POSTS AND TELECOMMUNICATIONS INSTITUTE OF TECHNOLOGY

Faculty of Information Technology

TERM PROJECT

COURSE: DATABASE

Lecturer:	Nguyen Dinh Hoa
Class:	E22CQCN03-B
Group:	2
Topics:	Design a database system for a University Dormitory
Member of group:	Duong Minh Quang Dang Tuan Phong Bui Minh Tung

Ha Noi – 2024

TABLE OF CONTENT

. Requirement Analysis

1. Real-life Scenario

Each year, there are many students enrolled in a university. In order to satisfy their needs of accommodation, universities usually set up 1 or more buildings as dormitories for them. To manage and control the information of such a high number of people, a well-developed database is essentially required. In this case, let assume that the university that we are working with has 3 dormitories called A, B and C. Here is what it will look like:

Firstly, we need Dormitory Manager Department staff to manage all the work, the Dormitory Manager Department staff will be split into 3 categories: office staff, guard and dormitory manager. They will have personal id, full name, salary, phone number (a staff can have multiple phone numbers), email address (only using one email for work), address, date of birth and status (still working or not), assigned dormitory and staff type.

Next, we need information about the dormitories in that university. It will consist of dormitory ID, location, number of floors, number of rooms and overall condition (new, old, stable, not stable)

After moving in, students will live in the assigned room which is managed by all the staff of the Dormitory Manager Department. The room details are Room ID (for example, 204-B, this means that room number is 04 and the room is located in floor 2 of dormitory B), amounts of students, condition, capacity and occupancy status. Each room will have a list of facilities, which has: item name, quantity and quality.

Students can also send requests to the manager and after receiving it, they can consider that. In request, we have request ID, content, type, status, completion date.

To register in a dormitory, a student must create a contract with the Dormitory Manager Department office staff. This contract consists of contract ID, contract start date, contract end date, contract status and penalty for violation.

After successfully registering into a room in the dormitory, student's information is obligated to update in the database system. Students will have: Student ID, student name (First Name, Middle Name, Last Name), date of birth, address, gender, email (provided by university), phone number(one student can only have one phone number), emergency contact.

After moving in, students will live in the assigned room which is managed by all the staff of the Dormitory Manager Department. The room details are Room ID (for example, 204-B, this means that room number is 04 and the room is located in floor 2 of dormitory B), amounts of students, condition, capacity and occupancy status. Each room will have a list of facilities, which has: item name, quantity and quality. Students can also send requests to the manager and after receiving it, they can consider that. In request, we have request id, type request, status request, content request and completion date.

After a month or a semester, each student will have a bill for electricity, water, internet usage and maybe some maintenance or even penalty. This will be managed by the office staff. The bill will consist of bill id, bill type, amount due, due date and payment status.

2. Database Applications Description:

The database will manage various aspects of a university dormitory:

- Dormitory Management:
    - Dormitory Manager Department Staff: Managing information for office staff, guards and dormitory managers
    - Dormitories: Information about the dormitories including location, number of floors, rooms and overall condition
- Student Management:
    - Registration and Contracts: Students must create contracts with office staff before moving in
    - Student Information: Personal details, academic information and contact details
    - Room Assignments: Assigning and managing student rooms
- Room Management:
    - Room Details: Room ID, capacity, occupancy status and condition
    - Facilities: Items in the room and their conditions
- Requests:
    - Maintenance Requests: Student can send maintenance requests which are managed by dormitory staff
    - Changing Room Requests: Student can send changing room requests which are managed by dormitory staff
- Billing and Payment:
    - Billing: Managing bills for utilities, maintenance and penalties
- Reporting

- Reports: Generating reports on room occupancy, student information, billing status and maintenance activities

3. **Applications to be Supported:**

Loads: Adding new students, rooms, dormitories, staff, contracts, requests and bills to the database

Updates: Modifying existing records such as updating student information, room assignments, staff details, maintenance and changing room request statuses and billing information

Retrievals: Fetching specific details such as student information, room assignments, staff information, maintenance requests, changing room request

Reports: Generating statistical reports on room occupancy, student payments, monthly revenue, active contracts and room conditions

I. Conceptual Design

1. Entity set

***Key Attribute will be Bold, Underline text**

· Dormitory Manager Department staff (**Personal ID**, first name, middle name, last name, salary, phone number, email address, address and status, assigned dormitory, staff type). Each of the staff will be split to either three types below:

- Guard (**Guard id)**
- Office (**Office id)**
- Dormitory Manager (**Manger id)**

· Dormitory (**Dormitory ID**, location, number of floors, overall condition, number of rooms)

· Student (**Student ID**, Student name (First Name, Middle Name, Last Name), date of birth, address, gender, email address, phone number, emergency contact)

· Contract (**Contract ID**, contract start date, contract end date, contract status, penalty for violation)

· Room (**Room ID**, amounts of students, condition, capacity, occupancy status, gender, floor of rooms)

· Facility (Item name, quantity, overall quality)

· Request (**Request ID,** content, type, status, completion date)

· Bill (**Bill ID**, bill type, amount due, due date and payment status)

2. Relationships:

· Student <register> Contract: one student has only 1 contract, and one contract must be owned by only 1 student -> 1 to 1 (both is fully participation).

· Student <have> Bill: one student may have many bills, but one bill only belonged to one student: 1 to many (both is fully participation)

· Student <live> Rooms: one student must live in one room, but one room can be lived in by many students or no students: many to one (only student is fully participation)

· Dormitory <own> Room: one dormitory can contain lots of rooms, but one room only belongs to a dormitory -> 1 to many (both is fully participation)

· Guard <protect> Dormitory: one guard protect one dormitory, and one dormitory is protected by one guard -> 1 to 1 (both is fully participation)

· Room <include> Facility: one room can have many types of facilities, but each type facility is unique in each room -> 1 to many (both is fully participation)

· Student <Send> Request to Manager: one student can send many requests and this will be managed by only one manager. But one manager can manage many requests of many students. (only Request is fully participation)

· Office staff <create> Contract: one staff can create multiple contracts, but a contract must be created by a staff -> 1 to many (only contract is fully participation)

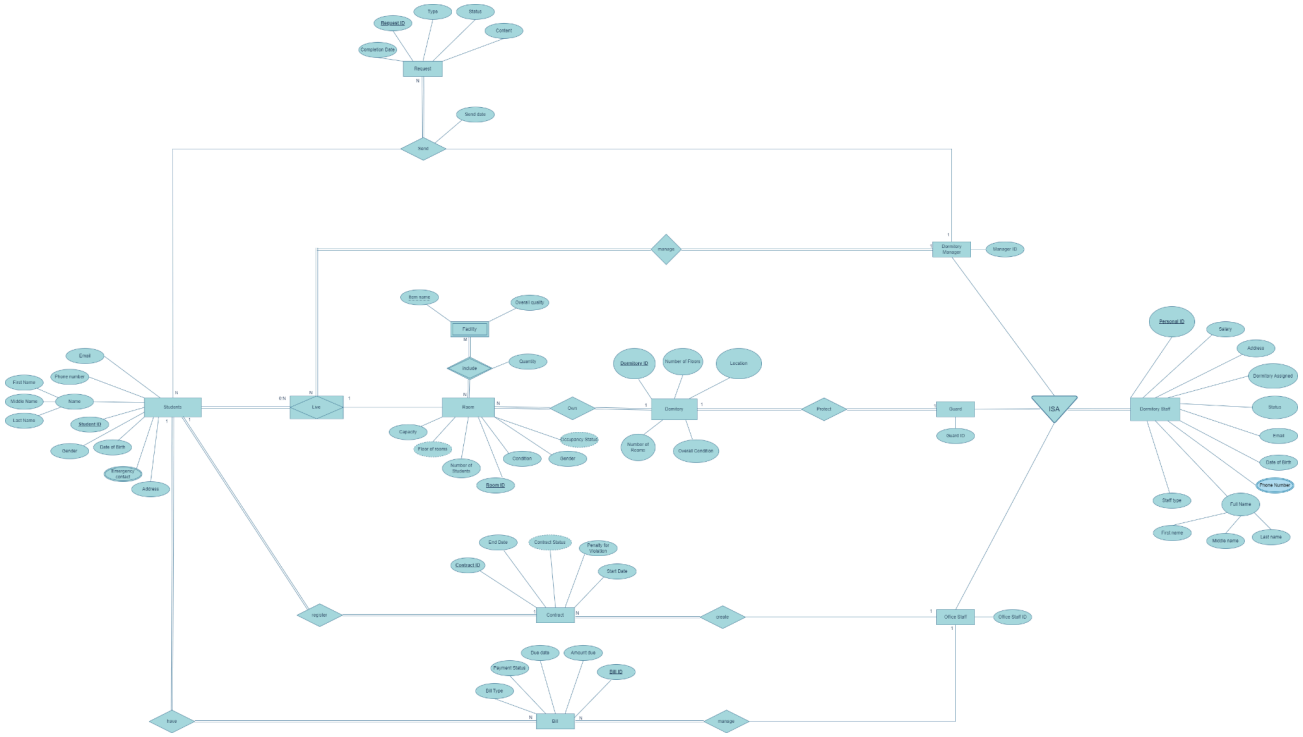
· Office staff <manage> Bill: one staff can manage many bills, but one bill must be managed by a staff -> 1 to many (only bill is fully participation)

· Dormitory manager <manage> Live: one dorm manager can manage the living process of many students in many rooms, but one student living in one room can only be managed by a dorm manager -> 1 to many (both is fully participation).

3. ERD Diagram

*CLICK THIS LINK TO SEE THE ERD DIAGRAM IN DRAW.IO:

https://app.diagrams.net/#G14do9Wkk_67FXeLc_ayfs2sA-lff48l6G#%7B%22pageId%22%3A%22a-EKnljrBsBY4bMRJUoA%22%7D



II. Logical Design

*Noted that Primary key will be Bold, Underline text; Foreign Key will in Bold, Italic, Dashed Underline text; and Partial Identifier of weak entity set will be normal, underline text)

Relational Schema:

Student (**Student ID**, First Name, Middle Name, Last Name, date of birth, address, gender, phone number, email address, emergency contact)

Student_Emegency_Contact (**Student ID**, emergency contact)

Dormitory Staff (**Personal ID**, first name, middle name, last name, salary, phone number, email,ID address, status, assigned dormitory, staff type)

Dormitory_Staff_PhoneNumber (**Personal ID**, phone number)

Dormitory manager (**Personal ID – DM**, manager ID)

Guard (**Personal ID – G**, guard ID)

Office staff (**Personal ID – O**, staff ID)

Dormitory (**Dormitory ID**, location, number of floors, overall condition, number of rooms, ***Personal ID - G***)

Room (**Room ID**, amounts of students, condition, capacity, gender, ***Dormitory ID***).

Bill (**Bill ID**, bill type, amount due, due date, payment status, ***Student ID***, ***Personal ID - O***)

Contract (**Contract ID**, contract start date, contract end date, contract status, penalty for violation, ***Personal ID - O***, ***Student ID***)

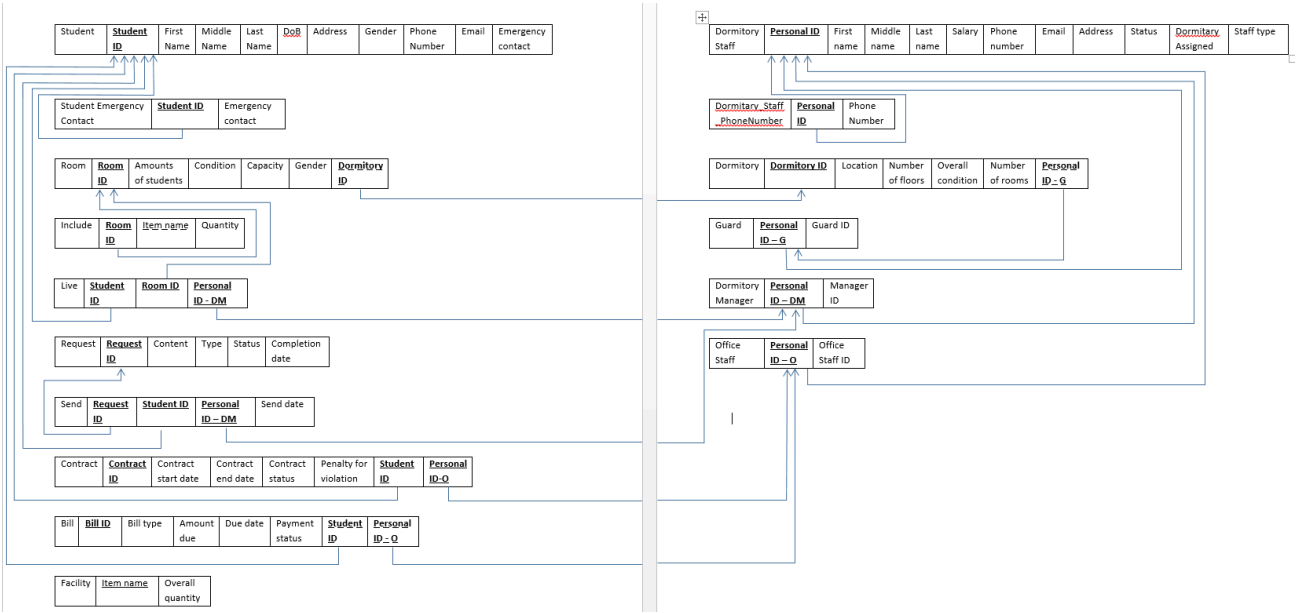
Request (**Request ID**, content, type, status, completion date)

Facility (Item name, overall quality)

Live (***Student ID***, ***Room ID***, ***Personal ID – DM***)

Send (***Student ID***, ***Personal ID - DM***, ***Request ID***, send date)

Include (**Room ID**, Item name, quantity)



III. Normalization

R1 = Student (**Student ID**, First Name, Middle Name, Last Name, date of birth, address, gender, phone number, email address)

F1 = {Student ID → First Name; Student ID → Middle Name; Student ID → Last Name; Student ID → Date of Birth; Student ID → Address; Student ID → gender; Student ID → phone number; Student ID → email address; Email Address → Student ID; Phone number→ Student ID} - Min cover

The relational schema includes atomic attributes => The schema is in 1NF.

Student ID, phone number and email address are the candidate keys, they are all single-attribute keys => The schema is in 2NF. (because all other non-prime attributes will fully depend on them)

No non-prime attribute is transitively dependent on any key of R => The schema is in 3NF.

R2 = Student_Emegency_Contact (**Student ID**; emergency contact)

F2 = {Student ID → emergency contact; emergency contact → Student ID} - Min cover

The relational schema includes atomic attributes => The schema is in 1NF.

emergency contact and Student ID ar e both candidate keys=> The schema is in 3NF

R3 = Dormitory Staff (**Personal ID**; first name; middle name; last name; salary; email address; staff type; status; assigned dormitory)

F3 = {Personal ID → First Name; Personal ID → Middle Name; Personal → Last Name; Personal ID → salary; Personal ID → Email Address; Personal ID → Status; Personal ID → Assigned Dormitory; Email Address → Personal ID; Personal ID → Staff type} - Min cover

The relational schema includes atomic attributes => The schema is in 1NF.

Personal ID and email address are the candidate keys, they are all single-attribute keys => The schema is in 2NF.

No non-prime attribute is transitively dependent on any key of R => The schema is in 3NF.

R4 = Dormitory_Staff_PhoneNumber (**Personal ID**, phone number)

F4 = {Personal ID → Phone number; phone number → Personal ID} - Min cover

The relational schema includes atomic attributes => The schema is in 1NF

Personal ID and phone number are both candidate keys=> The schema is in 3NF

R5 = Dormitory manager (**Personal ID – DM**, manager ID)

F5 = {Personal ID- DM → Manager ID, Manager ID → Personal ID-DM} - Min cover

The relational schema includes atomic attributes => The schema is in 1NF.

Personal ID-DM and Manager ID are candidate keys=> The schema is in 3NF

R6 = Guard (**Personal ID – G**, guard ID)

F6 = {Personal ID-G - > guard ID, guard ID → Personal ID-G} - Min cover

The relational schema includes atomic attributes => The schema is in 1NF.

Personal ID- and Guard ID are candidate keys=> The schema is in 3NF

R7 = Office staff (**Personal ID – O**, staff ID)

F7 = {Personal ID-O - > staff ID,staff ID → Personal ID-O} - Min cover

The relational schema includes atomic attributes => The schema is in 1NF.

Personal ID-O and Staff ID are candidate keys=> The schema is in 3NF

R8 = (**Dormitory ID**, location, number of floors, overall condition, number of rooms, *Personal ID - G*)

F8 = {Dormitory ID → location, Dormitory ID → number of floors, Dormitory ID → overall condition, Dormitory ID → number of rooms, Dormitory ID →Personal ID - G, Personal ID - G → Dormitory ID} - Min cover

The relational schema includes atomic attributes => The schema is in 1NF.

Dormitory ID and Personal ID - G are the candidate keys, they are all single-attribute keys => The schema is in 2NF.

No non-prime attribute is transitively dependent on Dormitory ID => The schema is in 3NF.

R9 = Room (**Room ID**, amounts of students, condition, capacity, gender,*Dormitory ID*). (occupancy status and floor of room are derived attributes)

F9 = {Room ID → amounts of students, Room ID → condition, Room ID → capacity, Room ID → occupancy status, Room ID → Gender, Room ID → floor of room, Room ID → Dormitory ID, amount of students, capacity → occupancy status} - Min cover

The relational schema includes atomic attributes => The schema is in 1NF.

Room ID is the candidate key, all non-prime attributes depend fully on Room ID => The schema is in 2NF.

No non-prime attribute is transitively dependent on Room ID => The schema is in 3NF.

R10 = Bill (**Bill ID**, bill type, amount due, due date, payment status, *Student ID*, *Personal ID - O*) - Min cover

F10 = {Bill ID → bill type, Bill ID → amount due, Bill ID →due date, Bill ID → payment status, Bill ID →Student ID, Bill ID →Personal ID - O}

The relational schema includes atomic attributes => The schema is in 1NF.

Bill ID is the only candidate key, it is a single-attribute key => The schema is in 2NF.

No non-prime attribute is transitively dependent on Bill ID => The schema is in 3NF.

R11 = Request (**Request ID**, content, type, status, completion date)

F11 = {Request ID -> content, Request ID -> type, Request ID -> status, Request ID -> completion date) - Min cover

The relational schema includes atomic attributes => The schema is in 1NF.

Request ID is the only candidate key, it is a single-attribute key => The schema is in 2NF. .

No non-prime attribute is transitively dependent on Request ID => The schema is in 3NF.

R12 = Facility (**Item name**, overall quality,Room ID)

F12 = {Item name, Room ID → overall quality) - Min cover

The relational schema includes atomic attributes -> The schema is in 1NF.

Room ID, Item name is the candidate key, overall quality depends fully on candidate key = > The schema is in 2NF.

No non-prime attribute is transitively dependent on the candidate key => The schema is in 3NF.

R13 = Live (**Student ID, Room ID**, ***Personal ID – DM***)

F13 = {Student ID, Room ID → Personal ID - DM, Student ID → Room ID}- Min cover

The relational schema includes atomic attributes => The schema is in 1NF.

Student ID is the only candidate key, it is a single-attribute key => The schema is in 2NF.

No non-prime attribute is transitively dependent on Student ID => The schema is in 3NF.

R14 = Send (**Student ID, Personal ID - DM, Request ID,** send date)

F14 = {Request ID → Student ID; Request ID, Student ID → Personal ID - DM} - Min cover

The relational schema includes atomic attributes => The schema is in 1NF.

Student ID is the only candidate key, it is a single-attribute key => The schema is in 2NF.

No non-prime attribute is transitively dependent on Student ID => The schema is in 3NF.

R15 = Include (**Room ID,** Item name, quantity)

F15 = {Room ID, Item name →quantity} -Min Cover

The relational schema includes atomic attributes => The schema is in 1NF.

Room ID, Item name is the only candidate key, quantity is fully depend on key=> The schema is in 2NF

No non-prime attribute is transitively dependent on the candidate key => The schema is in 3NF.