SubanKamo 300460671

<u>Question 1 [20 marks]</u>
<u>a) [8 marks] For every set of attributes (that is, for every subset of the attribute set) decide whether you can deduce that it is not a candidate key, assuming the shown instance is legal. Justify your answer.</u>
To be a candidate key must meet all conditions: unique (no duplicate), minimal(when multiple attribute - removing one attribute means remaining is no longer key), not null

{Height}: This set cannot be a candidate key since there can be multiple buildings with the same height in the table (This would make it difficult for us to obtain the correct records making it redundant as a candidate key).
For example, in the table, there are two buildings with a height of 530 meters.
[does not meet condition: unique]

{Name}: This set can be a candidate key since there are no buildings with the same name in the table (This would make it possible for us to obtain the correct records making it viable as a candidate key).

{City}: This set can be a candidate key since there are no buildings in the same city in the table (This would make it possible for us to obtain the correct records making it viable as a candidate key).
However this is unlikely as it defies logical assumption as there is likely to be multiple buildings in a city.

{Country}: This set cannot be a candidate key since there can be multiple buildings in the same country in the table (This would make it difficult for us to obtain the correct records making it redundant as a candidate key).
For example, in the table, there are two buildings in China and two in the United Arab Emirates.
[does not meet condition: unique]

{Year}: This set cannot be a candidate key since there can be multiple buildings built in the same year in the table (This would make it difficult for us to obtain the correct records making it redundant as a candidate key).
For example, in the table, there are two buildings built in 2019.
[does not meet condition: unique]

{Height, Country}: This set cannot be a candidate key since there can be multiple buildings with the same height and country in the table (This would make it difficult for us to obtain the correct records making it redundant as a candidate key).
For example, in the table, there are two buildings in China with the same height
[does not meet condition: unique]

{Height, Year}: This set can be a candidate key since there are no buildings with the same height and year in the table (This would make it possible for us to obtain the correct records making it viable as a candidate key).

{Year, Country}: {Height, Country}: This set cannot be a candidate key since there can be multiple buildings with the same year and country in the table (This would make it difficult for us to obtain the correct records making it redundant as a candidate key).
For example, in the table, there are two buildings in China with the same year.
[does not meet condition: unique]

{Any other set}: These sets cannot be a candidate key as {Name}, {City} and {Height, Year} are candidate keys, so sets containing these attributes are not minimal.
[does not meet condition: minimal]

b) [4 marks] For every remaining set of attributes (that is, for every set not ruled out as a candidate key in part a)), discuss whether you consider it a suitable candidate key? Justify your answer.
{Name}: This Name attribute is a suitable candidate key as logically there is a stronger assumption that building names are unique.
This means that it would be better at helping us identify a specific building in the database as it is likely to only return 1 result.

{City}: This City attribute is not a suitable candidate key as it defies logical assumption that there are likely to be multiple buildings in a city.
This means that it would be weaker at helping us identify a specific building in the database as it is likely to return multiple results.

{Height, Year}: It is not a suitable candidate key. As both Height and Year attributes have duplicates. It is not guaranteed that as we add more data to the database results from {Height, Year} will not have duplicates as the instances grow.
This means that it would be weaker at helping us identify a specific building in the database as it is likely to return multiple results.

c) [2 marks] Which of the candidate keys identified in part b) would you choose as the primary key? Justify your answer
This Name attribute is the more suitable primary key when compared to the City attribute or the Height and Year. This is because buildings' names are more likely to be unique than the city the buildings are in (logically we expect multiple buildings to originate from the same city) or the Height and year (logically we expect multiple buildings to be built with the same height in the same year).
Therefore the name attribute better satisfies the condition of uniqueness (primary keys cannot be duplicated) as it is less likely for two buildings to be of the same name than from the same city or to be built with the same height in the same year.
Therefore Name would be better to set as the primary key.

d) [2 mark] Add a new tuple for a building into the TALLBUILDING relation. How would you check that the primary key identified in part c) is still valid?
If the Name attribute is not null and does not exist in the TALLBUILDING relation.

e) [2 mark] Create a relation that shows for each country in the table above the country and the capital, i.e., use a relation schema with attribute set {Country, Capital}. How many records are in your relation?

The number of records in the relation will be 6, as there are six unique countries in the TALLBUILDINGS relation schema:

f) [2 mark]. Consider a relation schema with attribute set {City, Year} and assume that both attributes have a domain with 130 values each. What would be the maximum number of records in an instance of this relation schema
130^2= 16,900

Question 2 [10 marks]
1. Insert tuple (null, ‘4’, ‘01/01/2023’) into LISTENS
Cannot be added to the LISTENS relation as the primary key for the LISTENS relation is {UID, TID}.
which means that neither UID nor TID can be null.
Therefore, the insertion of this tuple violates the primary key constraint as UID is null.

2. Delete tuple (‘349’, ‘4’, null) from LISTENS
The tuple (‘349’, ‘4’, null) can be removed from the LISTENS relation as it exists in the relation and does not violate any constraints.

3. Insert tuple (‘6’, ‘Like a Rolling Stone’, ‘Bob Dylan’, ‘Folk’, ‘373’, ‘20/07/1965’) into TRACKS
The tuple (‘6’, ‘Like a Rolling Stone’, ‘Bob Dylan’, ‘Folk’, ‘373’, ‘20/07/1965’) can be added to the TRACKS relation as it does not violate any constraints and all attribute values are valid.

4. Insert tuple (‘64381’, ‘6’, ‘20/12/2023’) into LISTENS
The tuple (‘64381’, ‘6’, ‘20/12/2023’) can be added to the LISTENS relation as it does not currently exist in the relation and does not violate any constraints.

5. Delete tuple (‘12’, ‘Emma Wilson’, ‘emmawilson_23@companyname.co.uk’) from USERS
The tuple (‘12’, ‘Emma Wilson’, ‘emmawilson_23@companyname.co.uk’) can be deleted from the USERS relation as it exist in table but should not be deleted as it is referred to by the LISTENS relation through the foreign key UID.
Deleting this tuple would violate the referential integrity constraint.

6. Delete tuple (‘6’, ‘Never Gonna Give You Up’, ‘Rick Astley’, ‘Pop’, ‘215’, ‘27/07/1987’) from TRACKS
The tuple (‘6’, ‘Never Gonna Give You Up’, ‘Rick Astley’, ‘Pop’, ‘215’, ‘27/07/1987’) can be deleted from the TRACKS relation as it exists in the relation and does not violate any constraints.

7. Insert tuple (‘30’, ‘Andrew Lensen’, ‘andrew.lenson@vuw.ac.nz’) into USERS
The tuple (‘30’, ‘Andrew Lensen’, ‘andrew.lenson@vuw.ac.nz’) can be added to the USERS relation as it does not violate any constraints and all attribute values are valid.

8. Insert tuple (‘12’, ‘7’, ‘03/03/2023’) into LISTENS

The tuple (‘12’, ‘7’, ‘03/03/2023’) cannot be added to LISTENS because there is no track with TID value of 7 in the TRACKS relation.
This violates foreign key constraints.

<u>9. Insert tuple (‘0’,‘A Horse With No Name’, ‘America’, ‘Rock’, ‘248’, ‘12/11/1971’) into TRACKS</u>
The tuple (‘0’,‘A Horse With No Name’, ‘America’, ‘Rock’, ‘248’, ‘12/11/1971’) can be added to the TRACKS relation as it does not violate any constraints and all attribute values are valid.

<u>10. Insert tuple (‘1092’, ‘2’, ‘13/01/2023’) into USERS</u>
The tuple (‘1092’, ‘2’, ‘13/01/2023’) can be added to the LISTENS relation as the UID and TID values exist in the corresponding relations and the ListenDate attribute value is valid.

<u>Question 3 [20 marks]</u>
<u>a) [3 marks] For the relation schema Vet, identify all suitable candidate keys. Explain your answer. Which candidate key would you choose as the primary key? Justify your answer.</u>
To be a candidate key must meet all conditions: unique (no duplicate), minimal(when multiple attribute - removing one attribute means remaining is no longer key), not null

{VetRegistration}: The VetRegistration attribute could be a candidate key because it is unique for each vet. This attribute satisfies the uniqueness and minimality requirements for a candidate key, since no two vets should have the same registration number.

{FirstName, LastName }: FirstName and LastName could be a candidate key because together they can uniquely identify a vet. This attribute combination satisfies the uniqueness requirement, since no two vets should have the same first and last name.
However, it is not guaranteed that as we add more data to the database results from {FirstName, LastName } will not have duplicates as the instances grow.
This means that it would be weaker at helping us identify a specific vets in the database as it is likely to return multiple results.

Primary Key:
VetRegistration is more suitable to be the primary key.
This is because it satisfies both the uniqueness and minimality requirements which may not be met by FirstName and LastName as it is more likely for multiple vets to have the same first and last name than VetRegistration.

<u>b) [5 marks] For each of the relation schemas, identify all suitable foreign keys (if there are any). Explain your answer.</u>
>Assume VetRegistration is primary key of Vet schema
• Animal (AnimalID, Species, Name) with primary key {AnimalID}
-None, no relationship with other table

• Vet (FirstName, LastName, VetRegistration) with unknown primary key
-None, no relationship with other table

• Vaccinations (VaccineName, VaccineDescription, VaccinePrice) with primary key {VaccineName}
-None, no relationship with other table

• VaccinationHistory (AnimalID, VaccineName, Date, Location) with primary key {VaccineName, Date, AnimalID}
Foreign keys: AnimalID ⊆ Animal[AnimalID] and VaccineName ⊆ Vaccinations[VaccineName]
-AnimalID: This foreign key allows us to link vaccination records to specific animals.
-VaccineName: This foreign key allows us to track which vaccine was administered to each animal.

• VaccinationAppointment (AnimalID, VetRegistration1, VetRegistration2, VaccineName. Date) with primary key {VaccineName, Date, AnimalID}
Foreign keys: AnimalID ⊆ Animal[AnimalID] and VaccineName ⊆ Vaccinations[VaccineName] and VetRegistration1 ⊆ Vet[VetRegistration]and VetRegistration2 ⊆ Vet[VetRegistration]
-AnimalID: This foreign key allows us to link vaccination appointments to specific animals.
-VaccineName: This foreign key allows us to track which vaccine is scheduled to be administered during each appointment.
-VetRegistration1 & VetRegistration2: These foreign keys allow us to track which vets are scheduled to administer each vaccine.

c) [5 marks] Assume, the vet with name 'James Herriot' in the Vet relation retires. When deleting the record of this vet from the Vet relation, all the vaccinations performed by him should not be lost. How would you ensure this requirement? Explain your answer.
To prevent the loss of vaccination records performed by a retired vet, we can add a "Retired" attribute to the Vet table with a boolean value indicating whether the vet is retired.
Instead of deleting the record of a retired vet like "James Herriot", we can update their "Retired" attribute to "true" to retain their vaccination records in the database whilst indicating that they are no longer acting as a vet.

Alternatively, you could set the Vet ID value for vaccination performed by James Herriot to null/default for each VaccinationAppointment record using DELETE SET NULL or DELETE SET DEFAULT. This is so it will not be affected by cascading delete as the two tables are no longer connected through the foreign key VetID.

d) [2 marks] You notice that there are a lot of NULL values appearing in your new database that are not consistent with the "real world"/UoD. Identify two not null constraints you could add to make the database more meaningful.
1. The Animal relation schema should have a not null constraint on the Species attribute.

In the real world, every animal has a species, so it makes sense to enforce this constraint in the database.

2. The VaccinationAppointment relation schema should have a not null constraint on the VetRegistration1 and VetRegistration2 attributes.

Since in the real world it takes two vets to safely administer a vaccine, we should ensure that both vet registration numbers are included in every vaccination appointment record.

e) [5 marks] Murphy the Siamese elephant has sadly passed away. A zoo employee tries to delete Murphy from the database by deleting his AnimalID (which is '31') from the Animal table. However, they notice that all his vaccination records are still in the database. What went wrong, and how could they have prevented this?

>Assuming there is no foreign key constraint

The table Animal and VaccinationHistory does not have a relationship. Although they share the same attribute AnimalID they are not connected through a foreign key. Therefore when murphy is deleted from the Animal table it does not affect the VaccinationHistory table resulting in 'orphaned' records (records with murphy's animalID remaining in VaccinationHistory).

To prevent this issue, the database should have been designed with foreign key constraints between the tables, and with cascading delete constraints to ensure that all related records are deleted when the corresponding record in the Animal table is deleted.

>Assuming there is a foreign key

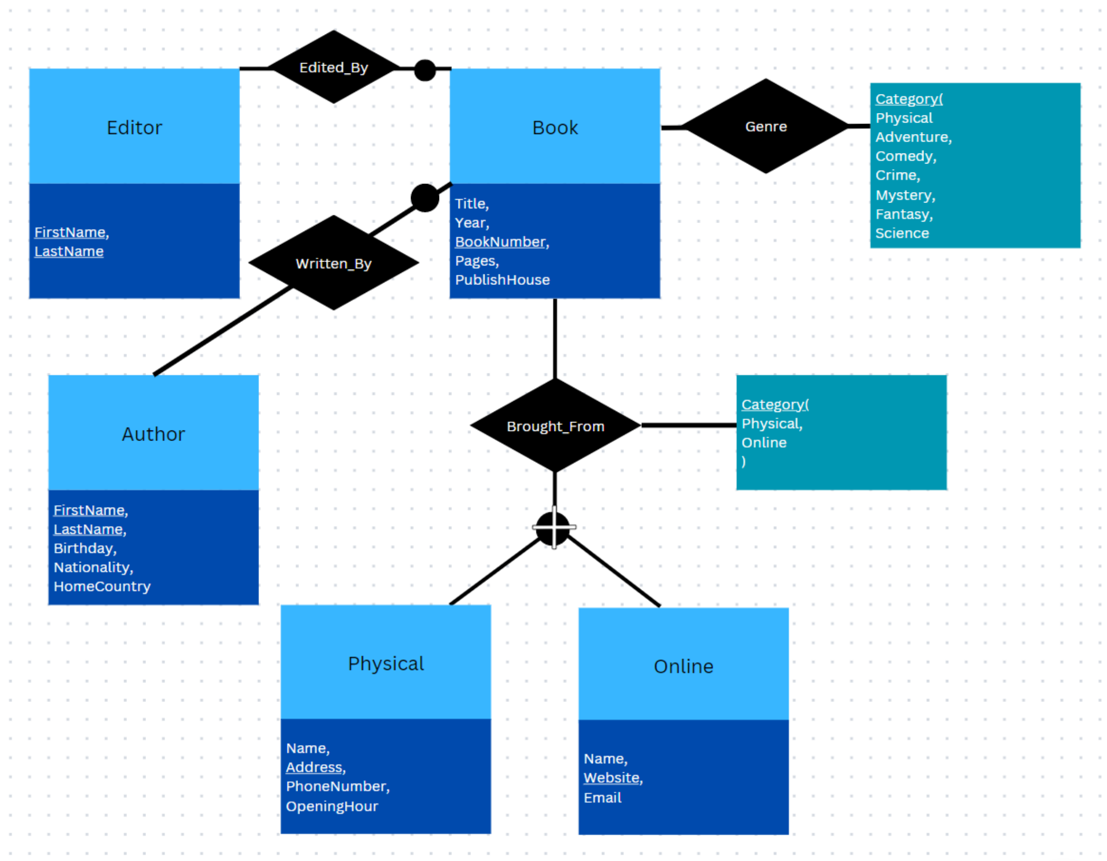
The table Animal and VaccinationHistory does not have cascading delete constraints configured. Therefore when murphy is deleted from the Animal table it does not affect the VaccinationHistory table resulting in 'orphaned' records (records with murphy's animalID remaining in VaccinationHistory).

To prevent this issue, the database should have been designed with cascading delete constraints to ensure that all related records are deleted when the corresponding record in the Animal table is deleted.

Question 4 [30 marks]

a) [24 marks] Draw an extended ER diagram for the database above. Write down the corresponding extended ER schema, including declarations of all the entity types (showing attributes and keys) and relationship types (showing components, attributes and keys).

SubanKamo 300460671



Entity
-Book
Attributes: (Title, Year, BookNumber, Pages, PublishHouse)
primary key {BookNumber}

-Editor
Attributes: (FirstName, LastName)
primary key {FirstName, LastName}

-Author
Attributes: (FirstName, LastName, Birthday, Nationality, HomeCountry)
primary key {FirstName, LastName}

-Physical
Attributes: (Name, Address, PhoneNumber, OpeningHour)
primary key {Address}

-Online
Attributes: (Name, Website, Email)
primary key {Website}

Relationship
-Written_By
Component: Book, Author

Attributes: none
PrimaryKey: {Book, Author}

-Edited_By()
Component: Book, Editor
Attributes: none
PrimaryKey: {Book, Editor}

-Bought_From
Component: Book, Physical, Online
Attributes: (category)
PrimaryKey: {Book, category}

-Genre
Component: Book
Attributes: (category)
PrimaryKey: {Book, category}

RDM
-Book
Attributes: (Title, Year, BookNumber, Pages, PublishHouse)
primary key {BookNumber}

-Editor
Attributes: (FirstName, LastName)
primary key {FirstName, LastName}

-Author
Attributes: (FirstName, LastName, Birthday, Nationality, HomeCountry)
primary key {FirstName, LastName}

-Physical
Attributes: (Name, Address, PhoneNumber, OpeningHour)
primary key {Address}

-Online
Attributes: (Name, Website, Email)
primary key {Website}

-Written_By
Attributes: (BookNumber, FirstName, LastName)
PrimaryKey: {BookNumber, FirstName, LastName}
ForeignKey: BookNumber ⊆ Book[BookNumber], FirstName ⊆ Author[FirstName],
LastName ⊆ Author[LastName]

-Edited_By()
Attributes: (BookNumber, FirstName, LastName)
PrimaryKey: {BookNumber, FirstName, LastName}

ForeignKey: BookNumber ⊆ Book[BookNumber], FirstName ⊆ Editor[FirstName], LastName ⊆ Editor[LastName]

-Bought_From
Attributes: (BookNumber, Category)
PrimaryKey: {BookNumber, Category}
ForeignKey: BookNumber ⊆ Book[BookNumber]

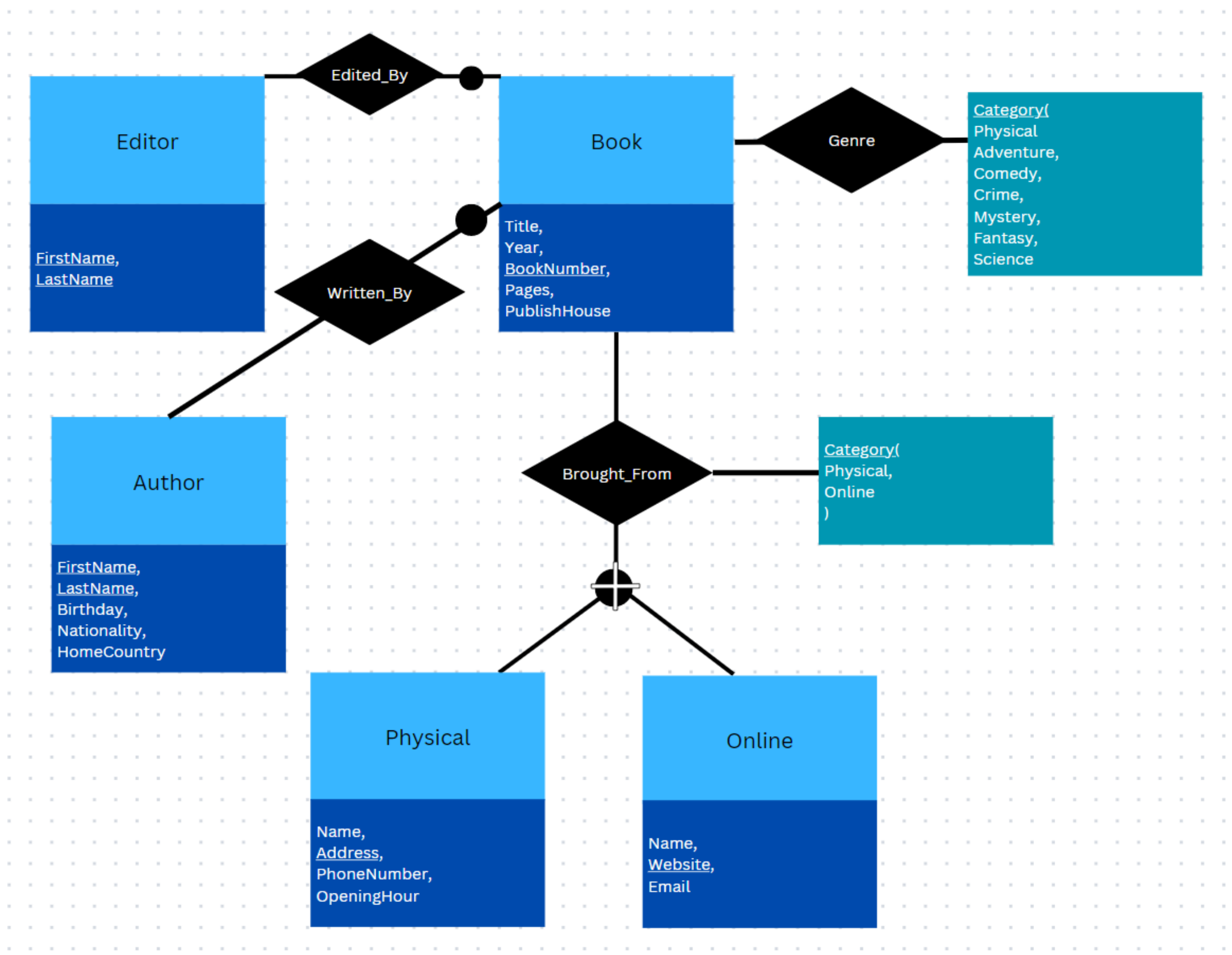
-Genre
Attributes: (BookNumber, Category)
PrimaryKey: {BookNumber, Category}
ForeignKey: BookNumber ⊆ Book[BookNumber]

b) [6 marks] There may be information, requirements or integrity constraints that you are not able to represent in your diagram. Give three examples of integrity constraints that have not been represented in your diagram.

1. Unique author/editor names: The ER diagram does not enforce a constraint that ensures that author and editor names are unique within their respective tables. This could potentially result in duplicate entries for authors or editors, which could cause confusion when querying the database.
2. Non-negative number of pages: The ER diagram does not enforce a constraint that ensures that the number of pages for a book is non-negative. This could potentially result in invalid data being stored in the database, which could lead to incorrect results when querying the database.
3. Required Connection Between Author and Editor: The ER diagram does not depict how a book without authors has at least one editor. This could potentially result in missing data in the database, which could lead to incorrect results when querying the database.

Question 5 [20 marks]
a) [5 marks] Present the extended ER schema of the extended ER diagram above.

SubanKamo 300460671



Entity
-Item
Attributes: (ItemId, Value, ItemDescription, Weight)
primary key {ItemId}

-Customer
Attributes: (Name, Address, Postcode, Country, Phone)
primary key {Address}

-Courier
Attributes: (Name, Phone, Id)
primary key {Id}

Relationship
-Expense
Component: Shipment
Attributes: (date, cost, Description)
PrimaryKey: {Shipment, Description}

-Shipment
Component: Item, Courier, From: Customer, To:Customer

Attributes: (dateSent, dateReceived, Status)
PrimaryKey: {Item, Courier, From: Customer, To:Customer}

b) [15 marks] Transform your extended ER schema into a relational database schema. In particular, list all the relation schemas in your relational database schema. For each relation schema, list all attributes, the primary key, the NOT NULL constraints, and the foreign keys.

-Item
Attributes: (ItemId, Value, ItemDescription, Weight)
primary key: {ItemId}
Not Null: ItemId
Foreign key: none

-Customer
Attributes: (Name, Address, Postcode, Country, Phone)
primary key {Address}
Not Null: Address
Foreign key: none

-Courier
Attributes: (Name, Phone, Id)
primary key {Id}
Not Null: Id
Foreign key: none

-Expense
Attributes: (ItemId, date, cost, Description)
PrimaryKey: {Description}
Not Null: itemID, Description
Foreign Key: itemID ⊆ Shipment[itemID], Shipment[itemID] ⊆ Item[itemID]

-Shipment
Attributes: (ItemId, Id, from_address, To_address, dateSent, dateReceived, status)
PrimaryKey: {itemID}
Not Null: itemID
Foreign Key: itemID ⊆ Item[itemID], Id ⊆ Courier[Id], from_address ⊆ Customer[from_address], To_address ⊆ Customer[To_address]