Table creation statements

Changelog

1. deleted videotape
2. added stock and videoID to movie

```sql
create table movies (
videoID int NOT NULL AUTO_INCREMENT,
title varchar(100) NOT NULL,
year int NOT NULL,
cat varchar(100) NOT NULL,
introduction varchar(255),
directors varchar(255),
producers varchar(255),
stock int NOT NULL,
PRIMARY KEY(videoID)
);
```

```sql
create table orderItems (
orderID int NOT NULL,
videoID int NOT NULL,
quantity int NOT NULL,
FOREIGN KEY(orderID) references customerOrders(orderID),
FOREIGN KEY(videoID) references movies(videoID)
);
```

```sql
CREATE TABLE customerOrders (
orderID int NOT NULL AUTO_INCREMENT,
dateOrdered DATE NOT NULL,
shippingDestination varchar(40),
period int,
status varchar(10),
returnTrackingInformation varchar(40),
email varchar(50) NOT NULL,
PRIMARY KEY(orderID),
FOREIGN KEY(email) REFERENCES customers(email)
);
```

```
CREATE TABLE customers (
email varchar(50) NOT NULL,
password varchar(50) NOT NULL,
loyaltyPoints int NOT NULL,
creditCardNum varchar(15),
address varchar(50),
phoneNumber varchar(50),
PRIMARY KEY (email)
);
```

```
CREATE TABLE managers (
email varchar(50) NOT NULL,
password varchar(50) NOT NULL,
PRIMARY KEY (email)
);
```

```
CREATE TABLE ratings (
criticName varchar(50) NOT NULL,
videoID int NOT NULL,
PRIMARY KEY(criticName),
FOREIGN KEY(videoID) REFERENCES movies(videoID)
)
```

Pre-inserted data

Movies

```
insert into movies (title, year, cat, introduction, directors, producers,
stock) values (
'The Thing',
1982,
'Horror',
'Kurt Russel and friends fight an alien',
'John carpenter',
```

```
'David Foster',
5
);
```

```
insert into movies (title, year, cat, introduction, directors, producers,
stock) values (
'Alien',
1979,
'Horror',
'Sigourney Weaver and friends fight an alien',
'Ridley Scott',
'Walter Hill',
5
);
```

Customers

note: @ symbol was rejected may need to tweak something here

```
insert into customers values (
'songwangatgmail.com',
'p455w0rd',
0,
'1234-1234-1234'
);
```

customerOrders

```
insert into customerOrders values (
'October 31st 2020',
'200 Crescent Avenue, Toronto Ontario',
20,
'Unpaid',
'Unspecified',
'songwangatgmail.com'
)
```

- Customer renting some movies for 20 days

- Need to also create seperate orderItems for this order
- ID is 0 for the orderID

orderItems

- adds two video tapes to song wangs order

```
insert into orderItems values (
0,
0
);
```

```
insert into orderItems values (
0,
1
);
```

Useful queries (some of this is deprecated because I removed the videotape entity)

1. Retrieve all the movies the service has regardless of the number of tape copies.

```
select * FROM movie group by title;
```

Returns

title	year	cat	introduction	directors	producers
The Thing	1982	horror	introduction	directors	producers
Alien	1979	horror	..	..	..

2. Retrieve the number of copies of a certain film regardless of rental status.

```
select count(title) FROM videotape WHERE title='Alien' AND
year='1979'
```

3. To retrieve all orderItems from an order.

```
SELECT * FROM orderItem WHERE orderID = [orderID];
```

4. Retrieve all available copies of films (everything that isn't rented)

```
select t1.title, t1.year, t1.videoID FROM videotape t1
WHERE videoID NOT IN (SELECT videoID FROM orderItem);
```

- This query is inefficient but works

5. Check if there is an available copy of a certain film. In this case, checks if there is a copy of alien.

```
select t1.title, t1.year, t1.videoID FROM videotape t1
WHERE t1.title ='Alien' AND t1.year = '1979' AND videoID
NOT IN (SELECT videoID FROM orderItem);
```

- This query is inefficient but works
- Returns all available video IDs of alien

6. Return all available movies with the number of available copies.

```
select t1.title, t1.year, COUNT(title) FROM videotape t1
WHERE t1.videoID NOT IN (SELECT videoID from orderItem)
```

```sql
GROUP BY t1.title, t1.year;
```

title	year	count
Alien	1979	1