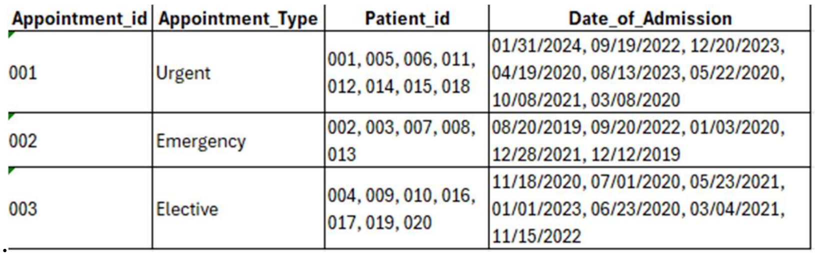
Normalization Form Report

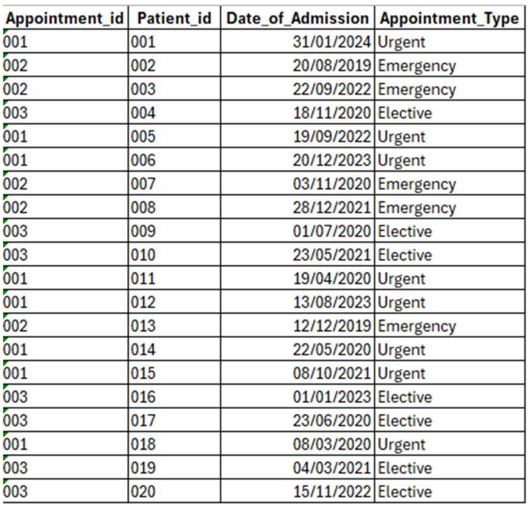
This report analyzes and normalizes an unstructured "appointment" table using the principles of database normalization: **1NF, 2NF, and 3NF**. The goal is to enhance data integrity, eliminate redundancy, and enable efficient querying in the healthcare system.

The original table stored multiple Patient_id and Date_of_Admission values in list format, which violates the rules of **First Normal Form (1NF).**



Appointment_id	Appointment_Type	Patient_id	Date_of_Admission
001	Urgent	001, 005, 006, 011, 012, 014, 015, 018	01/31/2024, 09/19/2022, 12/20/2023, 04/19/2020, 08/13/2023, 05/22/2020, 10/08/2021, 03/08/2020
002	Emergency	002, 003, 007, 008, 013	08/20/2019, 09/20/2022, 01/03/2020, 12/28/2021, 12/12/2019
003	Elective	004, 009, 010, 016, 017, 019, 020	11/18/2020, 07/01/2020, 05/23/2021, 01/01/2023, 06/23/2020, 03/04/2021, 11/15/2022

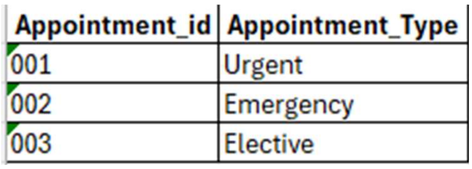
Using the principles of **1NF**, we transformed the table to ensure that each field contains only a single value. As a result, each row now represents a **single patient per appointment**, ensuring atomicity.



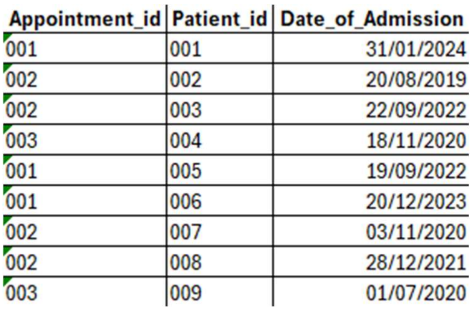
Appointment_id	Patient_id	Date_of_Admission	Appointment_Type
001	001	31/01/2024	Urgent
002	002	20/08/2019	Emergency
002	003	22/09/2022	Emergency
003	004	18/11/2020	Elective
001	005	19/09/2022	Urgent
001	006	20/12/2023	Urgent
002	007	03/11/2020	Emergency
002	008	28/12/2021	Emergency
003	009	01/07/2020	Elective
003	010	23/05/2021	Elective
001	011	19/04/2020	Urgent
001	012	13/08/2023	Urgent
002	013	12/12/2019	Emergency
001	014	22/05/2020	Urgent
001	015	08/10/2021	Urgent
003	016	01/01/2023	Elective
003	017	23/06/2020	Elective
001	018	08/03/2020	Urgent
003	019	04/03/2021	Elective
003	020	15/11/2022	Elective

To achieve **2NF**, we addressed the issue where Appointment_Type depended solely on Appointment_id, rather than on the full composite key (Appointment_id, Patient_id).

We resolved this by moving Appointment_Type into a separate table. The resulting appointments table now includes only data specific to each patient.

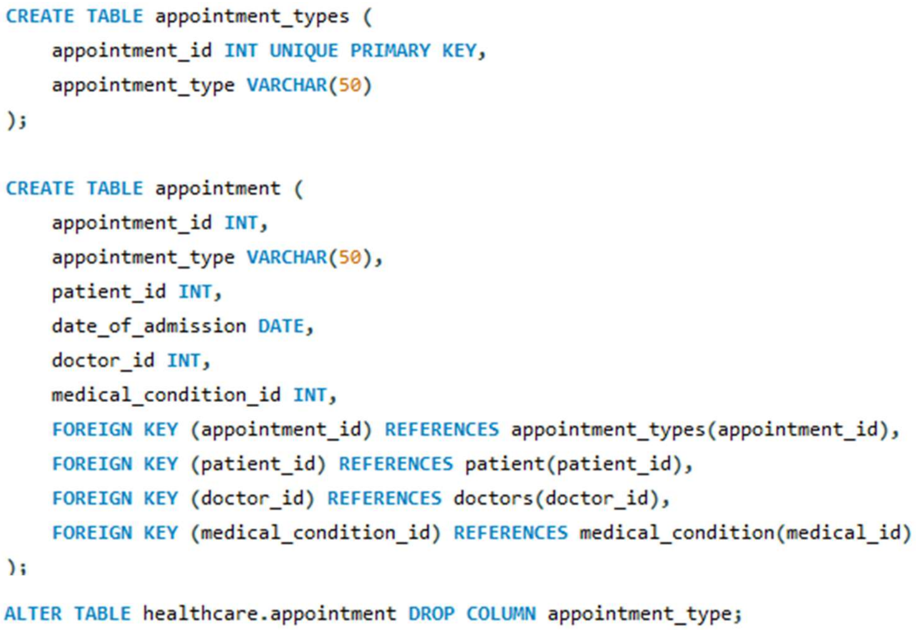


Appointment_id	Appointment_Type
001	Urgent
002	Emergency
003	Elective



Appointment_id	Patient_id	Date_of_Admission
001	001	31/01/2024
002	002	20/08/2019
002	003	22/09/2022
003	004	18/11/2020
001	005	19/09/2022
001	006	20/12/2023
002	007	03/11/2020
002	008	28/12/2021
003	009	01/07/2020

There are no transitive dependencies. All non-key attributes depend **only** on the primary key (Appointment_id, Patient_id). The table structure now satisfies the requirements of **3NF.** Below is the final normalized schema in SQL code format:



```
CREATE TABLE appointment_types (
    appointment_id INT UNIQUE PRIMARY KEY,
    appointment_type VARCHAR(50)
);

CREATE TABLE appointment (
    appointment_id INT,
    appointment_type VARCHAR(50),
    patient_id INT,
    date_of_admission DATE,
    doctor_id INT,
    medical_condition_id INT,
    FOREIGN KEY (appointment_id) REFERENCES appointment_types(appointment_id),
    FOREIGN KEY (patient_id) REFERENCES patient(patient_id),
    FOREIGN KEY (doctor_id) REFERENCES doctors(doctor_id),
    FOREIGN KEY (medical_condition_id) REFERENCES medical_condition(medical_id)
);
ALTER TABLE healthcare.appointment DROP COLUMN appointment_type;
```

The original, unnormalized appointment table had several structural issues, including non-atomic fields and redundant data. Through a systematic three-step normalization process, the database design now conforms to 1NF, 2NF, and 3NF. The result is a structure that reduces redundancy, enhances consistency, and supports scalable and efficient data management.