Part 1. Keys

Below is a relation R containing attributes A, B, C.

A	B	C
α	δ	ξ
β	δ	ψ
γ	η	ξ

(a) Identify all superkeys of R.

Response: $\{A, AB, BC, ABC\}$

(b) Identify all candidate keys of R.

Response: $\{A, BC\}$

Below is a relation called `Employee` containing some data about student employees at UCLA Recreation. The EmployeeID is created and controlled by UCLA Recreation and is always used by only one employee at a time, and is retired (never re-used) after an employee leaves the department. Each student employee also has a UID issued by the University.

EmployeeID (A)	EmployeeName (B)	UID (C)	ManagerID (D)
1	John Smith	123456789	9
2	Sarah Decker	242424246	3
3	John Smith	987654321	9
4	Jane Allison	112358132	1

Below is a relation called `Facility` containing data about recreation facilities and who manages each one. Each manager has an ID number that has the same format as the EmployeeID. A matter of fact, a ManagerID *is* an EmployeeID. The primary key of this relation is Facility (A).

Facility (A)	ManagerID (B)
Wooden Center	1
Sunset Canyon	3
Drake Stadium	9

(c) Identify all candidate keys of `Employee` in this instance.

Response: $\{A, C\}$

(d) Based on your answer from part (c), what is the full set of superkeys for `Employee` in this instance and why? *Be careful.* Explain.

Response: $\{A, C, AB, AC, AD, BC, CD, ABC, BCD, ABCD\}$. The candidate keys of `Employee` are $\{A, C\}$ so all supersets of A, C are superkeys.

(e) Which attribute or set of attributes would you choose to be the primary key of `Employee` in this instance and why? *Be careful.* Explain.

Response: EmployeeID, because it is the minimal candidate key and enforced to be unique by the problem statement (EmployeeID is always used by only one employee at a time and is retired (never re-used) after an employee leaves the department).

(f) Suppose an analyst decides to choose both `EmployeeID` and `UID` as a composite primary key for `Employee` since both are unique. Explain why this is not correct. **Hint:** Think about what this might imply about each attribute in the primary key and what data integrity issues can arise.

Response: Because `Employee` and `Facility` are related, if we use the chosen composite key, there is no possible foreign key from `Facility` into `Employee` since `Facility` does not have a `UID` attribute. So, referential integrity is broken and the two relations can no longer be linked by a foreign key.

(g) Identify all foreign keys in context of the problem (assume that all tables have more rows). How do these foreign keys help us ensure referential integrity in this context? Note that there may be more than you initially believe.

Response: Let $\mathcal{E}, \mathcal{F}$ be the relations for `Employee`, `Facility` respectively. Then $D \in \mathcal{E}$ and $B \in \mathcal{F}$ are foreign keys into $A \in \mathcal{E}$. Also, $A \in \mathcal{E}$ is a foreign key into $B \in \mathcal{F}$. By the problem statement, a ManagerID *is* an EmployeeID. Therefore, we ensure referential integrity because if we reference a ManagerID α in either relation, the employee with EmployeeID α must exist in `Employee`.

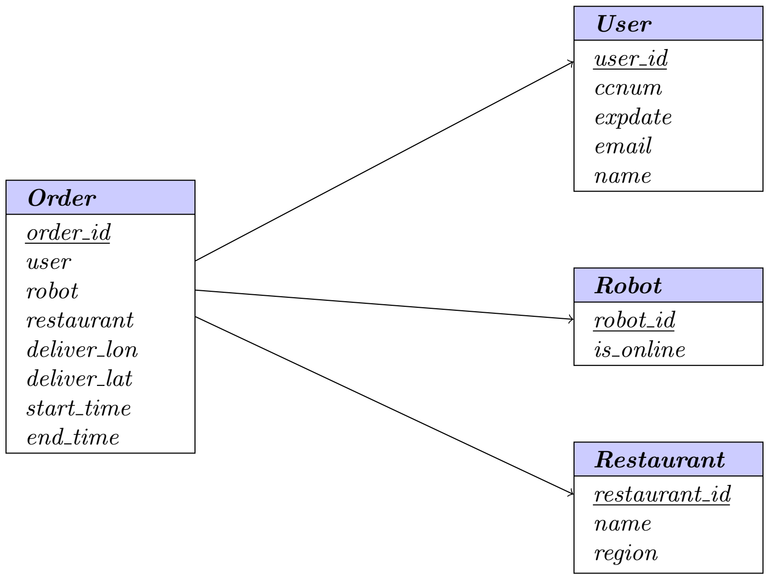
Part 2. Schema Diagram Suppose you are working for the data team at Starship, a San Francisco based startup that aims to revolutionize food delivery. How the Starship service works: a user installs an app on their phone and places an order at a participating restaurant within a geofenced region (e.g. UCLA campus). The user enters their credit card information, orders their meal and selects where they want the food delivered. A code is provided so that only the correct user can open the lid and retrieve their food. Due to limited supplies of robots, and high demand, the company wants to experiment with charging users based on the amount of time it takes to deliver the food from the restaurant to the user.

We have the following relations for managing robots, users and orders:

```
Robot(robot_id, is_online)
# robot_id is some value that identifies a robot.
# is_online just indicates whether or not the robot is able to be used.

User(user_id, ccnum, expdate, email, name)
# user_id is some value that identifies the user
# ccnum is the user's credit card number (hashed, hopefully!) and may be null, and may change.
# expdate is the credit card expiration date and may also be null, and may change.
# email is the user's email address.
# name is the, well, user's name

Order(order_id, user, robot, restaurant, deliver_lat, deliver_lon, start_time, end_time)
# order_id is some value that identifies the order.
# user uniquely identifies the user that placed the order.
# robot uniquely identifies the robot that delivered the user's food.
# restaurant identifies where the user ordered food.
# deliver_lon is the longitude of the delivery location
# deliver_lat is the latitude of the deliver location
# start_time and end_time represent the start and end times of the delivery.
# we will ignore the details of the order for now.

Restaurant(restaurant_id, name, region)
# restaurant_id is some value that identifies the restaurant.
# name is the name of the restaurant
# region is the name of the geofenced region where the restaurant is located (e.g. UCLA)
```

Hint: There is a lot written above; however, remember that we are interested in **keys**.

Exercise. Using these relations, draw the schema diagram. For an example of a schema diagram, see Figure 2.8 in the text (page 47), or the end of the Intro to the Relational Model section in the lecture slides for Lecture 2. Clearly denote primary and foreign keys (note that primary keys were intentionally withheld in the specification above).



The primary key for `Order` is `order_id`.
The primary key for `User` is `user_id`.
The primary key for `Robot` is `robot_id`.
The primary key for `Restaurant` is `restaurant_id`.
`user` in `Order` is a foreign key into `user_id` in `User`.
`robot` in `Order` is a foreign key into `robot_id` in `Robot`.
`restaurant` in `Order` is a foreign key into `restaurant_id` in `Restaurant`.