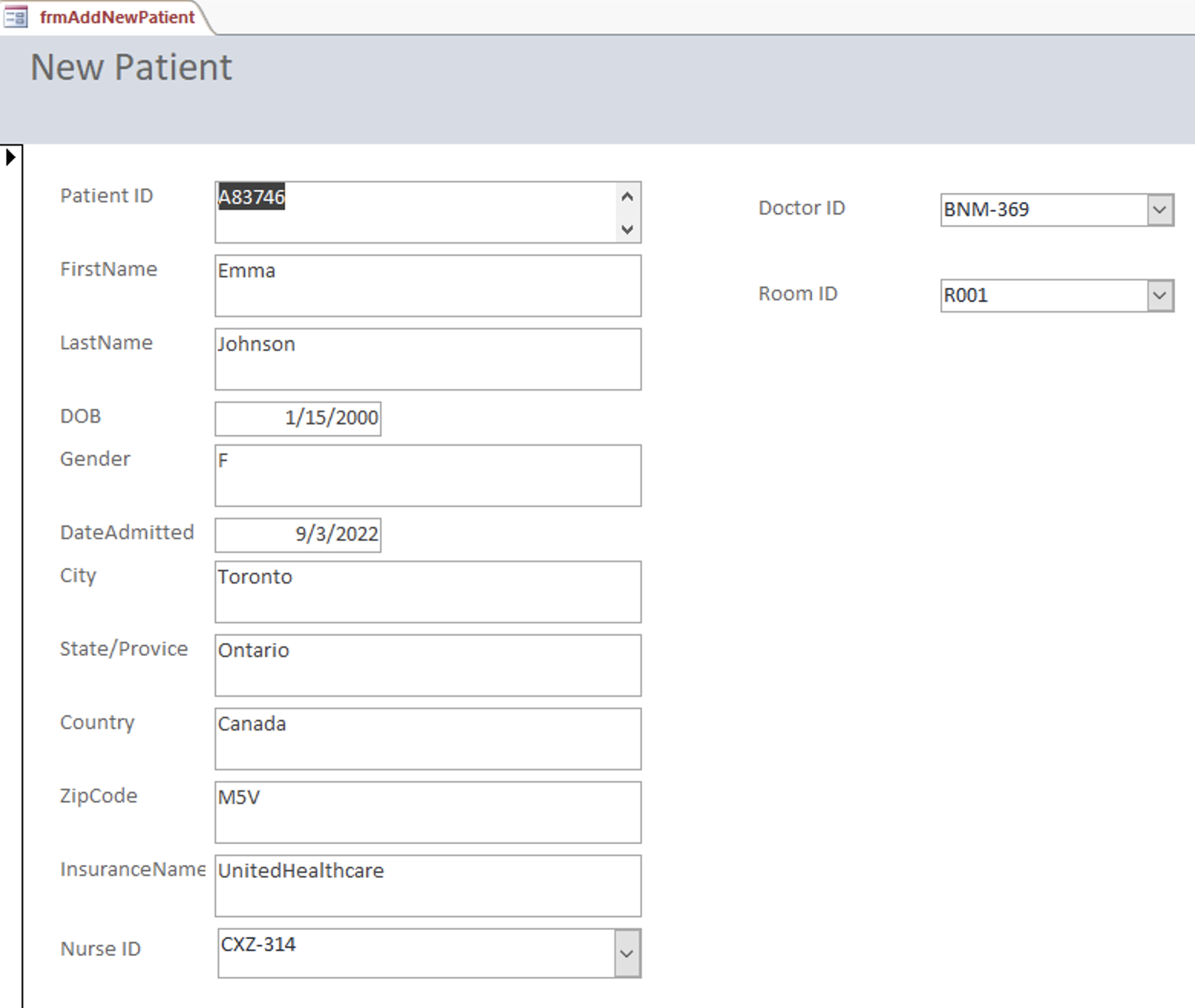


### Database Forms

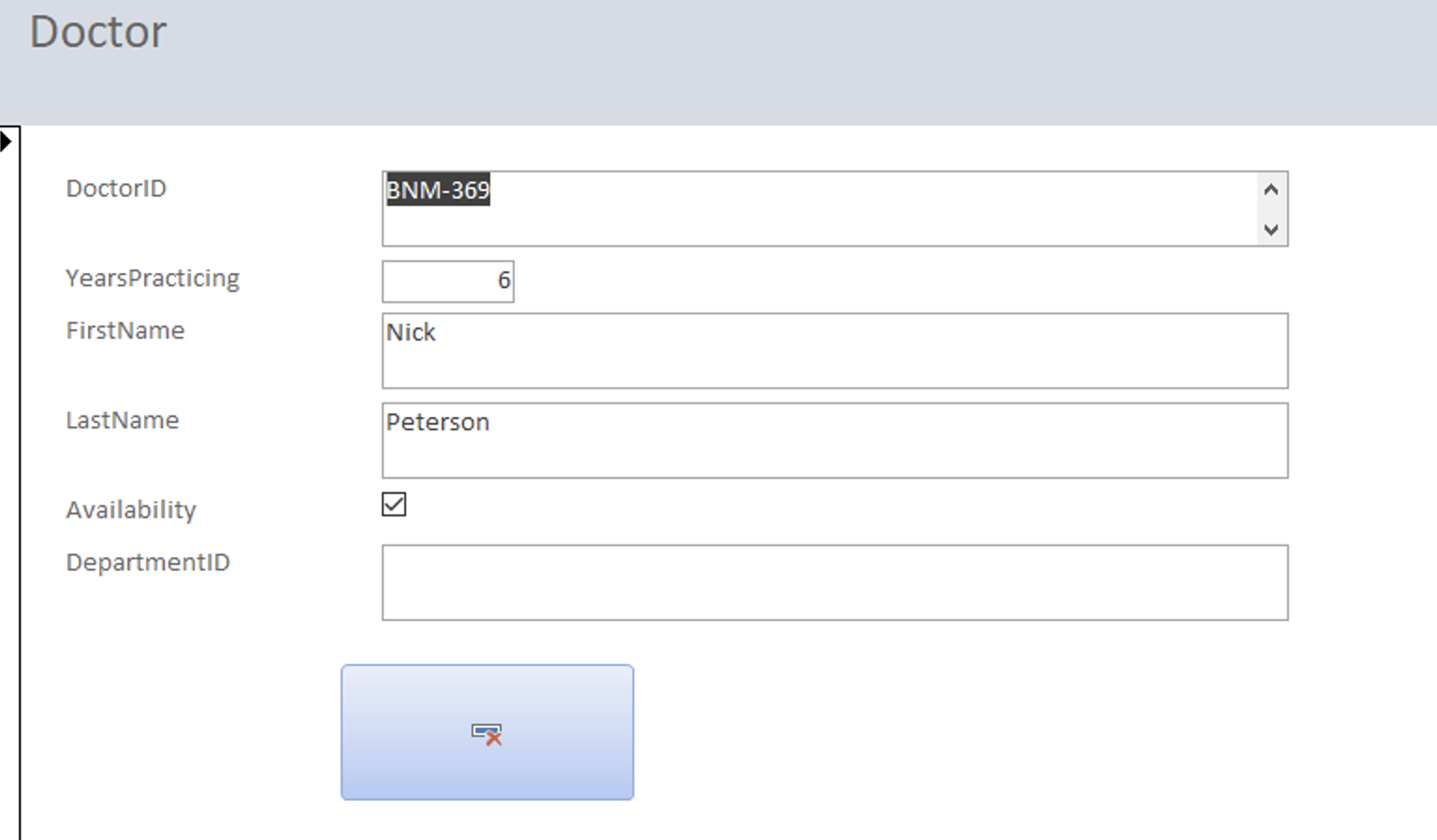
In addition to the switchboard, we have several forms within our Access database, each with its own specific purpose and accessible through our switchboard. Forms are an important component of a database as they provide users with a graphical interface to enter, view, and manipulate data. By using forms, users can interact with the database in a more user-friendly manner, and can quickly and easily add or modify data without having to directly manipulate tables. Forms can also help to ensure data accuracy by providing validation rules and data entry prompts. Overall, forms play a crucial role in enhancing the usability and functionality of a database.

Below are screenshots of the forms we created within our database:

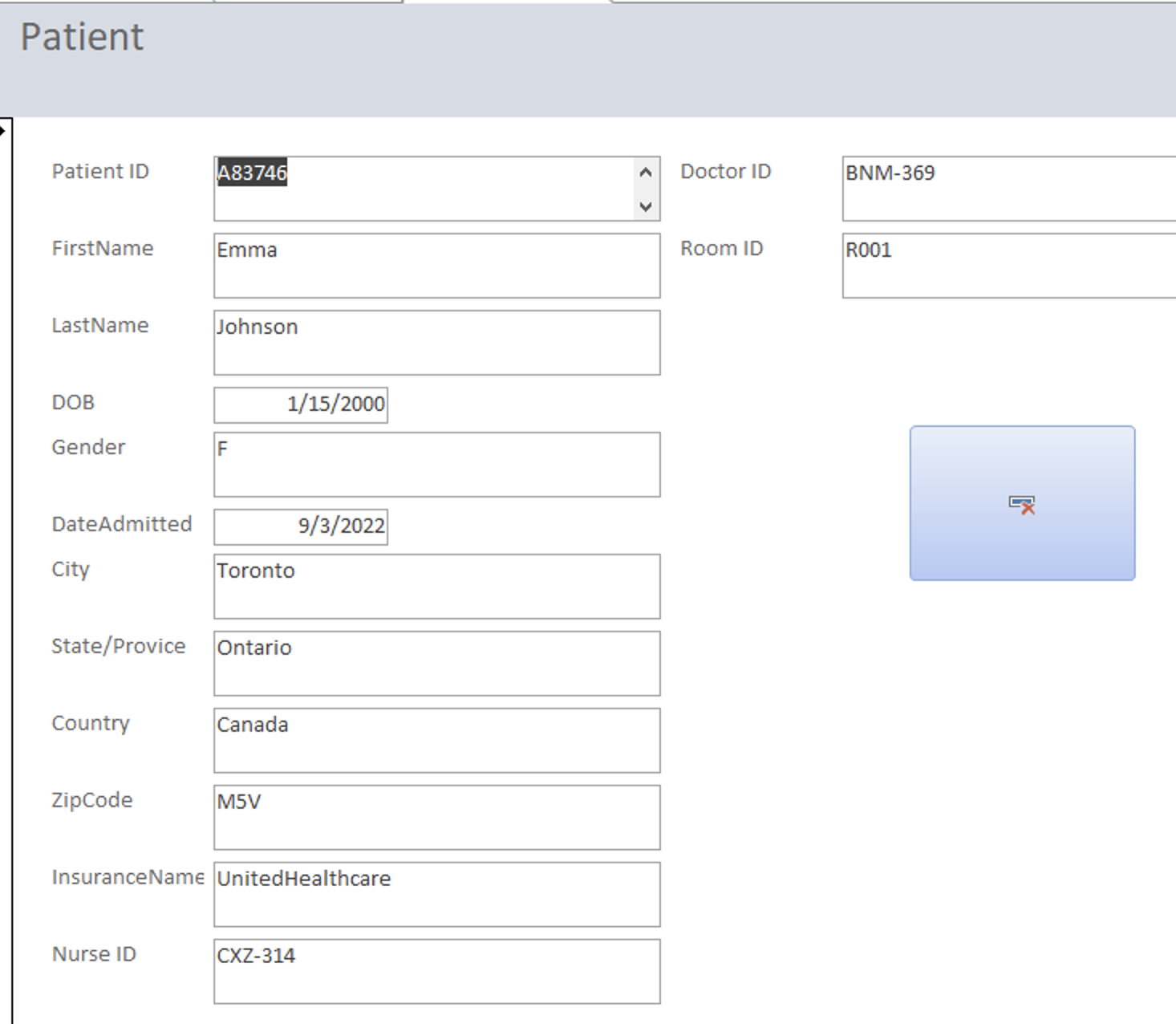
#### frmAddNewPatient



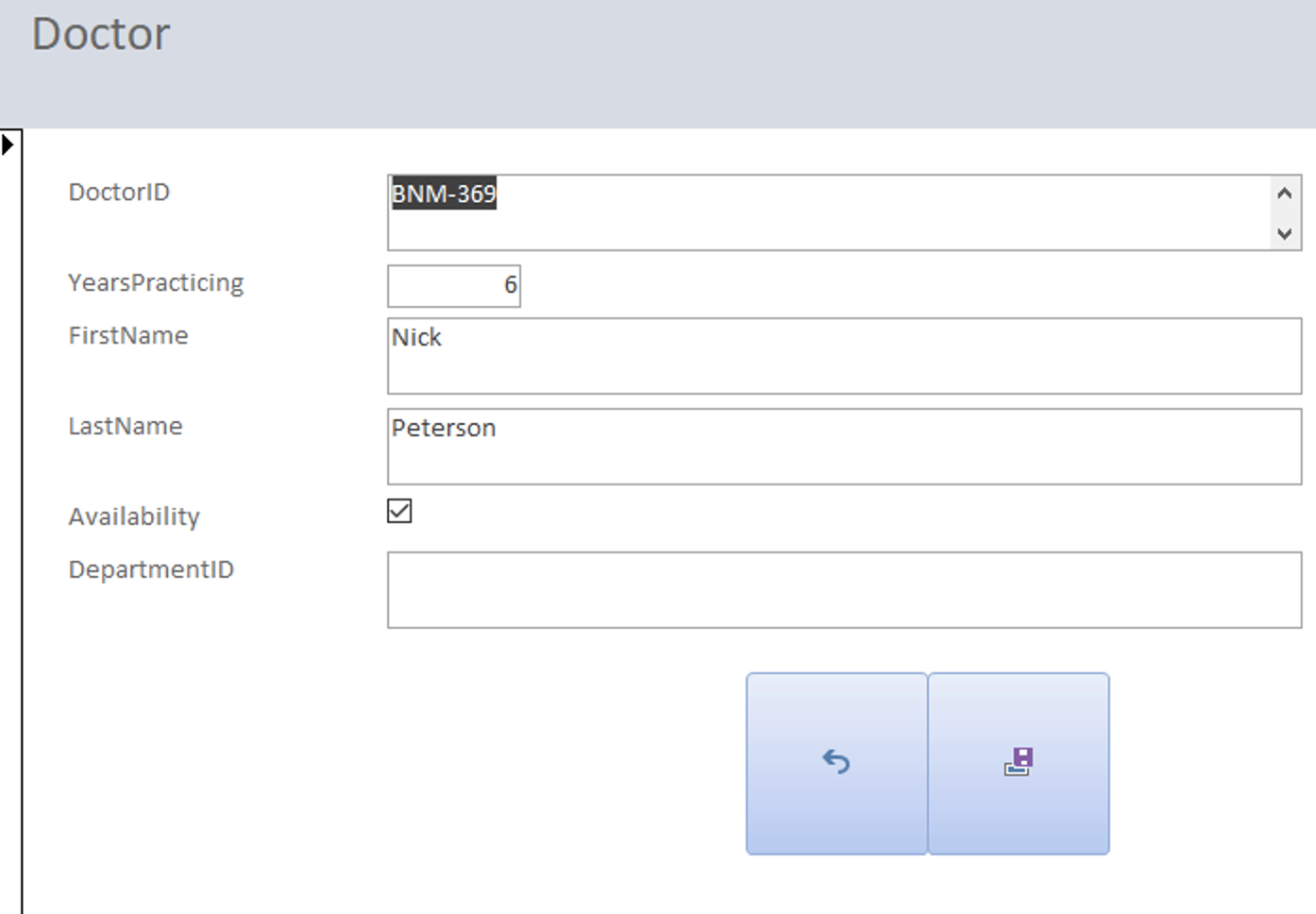
#### frmDeleteDoctor



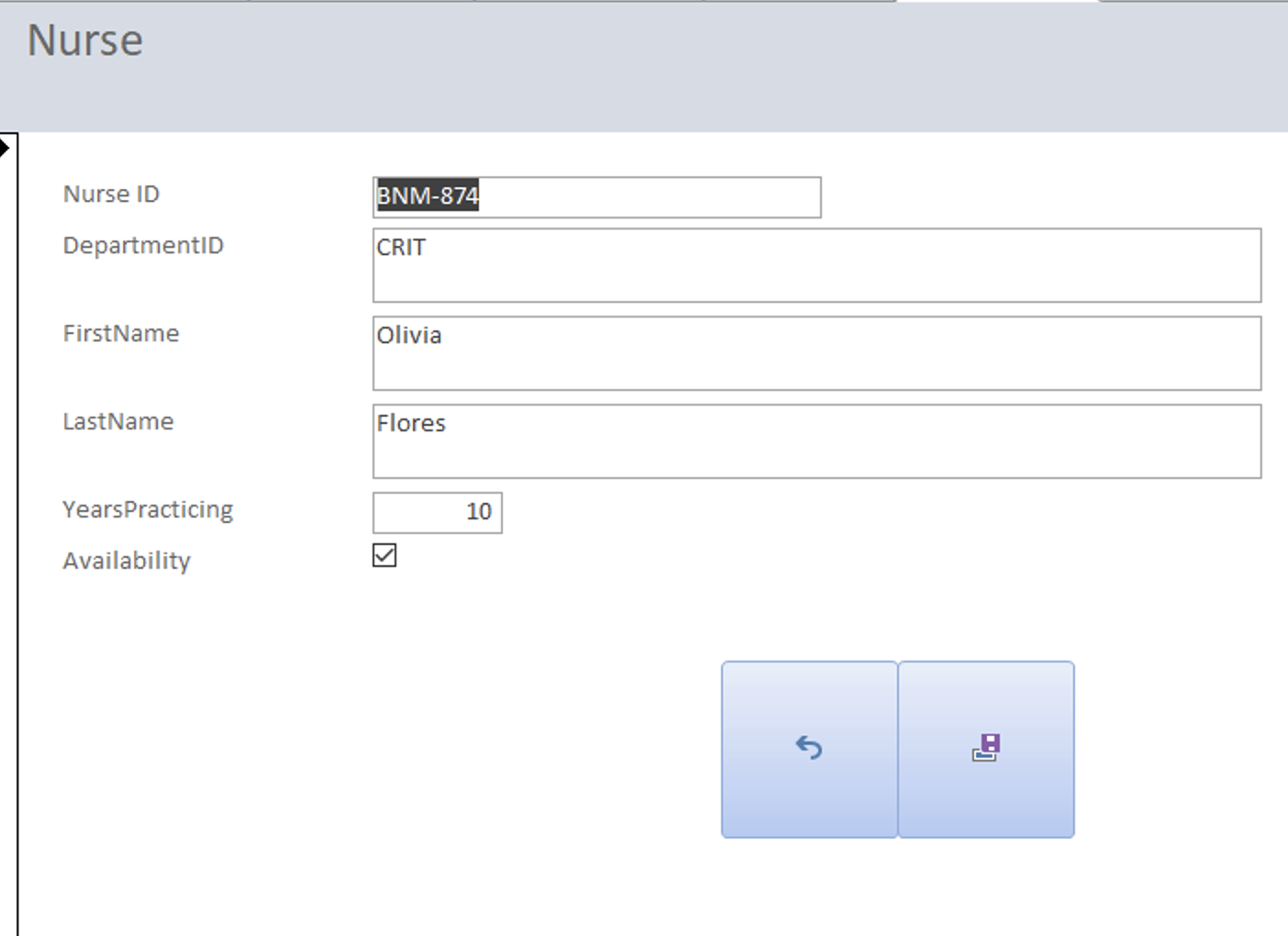
#### frmDeletePatient



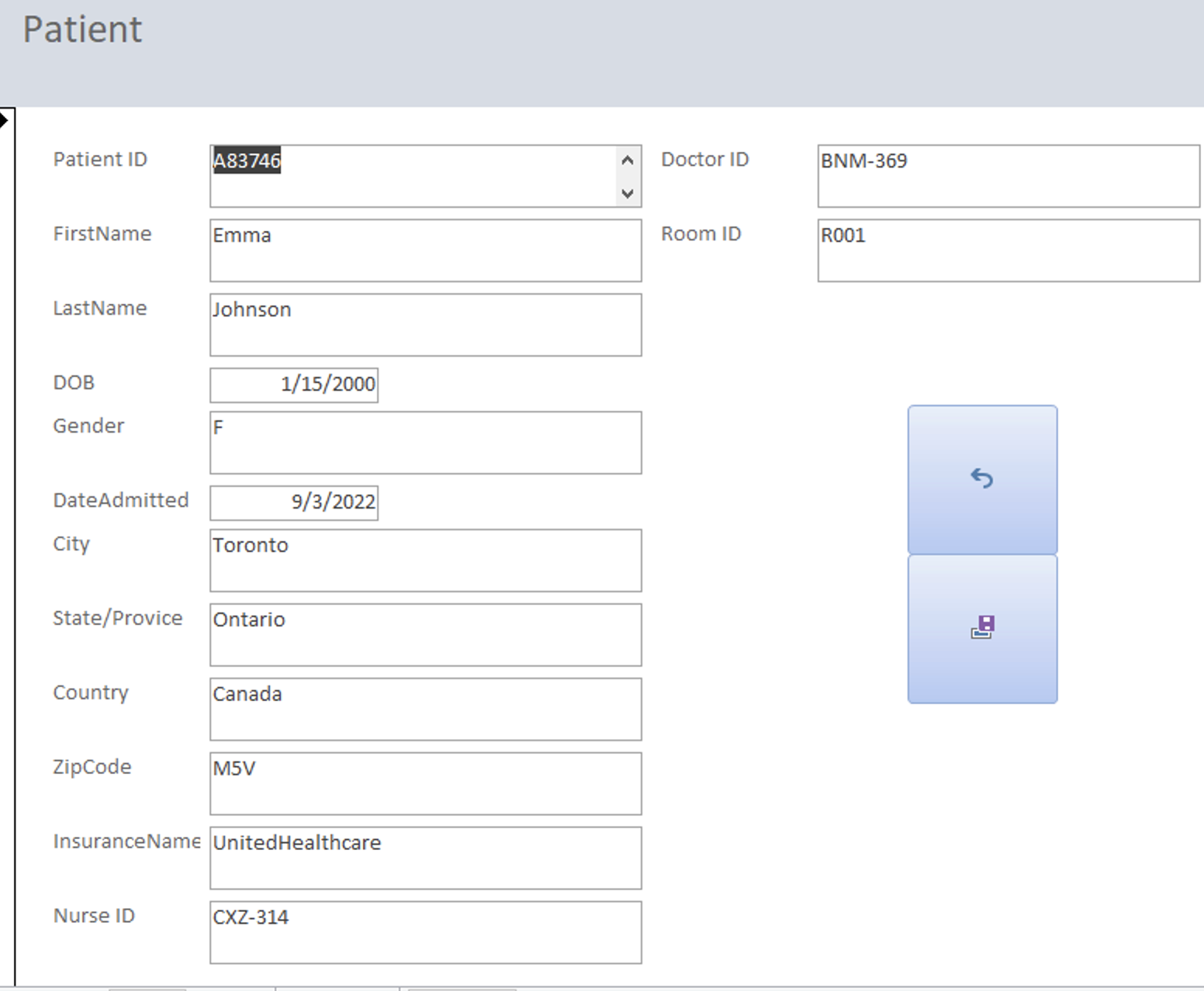
#### frmEditDoctor



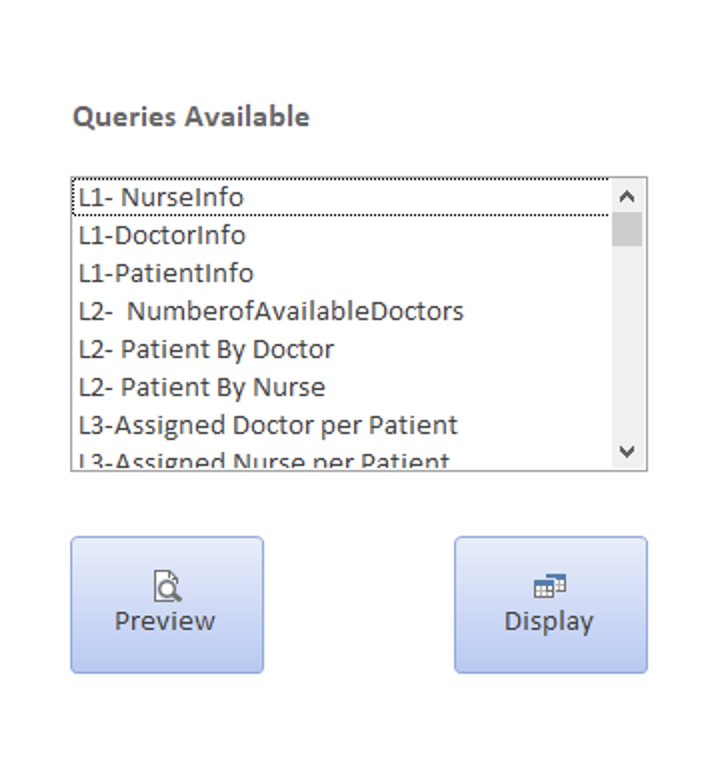
#### frmEditNurse



#### frmEditPatient



#### frmQueries

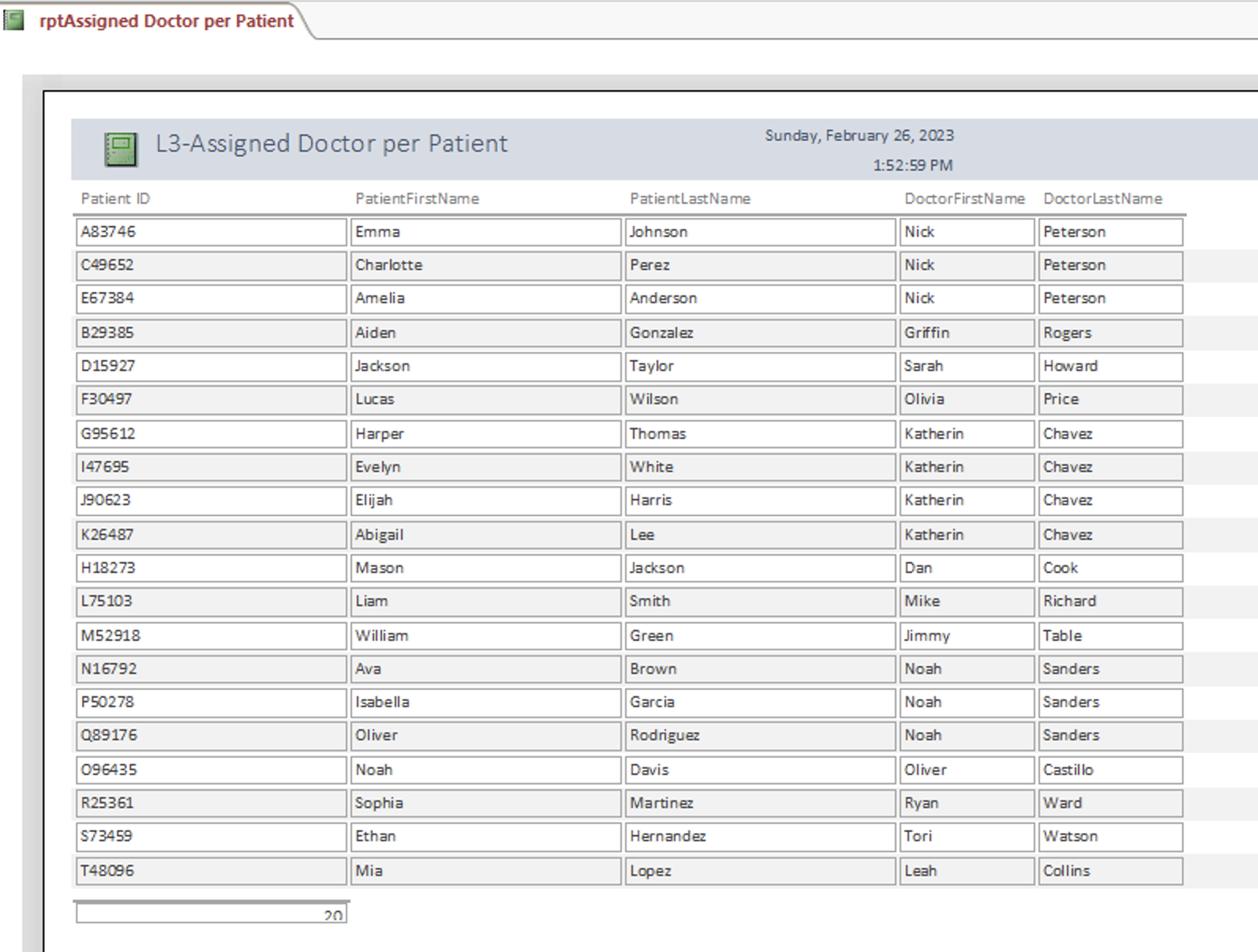


### Reports

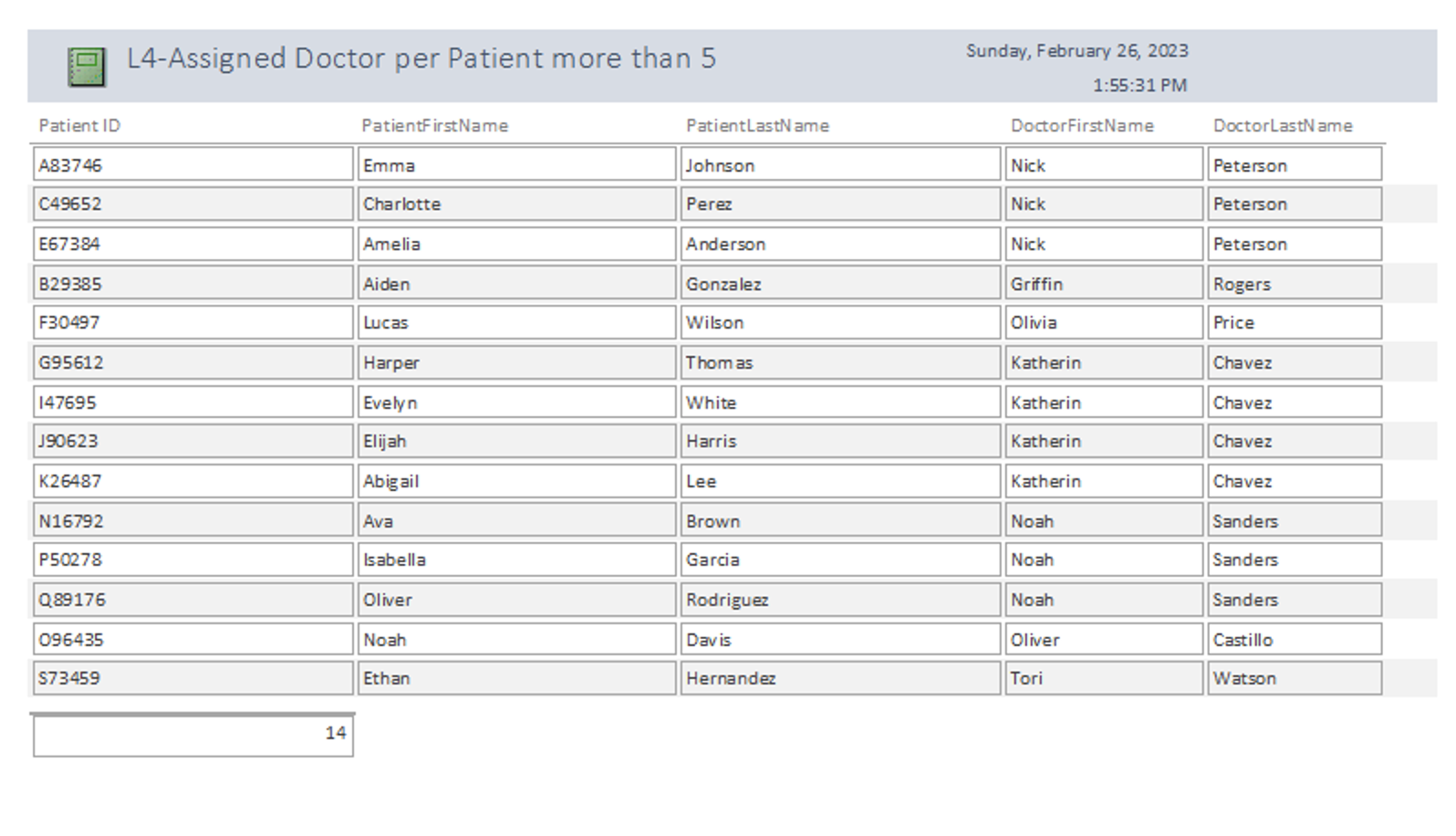
Reports are an important feature of a database as they allow users to quickly and easily analyze and summarize data in a meaningful way. With reports, users can generate customized summaries, calculations, and charts, which can be used to inform business decisions, track progress, or identify trends.

Below are screenshots of the reports we created within our database:

#### rptAssigned Doctor per Patient

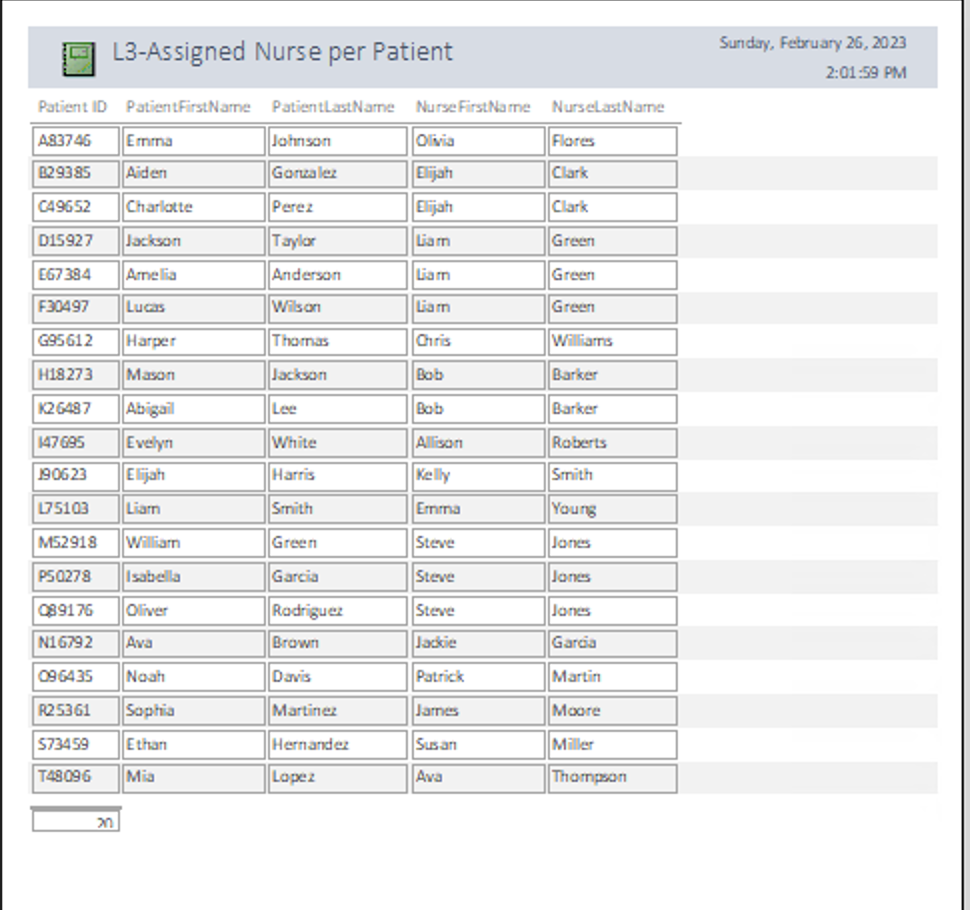


#### rptAssigned Doctor per Patient more than 5

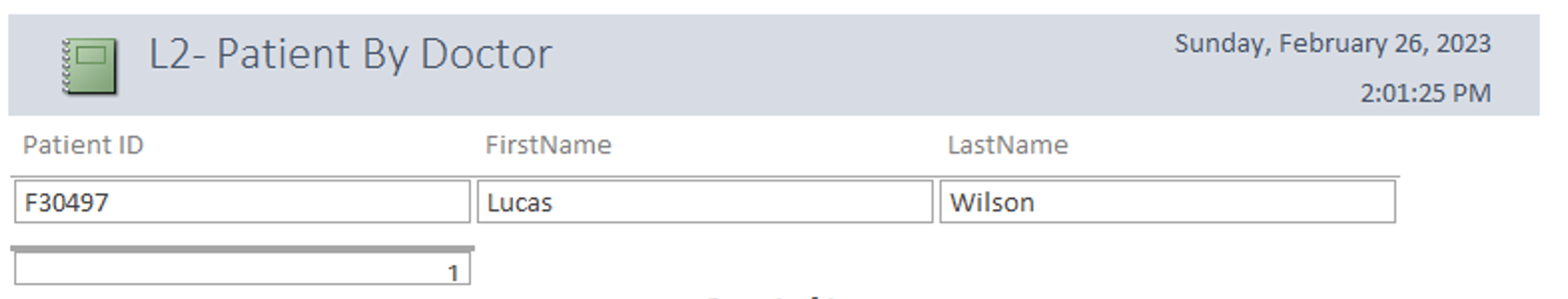


#### 

#### rptAssigned Nurse per Patient



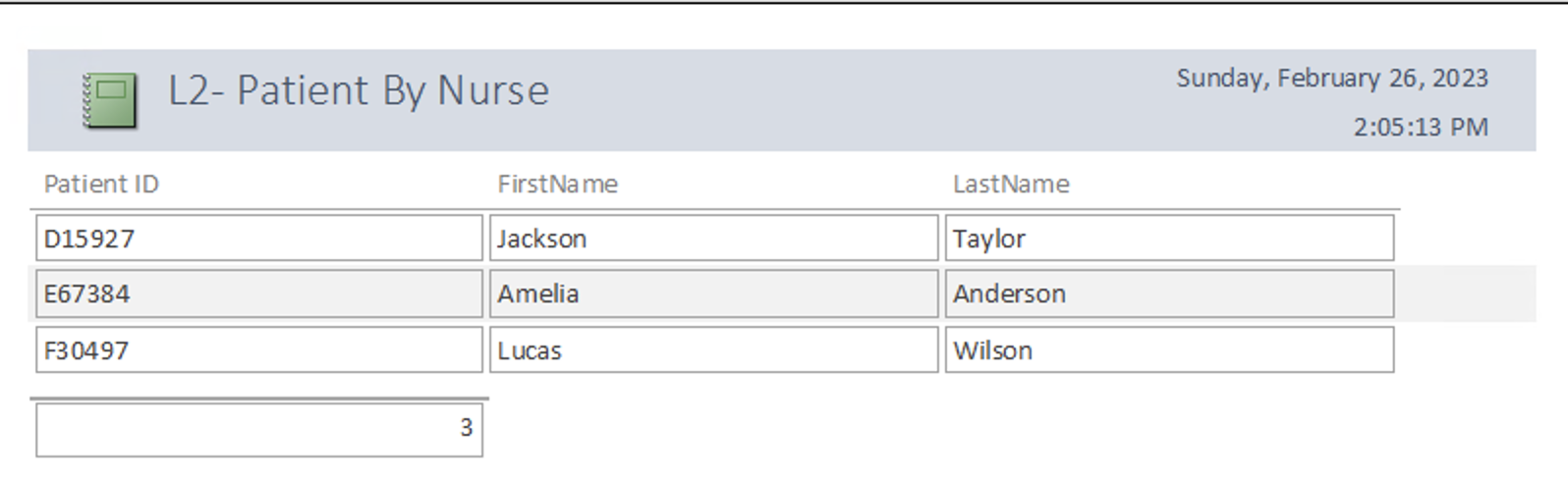
#### rptPatient by Doctor



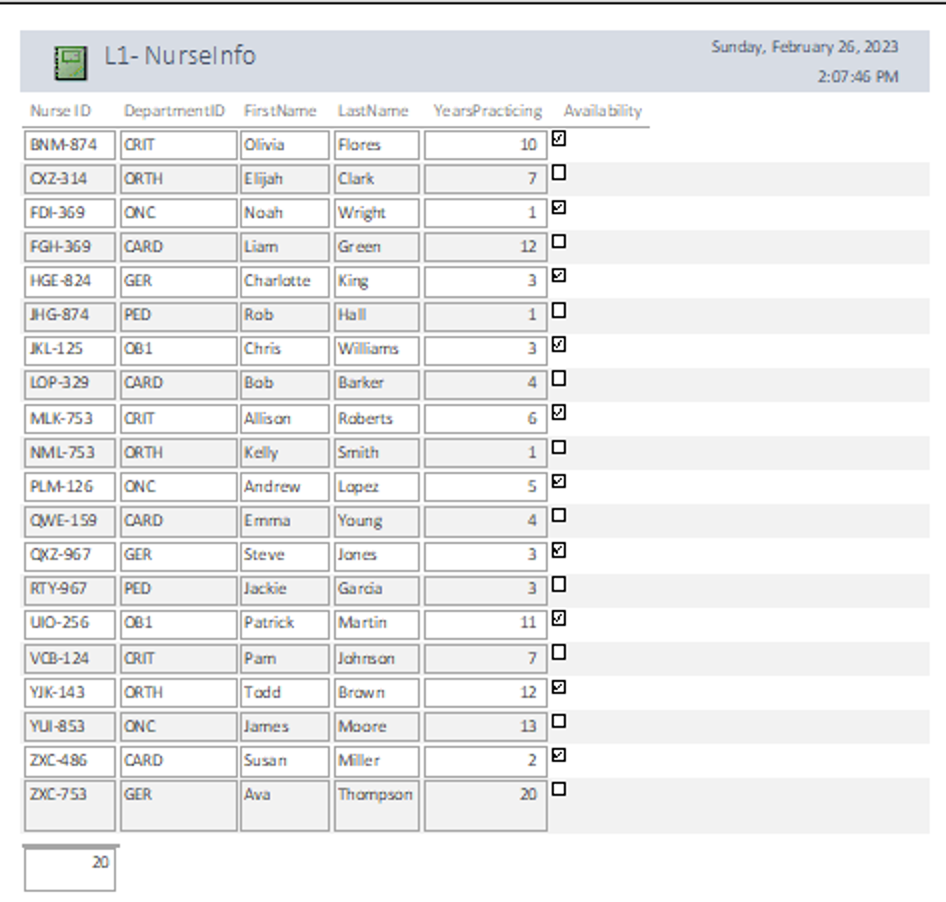
#### rptPatient Info



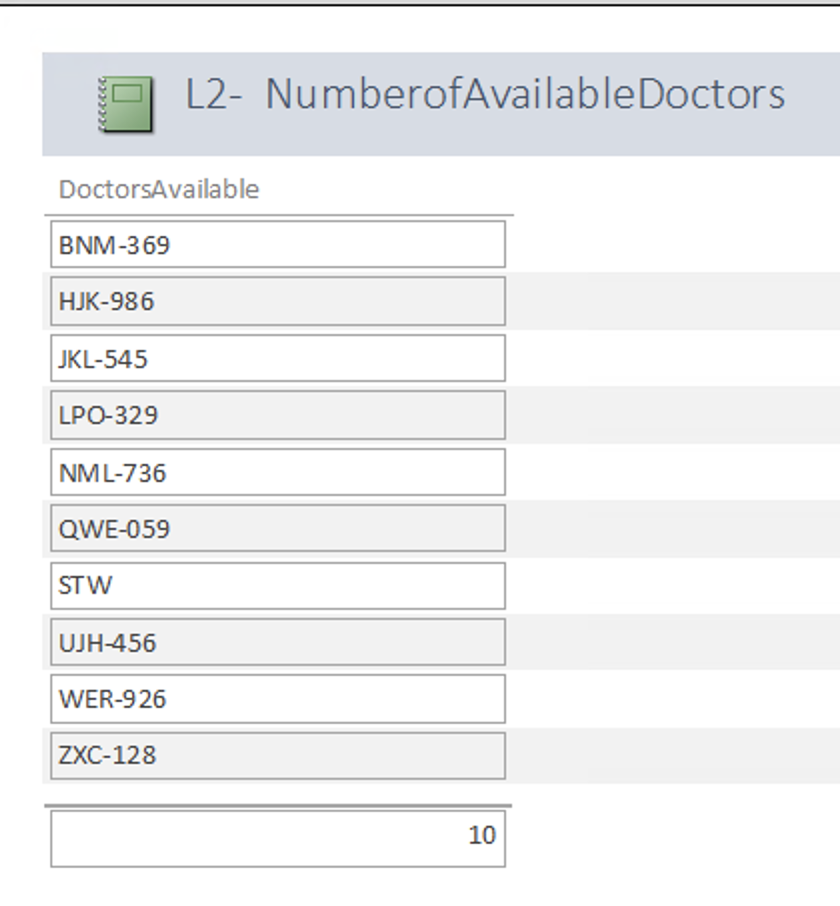
#### rptPatient by Nurse



#### rptNurse Info



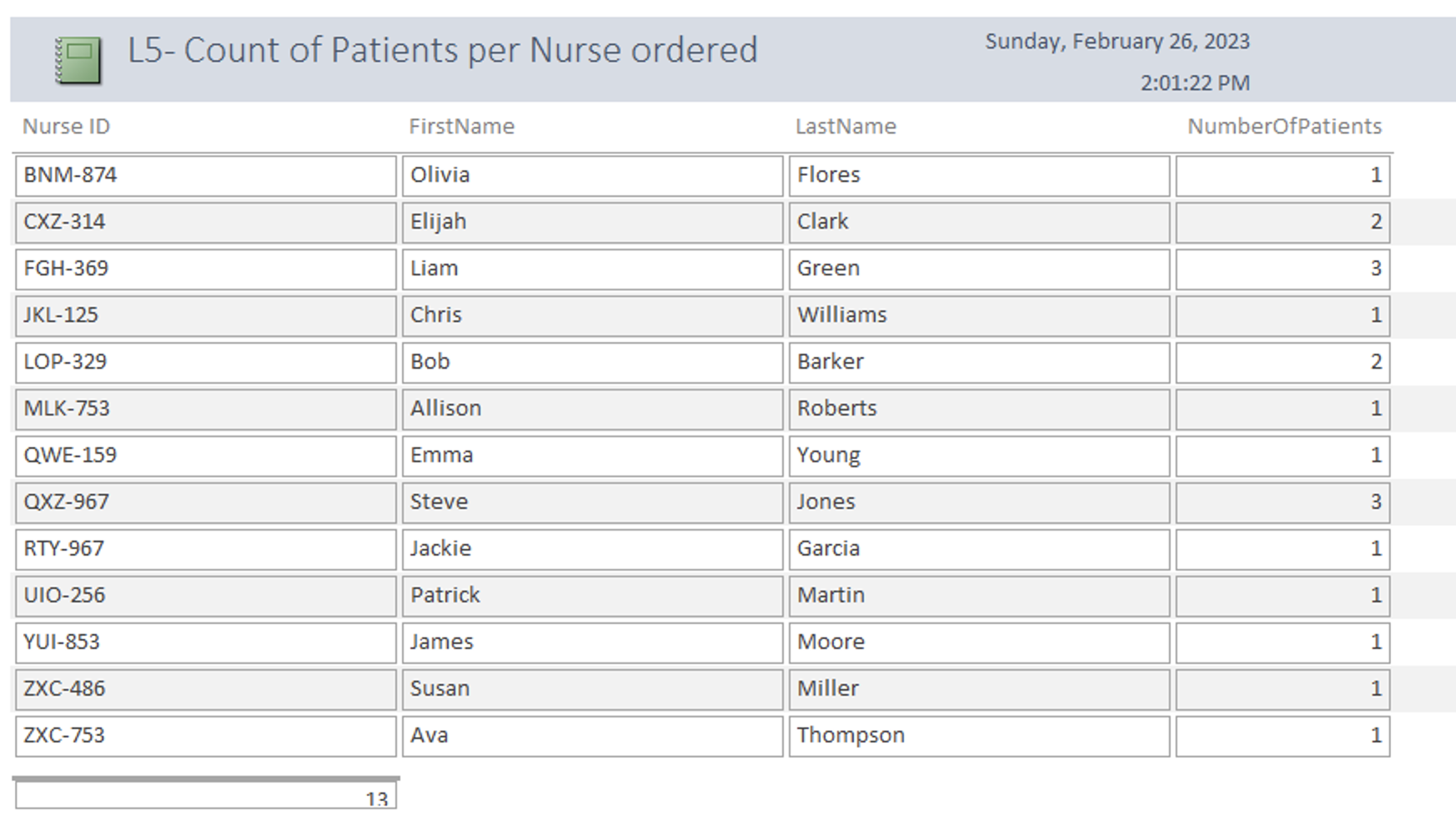
#### rptNumber of Available Doctors



#### rptDoctor Info

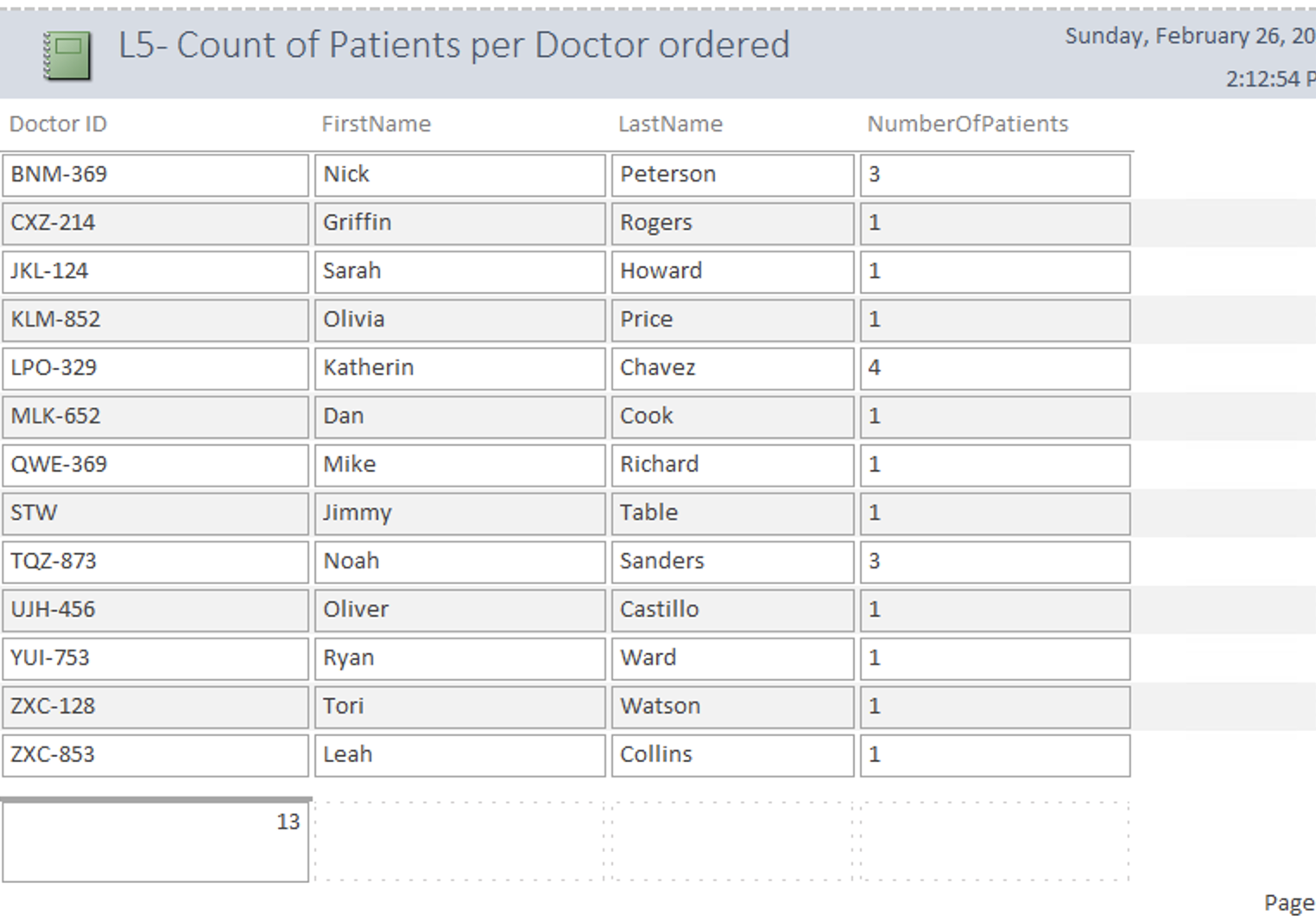


#### rptCount of Patients per Nurse Ordered



#### 

#### rptCount of Patients per Doctor Ordered



### Implementation Issues

During the implementation process of creating the database, several issues arose that required attention. First, while creating switchboards in our Access database, we encountered a significant challenge of ensuring that the switchboards were both meaningful to the users and properly operated. We discovered that creating a switchboard that was both user-friendly and efficient required careful consideration of the user's needs and behavior. During our first attempt at creating a switchboard in our database, we made the mistake of only providing a "view forms only" function. However, we soon realized that this approach was inadequate and made the switchboard unhelpful for users. This was because it limited the user's ability to interact with the database and perform necessary functions such as data entry, editing, and deleting. As a result, we had to redesign the switchboard to provide more comprehensive functionality that met the user's needs and expectations. Through a process of trial and error, we were able to create a switchboard that not only provided the necessary functionality but also improved the overall usability and user experience of our database.

We also faced challenges related to the proper use of foreign keys in our ERD and Conceptual Model. Initially, we made the mistake of including foreign keys where they were not necessary or appropriate, which resulted in significant data integrity issues. This led to problems such as redundant data, data inconsistencies, and difficulties in updating or deleting records. However, through careful analysis and reviewing provided feedback, we were able to identify and correct these errors in our database design, which improved the overall accuracy of our models. This experience highlighted the importance of proper planning and design in database development, as well as the need for ongoing testing and refinement to ensure the quality of the database.

Lastly, we encountered challenges in establishing appropriate relationships in our database due to our initial limited understanding of hospital operations. At first, we assumed that each patient would be assigned one nurse and one doctor. However, after learning about our domain further, we realized a patient could be treated by multiple doctors simultaneously

### Recommendations for ensuring data quality

To ensure data quality, it is recommended to establish data validation rules during the import process, which can help detect and correct any data issues. It is also essential to establish data quality standards and implement regular data quality checks to ensure that the data remains accurate and up-to-date. Additionally, it is important to provide user training and support to encourage good data entry practices.

We have also implemented several features in our DB switch that are designed to help ensure data quality.

In the Add sub-switchboard, the dropdown arrow allows the user to select the DoctorID, RoomID, and NurseID for a new patient. This feature ensures data quality by allowing the user to choose from existing records rather than manually inputting information that could potentially be incorrect or lead to duplicates. By selecting a predetermined ID from the dropdown, the user can be confident that they are accurately recording the data and avoiding any potential errors. This not only ensures the accuracy of the patient's information but also saves time by eliminating the need to search for or create new records for Doctors, Rooms, and Nurses.

In the Delete sub-switchboard, it is not possible to edit any individual information, but rather the entire form can be deleted. This is a deliberate design choice, as it mitigates the risk of accidental edits to critical information. By restricting the user's ability to make changes, the system ensures the integrity of the data and reduces the likelihood of errors or unintended modifications. This approach also adds an additional layer of security, preventing unauthorized changes to sensitive information. Overall, this design feature promotes data accuracy and helps to maintain the reliability of the system.

In addition to the aforementioned features, the form also includes an "Undo" button that allows the user to reverse any mistakes or unintended actions they may have taken. Having an "Undo" button is a useful feature because it allows users to quickly recover from any mistakes or accidental changes they may have made. This helps to reduce the risk of errors and also provides a sense of reassurance to the user that they can correct any issues that arise during their work.

Our "Save" button on each form is also important because it ensures that any changes made by the user are recorded in the system. This can be useful for maintaining accurate records and providing a clear audit trail of changes over time. Additionally, it helps to prevent accidental deletions or modifications, as users must consciously choose to save their changes before they take effect.

Overall, these features work together to promote data accuracy, prevent errors, and provide users with the tools they need to effectively manage and update their records.

### Space Estimate

The space required for our Access database will be influenced by several factors, such as the number of records, the size of the data fields, and the number of indexes used. Given that many US hospitals have over 10,000 patient beds and employ numerous doctors and nurses, we anticipate that our database will be relatively large. However, it is difficult to estimate the exact size of the database since each hospital is unique and may have varying amounts of data to store. As a result, we will need to carefully monitor the size of the database as it grows over time and implement appropriate strategies to optimize its performance and storage capacity.

### Choice of Indexes

Access uses indexes to speed up the process of finding and sorting records. When primary keys are defined, Access automatically creates indexes which are convenient. These indexes store the record locations based on the fields selected by the user. Indexing proved useful when searching for frequently accessed information such as patient names and doctors' availability. Instead of scanning through all records, indexing accelerates the process of retrieving data by locating records through the indexed fields.

### Backup and Recovery

Given that the data within our Access database contains critical information related to hospital patients and employees, it is imperative that we take steps to ensure its security and integrity. One of the most important measures we can take is to establish a regular backup schedule to ensure that the data is backed up frequently. This will help to minimize the risk of data loss in the event of system failures, human error, or other disasters. In addition, we will need to implement appropriate security measures to protect the database from unauthorized access or hacking attempts. This may include the use of firewalls, encryption, and access controls, as well as regular testing and updates to ensure that the database is secure. We recommend that backups be performed on a daily basis.