How Database Keys Will Save Cassie’s Life

Scenario 1: Primary Key

Narrator: It’s time to take medicine, Doctor Kushi prepared two medicines for Cassie number one and Cassie number two.

Kushi : (Looked at the same names on room number 1 and room number 2, had no idea which medicine is for which room, Walked to Cassie number one) Hey, Patient Cassie, time to take medicine.(Hand low blood pressure medicine to Cassie No.1)

Cassie : Thank you doctor. My heart beats so fast just now. (Drink water and take the medicine)

Kushi: (Walked to room number 2 and gave out medicine to Cassie No.2) Wake up, patient Cassie. Medicine time.

Cassie No2:（Rubbed her eyes) Thank you Kushi. I was almost faint just now. (Drink the water and take the medicine)

Cassie No.1 and Cassie No.2 fainted.

Kushi & Doctor: Emergency rescue

Narrator: We are sorry to see this tragedy happening, and right now we should take lessons to prevent it happening again. Here two Cassies have exactly the same information, they share the same name, the same address, the same height and the same birthday. Both doctors and assistants are not able to discern between them. Not only in hospitals, it can happen everywhere and lead to tragedies. It could be the case when you deliver items to the wrong address when users have the same name, or when you update grades to the wrong student when they have the same name.

To prevent it happening, we should have a unique key to identify the subject in a database. If we have an id number for each Cassie, the doctor and the assistant would not give out medicines wrongly. We can also locate the Cassie we are looking for more quickly without comparing multiple features. Primary key is the soul of a database,

Scenario 2: Foreign Key

Narrator: This morning, Cassie awoke with an agonizingly sore throat. Concerned, she scheduled an appointment with her doctor.

Cassie: "Dr. Khushi, every time I swallow, it feels like I'm swallowing needles!"

Dr. Khushi: "Let's take a look." (Peers into Cassie's mouth) "Ah, classic signs of strep throat. You'll need some medication."

Narrator: With prescription in hand, Cassie heads to her local CVS.

Cassie: "I'm here to collect medicine for Cassie Richter."

Kefeng: "One moment." (Retrieves the prescription and hands it over)

Cassie: "Finally! I hope this helps." (Eagerly takes the medicine and then collapses in a theatrical manner)

Narrator: But what went wrong? You see, when Dr. Khushi wrote that prescription, she referenced only the details of Cassie's current visit. Digging deeper into Cassie's health records, a glaring issue arises: an allergy to amoxicillin. Had the appointment and health history tables been interconnected, this tragedy could've been averted. This is where the magic of foreign keys in databases comes into play. To safeguard patients in the future, the hospital must introduce a foreign key in the appointments table, linking to the patients' comprehensive health profiles.

You might be wondering, what's this "foreign key" I speak of? In the realm of databases, a foreign key is a column or set of columns in one table that uniquely identifies a row of another table. In other words, it establishes a direct relationship between the data in two tables.

So, the next time you hear about "foreign keys" in a database, remember they're not just technical jargon. They're the unsung heroes, ensuring data integrity and preventing potential disasters.

Scenario 3: Foreign Key

Narrator: Dr. Xiao was a doctor at the USF Hospital until she suddenly decided to leave to work somewhere else right before Cassie arrives for an appointment with the doctor to pick up her medication for her blood pressure

Doctor: “I’ve been working at this hospital for such a long time, but I don’t think I’m enjoying this job anymore. I think I’m going to quit and work somewhere else.” (walks off-stage)

Narrator: Because Dr. Xiao no longer works at the hospital, her information is automatically deleted from the database. Just minutes after Dr. Xiao’s information is deleted from the doctors table, a table containing information on the doctors employed at USF, Cassie arrives for her appointment with Dr. Xiao

Cassie: (walks on stage) “Hi! I have an appointment scheduled for today to pick up my medication from Dr. Xiao.”

Hospital Receptionist: “Okay, no problem! Let me just look at the list for her appointments today.” (looks at laptop). “Oh no, it looks like she decided to move to another practice. All of her information, including the appointments that she’s scheduled have disappeared from our system! Unfortunately, this does mean that you will need to find someone else to provide your medication”

Cassie: “That’s too bad. I really need this medication as soon as possible though, so I’m willing set up another appointment”

Narrator: Cassie sets up another appointment scheduled for this Wednesday with another doctor practicing the same speciality that Cassie’s former doctor did. The next day, Cassie begins to experience serious symptoms associated with high blood pressure. Unfortunately, she doesn’t have the medication needed to manage these symptoms

Cassie: If only I had my medication, this wouldn’t be happening (collapses)

Narrator: What happened here? When Dr. Xiao’s information was deleted from the doctors tables when she left the hospital, this automatically deleted all of the appointments that she had scheduled from a separate appointments table. This is because the information stored in the doctor’s table is linked to the information stored in the appointments table through the use of doctor_id as a foreign key, allowing it to refer to the doctors table.

Unfortunately, the appointments table includes a clause allowing any deletions that occur in the doctors table to occur in the appointments table as well. As a result, any appointments that Cassie has with the doctor has been removed from the appointments table as well. This means that Cassie is unable to receive her medication, leading to tragic consequences. Therefore, it’s important that you place the proper constraints on a foreign key in order to ensure that important information remains consistent.

Scenario 4: A good design

Narration (Cassie):

Here we see properly setting a primary key in each table will allow us to identify a patient uniquely, even if patients have the same date of birth, name, or other information.

Then, setting foreign keys we see proper relationships and restrictions set. On update, we want all tables to have up to date information so a change in doctor or patient information should be reflected in appointments. On delete, we saw from scenario 3 that we want to restrict this relationship. This will ensure that if something is deleted from patients or doctors, the information remains in the appointments table.

With this proper database design, I can finally trust my doctors to prescribe me the correct, safe prescription, to save my life rather than kill me.