Customers are categorized by their respective IDs. A customer can buy 0 or as many tickets as they want. Tickets are associated to a minimum and maximum of 1 customer, because 1 ticket provides 1 entry for a customer to watch a movie.

Customers		
Primary Key	customer_id	SERIAL
	full_name	VARCHAR(100)
	date_visited	TIMESTAMP + TZ

Tickets		
Primary Key	ticket_id	SERIAL
	price	INTEGER
Foreign Key (customer)	customer_id	INTEGER
Foreign Key (movie)	movie_id	INTEGER

Every ticket is for one movie showing. If a movie_id represents a showing of one movie, this means that 1 ticket requires a minimum and maximum of 1 movie_id. On the flip side, a movie can sell 0 tickets (box office bomb), or infinite tickets (Summer blockbuster).

Movies		
Primary Key	movie_id	SERIAL
	title	VARCHAR(50)
	description	VARCHAR(100)

A movie can have 0 or many showings, perhaps depending on how popular it is or it's release date. However, one showing requires at least one movie_id, and since you can't show more than 1 movie at once, 1 is also the maximum.

To connect Movies and theatre showtimes together, I had to make a Movie Showings table.

Theatres		
Primary Key	theatre_id	SERIAL
	IMAX	BOOL
	seating	INTEGER
Foreign Key (movie)	movie_id	INTEGER
Foreign Key (showtimes)	showtimes	SET

A specific movie theatre can only show one movie at a time. So the minimum and maximum movie IDs it takes in is one. However, a movie can be shown in 0 or many theatres.

Movie Showings	
Foreign Key (movie)	movie_id
Foreign Key (theatres)	theatre_id
Foreign Key (showtimes)	showtimes

This table was created because the same movie can be shown in multiple theatres at multiple times. Therefore, to store data that tells us what theatres are playing what movies at specific times, I made a joined table with the movie showings table. It takes in 0 or many movies, 0 or many theatres, and 0 or many showtimes.