Relational Schema

User = (username, first\_name, last\_name, birthdate, address)

Employee = (username[fk1], taxID, hired, salary, experience)

fk1: username ->User. Username

Owner = (username[fk2], debt)

fk2: username ->User. Username

Pilot = (username[fk3], licence\_type, experience)

fk3: username ->Employee.username

Worker = (username[fk4])

fk4: username ->Employee.username

Restaurant = (name, spent, rating, location[fk5]

fk5: location -> Location.label

Service = (ID, name, revenue, location[fk6])

Fk6: location -> Location.label

Location = (label, x\_coord, y\_coord, space)

Ingredient = (barcode, iname, weight)

Drone = (serviceId[fk7], tag, fuel, cost, capacity, sales, weight, hover[fk12], flown\_by[fk13], swarmDroneID, swarmDroneID[fk20])

fk7: serviceId -> Service.ID

fk12: hover -> Location.label

fk13: flown\_by -> Pilot.username

fk20: swarmDroneID, swarmDroneID -> Drone.serviceId, Drone.tag

Fund = (Provider[fk8], Restaurant[fk9], invested, dt\_made)

fk8: provider -> Owner.username

fk9: Restaurant -> Restaurant.name

Contain = (DroneSId, DroneTag [fk16], ingredient[fk17], quantity, price)

fk16: DroneSId, DroneTag -> Drone.serviceId, Drone.tag

fk17: ingredient -> Ingredient.barcode

works\_for = (employee[fk18], employed\_by[fk19])

fk18: employee -> employee.username

fk19: employed\_by -> service.ID

Unhandled Constraints

* Ensure all users are either owners or employees
* Ensure each delivery service has at most a sole manager
* Ensure employee tax identifiers are stored using a “xxx-xx-xxxx” format
* Ensure each pilot has a valid license type to operate the drone safely
* Ensure pilots can’t fly drones for more than one delivery service at a time
* Ensure each delivery service employs one or more workers
* Ensure an employee cannot be a worker and a pilot at the same time
* Ensure a manager of a delivery service is also employed by that service
* Ensure an employee can manage only one delivery service at a time
* Ensure each drone belongs to a single service
* Each drone must be identified relative to the service it supports
* Ensure a drone can only be controlled by a single pilot at one time
* Ensure drones in a swarm always stay together
* Ensure each drone moving from one location to the next consumes fuel based on distance
* Ensure the home base is the only location where the drone can be refueled/restocked/repaired
* Ensure a drone can only move to a location if there’s enough space for it to maneuver safely
* Ensure restaurants can purchase ingredients from a drone only when it’s at the restaurant