**CS669 Term Project – Health Version**

****

**Database Schema Design for**

**Multi-Specialty Ambulatory Health Center**

Submitted By: Archana Balachandran

Dec 14, 2016

**Table of Contents:**

1. Entities
2. Structural Business Rules
3. Conceptual ERD
4. Logical ERD
5. Use case execution screenshots
6. Index creation and explanation
7. **Entities**

A comprehensive list of entities are listed below. For limiting the scope of the term project, only the most relevant entities will be selected for depicting the complete BUAHC system.

* Facility
* Building
* Physicians
* Patients
* Insurance Plans
* Wait\_List
* Department
* Ambulance
* Medical Record
* Prescription
* Bill
* Slots
* Appointment
* Medical\_Exams

1. **Structural Business Rules: Relationships, Optionality, Plurality**

In this section, the relationships between the entities, the optionality and plurality of these relationships are explored.

Facility and Building have a one-to-many relationship. .(mandatory)

A Facility must have at least one Building. A building belongs to only one Facility.

Building and Department share a one-to-many relationship. (optional)

A building may belong to one or more departments (Optional relationship). A department belongs to one Building.

Facility and Medical Records share a one-to-many relationship.(mandatory)

A facility must contain at least one medical record. A medical record belongs to one Facility.

Facility and Insurance plans share a many-to-many relationship. (optional)

A Facility may be covered by one or multiple insurance plans. An insurance plan may cover multiple facilities. (Optional relationships)

Insurance plan and Patient share a many-to-many relationship. (optional)

An insurance plan convers at least one patient. A patient may be enrolled in one or multiple insurance plans.

Patient and Appointment share a one-to-many relationship. (mandatory)

A patient books at least one appointment. An appointment is assigned to only one patient.

Appointment and Phycisians share a many-to-one relationship. (optional)

A physician may have multiple appointments. An appointment is assigned to only one Physician.

Department and Physician has a one-to-many relationship. (mandatory)

Department has at least one physician. A physician belongs to only one building.

Physician and Wait\_list has a one-to-one relationship. (mandatory)

A physician can have only one wait list and a wait list belongs to one physician.

Appointment and Prescription has a one-to-one relationship(optional)

An appointment may result in at most one prescription. A prescription belongs to one appointment.

Appointment and Medical\_exam share a one-to-many relationship(optional)

An appointment may result in multiple medical\_exams(optional). A medical exam belongs t only one Appointment.

Appointment and Bill has a one-to-one optional relationship. .(mandatory)

An appointment generates a bill. A bill belongs to one appointment.

Patient and Medical\_records share a one-to-many relationship.(mandatory)

A patient has at least one medical record with a doctor. A medical record belongs to one-patient.

A **patient** can have only one insurance plan

An **insurance plan** can have many patients enrolled.

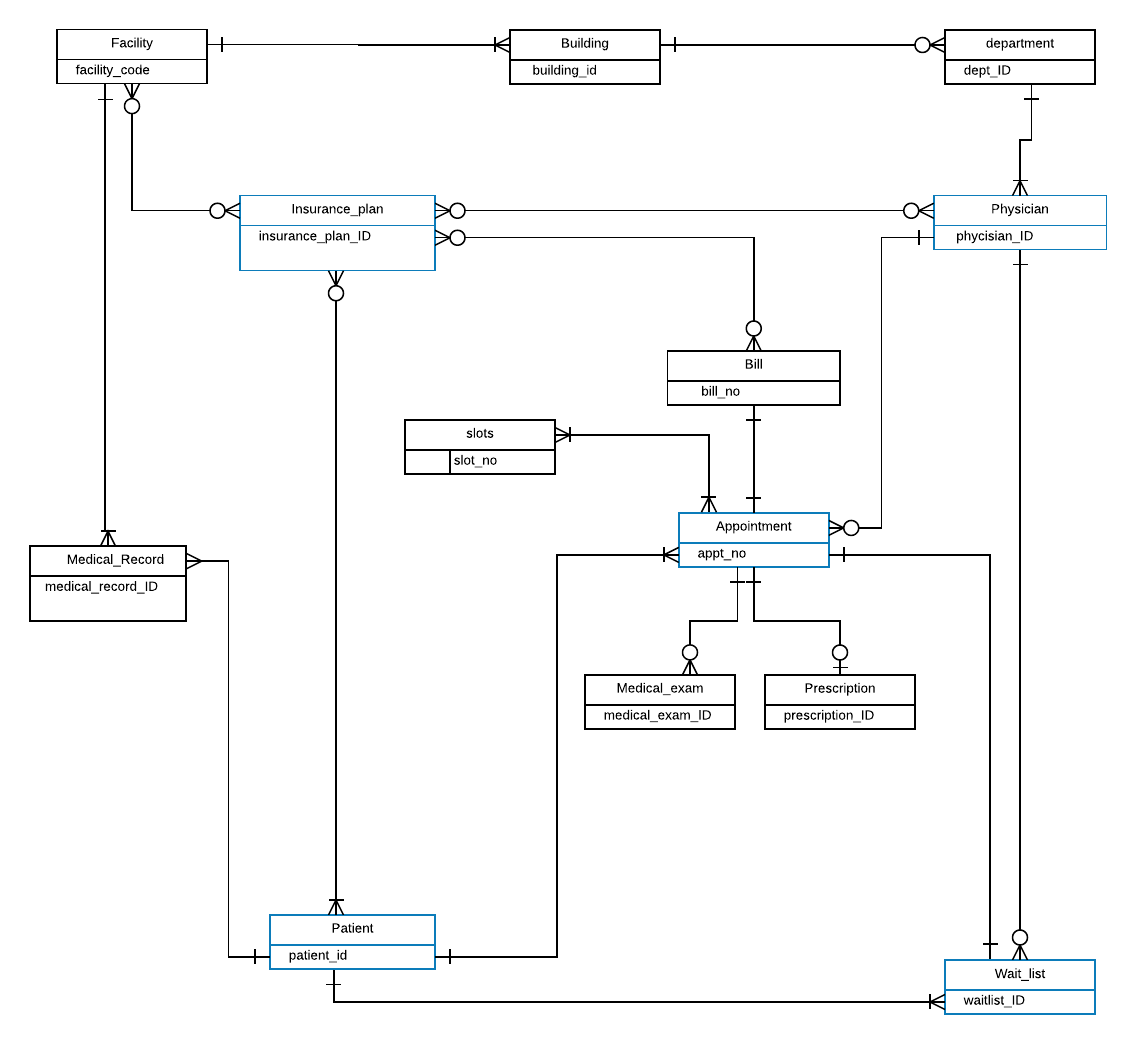
Each appointment can create one waiting\_list

Each waiting\_list entry belongs to one appointment.

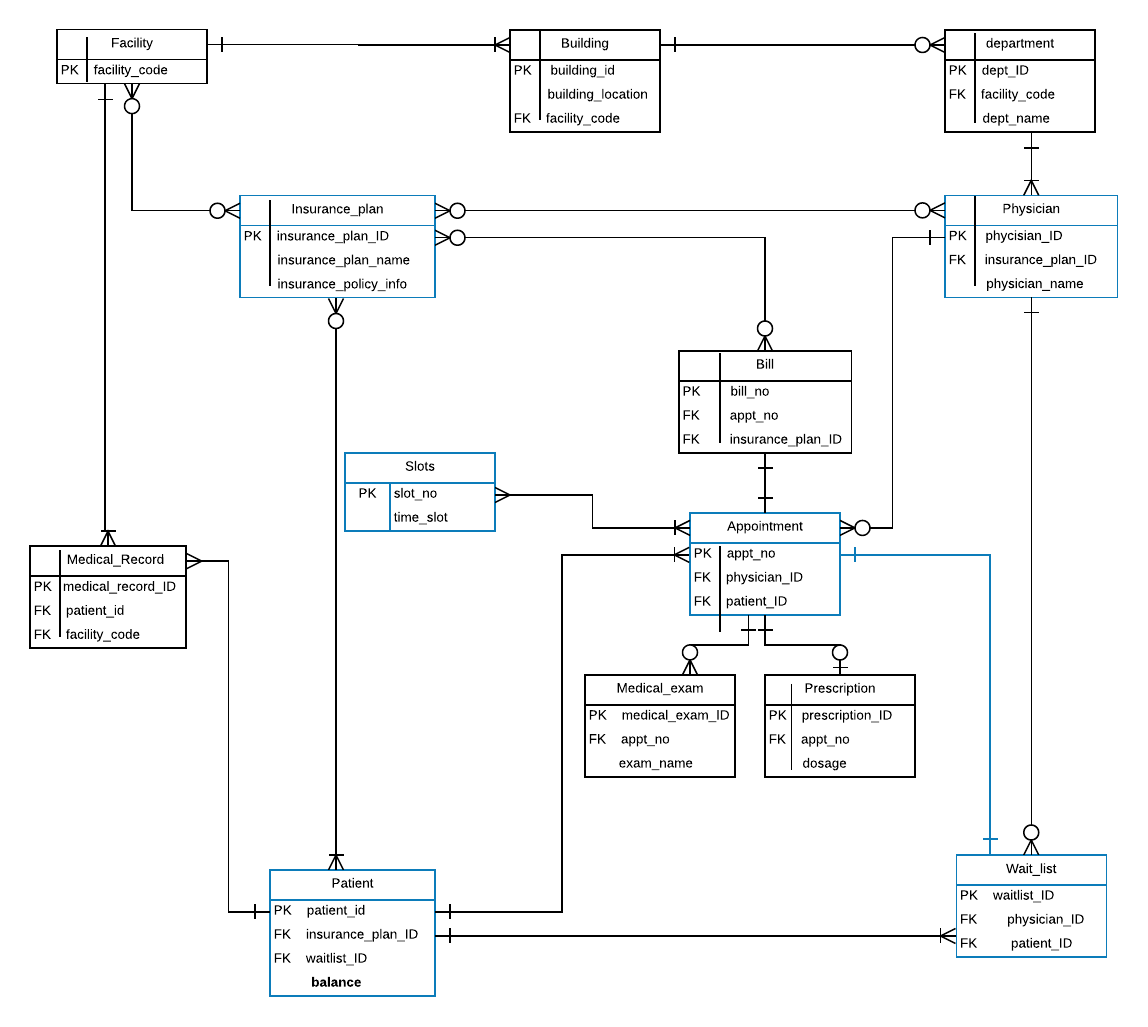
Each appointment has one slot.

Each slot may be assigned to multiple assignments.

1. **Conceptual Entity Relationship Diagram**



1. **Logical Entity Relationship Diagram**



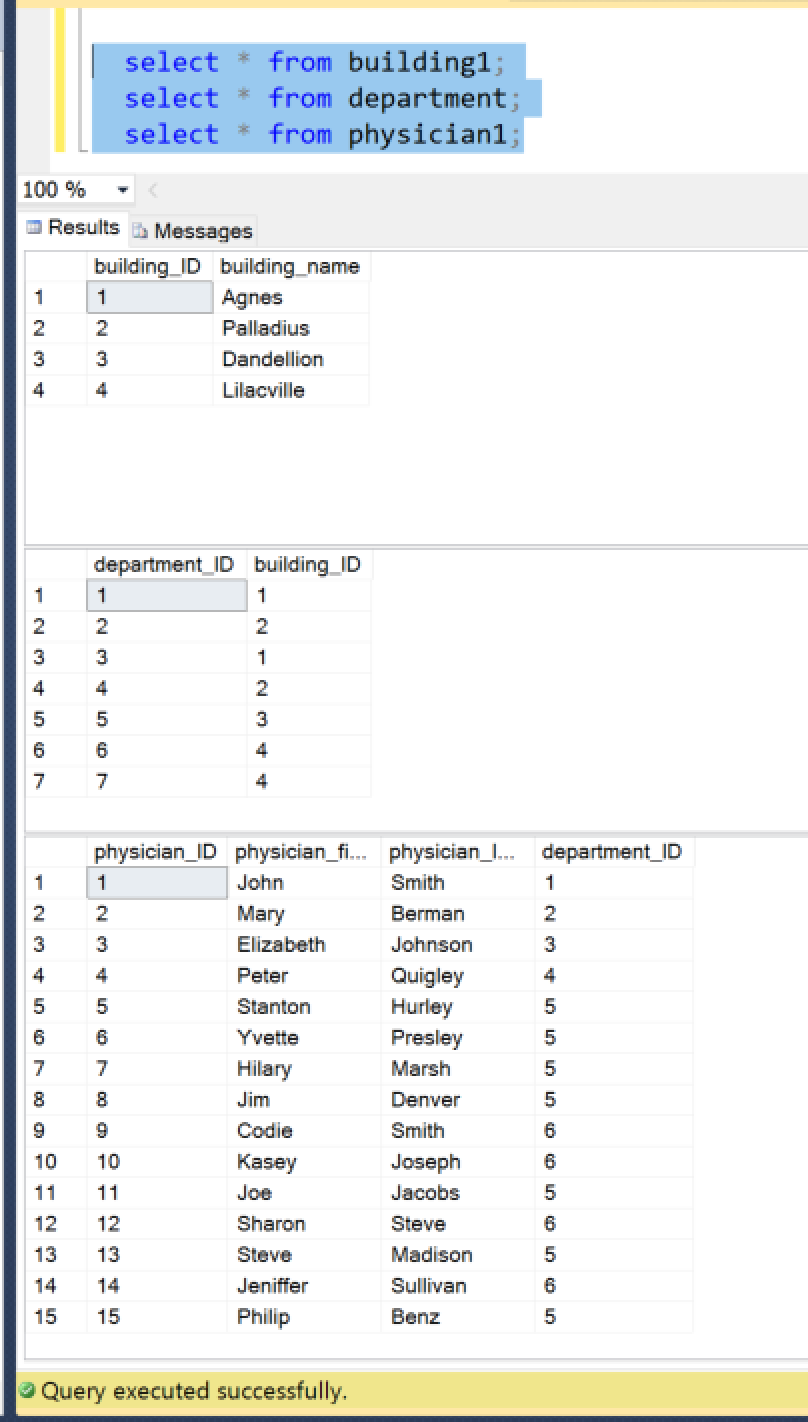
1. **Use case Execution**

**Tables used to satisfy each Use case:**

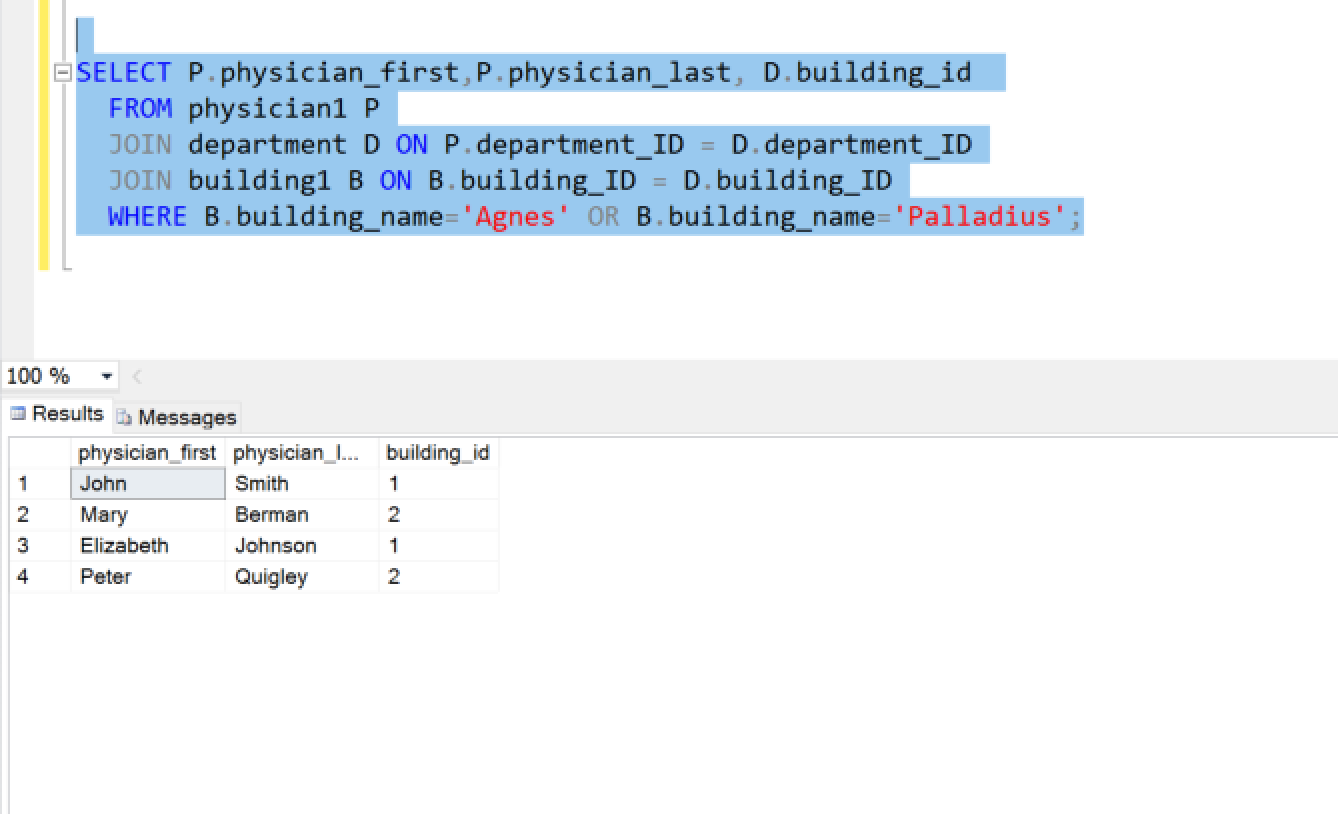
|  |  |  |
| --- | --- | --- |
| **Use case** | **Table** | **Columns** |
| #1 | Building,  physician | Building\_ID, building\_name  Physician\_first,physician\_last |
| #2 | Insurance,  patient | Insurance\_ID, insurance\_name  Insurance\_ID, patient\_first, patient\_last |
| #3 | Appointment  waitlist | Waitlist\_ID, physician\_ID, patient\_ID |
| #4 | Waitlist  Appointment  physician | waitlist\_ID, patient\_ID, physician\_ID  appointment\_ID, patient\_ID, physician\_ID, waitlist\_ID  physician\_ID, physician\_first, physician\_last |
| #5 | patient | balance, patient\_ID |
| #6 | patient | patient\_ID, insurance\_ID |
| #7 | Physician  Appointment | Physician\_ID, physician\_first, physician\_last, max\_capacity  Appt\_date, physician\_ID |
| #8 | Insurance  Patient | Insurance\_ID, insurance\_name, copay\_amt, number\_of\_enrollees  Insurance\_ID |
| #9 | Patient  Physician  Appointment | Patient\_first,patient\_last,physician\_ID  Physician\_first, physician\_last,physician\_ID  Physician\_ID,patient\_ID |
| #10 | appointment | Physician\_ID,patient\_ID |

**Use Case #1**

**Requirement**: Health center management requests the first and last names of all **physicians** that work in the ”Agnes” or “Palladius” **buildings**. Write a single query that retrieves this information for management.  The table structures for building, department and physician:



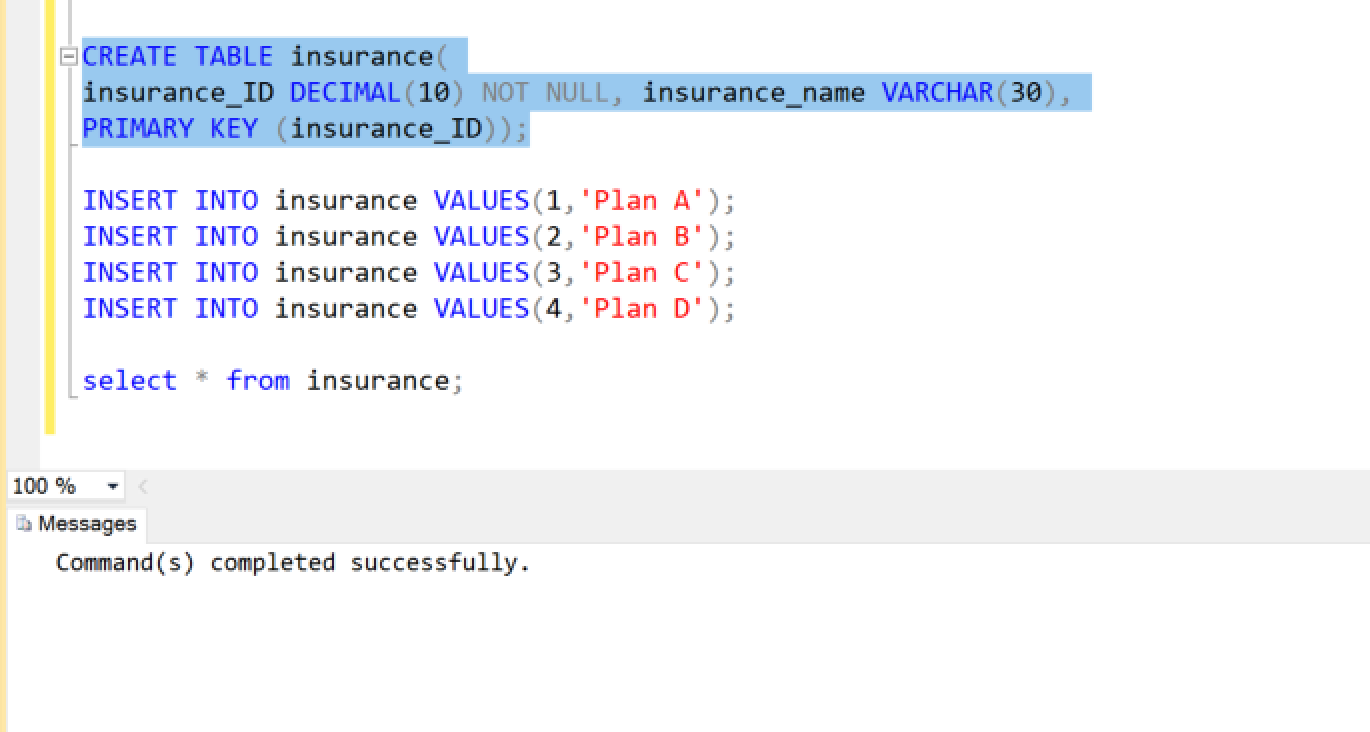
**Satisfying Use case 1:**

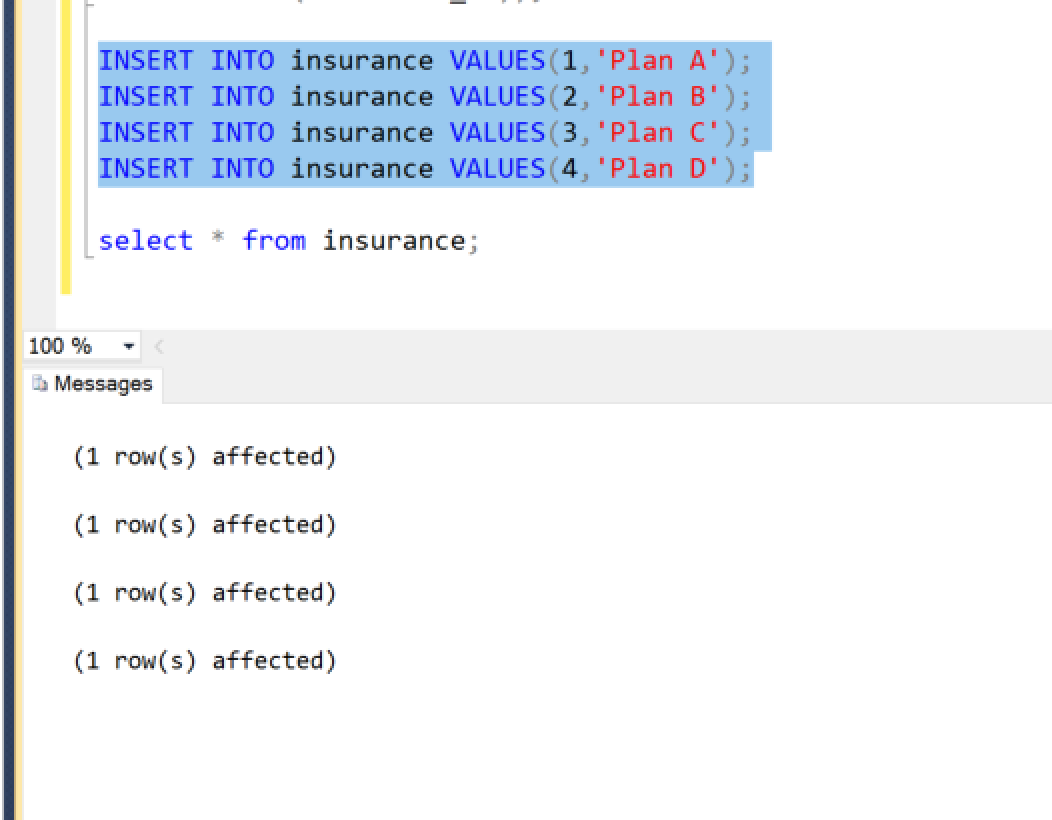


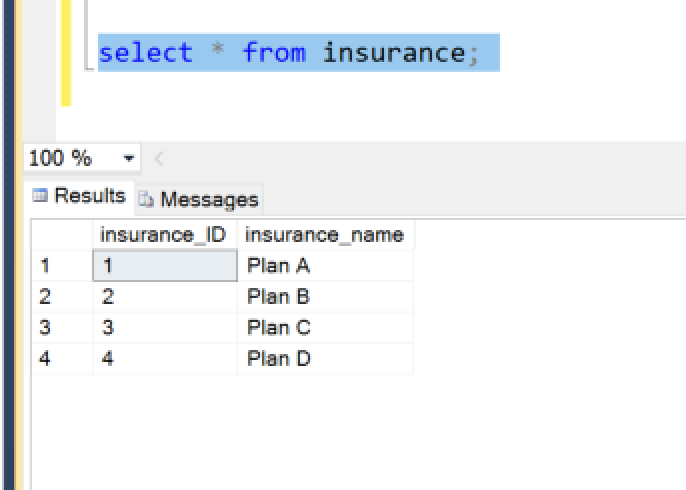
**Use Case # 2**

**Requirement**: Auditors request the names of all patients that currently have insurance, as well as the name of their current insurance plan. Write a single query that retrieves this information for the auditors.

Tables Patient,Insurance\_plans – check whether there is an insurance\_id NOT null, if so, for those NOT NULL values, refer insurance\_plans and list the insurance\_plan\_name. insurance\_id connects the tables.

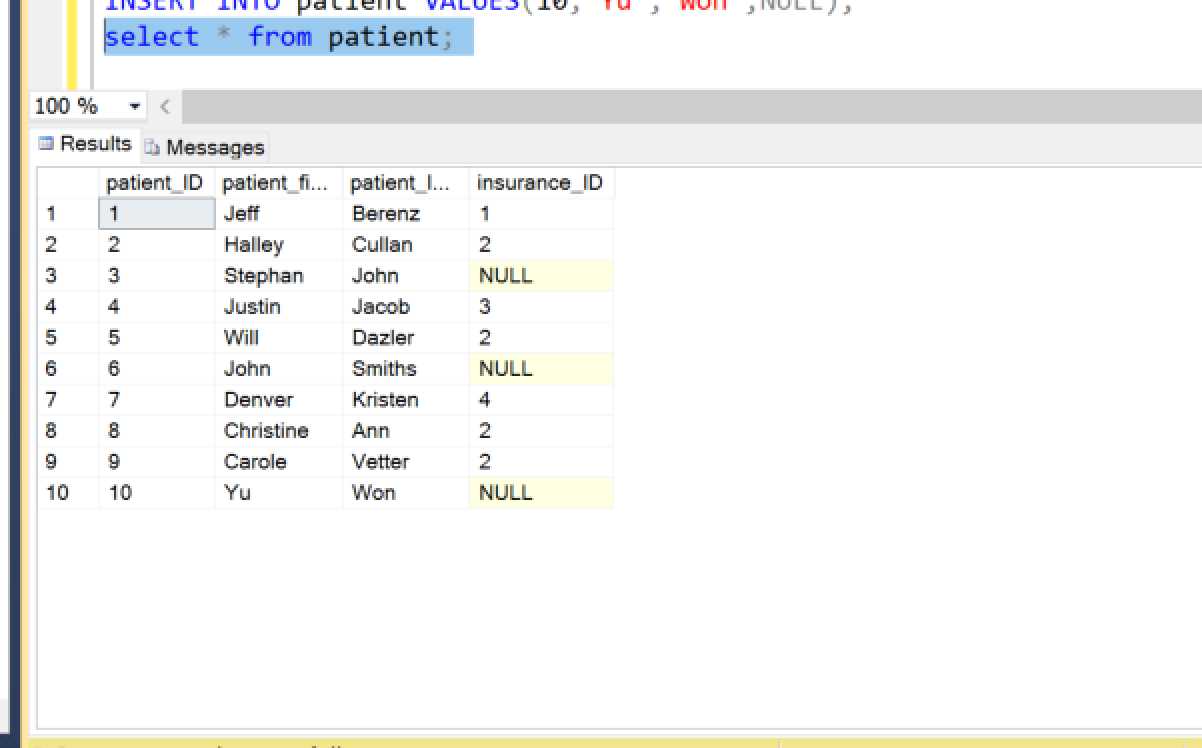




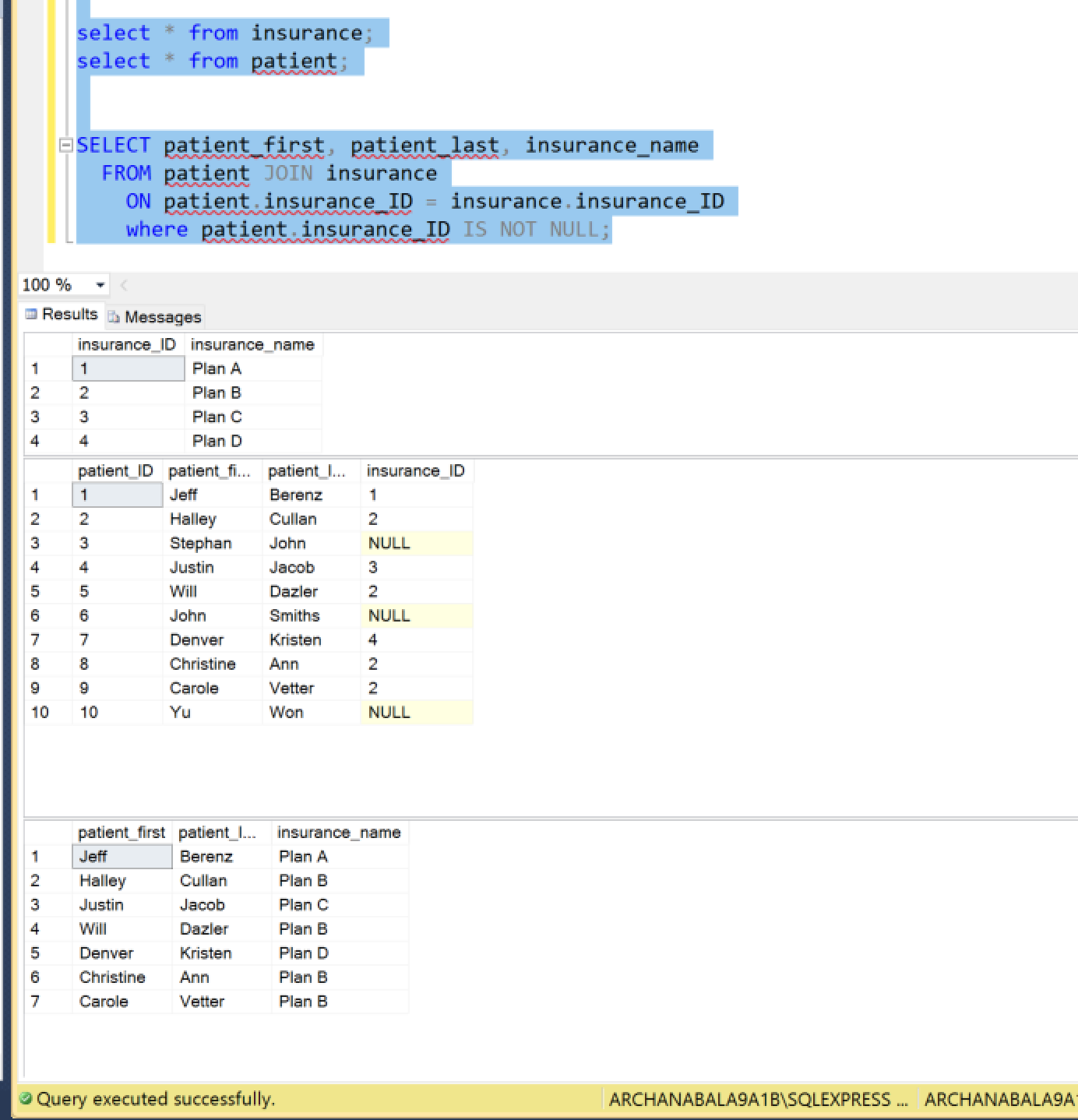






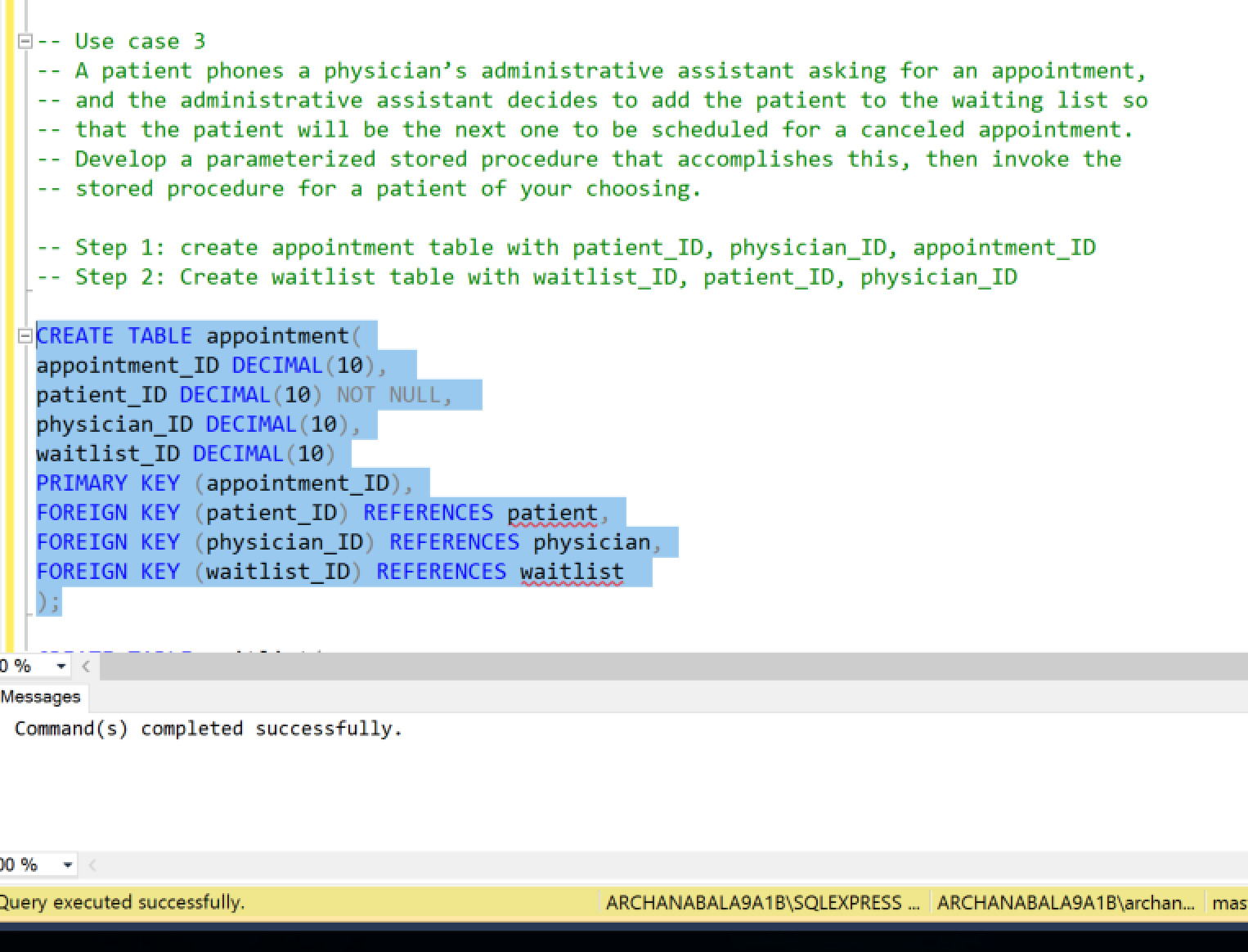


**Satisfying Use case 2:**

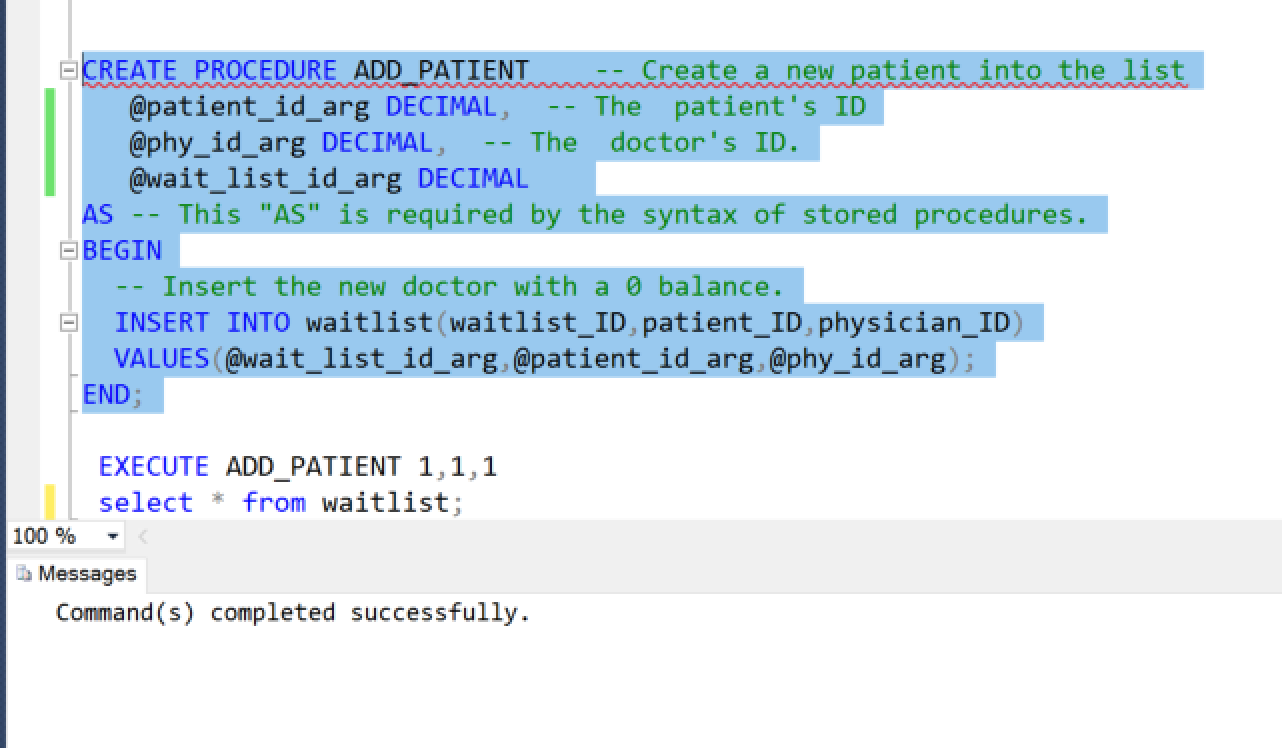


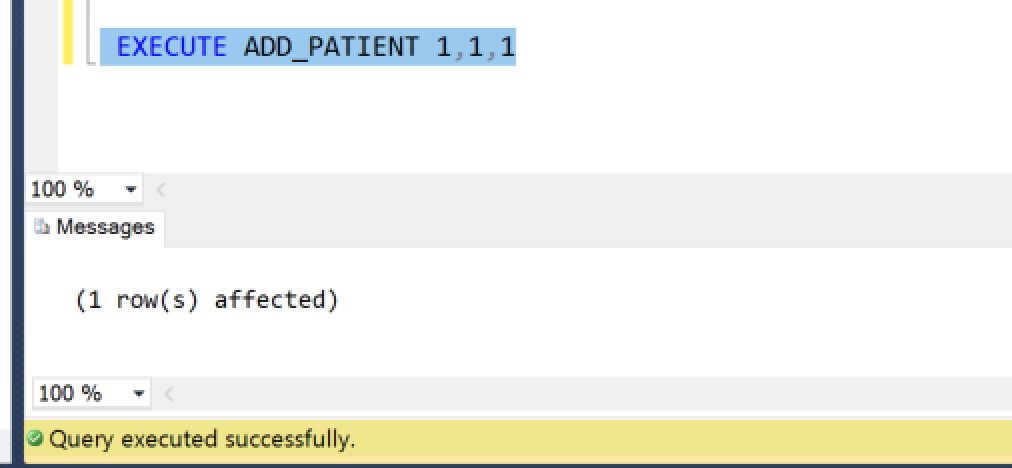
**Use Case #3**

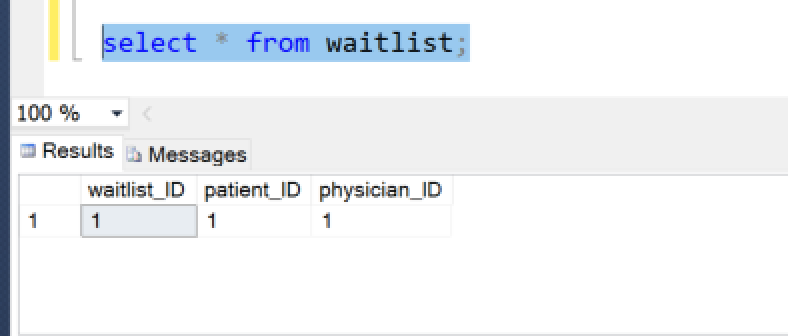
**Requirement**: A patient phones a physician’s administrative assistant asking for an appointment, and the administrative assistant decides to add the patient to the waiting list so that the patient will be the next one to be scheduled for a canceled appointment. Develop a parameterized stored procedure that accomplishes this, then invoke the stored procedure for a patient of your choosing.



**Satisfying Use case 3:**



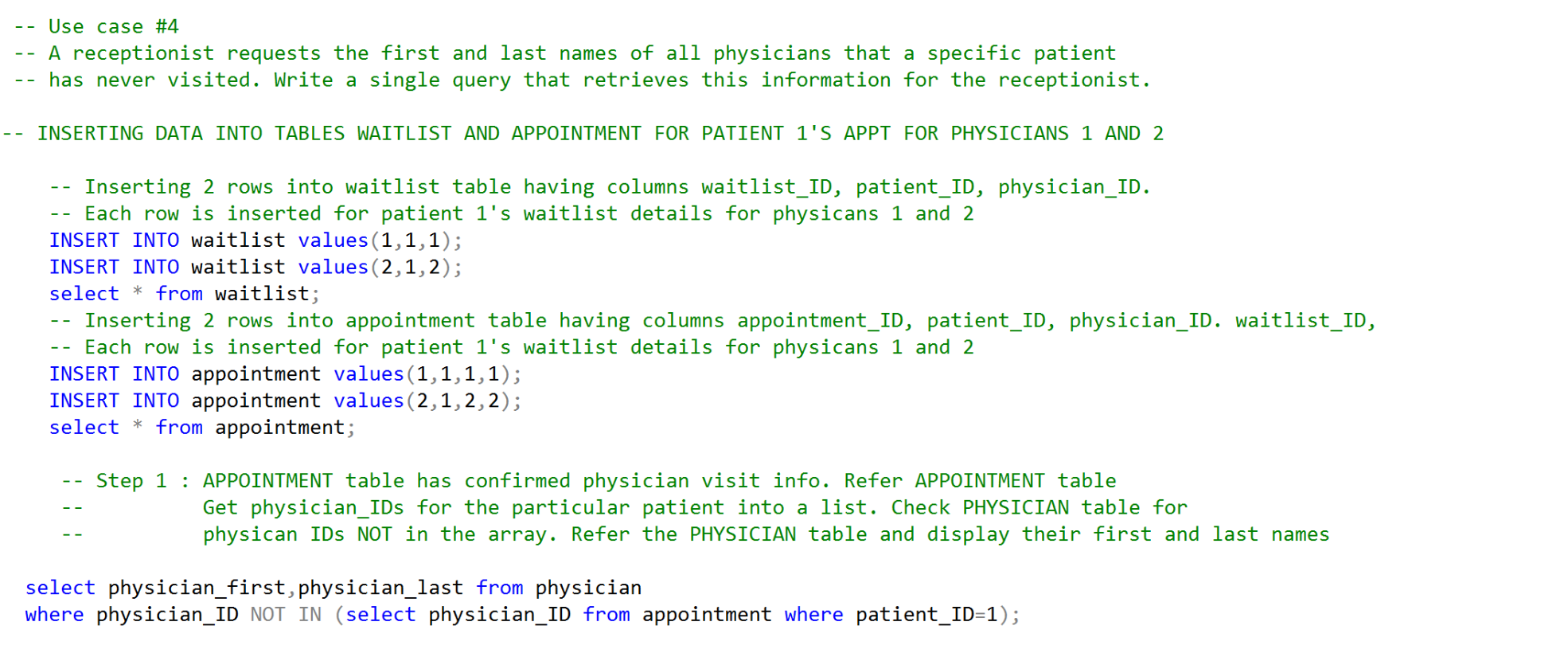




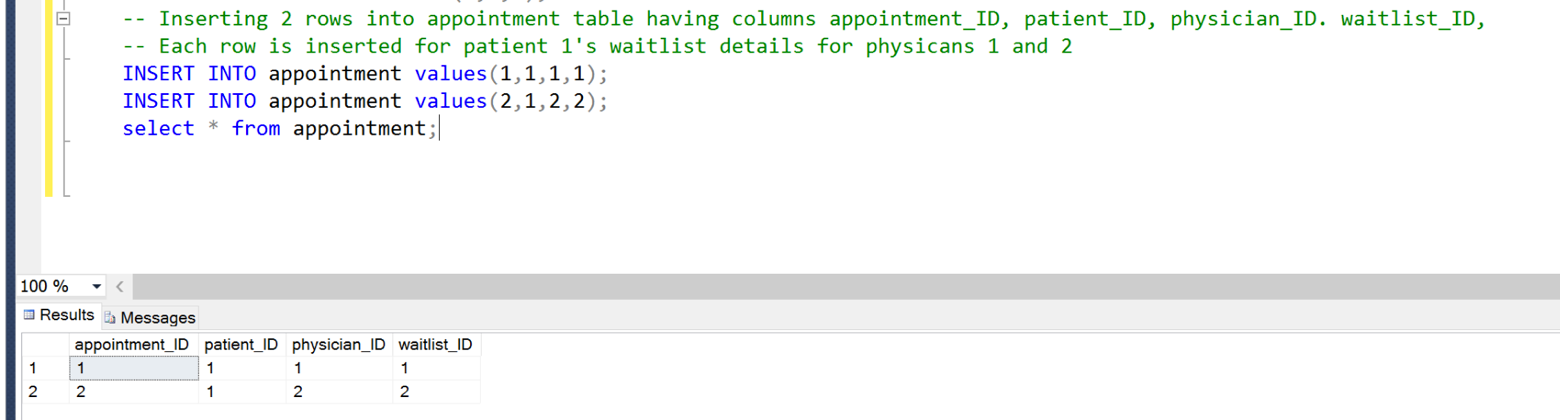
**Use case #4**

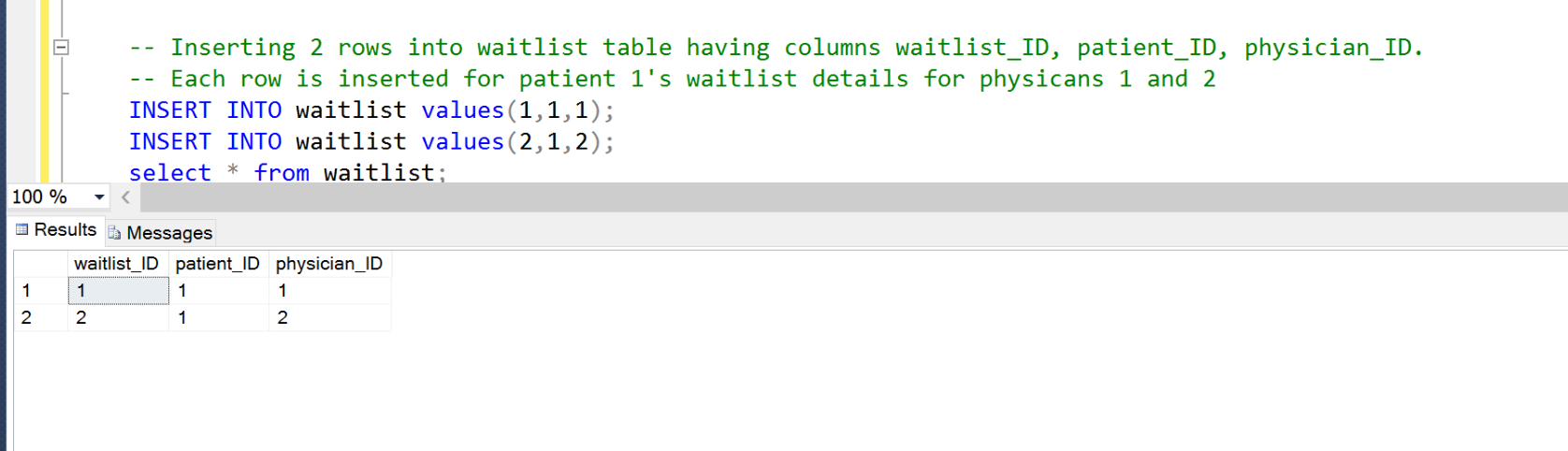
**Requirement**: A receptionist requests the first and last names of all physicians that a specific patient has *never* visited. Write a single query that retrieves this information for the receptionist.

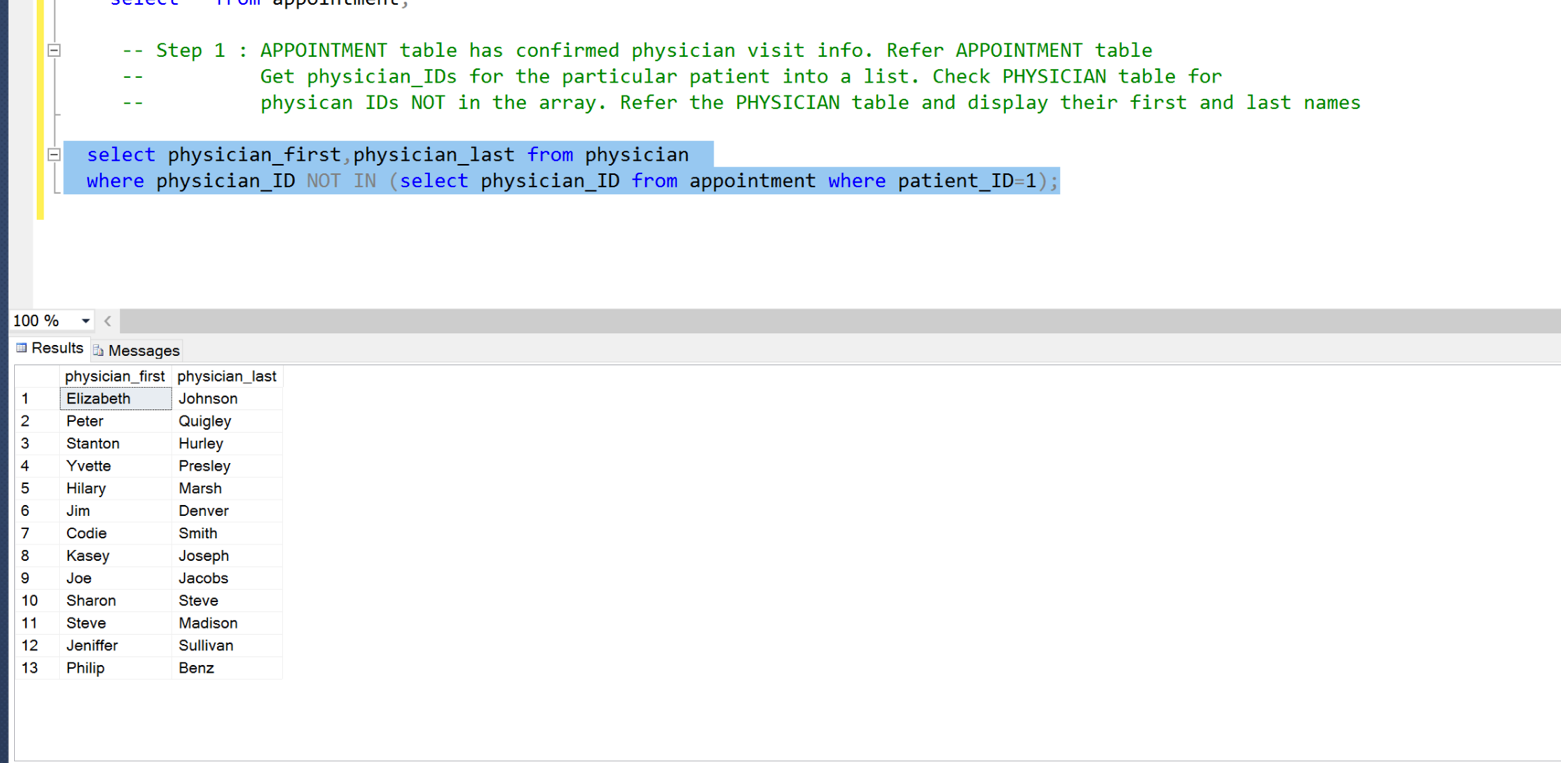
**Full Query:**



**Query Execution in parts:**







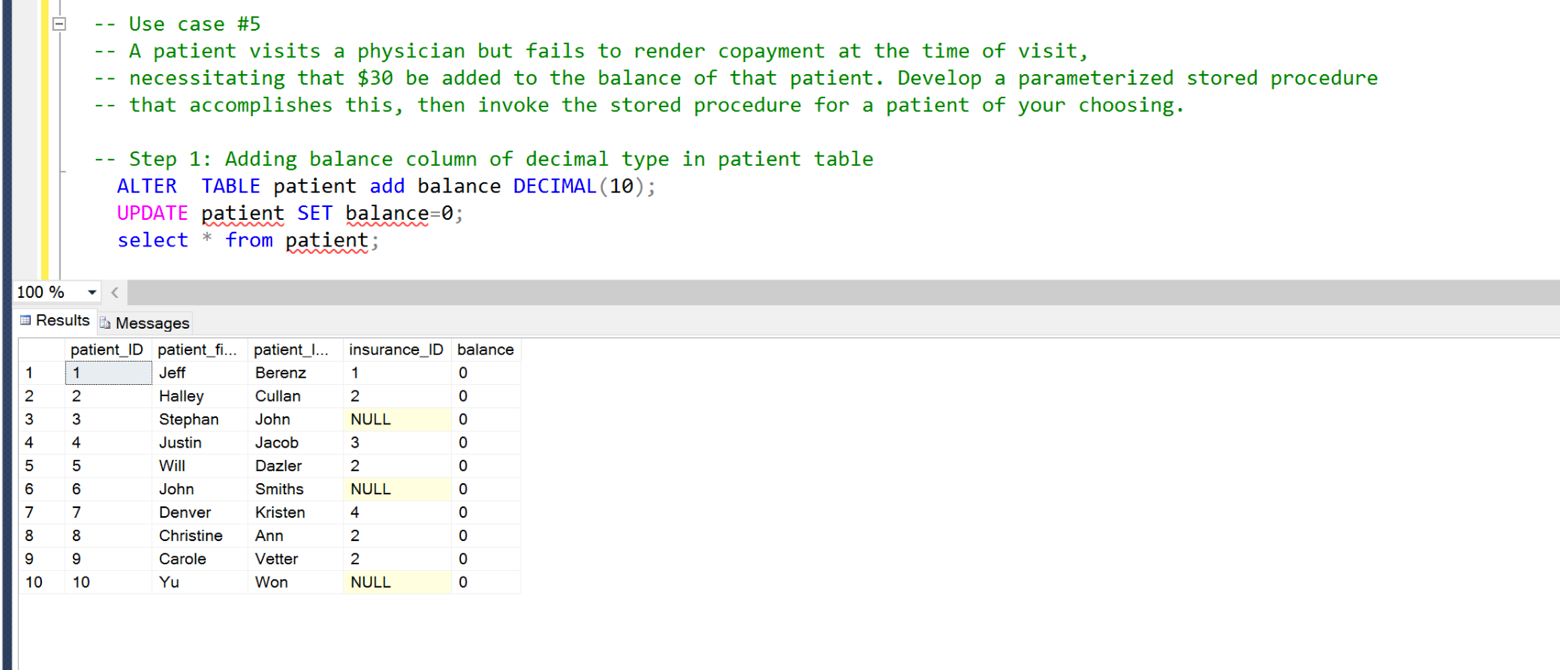
**Use case #5**:

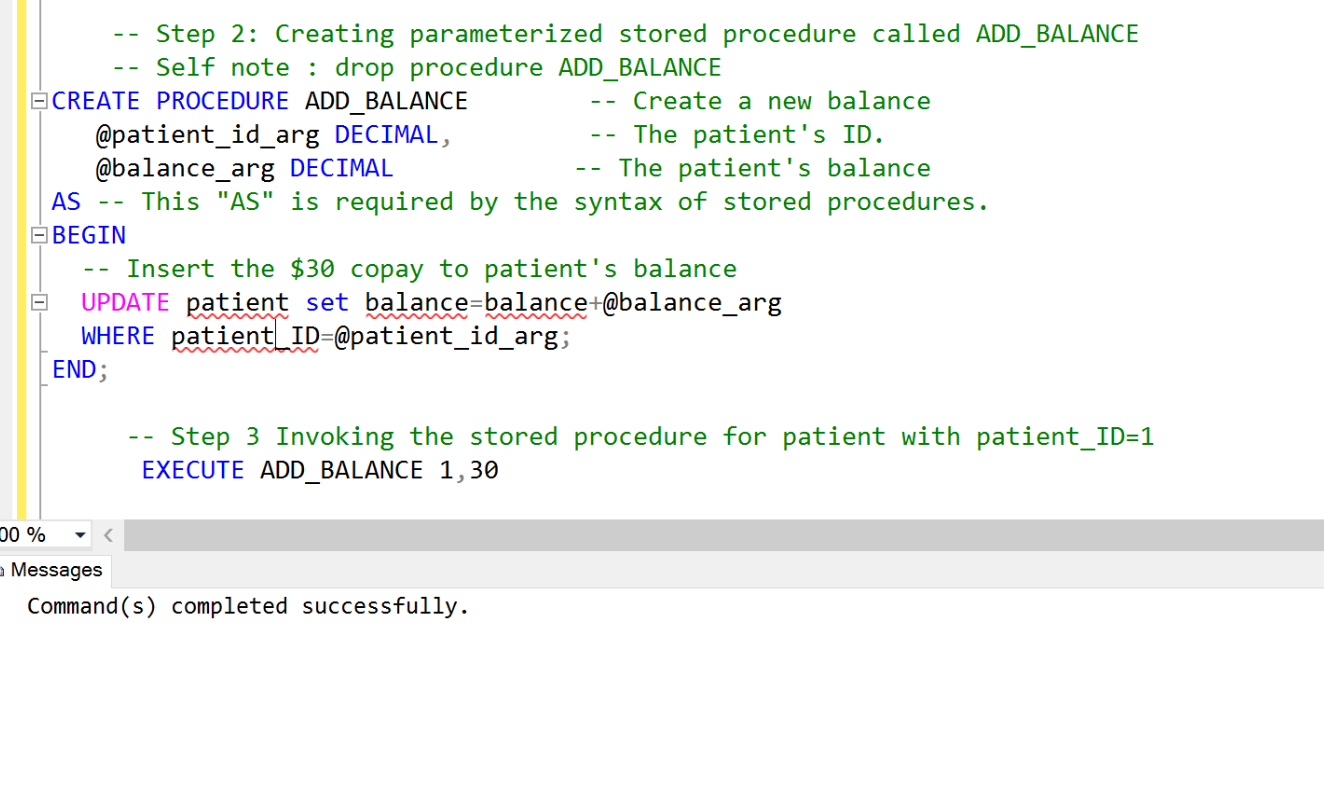
**Requirement**: A patient visits a physician but fails to render copayment at the time of visit, necessitating that $30 be added to the balance of that patient. Develop a parameterized stored procedure that accomplishes this, then invoke the stored procedure for a patient of your choosing.

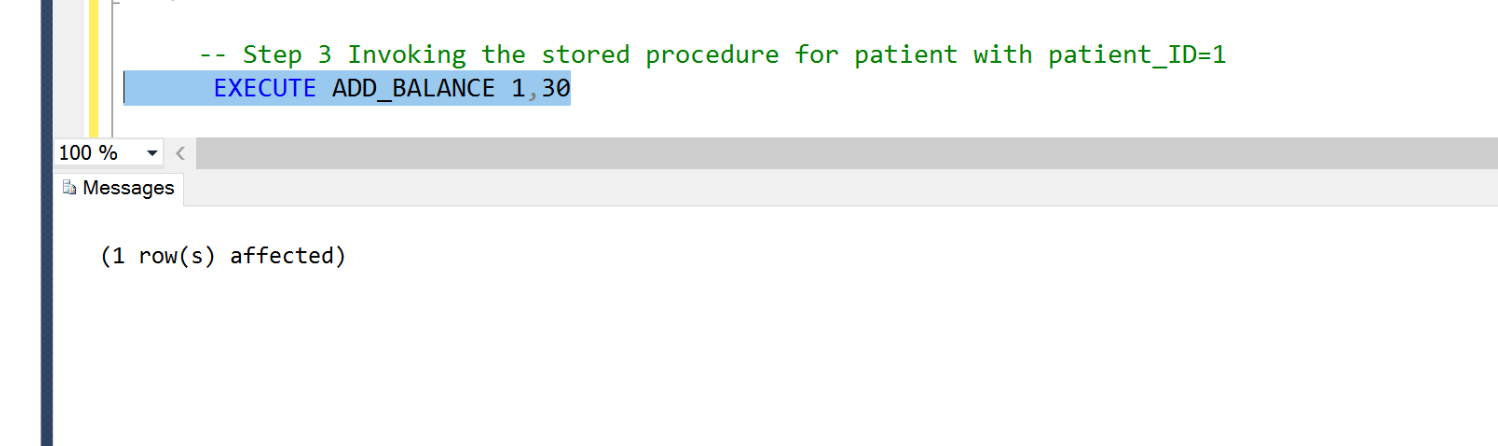
**Full Query:**

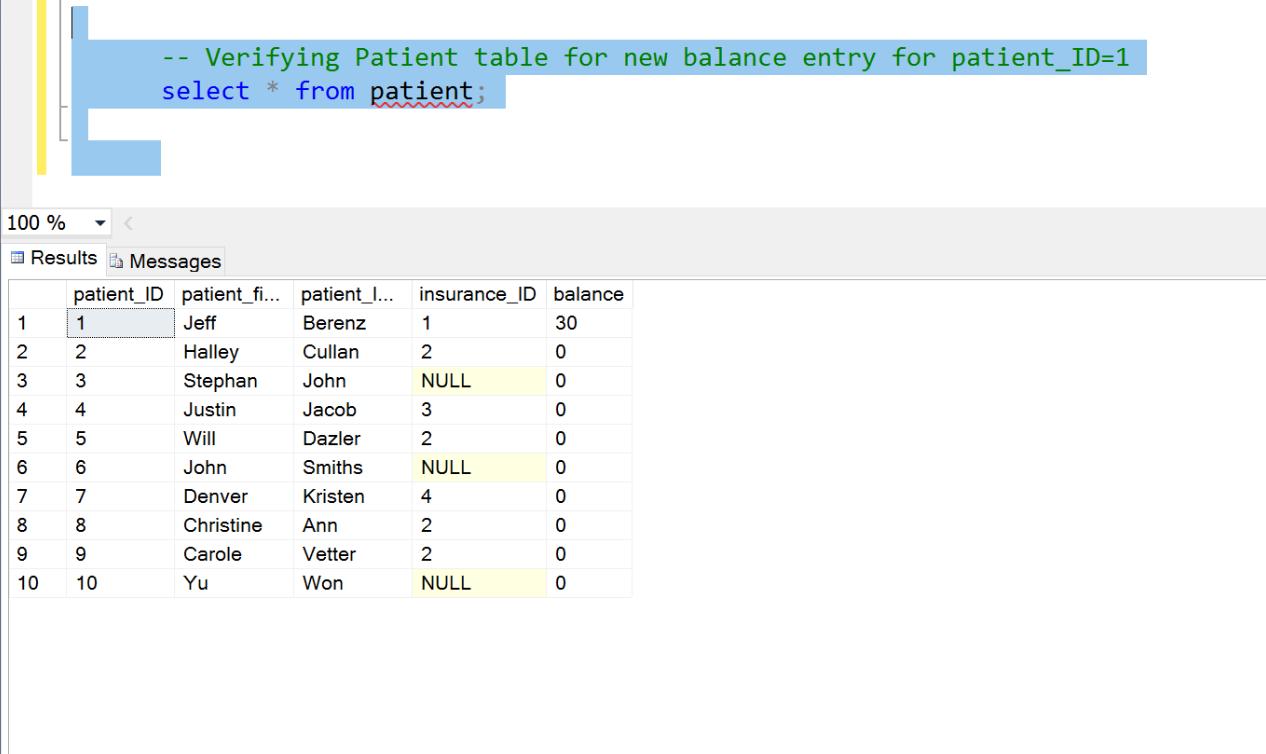


**Query Execution in parts:**





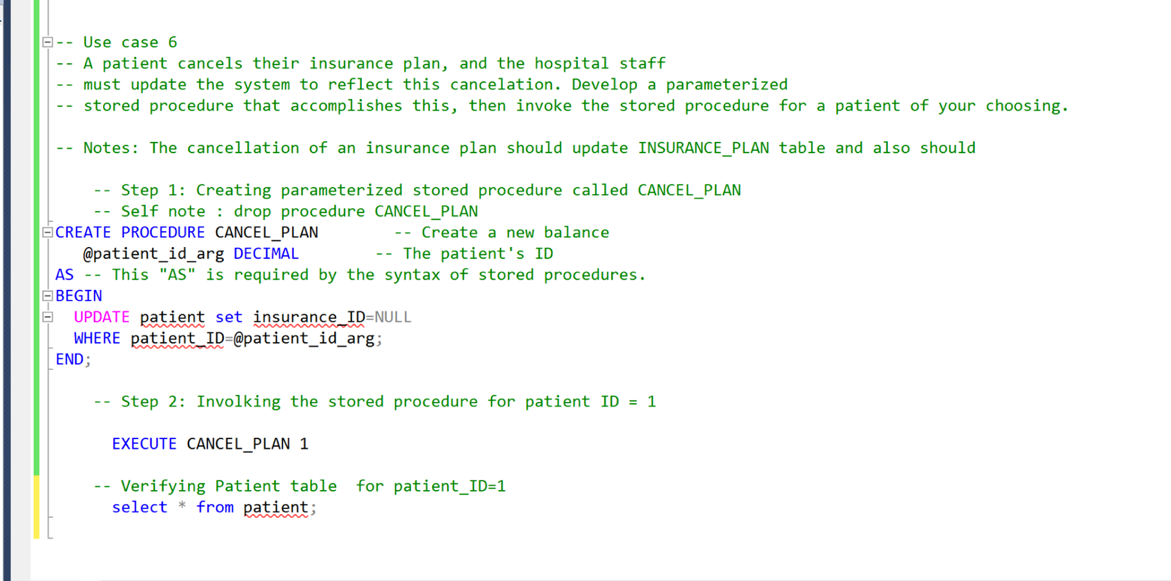




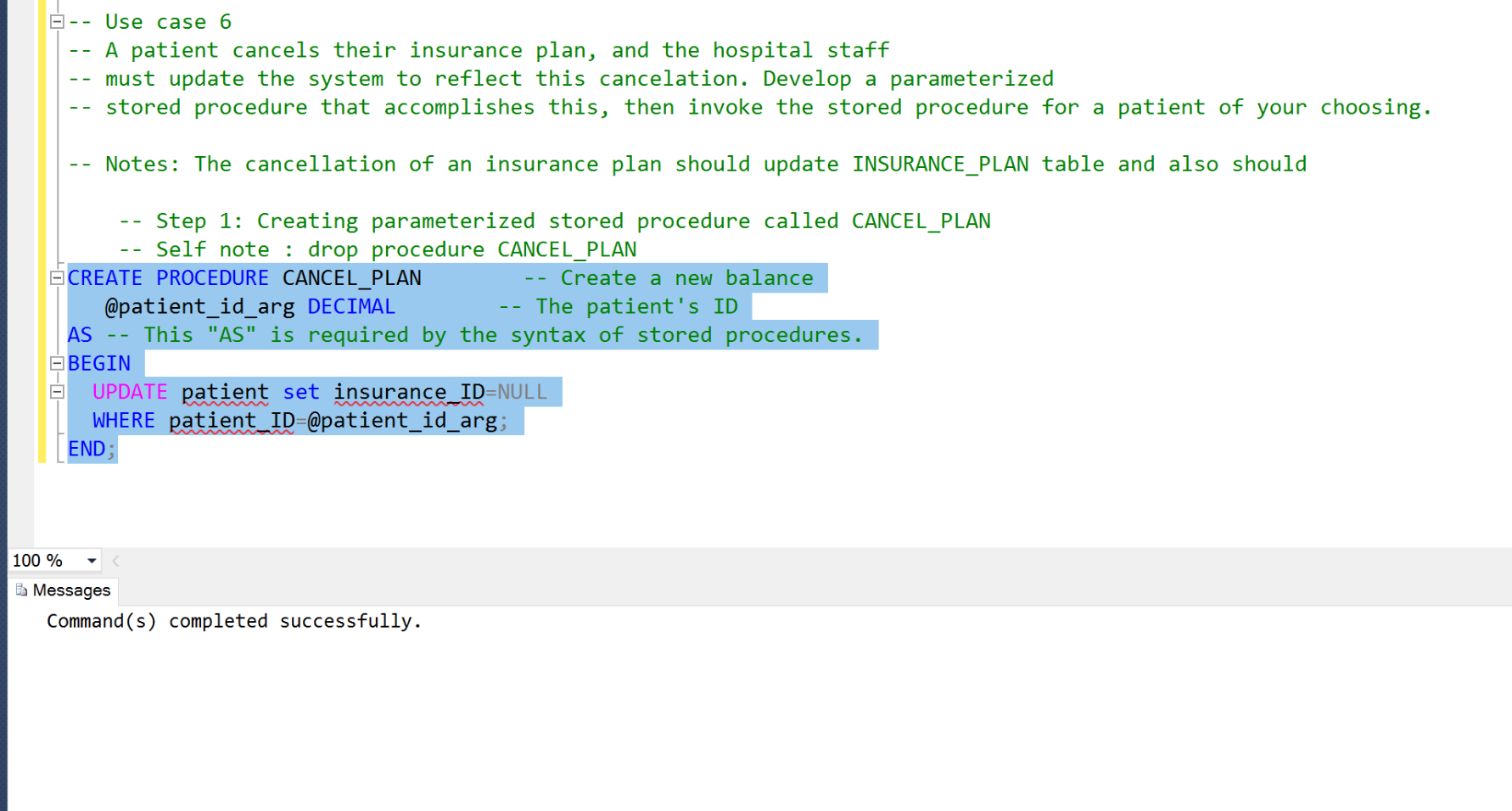
**Use case #6**:

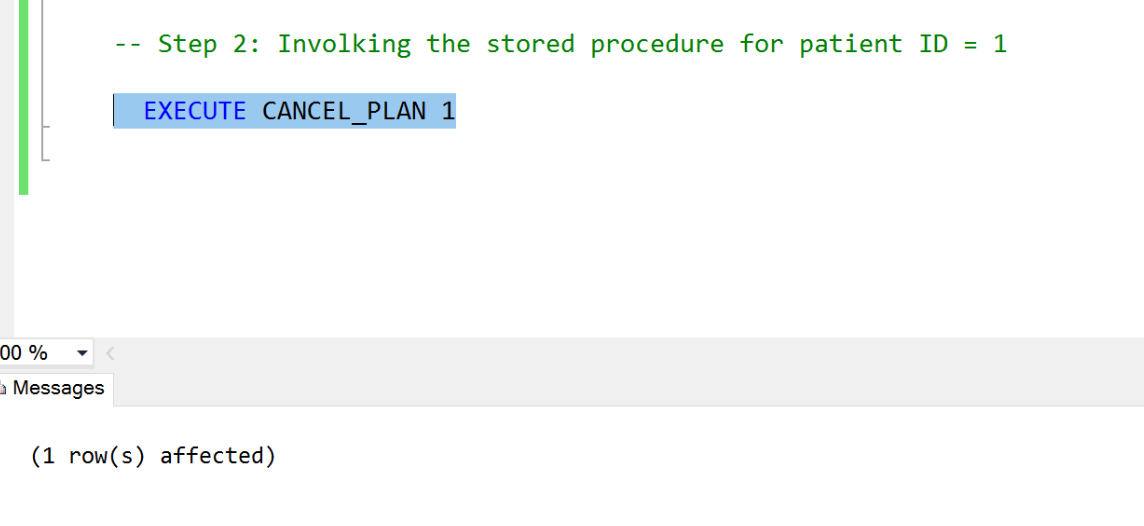
**Requirement:** A patient cancels their insurance plan, and the hospital staff must update the system to reflect this cancelation. Develop a parameterized stored procedure that accomplishes this, then invoke the stored procedure for a patient of your choosing.

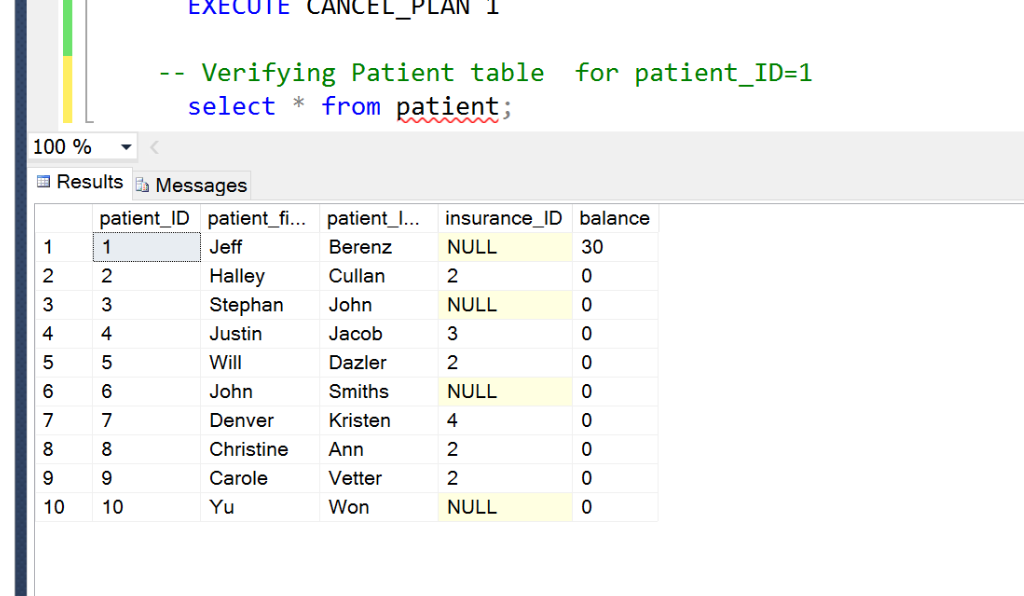
**Full Query:**



**Query Execution in parts:**

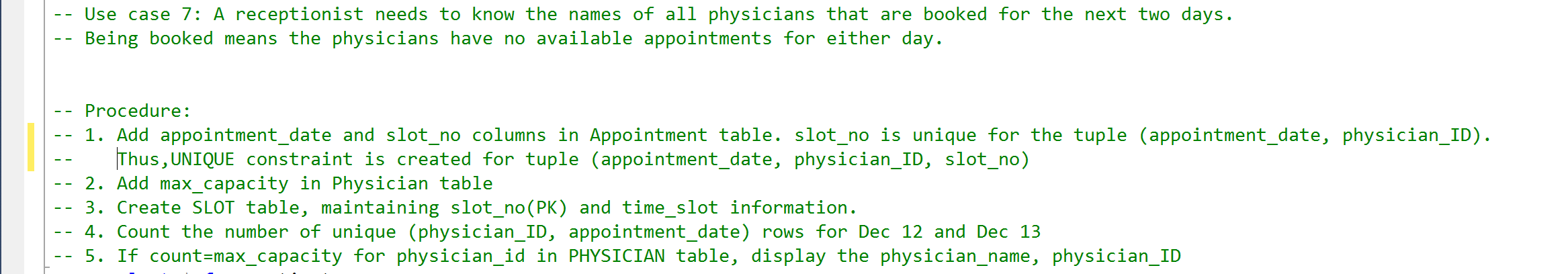


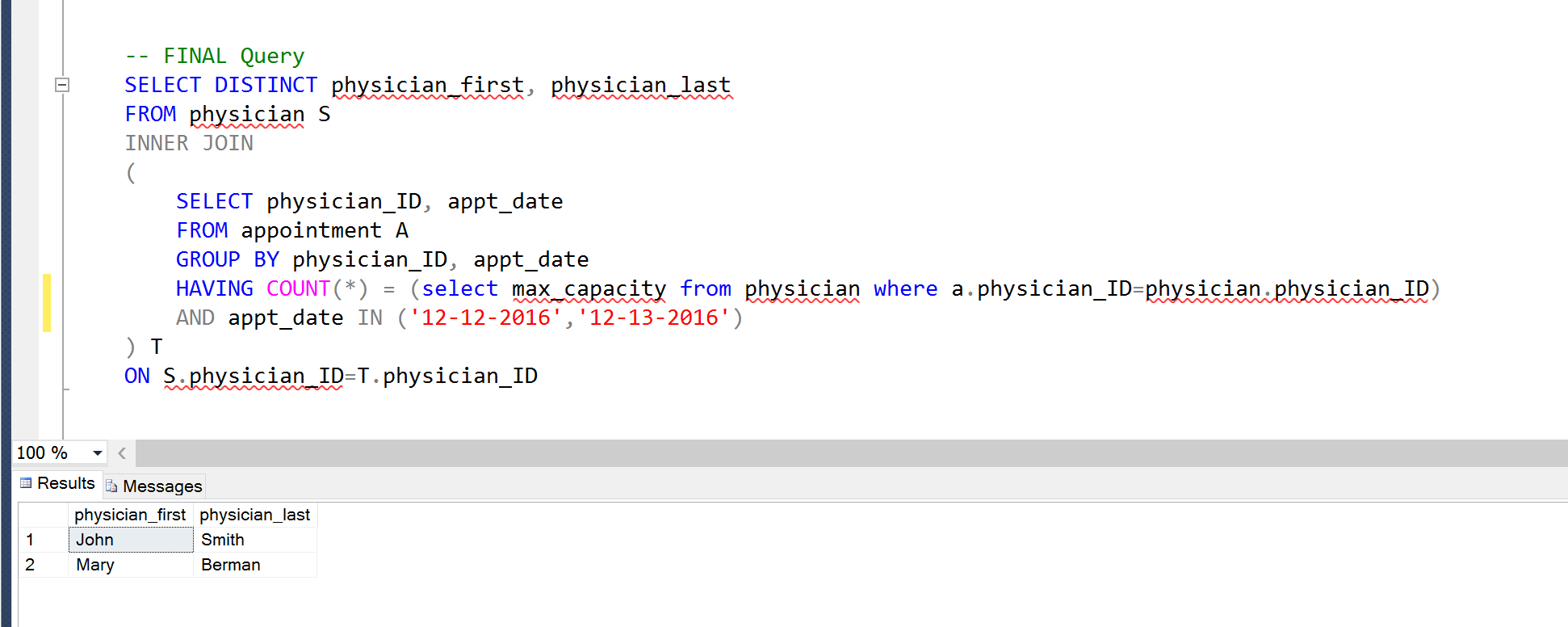




**Use case #7**:

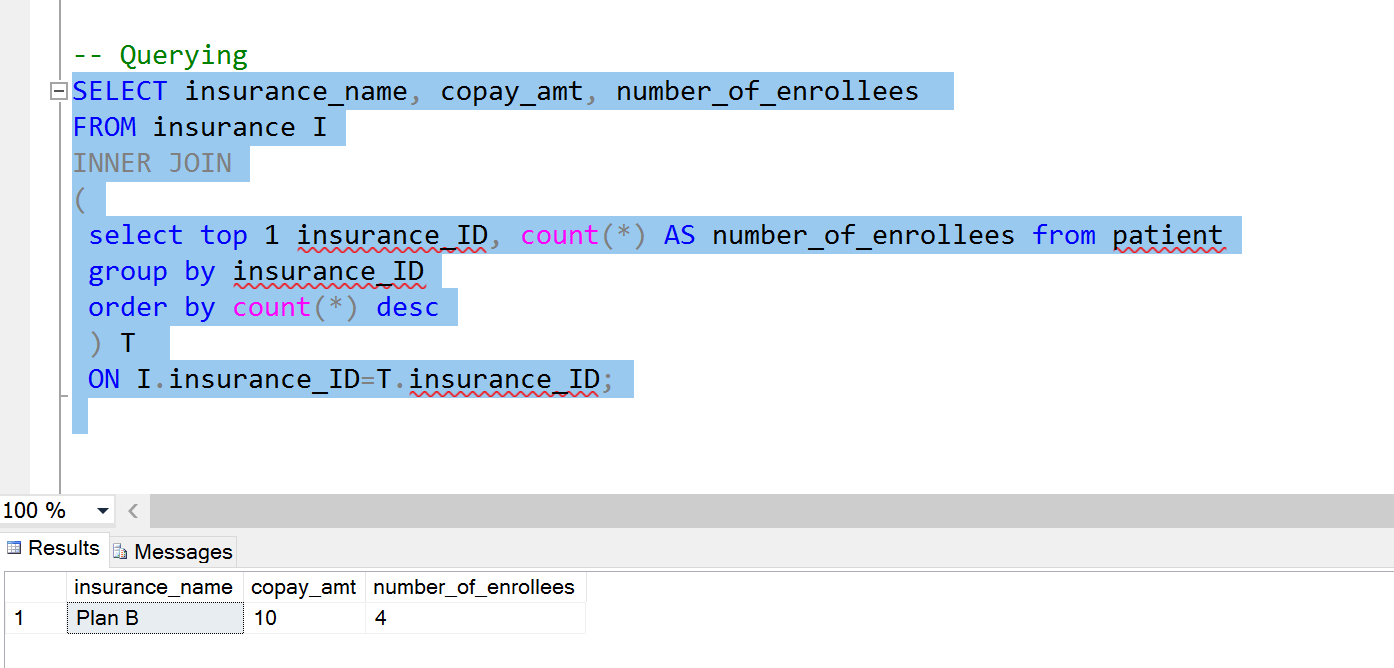
**Requirement**: A receptionist needs to know the names of all physicians that are booked for the next two days. Being booked means the physicians have no available appointments for either day. Write a single query that retrieves this information for the receptionist.





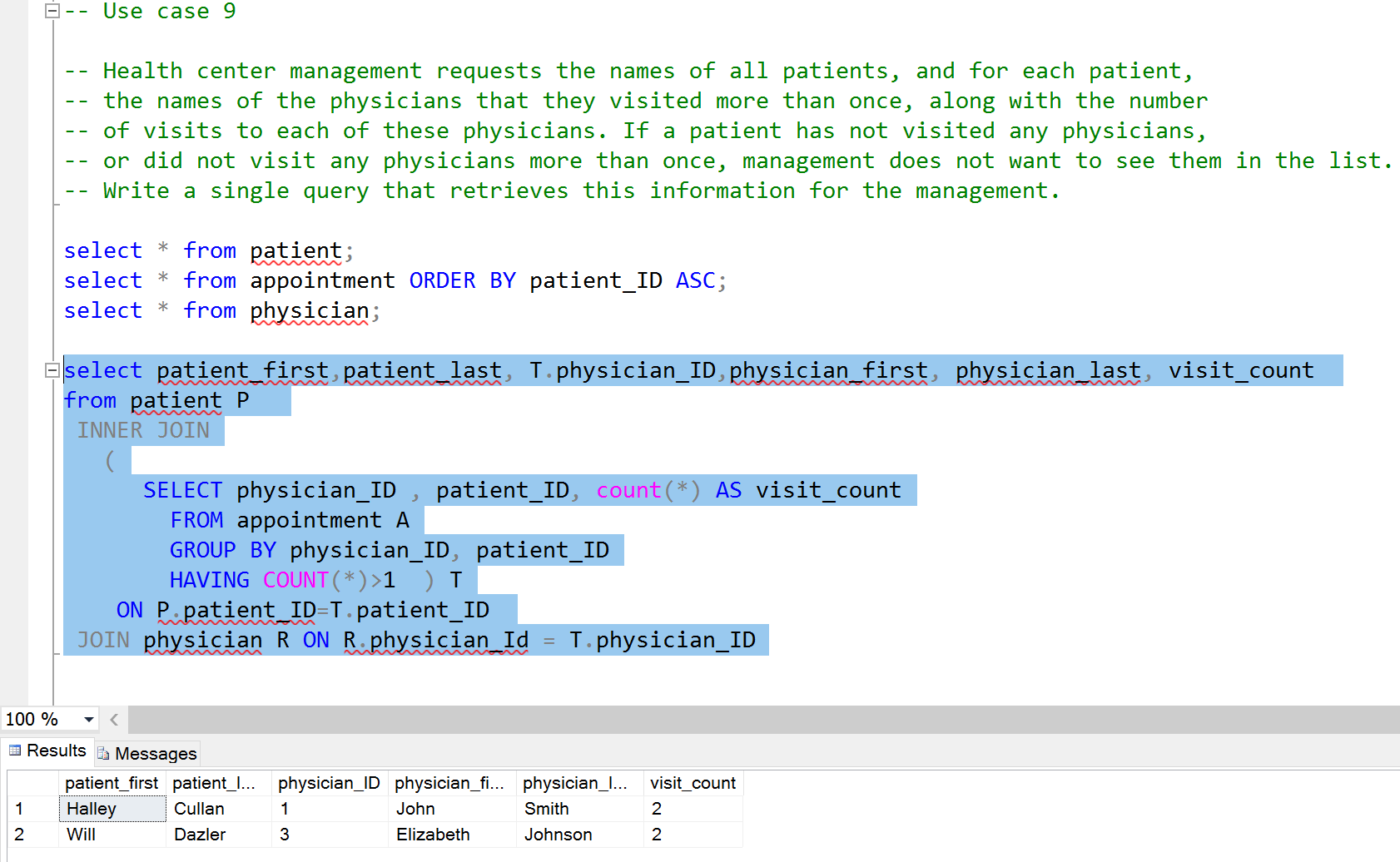
**Use case #8**:

**Requirement**: Health center management requests the insurance plan with the most patient enrollees, and for that plan, its name, required copayment amount, and the number of patient enrollees. Write a single query that retrieves this information for the management.



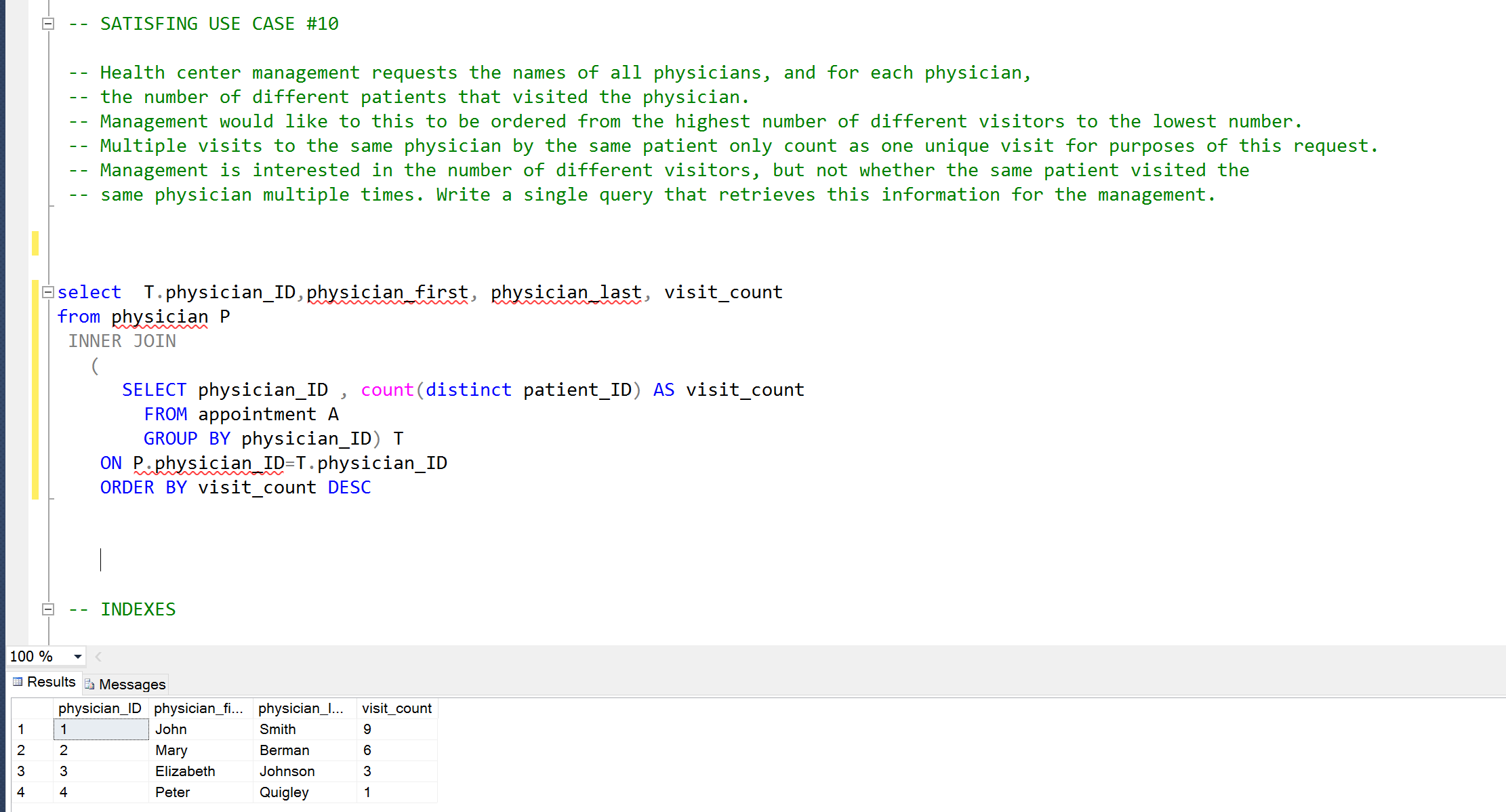
**Use case #9**:

**Requirement**:  Health center management requests the names of all patients, and for each patient, the names of the physicians that they visited more than once, along with the number of visits to each of these physicians. If a patient has not visited any physicians, or did not visit any physicians more than once, management does not want to see them in the list. Write a single query that retrieves this information for the management.

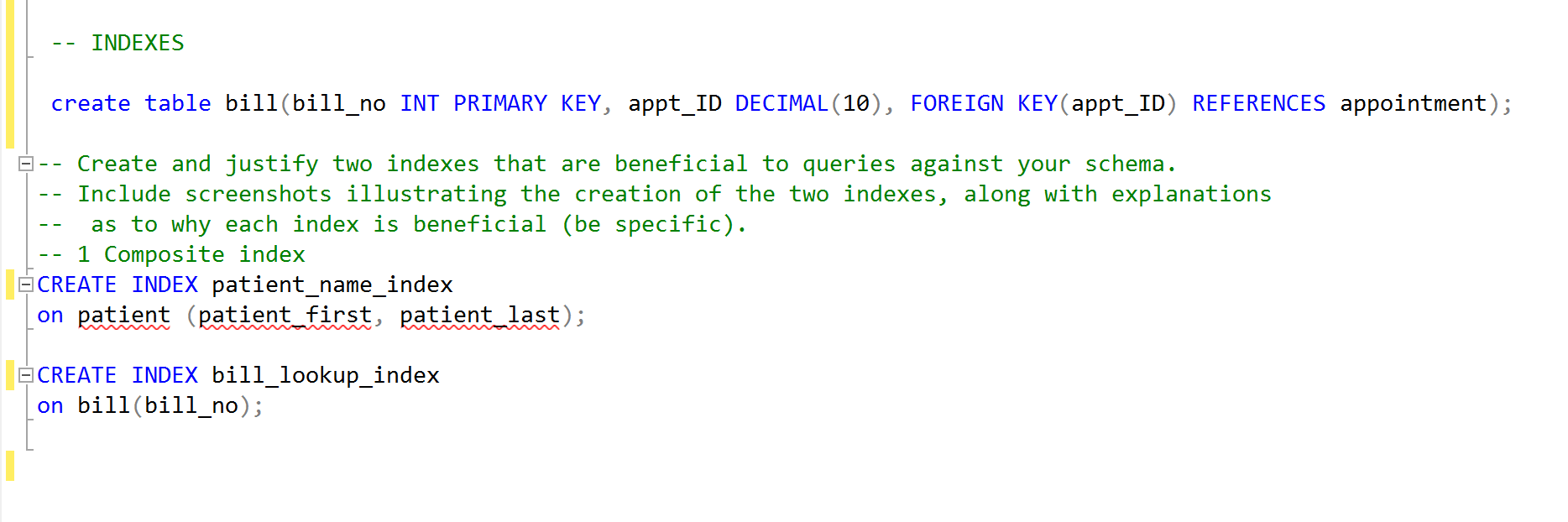


**Use case #10:**

**Requirement:** Health center management requests the names of all physicians, and for each physician, the number of different patients that visited the physician. Management would like to this to be ordered from the highest number of different visitors to the lowest number. Multiple visits to the same physician by the same patient only count as one unique visit for purposes of this request. Management is interested in the number of different visitors, but not whether the same patient visited the same physician multiple times. Write a single query that retrieves this information for the management.



**INDEXES**



Indexes are created in such a way that query performance is improved, therefore, they are used on large tables, and are avoided on tables with frequent update or insert operations (such as appointment table and waitlist table). Columns that are frequently altered/manipulated are also avoided for enhanced query performance.

The index created in Patient table is a composite index comprising of a patient’s first name and last name because typically, if a patient needs to be looked up in the system, their full name is entered to retrieve the best results because this composite index is most likely a unique value, and can boost performance when joins or searches are executed. It is safe to use the index on this table because the patient’s personal details such as name will not be changed, and the patient table is considerably large.

Another index is created on the bill table for the column bill\_no because this column will not be changed at all since the bill\_no values are unique to each appointment table entry. There will not be any update query, and will only have insert operations. This table is going to be a very large table as well, so the indexing will be very helpful.