**Order:** orderID (PK), customerID (FK), supplierID (FK), itemID (FK)

**Customers:** customerID (PK), cName, address

**Supplier:** supplierID (PK), sName, phone

I**tem:** itemID (PK), iName, price

**NewsLetter:** newsletterID (PK), nlName, customerID (FK)

**theoritical questions** - 7 sub true false questions

5a. read about the keys, and their relationships

5b. foreign key

5c. deadlock

Diagram

Description automatically generated

5d. performance of the query; optimal way? clauses? functions?

5e. where, order by, group by - their uses

5f. basic table structure - column, row

5g. indexes

**CURSOR CODE**

drop procedure if exists starRating;

delimiter $$

create procedure starRating()

begin

declare aName varchar(50);

declare num int;

declare cnt int;

declare sRating int;

declare prevnum int;

declare myCursor cursor for select concat(first\_name," ", last\_name) as fullname, count(film\_id) as count1, star\_rating from actor a inner join film\_actor fa

on a.actor\_id = fa.actor\_id

group by fullname

order by 2 desc;

open myCursor;

set num = 0;

set sql\_safe\_updates = 0;

myLoop:loop

fetch myCursor into aName, cnt, sRating;

if (cnt = prevnum) then

set sRating = num;

update actor set star\_rating = sRating where concat(first\_name," ", last\_name) = aName;

else

set num = num+1;

set sRating = num;

update actor set star\_rating = sRating where concat(first\_name," ", last\_name) = aName;

set prevnum = cnt;

end if;

end loop;

close myCursor;

set sql\_safe\_updates = 1;

end $$

delimiter ;

call starRating();

Graphical user interface, text, application

Description automatically generated