**Problem Set 1**

**Problem 1: A real-world database application**

1-1)

The train ticket booking could use a database system to store the data of passengers, trains, and bookings. All data values are stored as binary numbers. Central Processing Unit includes hardware for processing data stored in binary form, but it can only store a small amount of data. We can store programs and other data that are currently used in Memory, which only requires short access time, but the data will lose when the power is turned off. We can also use Secondary Storage like hard disks to store data for later use, but it requires long access times.

1-2)

Database system could translate the bytes on disk blocks and provide a logical presentation like tables of the data. It could make the users who may not be familiar with computer science easily understand. For example, Database system could build tables to record the information like the passenger's id, name, and birthday; the trains' number, origin, destination, time to departure, and availability of seats; and capture the relationship between the passengers and trains in a booking table. Therefore, the database system allows people to keep track of all booking information about the trains, including who booked the train tickets and which train the person was booked.

1-3)

Database system organizes the data on disk in ways that allow it to reduce the number of disk accesses, which can make the storage more efficient. We can use query language like SQL to add or modify the information of passengers and trains. It can also be used to retrieve data according to some criteria, like finding a passenger's booked train according to his id. Moreover, the transaction management could prevent the situation that a person booked a train ticket but there is actually no seat available on that train. By using transaction management, we ensure that all of the steps happen, or none do.

**Problem 2: Keys**

2-1)

id for Department

The id attribute in the table Department can be used to uniquely identify a row and is a minimal collection of attributes. We can use this value for quick retrieval.

id for Person

The id attribute in the table Person can be used to uniquely identify a row and is a minimal collection of attributes. We can use this value for quick retrieval.

(dept, person) for Chairs

The combination of dept and person can be used to uniquely identify a row. None of them is unnecessary because a department may have more than one people who chair it, or a person may chair multiple departments.

2-2)
Yes, because the combination can be used to uniquely identify a given row. The id alone is already unique because it is a primary key, so adding the office to it still makes it a unique identifier.

2-3)
No, because the office attribute is unnecessary. The id alone is already unique because it is a primary key, so we can remove the office and still be left with a key. Candidate key means none of the attributes are unnecessary.

2-4)
In the Chairs table:
dept is a foreign key that refers to the primary key (id) of the Department table.
person is a foreign key that refers to the primary key (id) of the Person table.

2-5)
Adding (100, mathematics, “8 Cummington St.”) to the Departement table would violate a uniqueness constraint because there is already another department with the id 100, and the id must be unique because it is a primary key.

2-6)
Adding (100, 5555) to the Chairs table would violate a referential integrity constraint because there is no person with id 5555 in the Person table. The value in a foreign-key column must match one of the values in the corresponding primary-key column.

**Problem 3: Schema for a hospital database**
3-1)
Adding a room_number attribute into the Patient table. It is a foreign key that refers to the primary key (number) of the Room table. Because a given patient is assigned to only one room, the id in the Patient table will still be unique and act as the primary key.

3-2)
It is necessary to create a separate table to capture the relationships between patients and doctors. As a given doctor can have multiple patients, and a given patient can have more than one doctor, there would be multi-valued attributes if we just add a patient attribute into the Doctor table or add a doctor attribute into the Patient table. However, each cell in the table must contain a single value.

3-3)
Treatment (<u>patient</u>, <u>doctor</u>)

In the Treatment table, patient is a foreign key that refers to the primary key (id) in the Patient table, and doctor is a foreign key that refers to the primary key (id) in the Doctor table. Each value of the patient attribute must match a value of the id attribute from the Patient table, and each value of the doctor attribute must match a value of the id attribute from the Doctor table.