Basic task

a. Candidate key:

A candidate key is a minimal set of attributes that can uniquely identify each tuple (row) in a relation (table). In other words, it is a set of attributes that, taken together, ensure that no two rows in the table have the same values for these attributes. Candidate keys are essential for maintaining data integrity and enabling efficient retrieval and manipulation of data. For example, in a table of students with attributes like student ID, name, and address, the student ID could be a candidate key as it uniquely identifies each student.

b. Composite key: A composite key is a key that consists of two or more attributes combined to uniquely identify a tuple in a relation. It is used when a single attribute is not sufficient to uniquely identify a row. For instance, in a table of orders where each order is identified by a combination of customer ID and order number, the combination of these two attributes forms a composite key.

c. Foreign key: A foreign key is an attribute or a set of attributes in one table that refers to the primary key of another table. It establishes a relationship between two tables and enforces referential integrity. For example, if there are two tables, one for customers and another for orders, the customer ID in the orders table would be a foreign key that references the customer ID in the customers table. This ensures that only valid customer IDs are used in the orders table.

d. Functional dependency: Functional dependency is a relationship between two sets of attributes in a relation. If attribute A determines attribute B, then it is said that B is functionally dependent on A. In other words, for a given value of A, there is only one possible value of B. For example, in a table of employees with attributes employee ID and department name, if an employee ID uniquely determines the department name, then there is a functional dependency between employee ID and department name.

2.

In the relational model, there are three important integrity rules/constraints:

1)Entity integrity:

Definition: Entity integrity ensures that each row in a table is uniquely identified by a primary key and that the primary key value cannot be null. This means that every entity in the database must be uniquely identifiable and cannot exist without a proper identifier. For example, in a table of employees, the employee ID might be the primary key. Each employee must have a unique employee ID, and this ID cannot be left blank.

2)Referential integrity:

Definition: Referential integrity enforces consistency between related tables. It ensures that values in a foreign key column in one table match the values of the primary key in another table. If a foreign key refers to a primary key in another table, then the value in the foreign key column must exist in the referenced primary key column. For instance, if there is a table of departments and a table of employees where the employee table has a foreign key referencing the department ID in the department table, every department ID in the employee table must correspond to an existing department ID in the department table.

3)Domain integrity:

Definition: Domain integrity constrains the values that can be entered into a particular column. It defines the valid data types, ranges, and formats for attributes in a table. For example, if a column is defined as an integer, only integer values can be entered. Or if a column is for dates, only valid date formats are allowed. Domain integrity also includes constraints like check constraints and default values. For instance, a check constraint might be used to ensure that values in a column are within a certain range.

3.

Table1:

1. Primary key is not unique, fo filmNo or directorNo, both of these could be.
2. DirectorNo753 isn’t in director table,we can’t know its name.
3. FilmNo008 ‘s genre is not given in this table.

Table2:

1. Inconsistent data formats, in the column PartNo, there are again numbers, upper and lower case letters, and symbols.
2. Inconsistent data formats, Aziz ‘s quantity in this table is ABC, clearly inconsistent with the facts.

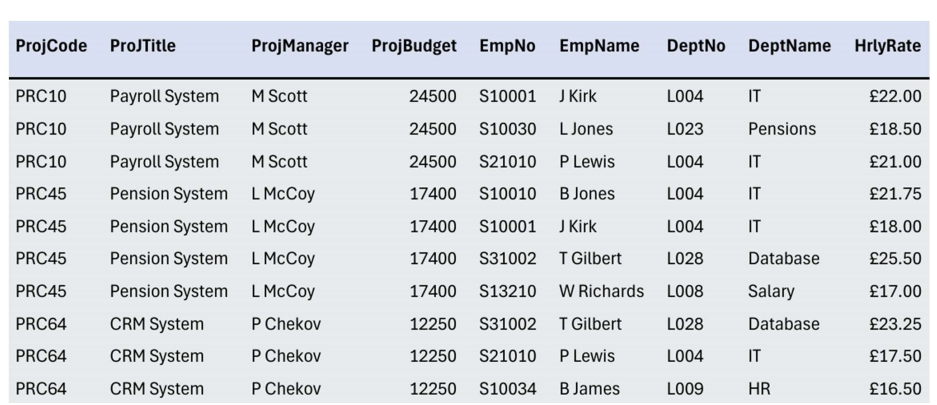
Table3:

1. Primary key is null, the song A kind of magic ‘s artist doesn’t know.

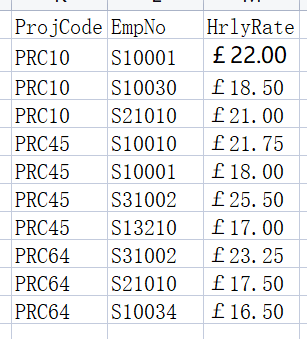
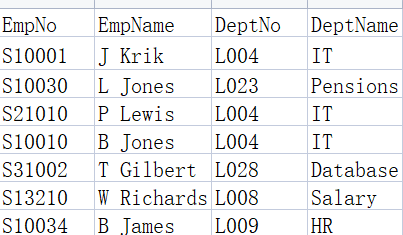
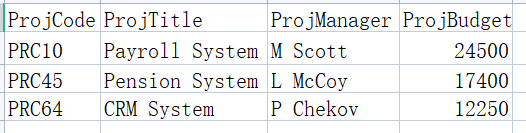
Medium task:

4.

1. Insertion exceptions: Insertion of new items which may have inconsistencies and insertion difficulties due to redundancy of existing data (e.g., there may be multiple employee records in the same row for the same item).
2. Deletion anomaly: Deletion of item ‘PrC10’ may result in the loss of related employee information (e.g., employee information sections such as S10001, S10030, etc.) because data related to the item may be deleted along with it.
3. Modification Exception: Moving ‘J Kirk’ from department ‘L004’ to department ‘L009’ requires modification of multiple rows of data due to data redundancy (the same employee may be in department information in different projects need to be modified), it is prone to data inconsistency.
4. 1NF: The title given already satisfies 1NF.



1. 2NF: Based on 1NF, partial function dependencies of non-primary attributes on codes are eliminated. For example, the departmental information and hourly wage of an employee may depend on the employee number rather than the combination of project number and employee number, which can further decompose the table structure.
2. 3NF: builds on 2NF by eliminating transfer function dependencies of non-primary attributes on codes. Ensure that each non-primary attribute is directly dependent on the code and does not pass dependencies through other non-primary attributes.



5.

a.Insertion exceptions: If you insert a new order, you need to repeat the insertion of customer information, such as address, etc., which is cumbersome and error-prone.

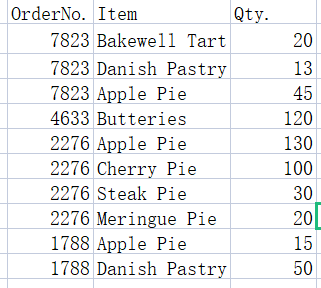
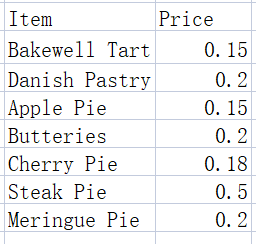
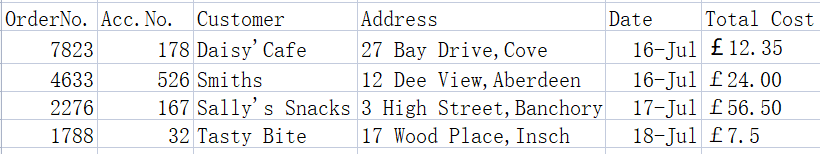
Update Exception: If a customer's address changes, it is necessary to update the address information in multiple order records.

Delete Exception: If an order is deleted, the customer information may be deleted by mistake (if the customer has only this one order record).

b.The data is broken down so that each order is treated as a row, and each attribute in each row contains only one value, with no repeating groups. For example, you can create an order table (Order No., Acc. No., Customer, Address, Date) and an order details table (Order No., Item, Qty., Price, Total Cost)

c.On the basis of 1NF, function dependencies are analysed to eliminate partial function dependencies of non-primary attributes on the code. For example, Item Price in the Order Details table may depend only on Item and not the whole Order No. The table structure can be further decomposed. For example, the Order Details table can be further split into the Order Item table (Order No., Item, Qty.) and the Item Price table (Item, Price).

d.Eliminate transfer function dependencies of non-primary attributes on codes based on 2NF. Ensure that each non-primary attribute is directly dependent on the code and does not pass dependencies through other non-primary attributes. For example, while the Order Commodity and Commodity Price tables above are already 3NF compliant, for the Order and Customer table sections, if there is some kind of indirect dependency between Customer and Order (e.g., the Customer category determines certain Order attributes), further decomposition is required to ensure 3NF compliance. assuming that there are no such complex relationships at present, the Order table (Order No., Acc. No. No., Acc. No., Customer, Address, Date) and Customer (Customer, Address) conform to 3NF (assuming for simplicity that Customer is the primary key of the Customer table, and that if a more appropriate unique identifier exists, it should be chosen as the primary key).



6.

1. Patient：（PatientNo，Surname，FirstNname）

Admission：（PatientNo，Admitted，Discharged，Ward）

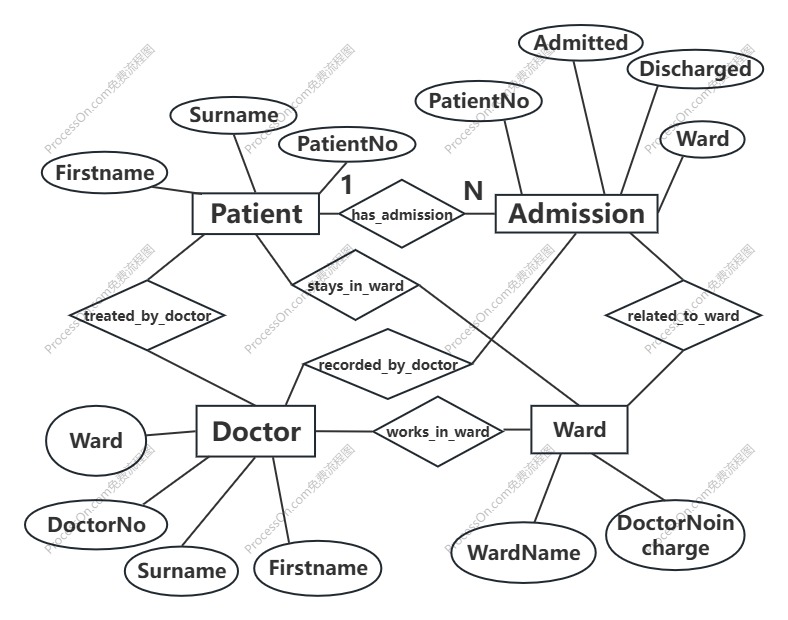
Doctor：（DoctorNo，Surname，FirstName，Ward）

Ward：（Ward，WardName，DoctorNo - InCharge）

1. 1)The Patient table and the Admission table establish a one-to-many relationship through PatientNo, which is a foreign key to the Admission table and corresponds to the primary key of the Patient table.

2)The Doctor table and the Ward table establish a one-to-many relationship through DoctorNo - InCharge and DoctorNo, which is a foreign key of the Ward table and corresponds to the primary key of the Doctor table.

3)The Patient table has no direct relationship with the Doctor table.

c.

Advanced task

7.

1. the functional dependencies

Table 1:

Order No.→Account No., Customer, Address, Date

Item→Quantity, Item Price

Table 2:

Student No→Name, Course, Course Duration

Course→Module No, Module name, Lecturer

b.

Table 1:

1NF: Separate the repeated rows. Use the order number as the association. Each order number corresponds to one row, and the order details are on another row.

2NF: Based on 1NF, analyze the functional dependency relationship and eliminate the partial functional dependency of non-primary attributes on the code. Since the item price may only depend on the item, the item-related information can be put in a separate table. 3NF: Based on 2NF, eliminate the transitive functional dependency of non-primary attributes on the code. Ensure that each non-primary attribute directly depends on the code and does not depend transitively through other non-primary attributes.

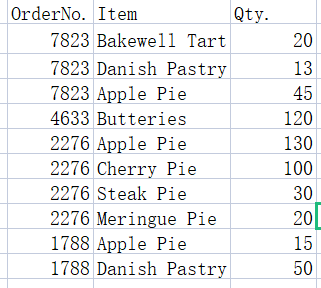
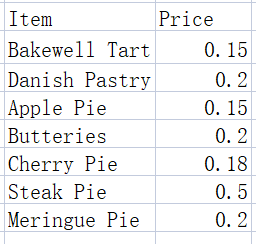
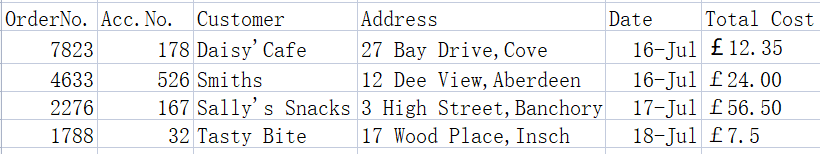
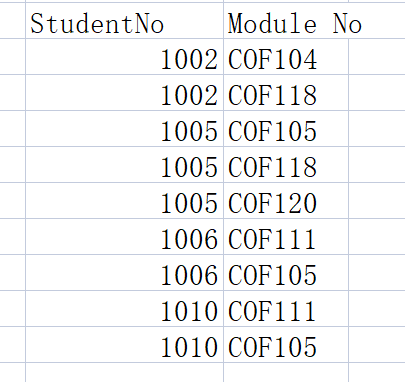
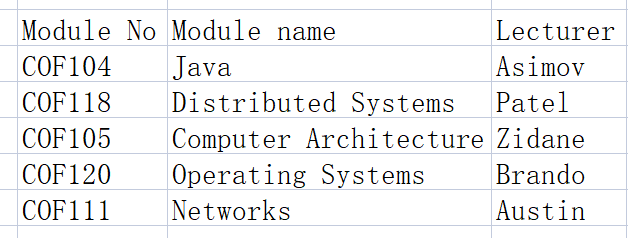
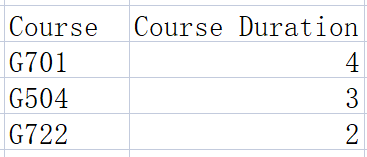
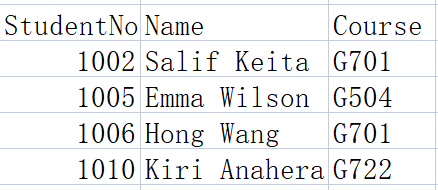


Table 2:

1NF: Separate the repeated rows. Use the student number as the association. Each student number corresponds to one row, and the course module-related information is on another row.

2NF: Based on 1NF, analyze the functional dependency relationship and eliminate the partial functional dependency of non-primary attributes on the key. Since the module name may only depend on the module number, the module-related information can be put in a separate table.

3NF: Based on 2NF, eliminate the transitive functional dependency of non-primary attributes on the key. Ensure that each non-primary attribute directly depends on the key and does not depend transitively through other non-primary attributes.



8.

a.the functional dependencies

Report ID, Reporting Period→Branch Code, Branch Name, Supervisor ID, Supervisor Name

Branch Code, Car Plate Nr→Car Type

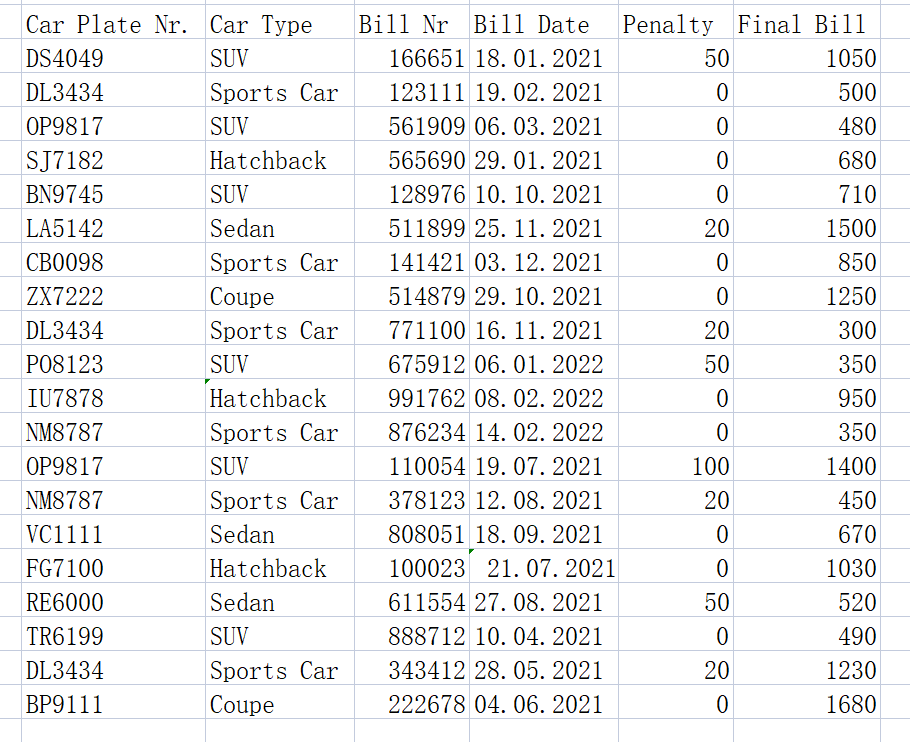
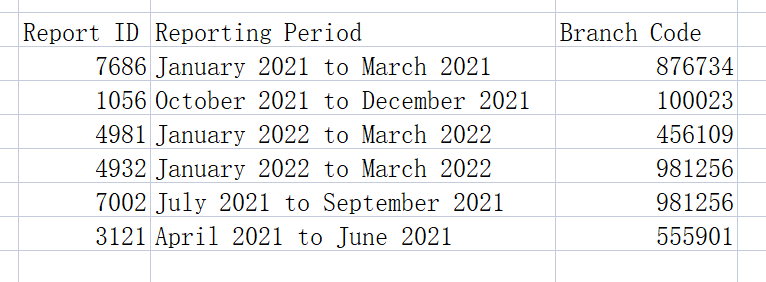
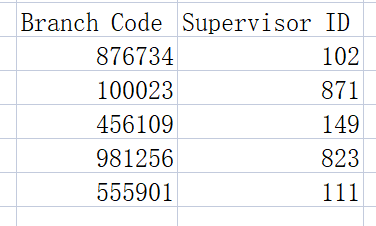
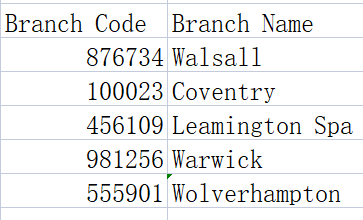
Bill Nr→Bill Date

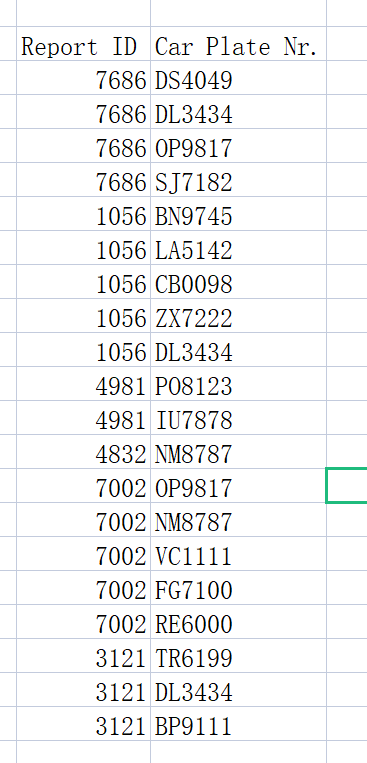
Car Plate Nr, Bill Nr→Penalty, Final Bill

b.

1NF: Eliminate repeating groups and decompose the data into a table structure that conforms to the first normal form. For example, a report table (Report ID, Reporting Period, Branch Code, Branch Name, Supervisor ID, Supervisor Name), a vehicle table (Branch Code, Car Plate Nr, Car Type), a bill table (Bill Nr, Bill Date), and a fine and bill details table (Car Plate Nr, Bill Nr, Penalty, Final Bill) can be created. 2NF: Based on 1NF, analyze the functional dependency relationship and eliminate the partial functional dependency of non-primary attributes on the key. For example, in the vehicle table, Car Type may only depend on Car Plate Nr, and the table structure can be further decomposed.

3NF: Based on 2NF, eliminate the transitive functional dependency of non-primary attributes on the key. Ensure that each non-primary attribute directly depends on the key and does not depend transitively through other non-primary attributes.





Challenge:

9.

a.

Boat ID→Marine equipment type, Servicing time, Maintenance tasks

Engineer ID→Servicing time, Man-hours

Task ID→Service (Software upgrade, Repair, Safety inspection, Other)

b.

1NF: Eliminate repeating groups and decompose the data into a table structure that conforms to the first normal form. For example, a boat table (Boat ID, Marine equipment type, Servicing time, Maintenance tasks), an engineer table (Engineer ID, Servicing time, Man-hours), and a task table (Task ID, Service) can be created.

2NF: Based on 1NF, analyze the functional dependency relationship and eliminate the partial functional dependency of non-primary attributes on the key. For example, in the engineer table, Man-hours may only depend on Engineer ID, and the table structure can be further decomposed.

3NF: Based on 2NF, eliminate the transitive functional dependency of non-primary attributes on the key. Ensure that each non-primary attribute directly depends on the key and does not depend transitively through other non-primary attributes.