CS2102 Report

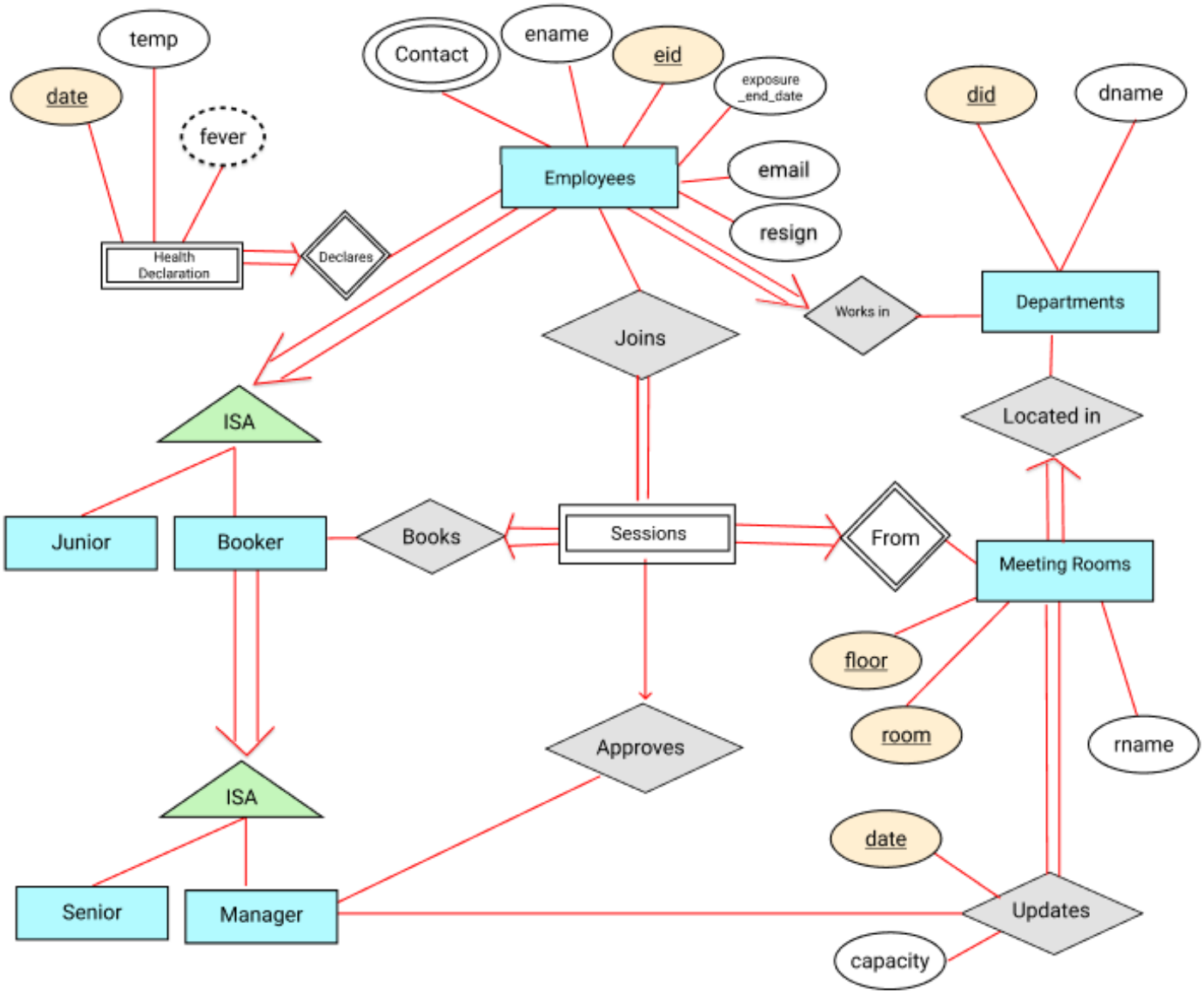
Group 88

[redacted]

1. Responsibilities

- [redacted]: ER Diagram, Health Functions, Populating Table, Report
- [redacted] ER Diagram, Basic Functions, Report
- [redacted]: ER Diagram, Admin Functions, Report
- [redacted]: ER Diagram, Core Functions, Report

2. ER Model



2.1 Design Decisions:

2.1.1 Employees

In the modern age, an employee may have multiple contact numbers. For instance, they may have landline and mobile phones. The use of multi-valued attributes is to reflect this but in practice, we will only allow up to three contact numbers:

1. Home phone number
2. Mobile phone number
3. Office phone number

2.1.2 Health Declaration

Health declaration is made into a weak entity set dependent on employees because each health declaration must be made by an employee. This allows the date to be a partial key. Additionally, fever is a derived attribute because we can set it automatically from temperature recorded.

2.1.3 Meeting Rooms

The meeting room has no attribute (derived or not) for capacity. An update provides the date as a partial key which can be used to distinguish the actual capacity at any given time. To ensure that a meeting room has a capacity, total participation constraints between meeting rooms and updates is added.

2.1.4 Booking and Session

A session is a weak entity set dependent on the meeting room to ensure that each meeting room can only have exactly one session with a session having a partial key of date and time. Here we have the assumption that time is the start time with each session lasting only one hour.

To ensure that a session must be created at booking time, a key and total participation constraints to books is added. Furthermore, to ensure that only a single approval is made, a key participation constraint to approve is added. Lastly, to partially ensure that the employee booking the room is to immediately join the meeting, a total participation constraints between employees and joins is added.

2.1.5 Junior, Senior and Manager

The two-level ISA hierarchy is used to first separate an employee that is allowed to book (i.e., a booker) and an employee that is not allowed to book (i.e., a junior). Secondly, we then separate a manager from a senior since only a manager can approve a meeting.

2.2 Constraints not captured:

1. Booking can only be made for future meetings
2. Employee can only join future meetings
3. Approval can only be made on future meetings
4. Each employee is given a unique email address
5. Constraints arising from having fever and being contacted

3. Schema

3.1 List of Schemas:

1. **Departments** with attributes *Department ID (did)* and *Department Name (dname)*
    a. *Department ID* as primary key

2. **Employees** with attributes *Employee ID (eid), Employee name (ename), Email (email), Home number (home), Phone Number (phone), Office Number (office), Resignation Date (resign), Department ID (did)*, and *last date of exposure (exposure_end_date)*
    a. *Employee ID* is the primary key
    b. *Department ID* is a foreign key that references **Departments**

3. **Junior, Senior, Manager and Booker**, all of which only have *Employee ID (eid)* as their sole attribute
    a. *Employee ID* is the primary key
    b. *Employee ID* is a foreign key that references **Employees**

4. **MeetingRooms** with attributes *Room Number (room)*, *Floor Number (floor)*, Room *Name (rname)*, *Department ID (did)*.
    a. (*Room Number*, *Floor Number*) is the primary key
    b. *Department ID* is a foreign key that references **Departments**

5. **Updates** with attributes *Date (date)*, *Room Number (room)*, *Floor Number (floor)*, latest set *room capacity (capacity)*, and *Employee ID of updating manager (eid)*
    a. (*Date*, *Room Number*, *Floor Number*) is the primary key
    b. (*Room Number*, *Floor Number*) is a foreign key that references **MeetingRooms**
    c. *Employee ID* is a foreign key that references **Manager**

6. **Sessions** with attributes *Timeslot, in 24 hour format (time), Date (date), Room Number (room), Floor Number (floor)*, *Employee ID of employee who booked the session (bid)* and *Employee ID of the employee who approved the session (approver).*
    a. *(Timeslot, Date, Room Number, Floor Number)* is the primary key
    b. *Employee ID* of booker is a foreign key that references **Booker**
    c. *Employee ID* of approver is a foreign key references **Manager**
    d. *(Room Number, Floor Number)* is a foreign key that references **MeetingRooms**

7. **Participants** with attributes *Employee ID (eid), Time (time), Date (date)*, *Room Number (room)* and *Floor Number (floor).*
    a. *(Employee ID, Time, Date, Room Number, Floor Number)* is the primary key
    b. *(Time, Date, Room Number, Floor Number)* is a foreign key that references **Sessions**

8. **HealthDeclaration** with attributes *Employee ID (eid)*, *Date (date), Temperature (temp)* and *Fever status (fever)*
    a. *(Employee ID, Date)* is the primary key
    b. *Employee ID* is a foreign key that references **Employees**
    c. *Fever Status* is a derived attribute from *Temperature*, which is TRUE if *Temperature* > 37.5 and FALSE otherwise

3.2 Non trivial designs:

1. Use of a **Participants** table to keep track of the participants within each session from the **Sessions** table
    a. To facilitate the checking of capacity when employees attempt to join meetings
    b. To facilitate contact tracing when one of the attendees catch a fever in the next three days

    This implementation is preferred over just adding an attribute *participants* to the **Sessions** table since employees in the same meeting room as a fever employee will be retrieved easily

2. **Sessions** and **Participants** store only the start time of the meeting, instead of the start and end times of the meeting. Additionally, it is assumed that each meeting starts and ends at the start of the hour (i.e. 0900 hrs, 1300 hrs, 1700 hrs, not 1230 hrs, 1645 hrs), and each session is one hour. That means
    a. A three hour meeting from 1200 hrs to 1500 hrs will be stored as three rows in the **Sessions** table, with *time* recorded as 12, 13 and 14 respectively
    b. An employee attending a three hour meeting from 1200 hrs to 1500 hrs will occupy three rows, with *time* recorded as 12, 13 and 14 respectively

c. A booker may have the flexibility to change a three hour meeting from 1200 hrs to 1500 hrs to a one hour meeting from 1200 hrs to 1300 hrs by simply unbooking the room from 1300 hrs to 1500 hrs, leaving one row in the **Sessions** table with *time* 12

3. **HealthDeclarations** store employees' temperatures over time so that the *non_compliance* function can find non-compliant employees by simply counting the number of rows in this table

4. A manager can reject a booking, using the *reject_meeting* function. This would then allow other bookers to book the room at that slot by removing said meeting from the **Sessions** table, instead of waiting for the initial booker to unbook the room

3.3 Constraints not captured:

1. Rooms from a removed department defaulting to HR (did = 0) upon deletion of department
2. Updating of room capacity can only be done by a manager from the room's department
3. Changing of room capacity will only be formalised in the future i.e a manager cannot change the capacity from a past date onwards.

Constraint 1 is captured in the *remove_department* function, 2 is captured in the *add_room* and *change_capacity* functions and 3 is captured in *change_capacity* function.

4. Triggers

1. **Updating Contact tracing (fever_check):** Trigger the execution of the function update contact_tracing after a personnel's health declaration indicates a fever positive on D-day. Firstly, the update_contact_tracing function will cancel all future bookings starting from D+1 made by the employee who had declared a fever, followed by a removal of the employee from all future bookings that he is going to participate in. Next, the function would update all the exposure_end_date of all close_contacts of this employee to D+7. Lastly, bookings made by these close contacts from D-day to D+7 would be removed. Close contacts will also be removed from all meetings from D-day to D+7. Note: Since employees can declare their temperatures at any time of the day, employees who are attending the same meeting as the fever personal any time on D-day will also be counted as close contacts.

2. **Employee resignation (resign_meetings)**: Created to remove employees from Sessions that they were supposed to participate in after they resign. After every update of the Employees table, the trigger checks if the update reflects a resignation, ie: having a resignation date under that employee's record. If so we delete records from the Participants table that represent sessions that the resignee was supposed to attend. Ie: eid = resignee eid, date same as or later than the resignation date. The trigger is to ensure that employees that are no longer part of the company should not be allowed to participate in further meetings.

3. **Changing of meeting room capacity (over_capacity)**: Created to remove Sessions that have participants count more than the updated capacity of that room when the capacity change comes into effect. After insert into Updates, we check whether the number of participants in each upcoming session that uses the recently updated room after the capacity change is effected is over the updated capacity. If yes, we remove the session. This is to ensure that the number of participants in every session that takes place meets the most up-to-date available capacity limit without affecting those sessions that take place before the official capacity update day.

5. Analysis of Normal Forms

1. **Departments**
    - By default, a table with 2 attributes is in BCNF.
    - Thus, **Departments** is in 3NF.

2. **Employees**
    - Keys: $\{eid\},\ \{email\}$
    - Prime Attributes: $\{eid,\ email\}$
    - Non-trivial and decomposed FDs on **Employees:**
        - All the FDs here are of the form
          $\{eid\} \rightarrow \{ename,\ email,\ home,\ phone,\ office, resign,\ exposure\ \ end\ date\}$
          or
          $\{email\} \rightarrow \{eid,\ ename,\ home,\ phone,\ office, resign,\ exposure\ end\ date\}$
        - In both instances, the left hand side is a super key
    - Thus, **Employees** is in 3NF.

3. **Junior, Senior, Manager** and **Booker**
    - No functional dependencies exist across the four tables since they each have only 1 attribute *(eid)*.
    - Thus, **Junior, Senior, Manager** and **Booker** are in 3NF by default.

4. **MeetingRooms**
    - Key: $\{room,\ floor\}$
    - Prime Attributes: $\{room,\ floor\}$
    - Non-trivial and decomposed FDs on **MeetingRooms:**
        - $\{room,\ floor\} \rightarrow \{rname\}$
            - $\{room,\ floor\}$ is a superkey
        - $\{room,\ floor\} \rightarrow \{did\}$
            - $\{room,\ floor\}$is a superkey
    - Thus, **MeetingRooms** is in 3NF.
5. **Updates**
    - Key: $\{room,\ floor,\ date\}$
    - Prime Attributes: $\{room,\ floor,\ date\}$
    - Non-trivial and decomposed FDs on **Updates:**
        - $\{room,\ floor,\ date\} \rightarrow \{capacity\}$
            - $\{room,\ floor,\ date\}$ is a superkey
        - $\{room,\ floor,\ date\} \rightarrow \{eid\}$
            - $\{room,\ floor,\ date\}$is a superkey
    - Thus, **Updates** is in 3NF.
6. **Sessions**
    - Key: $\{time,\ date,\ room,\ floor\}$
    - Prime Attributes: $\{time,\ date,\ room,\ floor\}$
    - Non-trivial and decomposed FDs on **Sessions:**
        - $\{time,\ date,\ room,\ floor\} \rightarrow \{bid\}$
            - $\{time,\ date,\ room,\ floor\}$ is a superkey
        - $\{time,\ date,\ room,\ floor\} \rightarrow \{approver\}$
            - $\{time,\ date,\ room,\ floor\}$is a superkey
    - Thus, **Sessions** is in 3NF.
7. **Participants**
    - Key: $\{eid,\ time,\ date,\ room,\ floor\}$
    - Prime Attributes: $\{eid,\ time,\ date,\ room,\ floor\}$
    - Since all attributes are prime attributes, **Participants** is in 3NF.

8. **HealthDeclaration**
    - $FD = \{\{temp\} \rightarrow \{fever\}, \{eid,\ date\} \rightarrow \{temp\}\}$
    - $\{temp\}^{+} = \{temp,\ fever\}$
    - Since $\{temp\}^{+}$is not trivial and $\{temp\}^{+} \neq \{eid,\ date,\ temp,\ fever\}$
    - $\{temp\} \rightarrow \{fever\}$ violates the BCNF property, and is not in 3NF
    - Thus we can decompose the table into $R1(eid,\ date,\ temp)$ and $R2(temp,\ fever)$which both satisfies the BCNF property.

6. Reflection

- Allan: The project has provided an opportunity for us to work with SQL language in a practical scenario. Through this project, I have learnt to plan, conceptualise and refine a database. In this long arduous journey, from creating the ER diagram, to conceptualizing the schema, implementing the functionalities and finally thinking of every possible scenario where a certain imperfection in implementation could cause inconsistency in the database, it was indeed a rewarding experience for my first database project (despite it being a rather small project). I enjoyed the process and would be looking to do more database projects as in the future.
- Gordon: The project has allowed us the opportunity to put theory learnt in the lectures into practical use as well as collaborative programming using Github. Being new to the SQL language, understanding the syntax and functionalities of it was an initial struggle but got better after more practice and googling while working on the project. It also highlights the importance of thorough planning especially when there are many components.
- Jasin: The team faced difficulties having to code the functions having been taught very simple examples in lectures, for triggers in particular. Of course, this has allowed us to gain a better understanding of the formatting and using of these functions. Another prominent difficulty was also having to account for all the pieces of information given to us (in the project.pdf file) when implementing the schemas, functions and ER diagram. I guess the lesson from this would be that the implementation of a database requires the person implementing it to have a clear idea of how the processes in the database works e.g a change in a table prompts what other changes in which other tables, as well as having rational thinking when accounting for ‘unlikely’ events.
- Keith: The project allowed me to apply what was taught in lecture to use. Only after writing the SQL queries hands-on, I realise I may not have understood certain concepts that well. With the project, I now have a better understanding of how schemas and relations between entities and attributes are important, and planning a database that is representative of the information we have.