**BASIC TASK**

1.

Candidate Key：

Definition：In a relational database, a candidate key is one or a set of attributes that can uniquely identify each tuple (row) in a relation (table). A relation may have multiple candidate keys.

Composite Key：

Definition：A composite key is a key formed by the combination of two or more attributes. The combination of these attributes can uniquely identify a tuple in a relation.

Foreign Key：

Definition：In a relational database, a foreign key is one or a set of attributes in a relation (table) that references a candidate key (usually the primary key) in another relation (table).

Functional Dependency：

Definition：In a relational database, functional dependency is a concept that indicates that in a relation (table), the value of one attribute or a set of attributes can determine the value of another attribute.

2.

Entity Integrity

Definition: The entity integrity rule states that in a relation (table), the primary key of a relation cannot have null values. A primary key is used to uniquely identify each tuple (row) in a table.

Referential Integrity

Definition: Referential integrity concerns the relationships between different relations (tables). It requires that the value of a foreign key in a referencing relation (a table that refers to another table) must either match the value of the primary key in the referenced relation (the table being referred to) or be null.

Domain Integrity

Definition: Domain integrity refers to the validity of the data values entered into a particular attribute (column) of a relation. Each attribute in a table has a defined domain, which is a set of all possible values that the attribute can take.

3.

Table 1 :

Violates the principle of primary key uniqueness. "filmNo" is set as the primary key, but there are duplicate values in the table. The primary key should uniquely identify each row in the table, so there is a violation of relational integrity rules here.

Table 2:

Similarly violates the principle of primary key uniqueness. "Supplier" is set as the primary key, but there are duplicate values in the table.

Table 3 :

Also violates the principle of primary key uniqueness. "Song" is set as the primary key, but there are duplicate values in the table. Therefore, there is also a violation of relational integrity rules here.

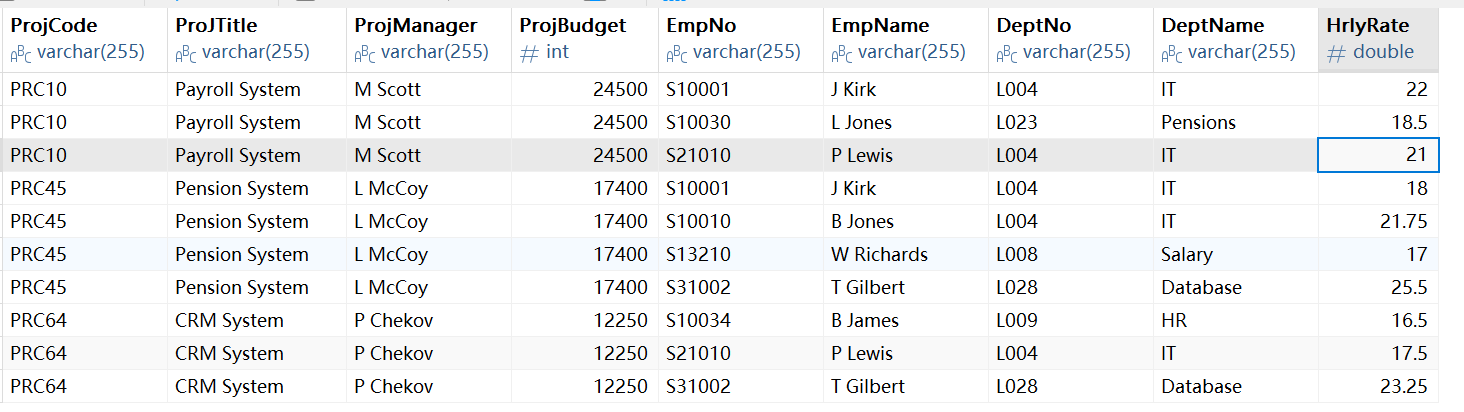
**MEDIUM TASKS**

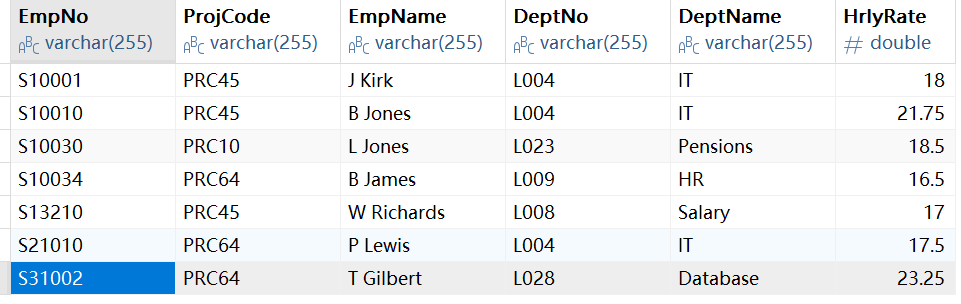
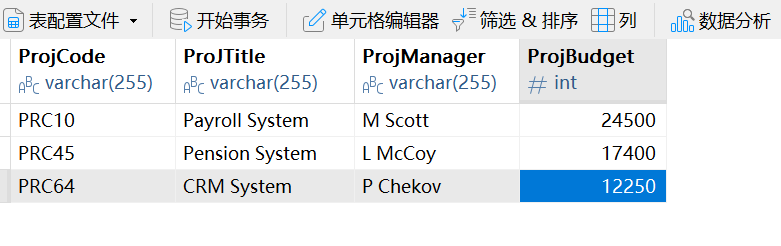
4.

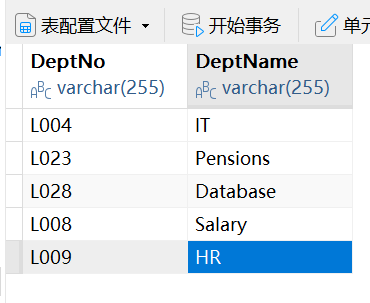
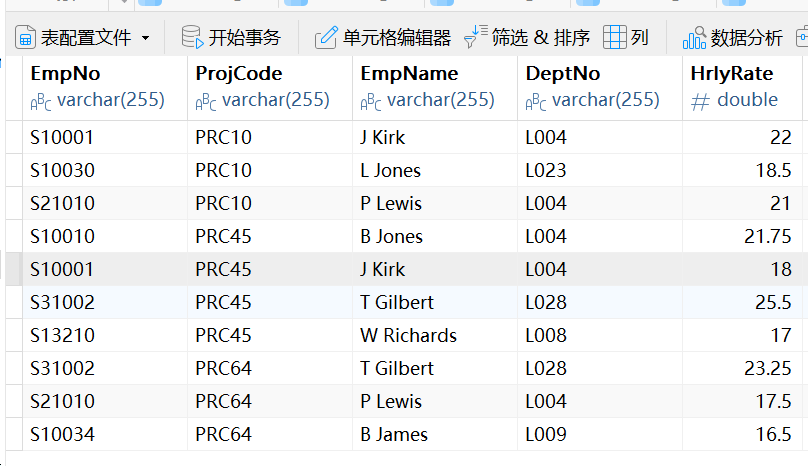
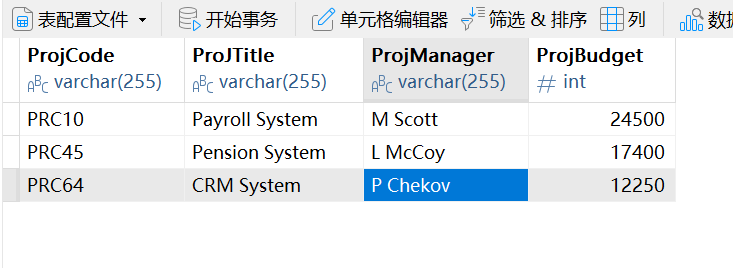
a. Insertion of new project exception:  
If an attempt is made to insert a new project with the project code (ProjectCode) of PrC30, and this code already exists in the table, the insertion operation will fail because ProjectCode is the primary key and must be unique. This violates the uniqueness constraint of the database.

b. Deletion of project exception:  
If the project with the project code (ProjectCode) of PrC10 is deleted, then all the employee records related to this project will also be deleted. This may cause confusion in project management because all the historical data and related information of this project will be permanently lost. Moreover, if other systems or modules rely on these project data, the deletion operation may also lead to system errors or data inconsistency.

c. Modification of employee department exception:  
If the employee is moved from department L004 to department L009, but the department number (DeptNo) and department name (DeptName) of this employee in the Project-Employee table are not updated, the data in the table will be inconsistent. This will lead to errors or misleading results when querying the information of this employee. To avoid this situation, the department information of the employee must be updated simultaneously.

d. Derive 1NF:  


e. Derive 2NF:  


f. Derive 3NF:  


5.

a.

Insertion anomaly

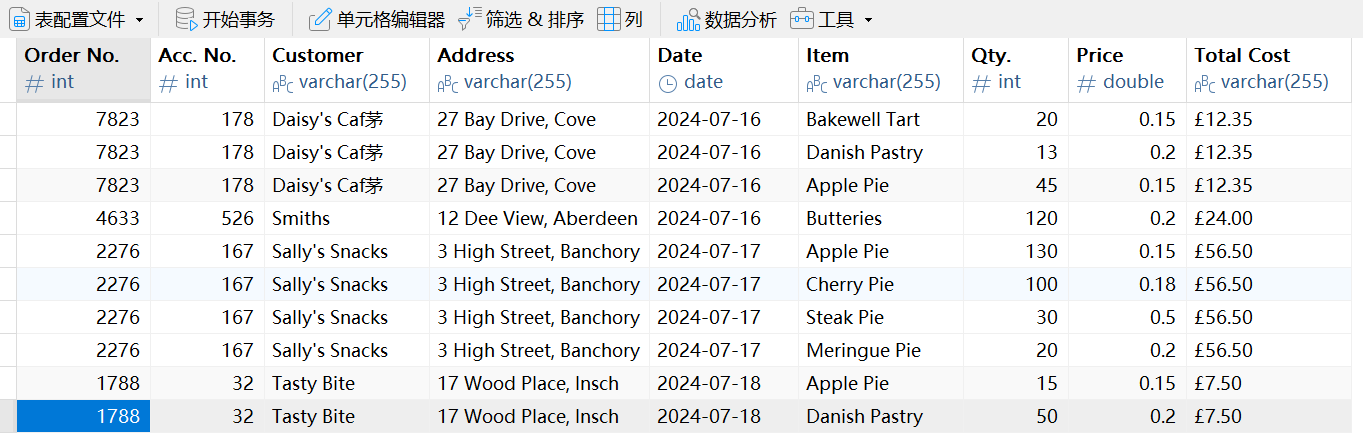
If a new customer is to be added but the customer has not placed an order yet, since the order number (Order No.) is part of the primary key, customer information cannot be inserted without an order number because the primary key cannot be null.

Update anomaly

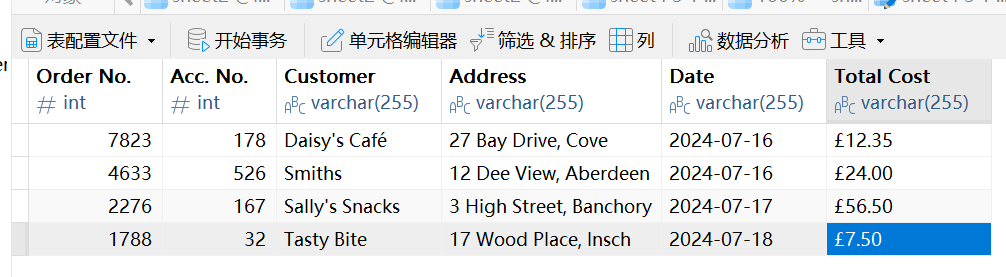
If a customer's address changes, since each order records the customer's address, multiple records (each order record of this customer) need to be updated, which can easily lead to data inconsistency.

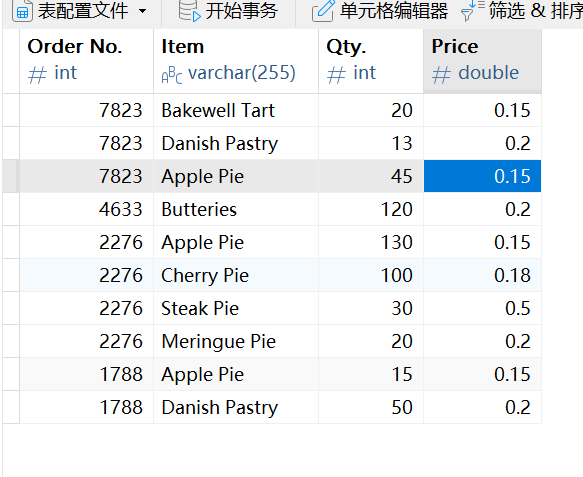
Deletion anomaly

If an order is deleted, related customer information (such as customer address) may be accidentally deleted because customer information and order information are mixed together.

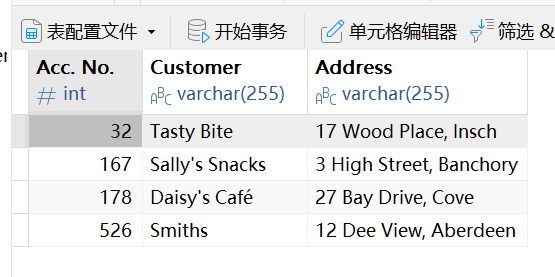
b.

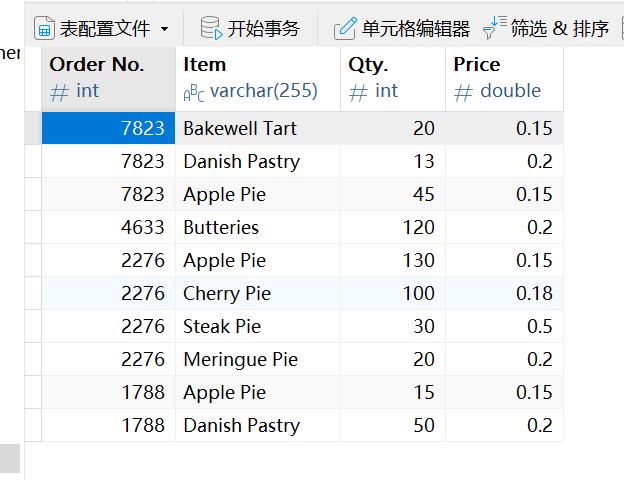
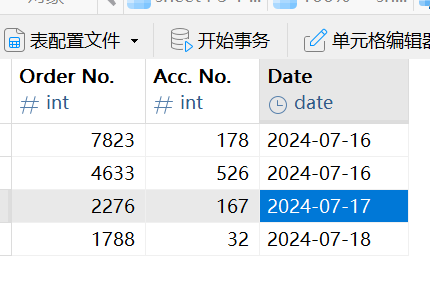
c.





d.





6.

a.

Table name: Patient

Column names and possible data types:

PatientNo (possibly a string type, used to uniquely identify a patient)

Surname (string type, patient's last name)

FirstNname (string type, patient's first name)

Table name: Admission

Column names and possible data types:

PatientNo (corresponds to PatientNo in the Patient table, possibly a string type)

Admitted (date type, indicating the admission date)

Discharged (date type, indicating the discharge date)

Ward (string type, indicating the ward)

Table name: Doctor

Column names and possible data types:

DoctorNo (possibly a numeric type, used to uniquely identify a doctor)

Surname (string type, doctor's last name)

FirstName (string type, doctor's first name)

Ward (string type, indicating the ward where the doctor is located)

b.

Identify the relationships between the tables

Patient table and Admission table:

The relationship is a one-to-many relationship. One patient (Patient) can have multiple admission (Admission) records.

The link is through PatientNo (primary key) in the Patient table and PatientNo (foreign key) in the Admission table.

Doctor table and Ward table:

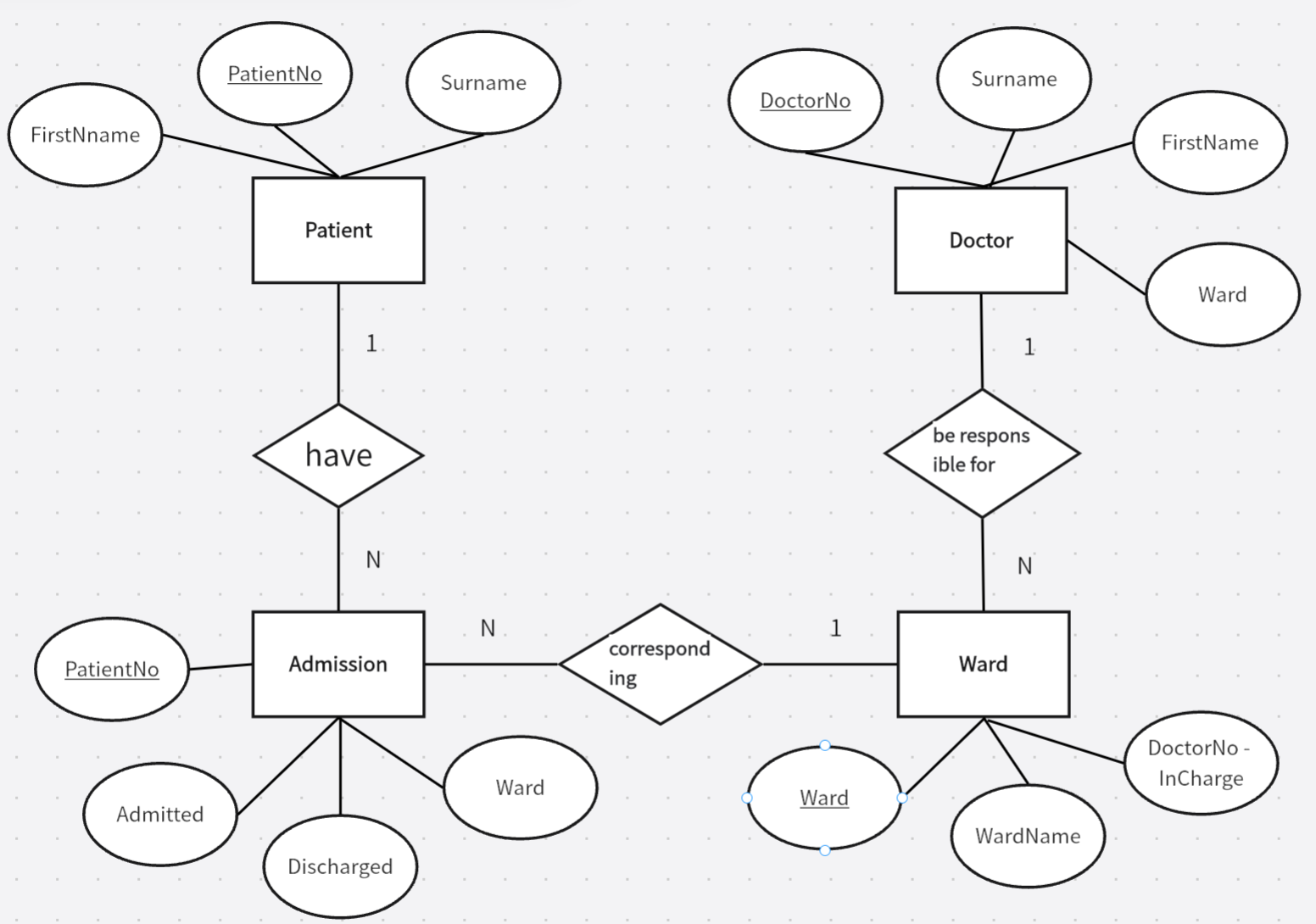
The relationship is a many-to-one relationship. Multiple doctors (Doctor) can belong to one ward (Ward).

The link is through Ward (ordinary field) in the Doctor table and Ward (primary key) in the Ward table, and through DoctorNo - InCharge (foreign key) in the Ward table and DoctorNo (primary key) in the Doctor table.

Admission table and Ward table:

The relationship is a many-to-one relationship. Multiple admissions (Admission) can correspond to one ward (Ward), through Ward (ordinary field) in the Admission table and Ward (primary key) in the Ward table.

c.



**ADVANCED TASKS**

7.

a.

For Table 1 (Orders):

Order No. -> Account No., Customer, Address, Date (An order number determines the account number, customer details, address and date.)

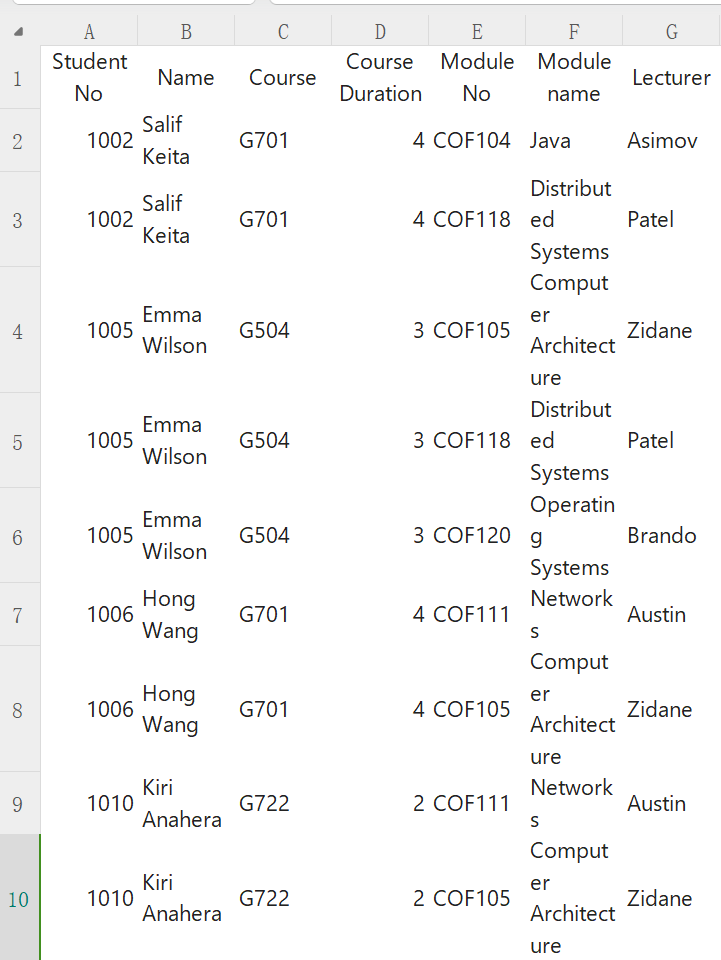
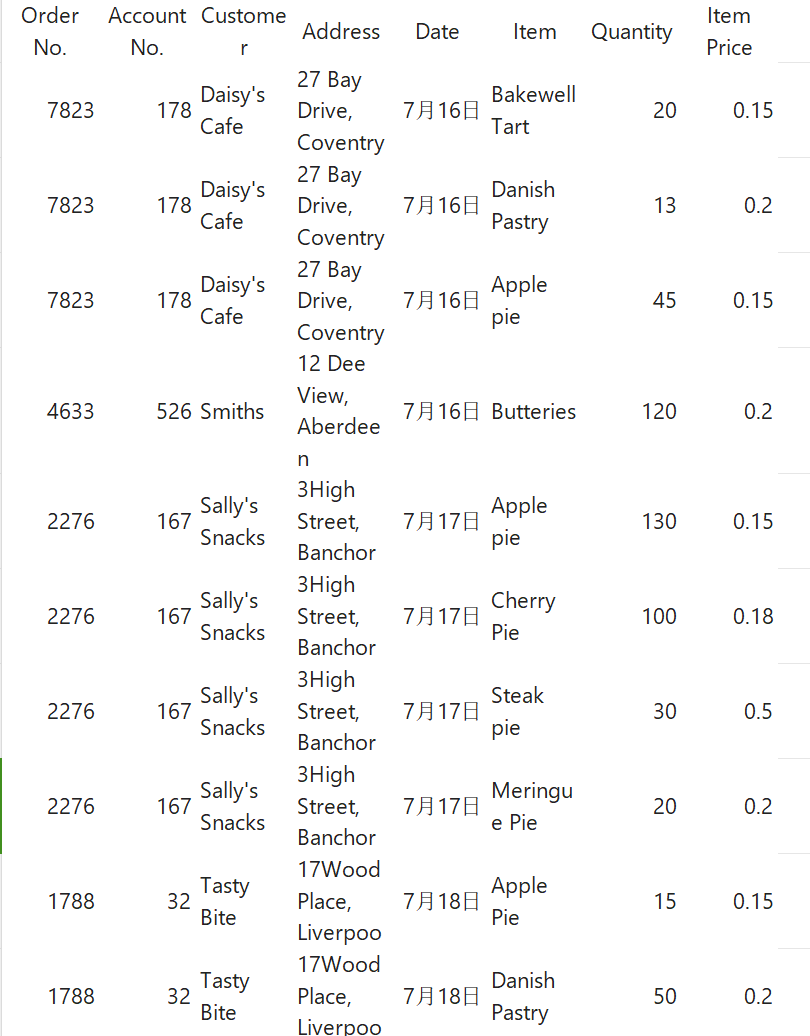
Order No. + Item -> Quantity, Item Price (The combination of order number and item determines the quantity and item price.)

For Table 2 (Student Information):

Student No. -> Name, Course, Course Duration (A student number determines the student's name, course and course duration.)

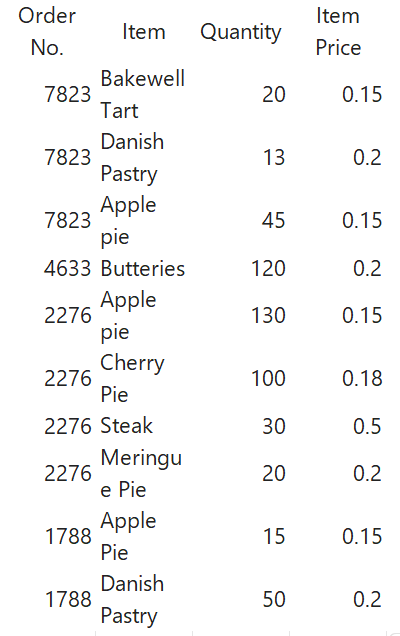
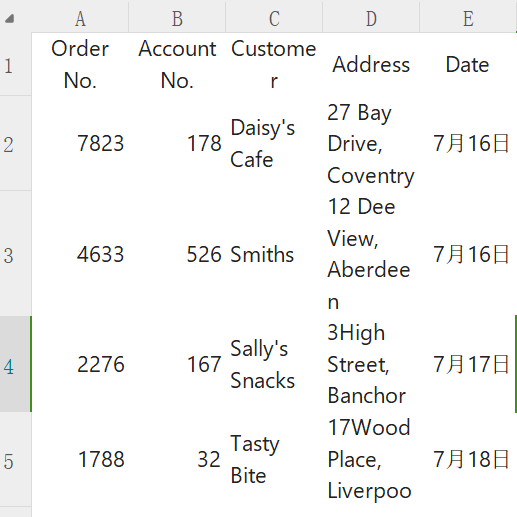
Student No. + Module No. -> Module name, Lecturer (The combination of student number and module number determines the module name and lecturer.)

b.

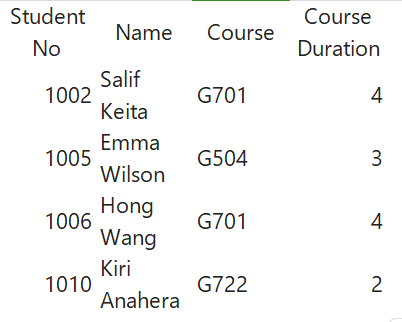
1NF

2NF

①



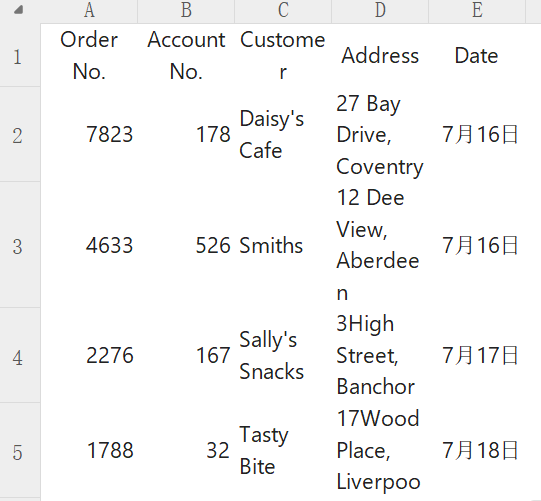
②

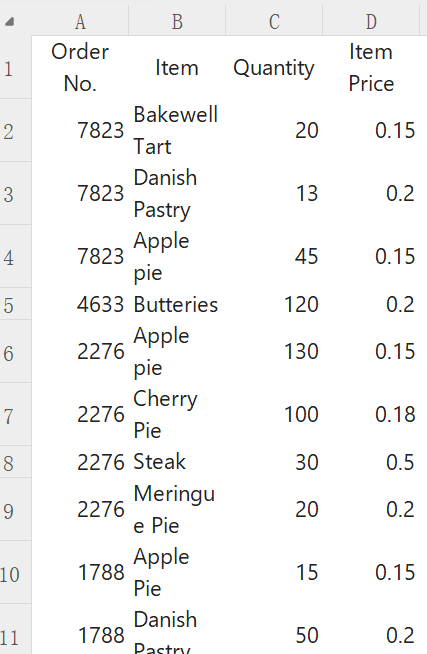




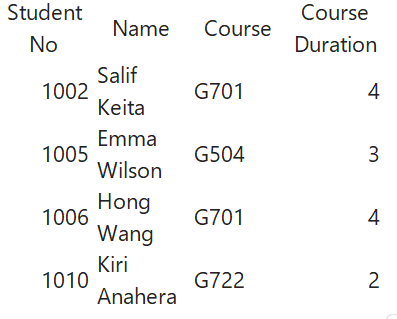
3NF

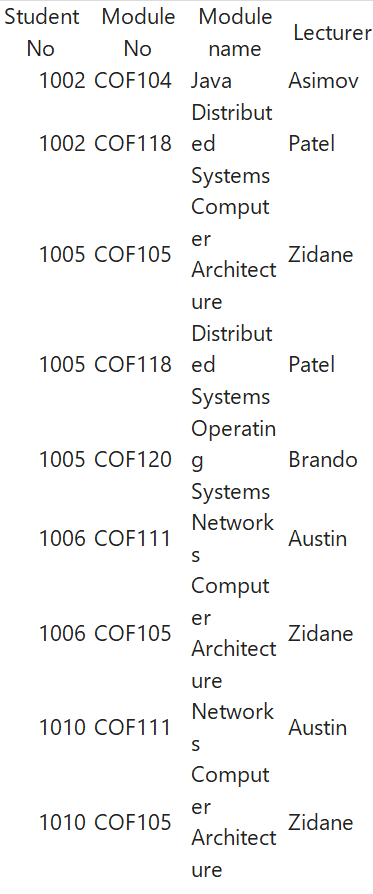
①





②





8.

a.

Report ID: Uniquely identifies each report entry. So, all other attributes are functionally dependent on Report ID. For example, Reporting Period, Branch Code, etc. depend on the specific Report ID.

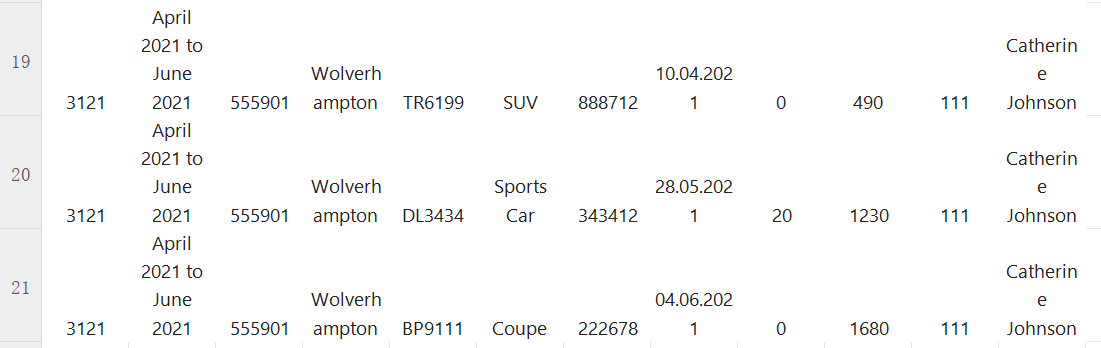
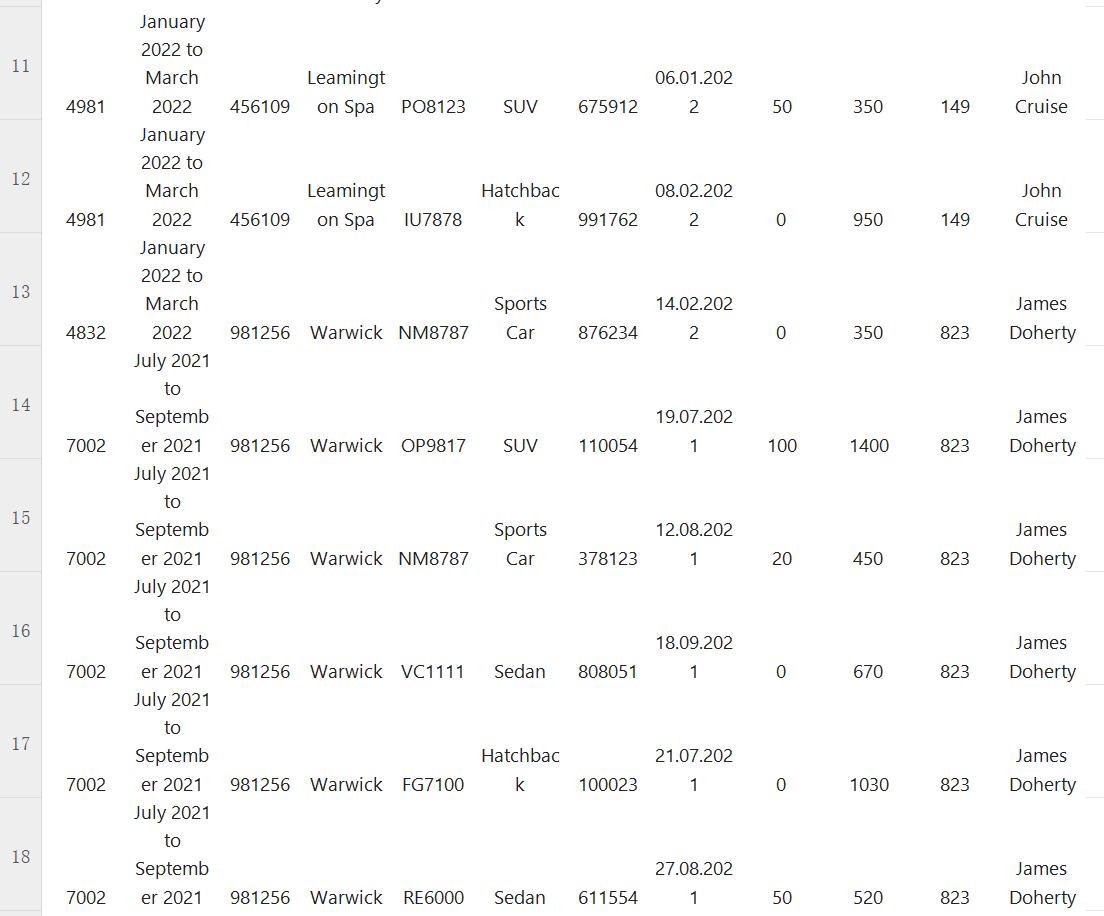
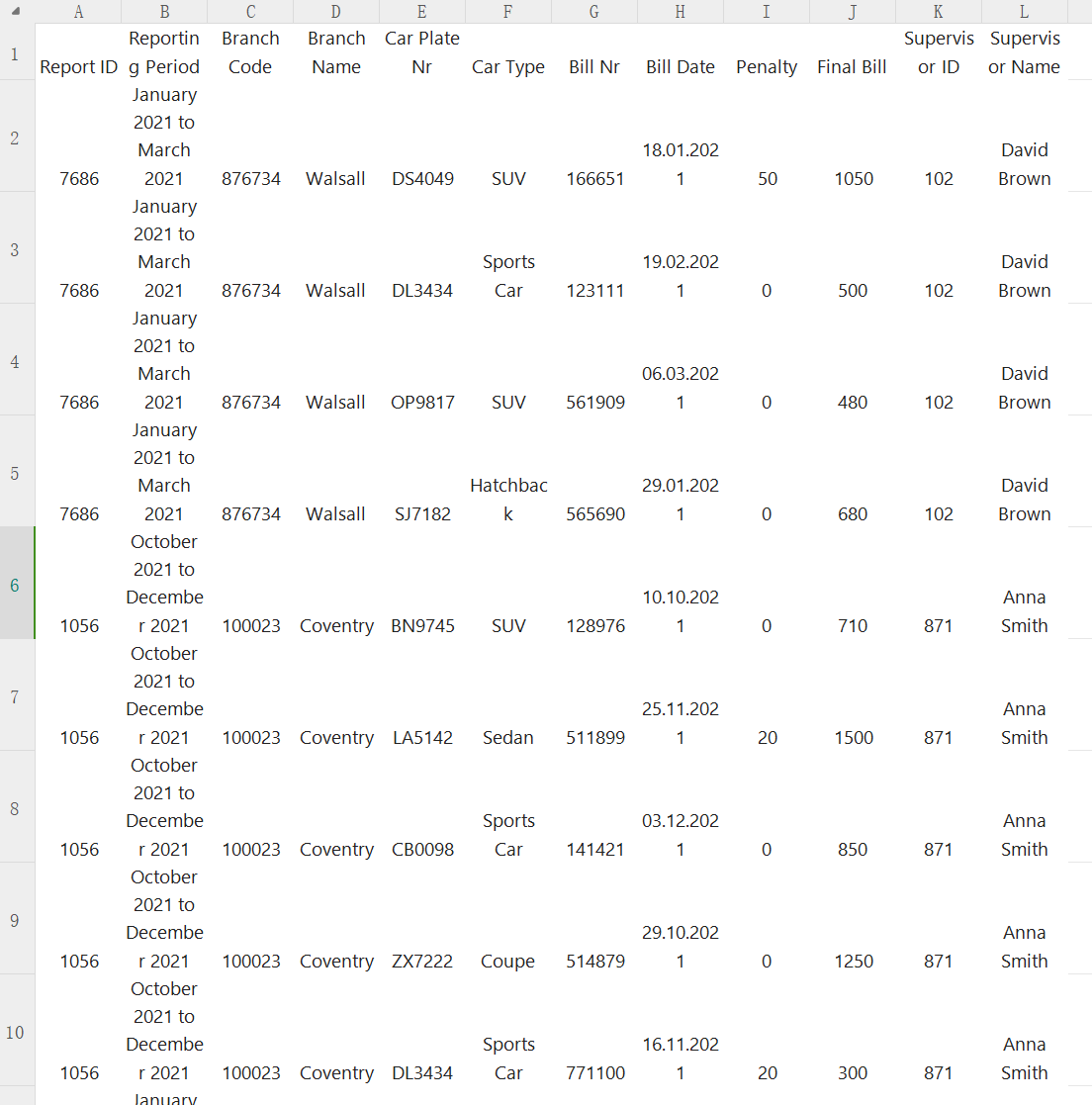
Branch Code: Determines Branch Name, Branch Supervisor ID, and Supervisor Name. That is, for a given Branch Code, there is only one corresponding Branch Name, one Branch Supervisor ID, and one Supervisor Name.

Car Plate Nr: Determines Car Type. Each car with a specific plate number has a unique type.

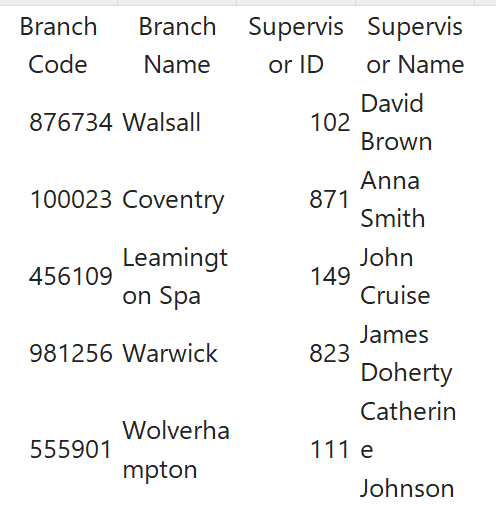
Bill Nr: Determines Bill Date, Penalty, and Final Bill Amount. A specific bill number is associated with a particular bill date, penalty amount, and final bill amount.

b.

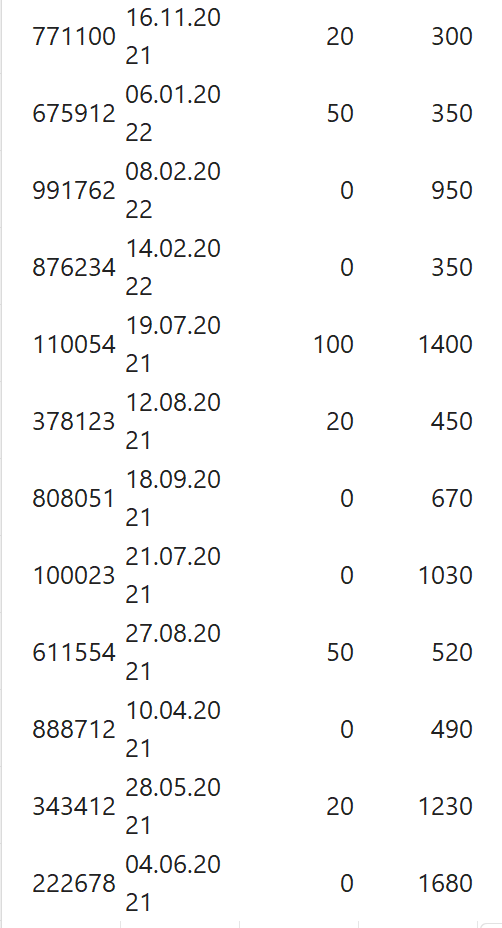
1NF

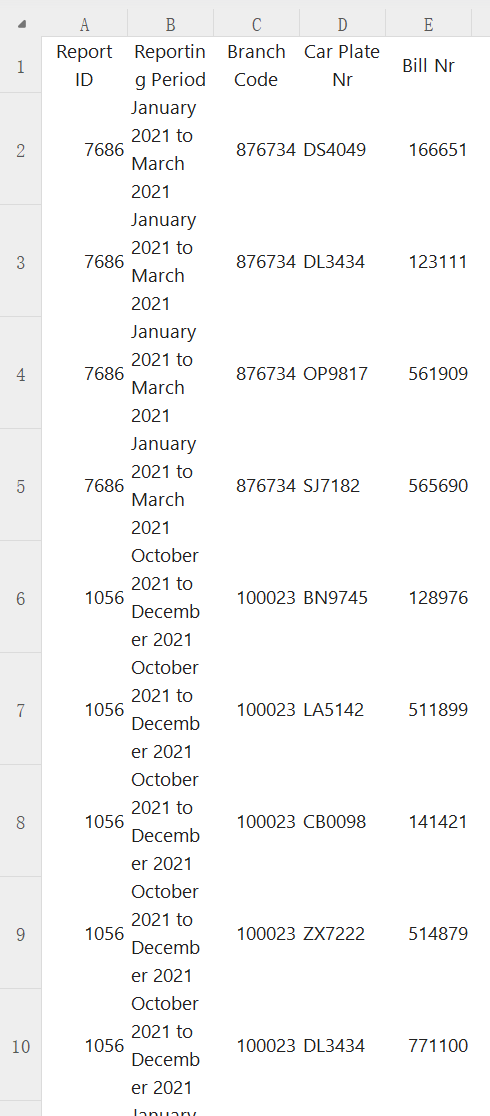


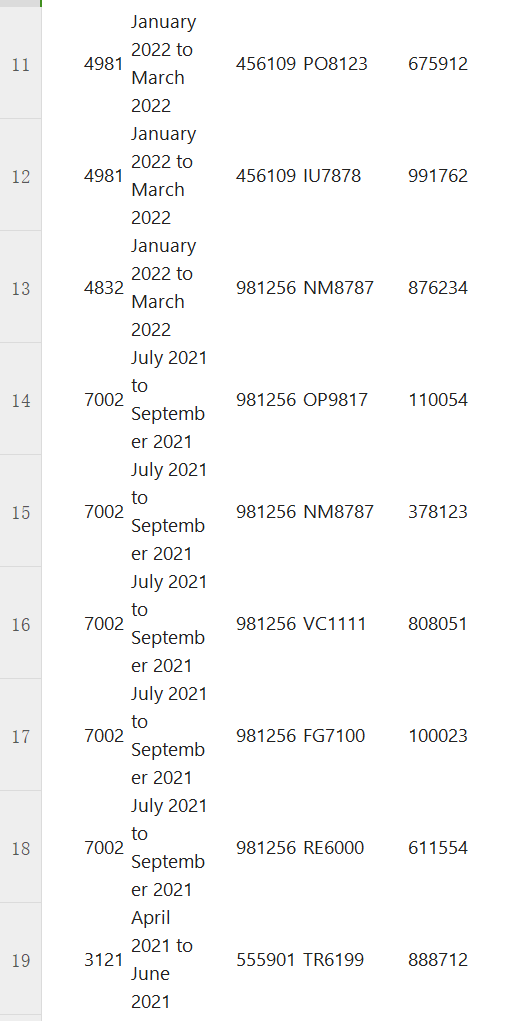
2NF

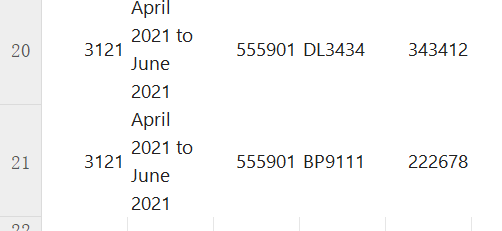




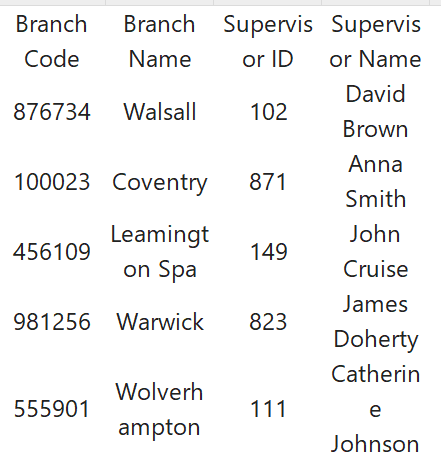




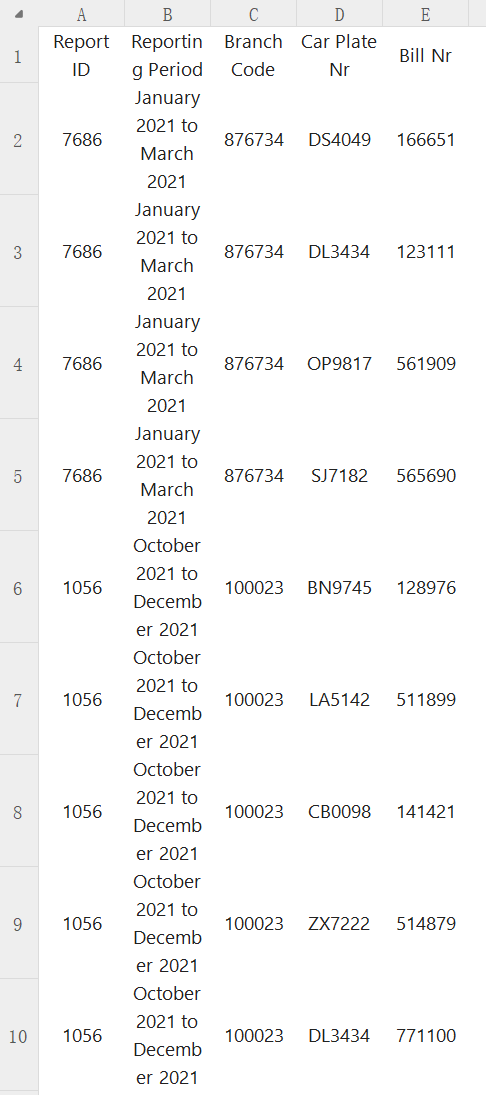


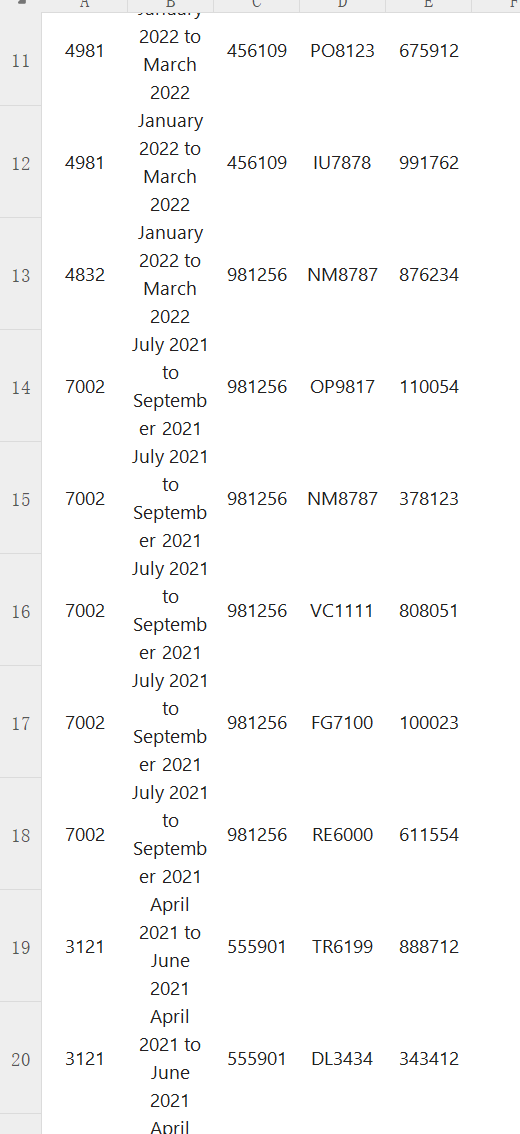


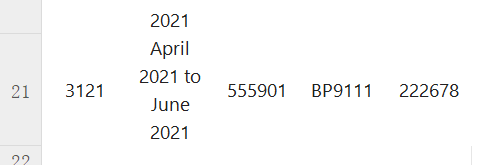
3NF











**CHALLENGE YOURSLEF**

9.

a.

Logical Model:

Entities: Boat, Engineer, Contractor, Task, ServiceRecord.

Attributes:

Boat: boat\_id, marine\_equipment\_type, maintenance\_tasks.

Engineer: engineer\_id, name.

Contractor: contractor\_id, name, skill.

Task: task\_id, task\_type (Service/Software upgrade/Repair/Safety inspection/Other).

ServiceRecord: record\_id, boat\_id, engineer\_id, task\_id, servicing\_time, man\_hours, service\_date.

Physical Model:

Tables:

boats (boat\_id, marine\_equipment\_type, maintenance\_tasks).

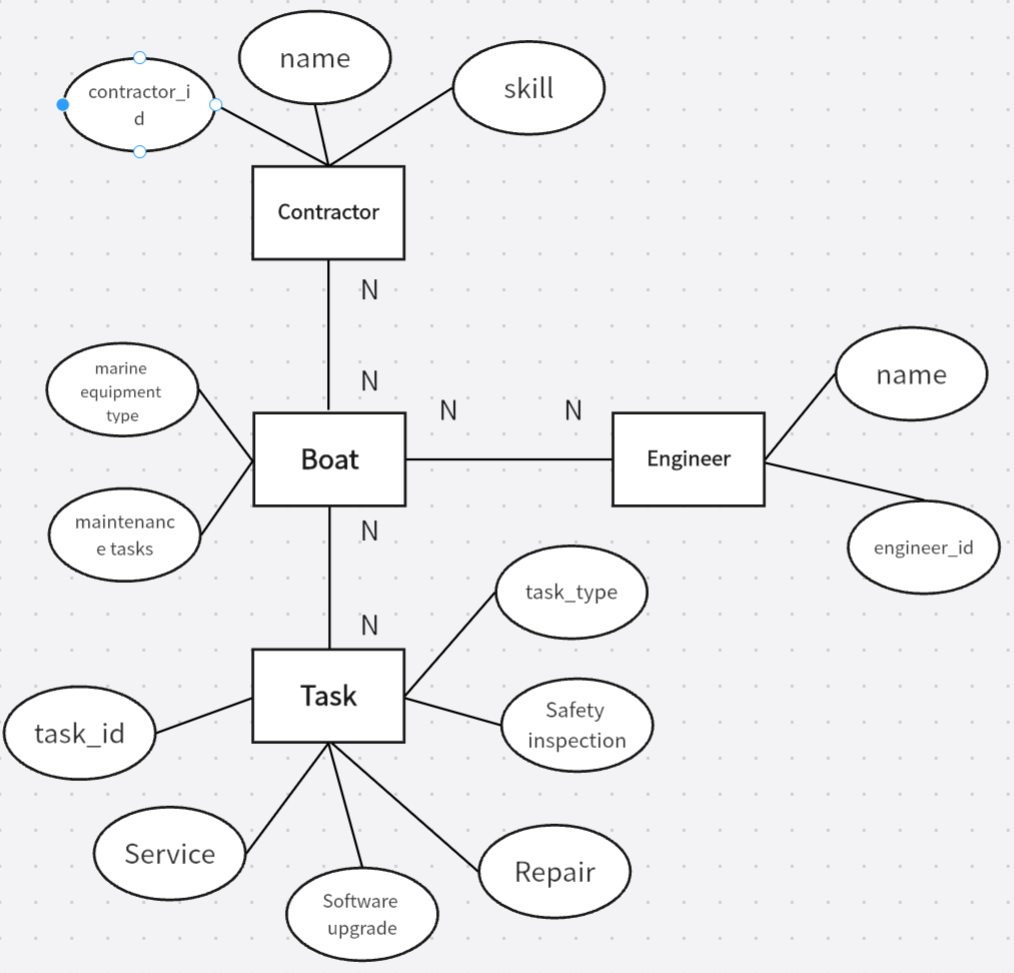
engineers (engineer\_id, name).

contractors (contractor\_id, name, skill).

tasks (task\_id, task\_type).

servicerecords (record\_id, boat\_id, engineer\_id, task\_id, servicing\_time, man\_hours, service\_date).

b.



c.

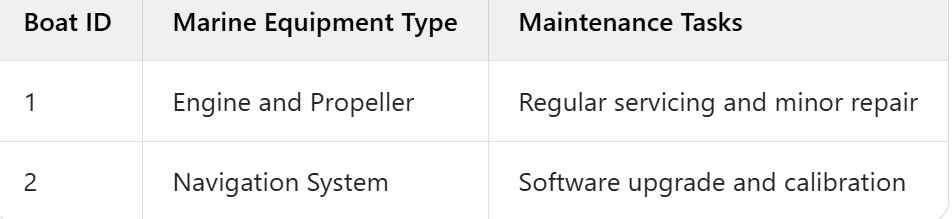
1.In the boats table, boat\_id determines marine\_equipment\_type and maintenance\_tasks.

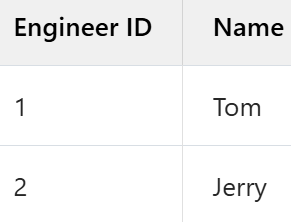
2.In the engineers table, engineer\_id determines name.

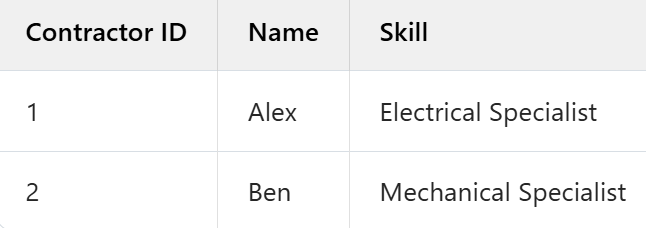
3.In the tasks table, task\_id determines task\_type.

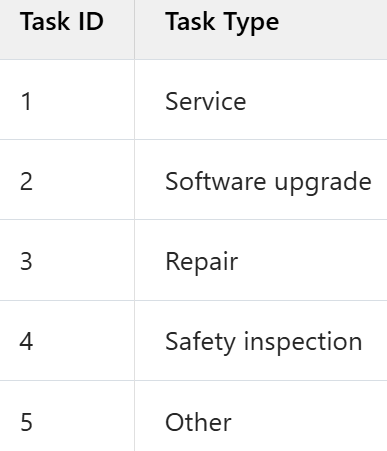
4.In the servicerecords table, (boat\_id, engineer\_id, task\_id) determines servicing\_time, man\_hours, and service\_date.

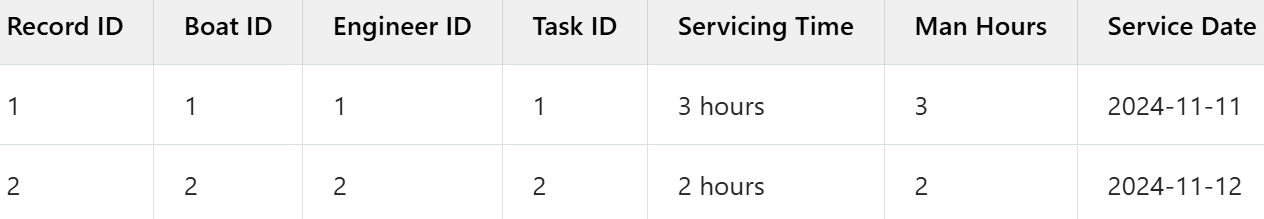
d.





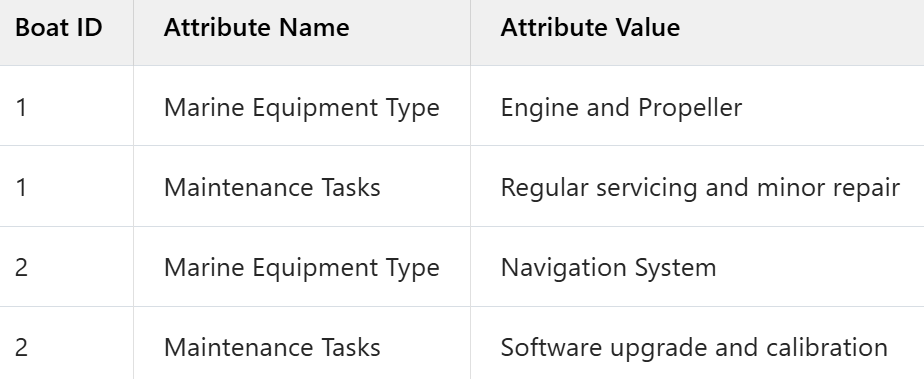


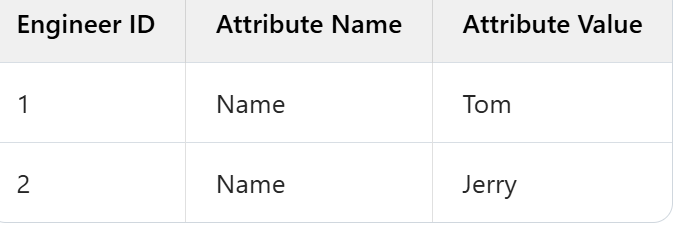


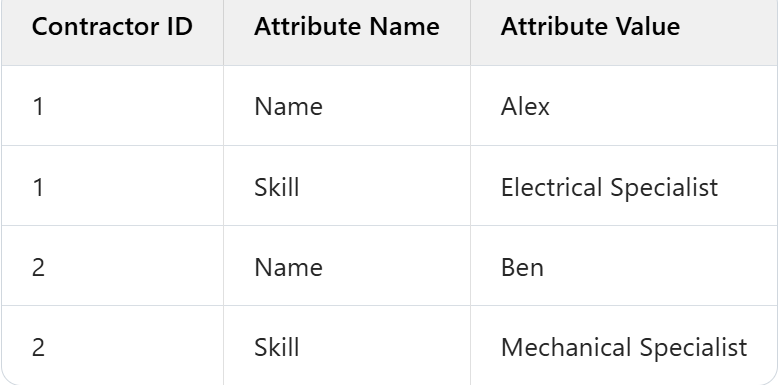


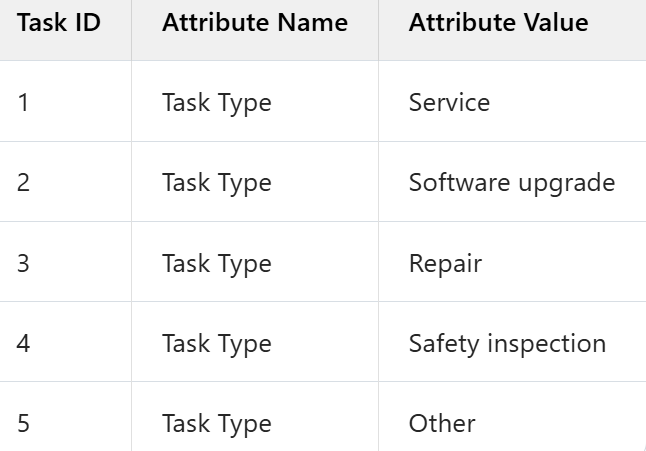
e.

1NF



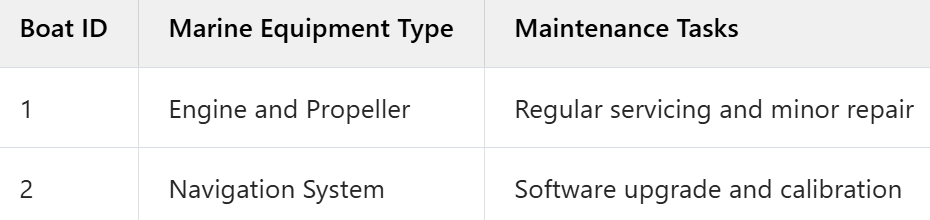


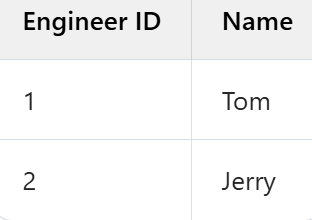


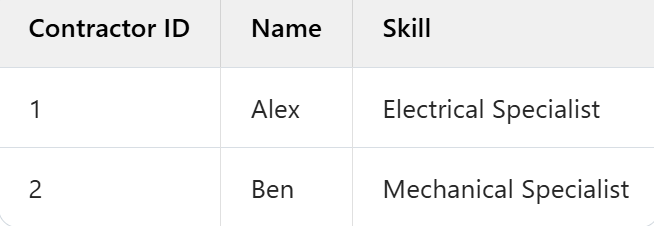


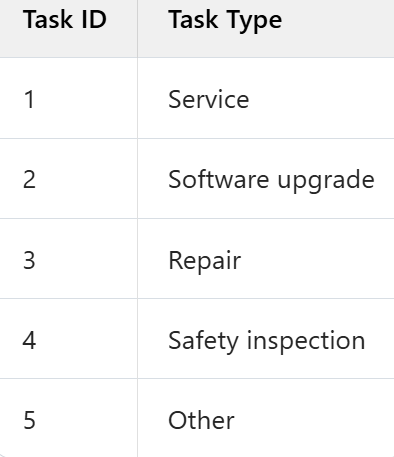


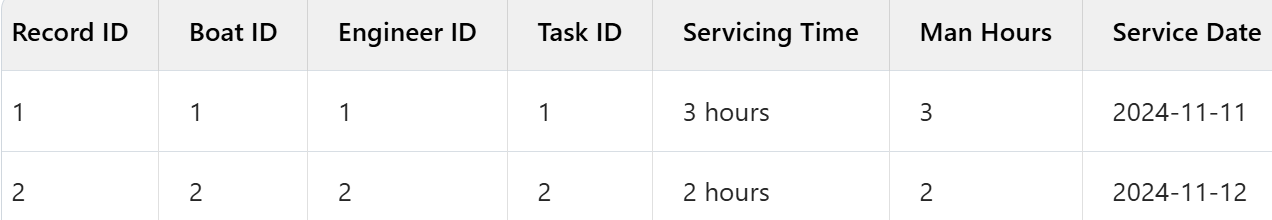
2NF











3NF

