COMP518 Assignment 1
Saeth Wannasuphoprasit
Student ID: 201585689

1 Question One (50%=50 marks)

Consider the following relational database schema where the underline attributes are the primary keys,

- Lecturer(<u>lId</u>, name, title, address)
- Student(<u>sId</u>, name, major)
- Course(<u>cId</u>, cName, lId)
- Enrolment(<u>cId, sId</u>)

Write relational algebra expressions for the following queries.

1. **(5 marks)** Find the lecturers who teach at least one course. Show the lIds of those lectures.

Case 1: no null in LId in Course
We can directly project lecturer Id from Course to get the result.

$\pi_{lId}(Course)$

Case 2: LId in Course has some null values
We join Course with Lecturer (with LId) to make sure that there is no null value in lecturer Id and project only LId to get the final result.

$\pi_{lId}(Course \bowtie Lecturer)$

2. **(6 marks)** Find the students who have the same name as some of lecturers. Show the IDs and names of the students.

Assumption: no null values in Lecturer.name and Student.name
First, we find the duplicate values of names from both Lecturer and Student.

$\pi_{name}(Lecturer) \cap \pi_{name}(Student)$

Then, we join the result with Student (with name) and project only sId and name to get the sId and name of students who have the same name as some of the lecturers.

$\pi_{sId, name}(Student \bowtie (\pi_{name}(Lecturer) \cap \pi_{name}(Student)))$

3. **(8 marks)** Find the students who do not enrol in any course taught by 'John Smith'. Show the names of those students.

We join Lecturer and Course (with LId) and join them with Enrolment (with cId) and join them with Student (sId).

Note: rename name in Student to StudentName because we want to natural join Student with other relations with only sId. Moreover, with the selection condition 'name ≠John Smith' and projection the table with StudentName, we will get the names of students who do not enroll in any courses taught by John Smith.

$\pi_{StudentName}(\rho s(2 \rightarrow StudentName)(Student) \bowtie Enrolment \bowtie (Course \bowtie \sigma_{name \neq John\ Smith}(Lecturer)))$

4. **(10 marks)** Find the courses for which there are **not both** students with name 'John' or 'Alice' who enrol in. That is, that in any course appeared in the outcome, there may be a student with name 'John' or a student with name 'Alice', but there should not be both of them. Show the cIds of those courses.

Case 1: courses can have other students apart from John and Alice

1. By joining Student with Enrolment (with sId) and defining selection conditions as 'name = John' and 'name = Alice', we will get the table describing all courses learned by John and Alice.

2. Then, we project cId and intersect them together to find the cId that is learned by both John and Alice.

3. Finally, we can get the desired result by subtracting the whole cId in Enrolment table with the previous value described above.

$\pi_{cId}(Enrolment) - (\pi_{cId}(Enrolment \bowtie \sigma_{name = John}(Student)) \cap \pi_{cId}(Enrolment \bowtie \sigma_{name = Alice}(Student)))$

Case 2: course can not have other students apart from John or Alice

Same as the above solution but this time we subtract the union operation with the intersection operation to get the cId of all students whose names are only John or Alice.

$$(\pi_{cId}(Enrolment \bowtie \sigma_{name = John}(Student)) \cup \pi_{cId}(Enrolment \bowtie \sigma_{name = Alice}(Student)))$$
$$-(\pi_{cId}(Enrolment \bowtie \sigma_{name = John}(Student)) \cap \pi_{cId}(Enrolment \bowtie \sigma_{name = Alice}(Student)))$$

5. **(10 marks)** Find the students who have the same name as some other student. Show the names of the students.

We rename one Student table as ‘S’ and cross it with another Student table to match all possible combinations.

Then, we use the selection condition ‘S.name = Student.name’ and project only S.name to find the names of students who have the same name as some other student.

$$\pi_{S.name}(\sigma_{S.name = Student.name}(\rho S(Student) \times Student))$$

6. **(11 marks)** Find the students who take all the courses taught by ‘David Cheung’. Show only the IDs of these students.

First, we find cId which are taught by David Cheung by joining Lecturer with Course (with LId), selecting name of Lecturer table = David Cheung, and projecting only cId.

Then, we divide Enrolment table (cId, sId) with the previous result (cId) to get all the sId who take all courses taught by David Cheung.

$$Enrolment \div \pi_{cId}(Course \bowtie \sigma_{name = David\ Cheung}(Lecturer))$$

2 Question Two (50% = 50 marks)

According to the given information, the action plan to design the ER, EER diagram is shown below.

1. define entity types
organization, virtual machine, user account, developing account, customer account, e-product, music, book, movie, image

2. identify relationship types/ and superclass with subclass
2.1 organization maintain virtual machine
2.2 organization affiliate developing account
2.3 virtual machine run e-product
2.4 customer account has virtual machine
2.5 customer account purchase e-product
2.6 developing account develope e-product
2.7 user account consists of developing account and customer account
2.8 e-product consists of music, book, movie, and image

3. determine multiplicity constrains
3.1.1 each organization can maintain virtual machine from 0 (not having a virtual machine) to many (0..*)
3.1.2 each virtual machine must be maintained by only 1 organization (1..1)

3.2.1 each organization can affiliate developing account from 0 (it has no developing account) to many (0..*)
3.2.2 each developing account must be affiliated by only 1 organization (1..1)

3.3.1 each virtual machine can run e-product from 0 (not run at all) to many (0..*)
3.3.2 each e-product can be run by virtual machine from 0 (not being run at all) to many (0..*)

3.4.1 each customer account can have virtual machine from at least 1 to many (1..*)
3.4.2 each virtual machine can be owned by 0 (no belonging customer at all) or 1 customer account (0..1)

3.5.1 each customer account can purchase e-product from 0 (not buy anything) to many (0..*)
3.5.2 each e-product can be purchased by 0 (no one buys it) to many customer account (0..*)

3.6.1 each developing account can develop 0 (not creating anything) to many e-product (0..*)
3.6.2 each e-product must be developed by only 1 developing account (for copyright reasons) (1..1)
or
each e-product can be developed by at least 1 to many (more than 1 developers agree to build the same e-product) developing account. However, due to copyright reasons, I think it would be more appropriate to choose the above case.

3.7 user account must belong to either developing account or customer account and it can not be both account types at the same time (if developing accounts want to buy some e-products, they need to utilize customer accounts. In the same way, if customer accounts want to create e-products, they need to create another developing account to do so.) {mandatory, or}

3.8 e-product must belong to the set of {music, book, movie, image} and it can only be one type at the same time (for example, it can not be both music and book at the same time) {mandatory, or}

4. check for fan and chasm traps
4.1 virtual machine/ organization/ developing account

A developing account can only be affiliated with 1 organization. However, some organization may not have their own virtual machines. So, this is a chasm trap. We need to create another relationship type that links between developing account and virtual machine (rent relationship from developing account to virtual machine) so that the developing account can create their e-products via virtual machines.

4.2 virtual machine/ e-product/ developing account

A virtual machine can run 0 to many e-products and each e-product can only be developed by 1 developing accounting. For example, 1 virtual machine develop 2 products (a and b). a may be run by 5 virtual machines, b may be run by 0 virtual machine. So, this will work fine.

4.3 virtual machine/ e-product/ customer account

Each e-product can be run by 0 to many virtual machines and customer account can purchase 0 to many e-products. For example, 1 customer account purchases 2 e-products (a and b). a can be run by 1 to many virtual machines, while b is run by 0 virtual machine (the customer just purchased it and has not run it at all). So this will work fine.

4.4 organization/ developing account/ e-product

Each developing account must be affiliated by 1 organization and it can develop 0 to many e-products. So, we can easily link between e-products and their organization. For example, each e-product can be created only by 1 developing account and that developing account must be affiliated by only 1 organization. So, this will work fine.

4.5 customer account/ e-product/ developing account

Each e-product can only be developed by 1 developing account and it can be purchased by 0 to many customer accounts. So, we can easily link between customer account and developing account which create e-product the customer account purchased. For example, one e-product created by 1 developing account can be purchased by no one or many customers. So, this will work fine.

4.6 organization/ virtual machine/ customer account

Each virtual machine can be maintained by only 1 organization and it can be owned by 0 to 1 customer account. So, we can easily link between customer account and organization which provide virtual machines for customer account. For example, 1 virtual machine is maintained by 1 organization, and that virtual machine can be owned by 1 customer account, 1 developing account, or no one. So, this will work fine.

5. Identify attributes for each entity type, primary keys, and foreign keys.

The pictures below show the design of the tables which states all attributes and their specification (null, not null).

Note that primary key = PK and foreign key = FK

virtual machine			
virtual machine ID	specification	organization ID	account ID
1234	4	444	123
1235	3	555	234
2136	2	666	345
not null	not null	not null	null

PK = virtual machine ID
FK = organization ID, account ID

e-product								
product ID	price	discount	deal detail	total price	release date	developer ID	customer ID	running virtual machine ID
123	2	5	str	4	1/2/2023	345	34, 45, 66	234, 456, 778
124	3	5	str	3	2/3/2023	234	45	234
324	4	5	str	2	3/4/2023	546	34,22	234, 567
not null	not null	null	null	not null	not null	not null	null	null

PK = product ID
FK = developer ID, customer ID (can be more than 1), running virtual machine ID (can be more than 1)

organization		
organization ID	name	location
123	str	str
234	str	str
345	str	str
not null	not null	not null

PK = organization ID
FK = no

user account				
account ID	personal info	user name	password	account type
123				
234				
435				
not null	not null	not null	not null	not null

PK = account ID
FK = no

developing account
organization ID
546
234
567
not null

PK = no
FK = organization ID

customer account	
credit card no	address
1234	str
2345	str
3456	str
not null	not null

PK = no
FK = no

music		
singer name	genres	duration
a,b,c	rock	3.45
h	pop	4.23
not null	not null	not null

PK = no
FK = no

book		
author name	title	page
a,b,c		45
g		23
not null	not null	not null

PK = no
FK = no

movie		
movie name	director name	type
		drama
		horror
not null	not null	not null

PK = no
FK = no

image	
image name	dimension
	128x128
	256x256
not null	not null

PK = no
FK = no

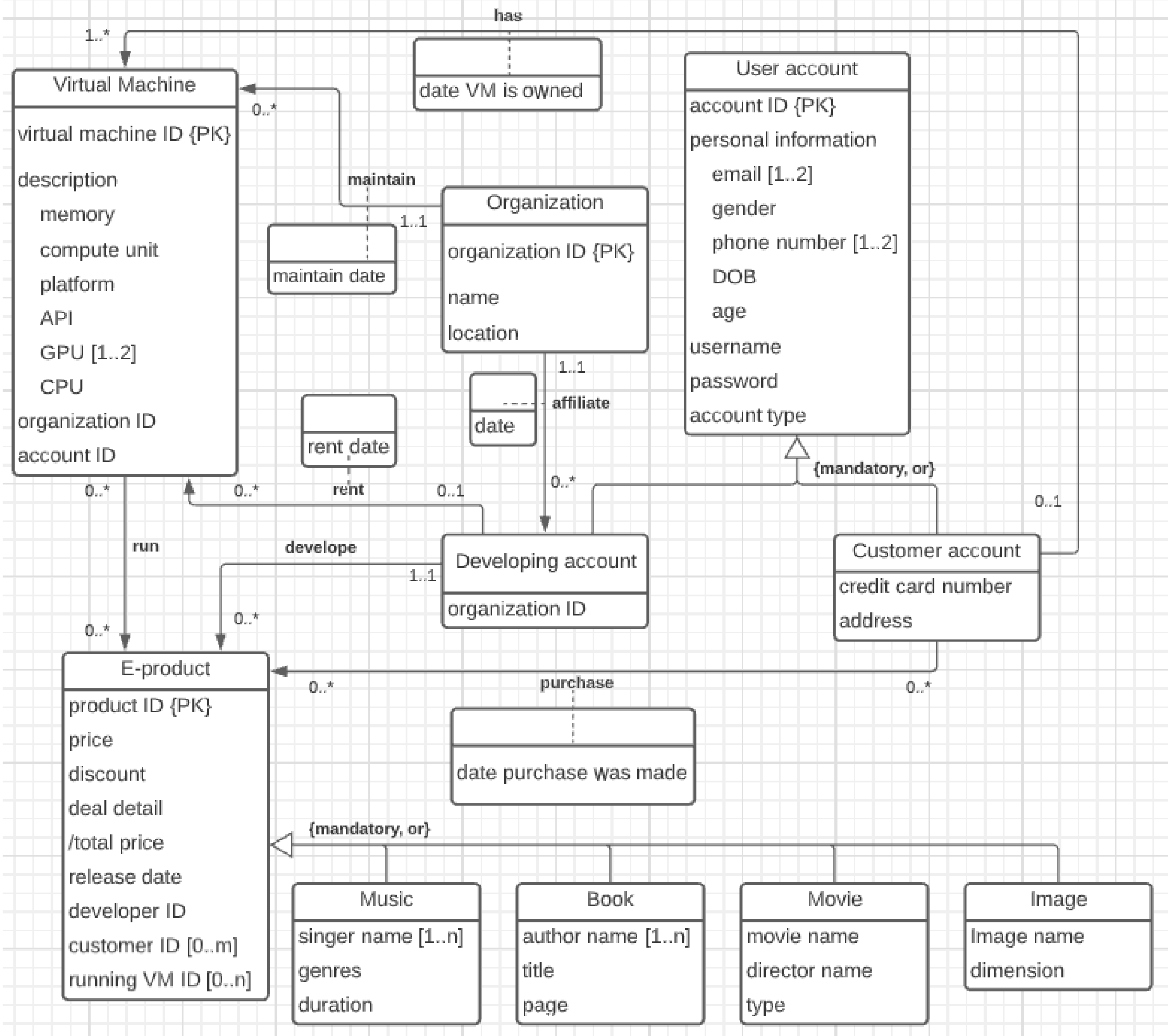
Attribute of relations

1. maintain relation has maintain date
2. has relation has date that a virtual machine is owned by a customer account
3. affiliate has date when a developing account is affiliated to an organization
4. rent has date when a developing account rent a virtual machine
5. purchase has date when a customer purchases an e-product

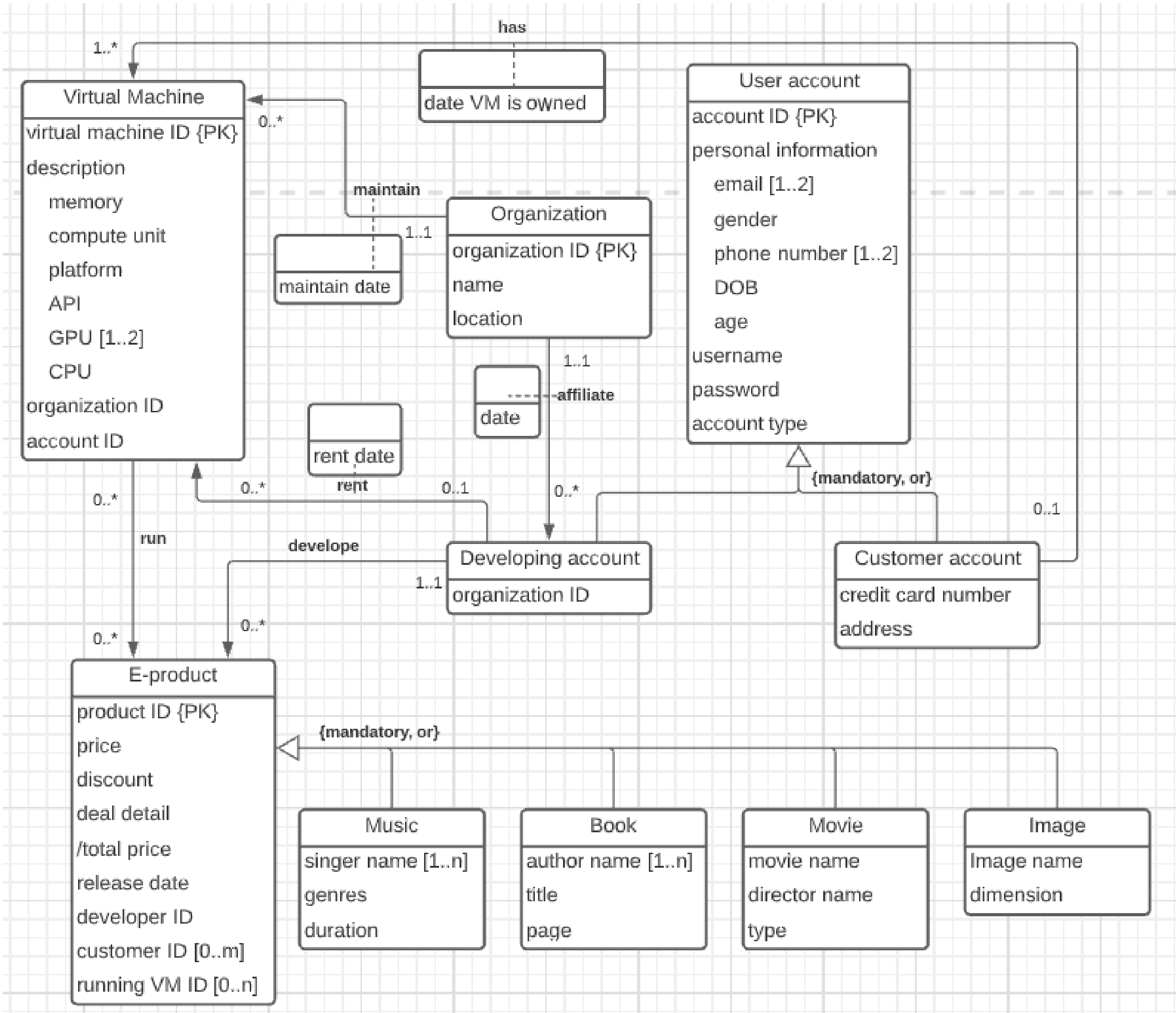
6. possible ER and EER designs.

Note that
[1..2] means that value can be either 1 data or 2 data.
[1..n] means that value can be from 1 data to n data.
[0..m] means that value can be from 0 data to m data.
/ means the derived attribute.

Design 1: based on the specifications described before.



Design 2: like design 1 but the purchase relation is eliminated because we might assume that when customer accounts use their virtual machine to run e-products, they must purchase those e-products. The ER, EER diagram of design 2 is shown below.



7. Choosing the best ER, EER design

Design 2 may have an error. This is because some customer accounts may purchase some e-products but they just leave it there without running it via their virtual machine. So, we need the purchase relation and the best design is design 1.