Group 6

Nguyễn Huy Hoàng – 20194433

Phùng Hà Nguyên – 20190087

Hoàng Nguyễn Minh Nhật – 20194445

Nguyễn Quang Minh – 20194444

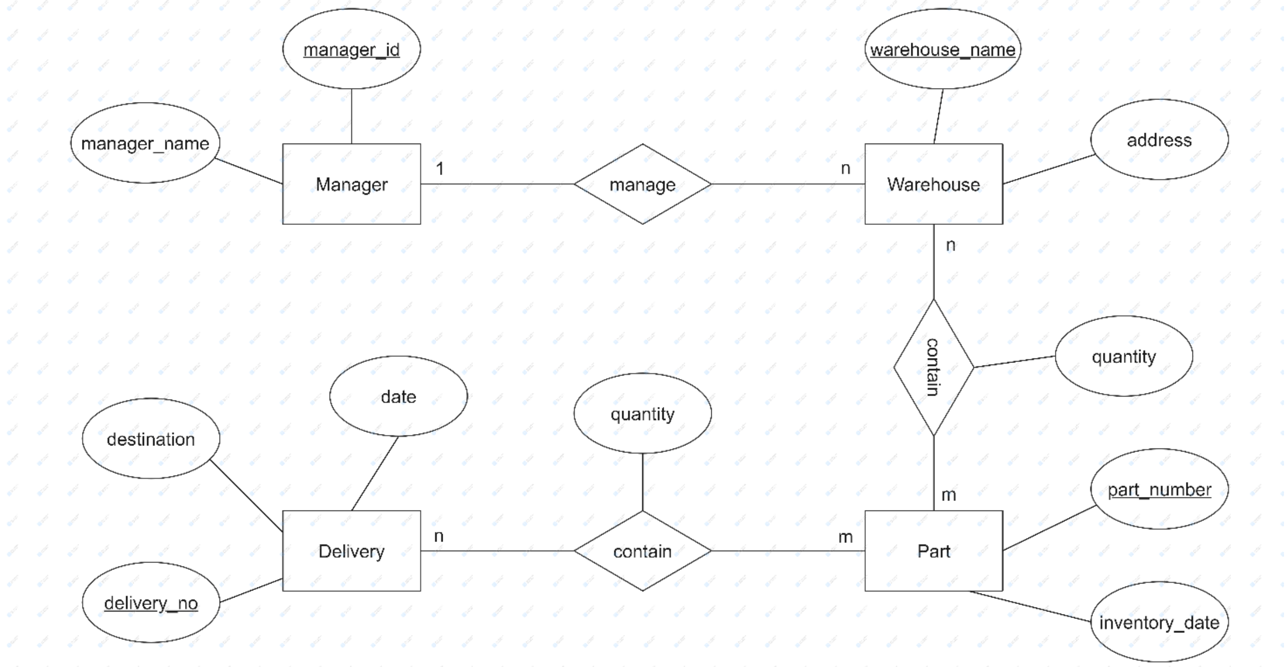
EXERCISE LIST NUMBER 6

Exercise 1:

1. Schema
   MANAGER (**manager_id**, manager_name)
   WAREHOUSE (**warehouse_name**, warehouse_address, **manager_id** #)
   PART (**part_no**, supplier_name, inventory_date)
   PART_IN_WAREHOUSE (**part_no** #, **warehouse_name** #, qty_on_hand)
   DELIVERY (**delivery_no**, delivery_date, delivery_destination)
   DELIVERY_DETAILS (**delivery_no** #, **part_no** #, delivery_qty)

2. ERD

Why do you need manager_id as part of the PK of the table WAREHOUSE?
Also, inventory_date should be in the PART_IN_WAREHOUSE table (the inventory date als
depends on the warehouse)



Good, except that inventory_date should be an attribute of the CONTAIN relationship between PART and warehouse (the inventory date also depends on the warehouse, not only on the part)

Exercise 2:

1. Move the attribute *title_id* in the DOMAIN table and add the attribute domain_id in the TITLES table.

   AUTHOR (**author_id**, first_name, last_name)

   TITLES (**title_id**, name, author_id #, domain_id #)

   DOMAIN (**domain_id**, name)

   READERS (**reader_id**, first_name, last_name, address, city_id, city_name, phone)

   BORROWING (**borrowing_id**, reader_id #, title_id #, date)

   HISTORY (**reader_id** #, **title_id** #, date_of_borrowing, date_of_returning)

   OK, but the HISTORY PK is not unique…

2. They are not because there is Transitive Dependency between city_id → city_name.

   Very good

3. Schema in 3NF:

   AUTHOR (**author_id**, first_name, last_name)

   TITLES (**title_id**, name, author_id #, domain_id #)

   DOMAIN (**domain_id**, name)

   READERS (**reader_id**, first_name, last_name, address, city_id #, phone)

   CITY (**city_id**, city_name)

   BORROWING (**borrowing_id**, reader_id #, title_id #, date)

HISTORY (**reader_id** #, **title_id** #, date_of_borrowing, date_of_returning)

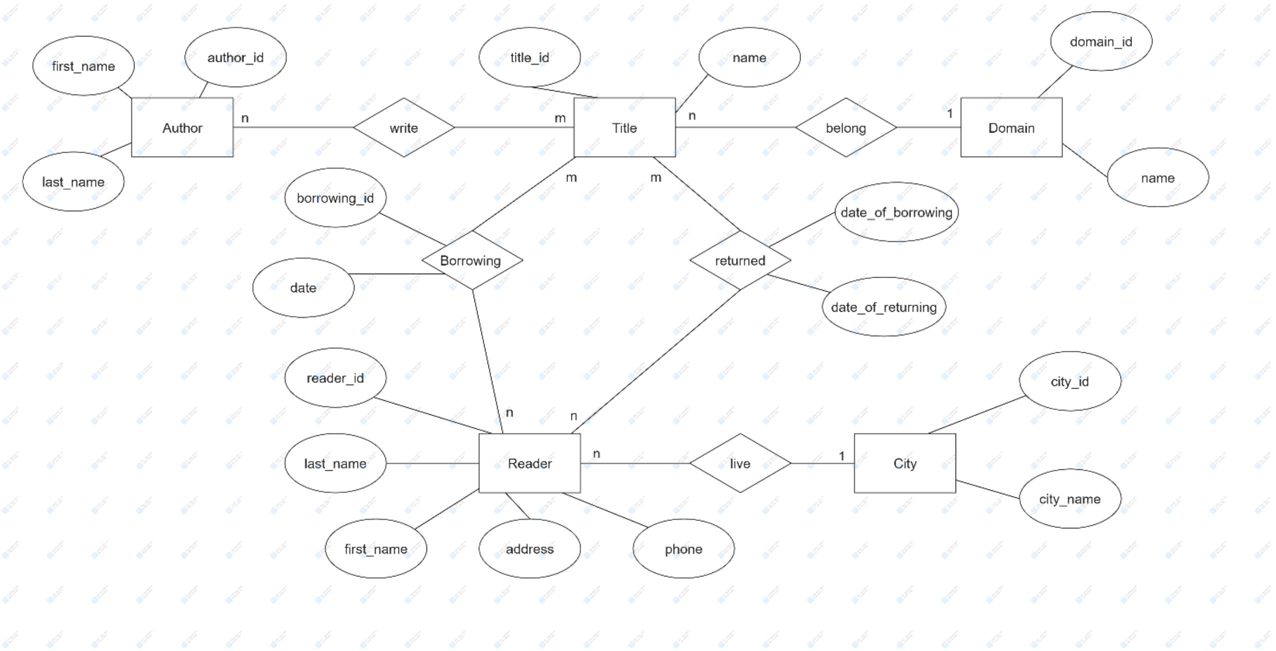
4. Add a relationship table WRITING with 2 attributes *title_id (PK)* and *author_id*.
   AUTHOR (**author_id**, first_name, last_name)
   WRITING (**title_id** #, author_id #)
   TITLES (**title_id**, name, domain_id #)
   DOMAIN (**domain_id**, name)
   READERS (**reader_id**, first_name, last_name, address, city_id #, phone)
   CITY (**city_id**, city_name)
   BORROWING (**borrowing_id**, reader_id #, title_id #, date)
   HISTORY (**reader_id** #, **title_id** #, date_of_borrowing, date_of_returning)

OK for the idea of adding writing, but in your implementation the WRITING and HISTORY PKs are not unique..

5. Yes, It is a good idea. Because when using one single BORROWING table with a *date_of_returning* column, you will have to update the table manual whenever a reader returns a book. Meanwhile, with the trigger will automatically updates the information.

It is indeed a good idea, but your argument is not necessarily true.

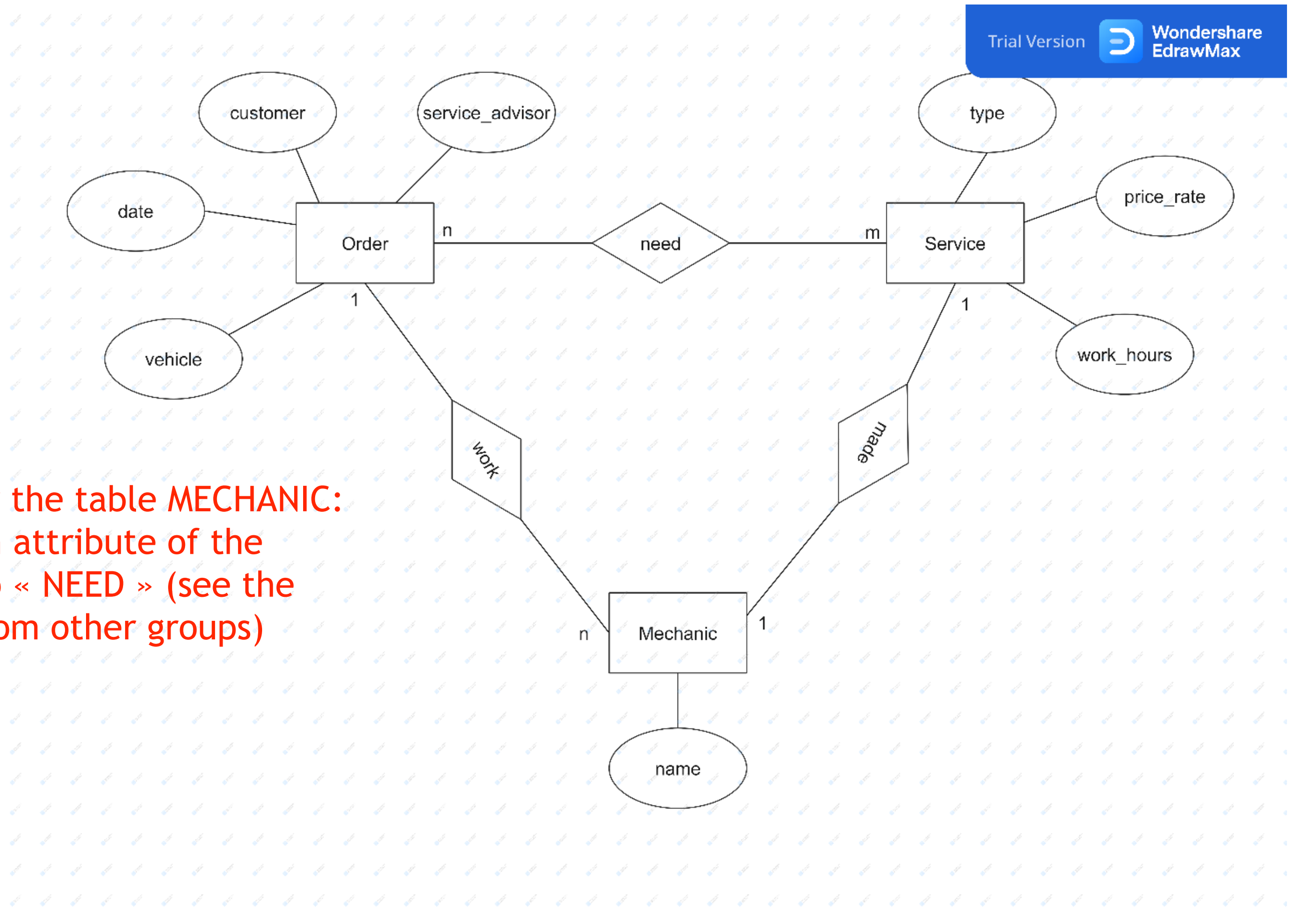
6. ERD:



Very good

Exercise 3:

1. ERD:



No need for the table MECHANIC: it can be an attribute of the relationship « NEED » (see the solutions from other groups)

2. Relational schema:

   ORDER (**order_id**, customer, vehicle, date, advisor)
   SERVICES (**service_id**, type, price_rate, work_hours)
   ORDER_NEED (**order_id** #, **service_id** #)
   MECHANIC (**mechanic_id**, mechanic_name, **order_id** #)
   SERVICE_MADE (**mechanic_id** #, **service_id** #)

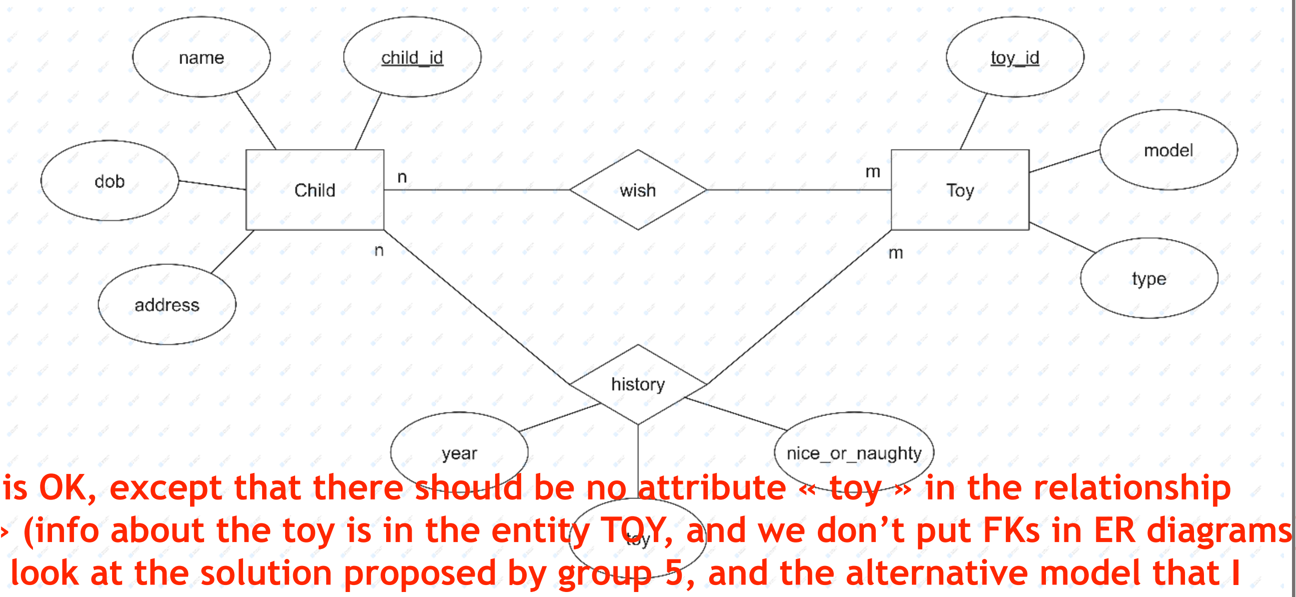
This is not a good schema: each mechanic can work on one order maximum!

3. The schema in 3NF:

   CUSTOMER (**cust_id**, cust_name, vehicle)
   ORDER (**order_id**, **cust_id** #, date, advisor)
   SERVICES (**service_id**, type, price_rate, work_hours)
   ORDER_NEED (**order_id** #, **service_id** #)
   MECHANIC (**mechanic_id**, mechanic_name, **order_id** #)
   SERVICE_MADE (**mechanic_id** #, **service_id** #)

<u>Exercise 4:</u>

1. ERD:



This model is OK, except that there should be no attribute « toy » in the relationship « HISTORY » (info about the toy is in the entity TOY, and we don't put FKs in ER diagrams). Just have a look at the solution proposed by group 5, and the alternative model that I propose.

2. Relational schema:
   CHILDREN (**child_id**, name, address, DOB)
   TOY (**toy_id**, model, type)
   HISTORY (**year**, **child_id** #, nice_or_naughty, **toy_id** #)
   WISH_LIST (**year**, **child_id** #, **toy _id** #)

OK.

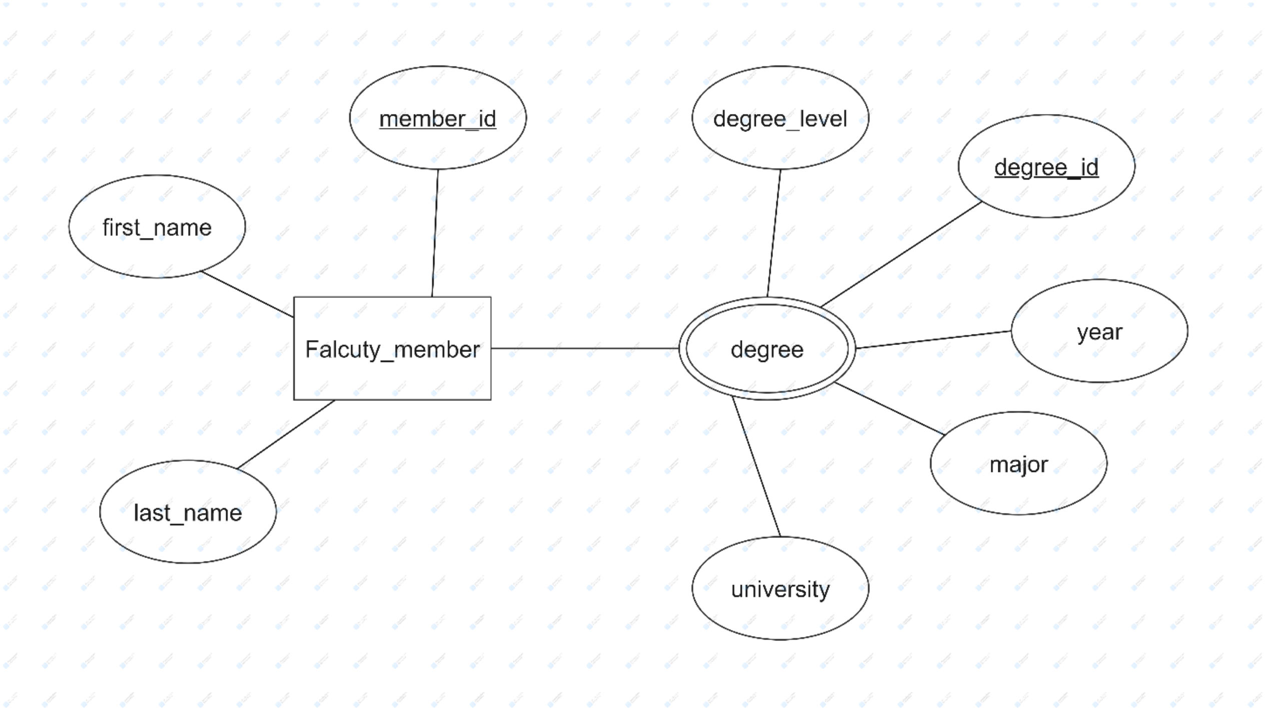
3. The schema is in 3NF.

Exercise 5:

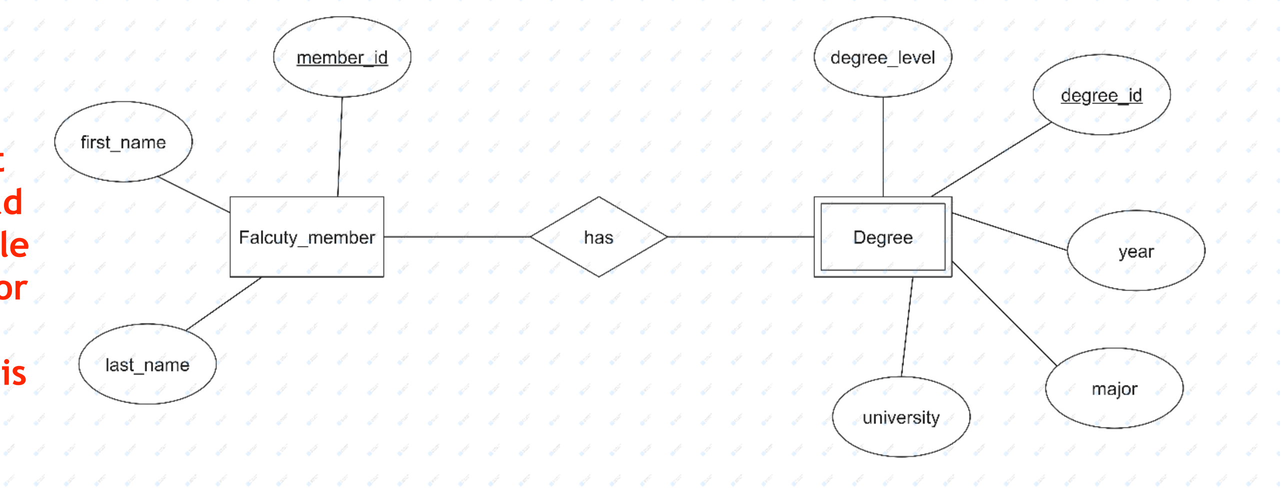
1. Main drawback:
   + The way this design combines all information into one table makes it bulky and when there is a need of modify or query, it would be inconvenient.
   + This design brings a large amount of wasted memory because there would be a lot of NULL attributes.

True. And also,
- the attribute 'degree' is not simple: it is composite -> not in 1NF
- What if an employee has 4 degrees (for instance 1 Bachelor, 1 Master and 2 PhDs??)

2. ERDs:



Very good



Very good, except that there should be no double rectangle for the entity DEGREE (it is not a weak entity)

3. In the two diagrams, the second one might be more memory-saving in order to store GPA for each degree.

Very good

4. Relational schema:
MEMBER (**member_id**, first_name, last_name)
DEGREE (**degree_id**, year, major, university, degree_level)
DEGREE_DETAIL (**member_id** #, **degree_id** #, GPA)

Very good

5. Yes, it is in 3NF because its attributes are atomic and cannot be broken into smaller tables (No Transitive).

OK