Brandon Wheeless & Peter Sun

CSE3330 Project 1

How do these parking lots work?

Default: The parking lots have one entry point. The entry point has 3 employees each, checking incoming cars for their tickets. Assuming all people in each car do have tickets, they park in a parking spot. Cars can have handicap signs so those are able to park in handicap spots. However, this does not mean they have to and there are cars with handicap signs that just park in normal spots as well.

Define and organize the data

Employee: One entry point has many employees

Entrypoint: One entry point has one parking lot and vice versa

Event: One event can only take place at one stadium but one stadium has many events

Parking_lot: One parking lot needs one entry point to let people in. Also, one stadium has many parking lots but one parking lot can only have one stadium

Parking_space: One parking space has one car and one car has one parking space, one parking space has one parking lot but one parking lot has many parking spaces

Stadium: 1:many relationship with parking lot, 1:many with event

Ticket: One ticket can only have one vehicle and one vehicle can only have one ticket

Vehicle: 1:1 relationship with parking space, 1:1 relationship with ticket

Define the attributes

Employee
* First name (varchar(10))
* Last name (varchar(10))
* PRIMARY KEY: First and last name Since this combination of names is unique to each individual
* Entry point (char) (FOREIGN KEY) Since employees are always working at a specific entry point and one entry point has many employees

Entrypoint
* parking_lot_id (char) (FOREIGN KEY) Since one entry point has one parking lot and vice versa
* entrypoint_id (char) (PRIMARY KEY) Since this is how each entry point is uniquely defined
* entrypoint_name (varchar(20))

Event
* event_id (serial4) (PRIMARY KEY) Because this is used to keep track of each individual event
* event_type (varchar)

* stadium_id (serial4) (FOREIGN KEY) Because each event can only take place at one stadium but one stadium has many events

Parking_lot
* name (char) (PRIMARY KEY) Because each parking lot needs a unique identification
* entrypoint_id (serial) (FOREIGN KEY) Because one parking lot needs one entry point to let people in
* stadium_id (serial) (FOREIGN KEY) Because one stadium has many parking lots but one parking lot can only have one stadium

Parking_space
* spot_number (int) (PRIMARY KEY) To identify each unique spot
* is_available (bool)
* is_handicap (bool)
* vehicle_id (varchar) (FOREIGN KEY) Because one parking space has one car and one car has one parking space
* lot_id (char) (FOREIGN KEY) Because one parking space has one parking lot but one parking lot has many parking spaces

Stadium
* id (serial4) (PRIMARY KEY) To identify each stadium
* address (varchar(30))
* name (varchar(15))

Ticket
* id (int4) (PRIMARY KEY) To identify each unique ticket
* vehicle_id (varchar) (FOREIGN KEY) Because one ticket can only have one vehicle and one vehicle can only have one ticket

Vehicle
* license_plate (varchar(8)) (PRIMARY KEY) To identify each unique vehicle
* type (varchar(5))
* is_handicap (bool)

Think outside the box

Scenario 1: Consider the event that parking lots produce high amounts of traffic due to only having 1 entry point that also acts as the exit. For future renovations to the parking lots, the owner may consider adding additional entry points for each lot so ensure the system can support adding new entry points with corresponding data and rules that come with it.

Scenario 2: For future developments of the parking lot, some customers may enjoy the ease of valeting a car at the front of an entry point compared to searching for a designated spot. For these individuals design and implement a valet driver that has a many to many relationships with vehicles. These valet drivers will drive multiple cars and these cars will have multiple drivers.