**Project One:**

**Possible Entities:**

1. **Stadium** (Holds all of the Below)
2. **Event** (Takes place in Stadium)
3. **Parking\_Lot** (In Stadium, has Spaces)
4. **Parking\_Space** (Spaces within Lot)
5. **Employee** (Information, First\_name, Last\_name)
6. **Vehicle** (Information, Type)
7. **Ticket** (Fan, Vehicle, Etc.)
8. **Schedule** (Work Schedules for Employees)
9. **Fan** (Information relating to the Above)

**Possible Relationships: (\* Means Completed)**

1. Stadium and Event **(1: n)** *The stadium can only hold multiple events. \**
2. Stadium and Parking\_Lot **(1: n)** *One stadium can have many parking lots. \**
3. Event and Ticket **(1: n)** *An event will sell many tickets. \**
4. Event and Schedule **(1: n)** *An event has many different schedules. \**
5. Ticket and Parking\_Lot **(1: n)** *A parking lot has many associated tickets. \**
6. Parking\_Lot and Parking\_Spot **(1: n)** *A parking lot has many parking spots. \**
7. Parking\_Lot and Employee **(1: n)** *A parking lot has many employees working. \**
8. Parking\_Lot and Schedule **(1: n)** *A parking lot has many different schedules. \**
9. Employee and Schedule **(n: n)** *Many employees can have many schedules. \**
10. Employee and Parking\_Spot **(1: n)** *An employee allocates multiple parking spots. \**
11. Vehicle and Parking\_Lot **(1: 1)** *A car will be in a parking lot. \**
12. Vehicle and Parking\_Spot **(1: 1)** *A car will be in a parking spot. \**
13. Fan and Ticket **(1: 1)** *A fan will have one ticket. \**
14. Fan and Vehicle **(1: 1)** *A car will be associated with a fan. \**

**Define Attributes:**

**Stadium: (stadium)** *Domain: Highest Level Entity, Building*

* id SERIAL PRIMARY KEY *Simple Incrementing ID*
* name VARCHAR(50)
* address VARCHAR(50)
* capacity INTEGER

**Event: (event)** *Domain: Stadium*

* id INTEGER PRIMARY KEY *Simple ID (Not Incremented)*
* name VARCHAR(30)
* date DATE
* num\_sold INTEGER
* stadium\_id REFERENCES stadium(id) *Foreign Key \**

**Parking Lot: (parking\_lot)** *Domain: Event, Carpark*

* id SERIAL PRIMARY KEY *Simple Incrementing ID*
* num\_spots INTEGER
* stadium\_id REFERENCES stadium(id) *Foreign Key \**

**Schedule: (schedule)** *Domain: Event, Employee Work/Job*

* schedule\_num INTEGER PRIMARY KEY *Schedules need to be Identifiable*
* start\_time TIME
* end\_time TIME
* lot\_id REFERENCES parking\_lot(id) *Foreign Key \**
* event\_id REFERENCES event(id) *Foreign Key \**

**Employee: (employee)** *Domain: Parking\_Lot, Job Location, Parking Assistance*

* id INTEGER
* name VARCHAR(50)
* PRIMARY KEY (id, name) *Composite Key => Name gives more Information*
* schedule\_num REFERENCES schedule(schedule\_num) *Foreign Key \**
* lot\_id REFERENCES parking\_lot(id) *Foreign Key \**

**Parking Spot: (parking\_spot)** *Domain: Parking\_Lot*

* spot\_num INTEGER PRIMARY KEY *Spot Numbers Usually Start Parking\_Lot ID*
* is\_available BOOLEAN
* is\_handicap BOOLEAN
* employee\_id INTEGER
* employee\_name VARCHAR(50)
* FOREIGN KEY (employee\_id, employee\_name) REFERENCES employee(id, name)
* lot\_id REFERENCES parking\_lot(id) *Foreign Key \**

**Ticket: (ticket)** *Domain: Event, Specific Event Tickets*

* ticket\_num INTEGER PRIMARY KEY *As in Stadium Seat Location (538)*
* price FLOAT
* event\_id REFERENCES event(id) *Foreign Key \**
* lot\_id REFERENCES parking\_lot(id) *Foreign Key \**

**Vehicle: (vehicle)** *Domain: Automobiles, Parking\_Lot*

* license VARCHAR(30)
* type VARCHAR(30) *(Coupe, Sedan, Hatchback, SUV, Minivan, Van, Truck, RV)*
* PRIMARY KEY (license, type) *Composite Key => Type help in Identification*
* lot\_id REFERENCES parking\_lot(id) *Foreign Key \**
* spot\_num REFERENCES parking\_spot(spot\_num) *Foreign Key \**

**Fan: (fan)** *Domain: People, Ticketed People*

* ssn INTEGER PRIMARY KEY *Something that everyone can be Identified by*
* name VARCHAR(50)
* car\_license VARCHAR(30) NOT NULL
* car\_type VARCHAR(30) NOT NULL
* FOREIGN KEY (car\_license, car\_type) REFERENCES vehicle(license, type)
* ticket\_num REFERENCES ticket(ticket\_num) *Foreign Key \**

**Think Outside The Box:**

**Scenario One:** Employees have many jobs, whether that be in the Parking Lot or in Concessions. These jobs include schedules, names, prices, and more. Basically, one could add a Job object that references schedule and employee.

**Job: (job)**

* id INTEGER PRIMARY KEY *Simple ID (Not Incremented)*
* title VARCHAR(30) *Title of Job*
* type VARCHAR(30) *Type of Work (Sales, Concessions)*
* *FOREIGN KEY From Employee to Job* **(1: n)**
* schedule\_num REFERENCES schedule(schedule\_num) *FOREIGN KEY* **(n: n)**

Of course, in doing this, we would need to add more relations in the other entities listed.

**Scenario Two:** Some Fans do not have vehicles or take public transportation to get to the Event. This means that many Fans will have tickets that do not have a lot number associated to them. To account for this, one could create a Transportation Entity that allows the database to store Fans with Cars, and leave out those taking other means of Transportation.

**Transportation: (transportation)**

* id VARCHAR(30) *Bus Route Number, Train Route Number*
* type VARCHAR(30) *Type of Transportation*
* PRIMARY KEY (id, type) *Composite Key => Most possible Information*
* *FOREIGN KEY From Fan to Transportation* **(n: n)**

Of course, in doing this, we would need to add more relations in the other entities listed.

**Extra Questions:**

1. **How many total tickets have been sold for event 1?**

SELECT COUNT(\*) FROM ticket;

1. **How many schedules can employees have?**

SELECT COUNT(\*) FROM schedule;

1. **Which employees have been assigned to which schedules, and what are times?**

SELECT name, employee.schedule\_num, start\_time, end\_time FROM employee

JOIN schedule on employee.schedule\_num = schedule.schedule\_num;