Drone Dispatch! Express Delivery

CS 4400: Introduction to Database Systems
Course Project: Spring 2024 Semester

TEAM 9

List of Assumptions

Aspect of Scenario Description	Assumption/Clarification	Design Choice(s) and Impact
• A drone must be identified relative to the specific store that it serves.	Drone can be uniquely identified using a combination of store (owner) and drone number (may be same for different drones)	DRONE is a weak entity with an identifying relationship OWNS with STORE. It has a partial key Drone_number which together with the Store_id key of STORE makes a composite key for DRONE.
• Each drone pilot can control one drone at a time • A drone must be controlled by one pilot at a time	Drone pilot can be idle but Drone cannot be idle. Consequently, a drone must be assigned to at least one order.	The DRONE_PILOT-DRONE PILOTS relationship has cardinality 1:1 with partial participation of DRONE_PILOT and full participation of DRONE. The DRONE-ORDER ASSIGNED_TO relationship has cardinality 1:N with full participation of DRONE and ORDER.
• Each drone has been purchased/sponsored by a single store, and is used to serve (e.g., carry orders for) that store alone	There's a possibility that a store doesn't own any drones.	The STORE-DRONE OWNS relationship has partial participation of STORE.
• An order will consist of one or more lines, where each line represents a certain quantity (i.e., one or more) of a product being ordered at a given unit price • Each product can be listed at most once on a given order	Each entry of a product in an order (assumed to be non-empty) is uniquely identified by the composite key (Receipt_id,Barcode). This ensures that one product can be in multiple orders but exists at most once in a particular order.	Instead of adding a separate ORDER_LINE entity for each entry in ORDER, the ORDER-PRODUCT CONTAINS relationship has cardinality M:N with partial participation of PRODUCT and full participation of ORDER. The relationship has two stored attributes - Quantity and Unit_price, and one derived attribute Cost (Quantity*Unit_price)

• We must keep track of the number of trips that a drone has remaining before it needs maintenance	It is only important to track the number of trips remaining before maintenance and this value is updated with each trip.	A stored attribute- Trips_remaining for the DRONE entity.
• We must also keep track of the number of successful deliveries (as an integer) for each pilot	We need to update the number of successful deliveries after each successful trip for a pilot.	A stored attribute- Successful_deliveries for the DRONE_PILOT entity. This gets updated each time an order is successfully delivered.
• Our system must be able to calculate and display the cost of an order as the total cost of each line • Our system must be able to calculate and display the total cost for all of the outstanding orders for each customer • Our system must be able to calculate and display the weight of an order as the total of the weight of each line, which is the weight of the individual product multiplied by the quantity purchased.	The outstanding cost of orders for each customer can be calculated from all of their outstanding orders. An order is either outstanding or successful (unsuccessful is not an option). Total cost and weight of all orders can be calculated.	A stored attribute- Status for the ORDER entity which shows whether an order is outstanding or not. Two derived attributes- Total_cost (sum of Cost of all products in an order) and Total_weight (sum of Quantity*Weight for all products in an order) for the ORDER entity. A derived attribute- Cost_outstanding for the CUSTOMER entity which is the sum of Total_cost of all outstanding orders. Credit must be more than or equal to Cost_outstanding.
• Our system must be able to calculate and display the total weight (i.e., payload) for all of the orders being delivered by each drone.	We don't need to explicitly track the payload as it can be calculated from the sum of orders' Total_weight attribute.	A derived attribute- Payload for the DRONE entity which is the sum of Total_weight of all its orders. This must be less than or equal to its Capacity.
• Our system must be able to calculate and display the incoming revenue for each store as the total cost of all orders currently being delivered by drones on behalf of that store.	The revenue only comes from delivery of orders. There is no other source.	A derived attribute- Revenue for the STORE entity which is the sum of Total_cost of all its orders.