**Exercise 1. Attribute Closure**

Relation (A,B,C,D,E,G) has following functional dependencies:

AB → C  
C → A  
BC → D  
ACD → B  
D → EG  
BE → C  
CG → BD  
CE → AG

Task: Build Attribute Closure (BD)+

**Решение:**

1. (BD)+ сначала = {B, D}
2. D → EG, D содержится в {B, D} => добавляем EG => (BD)+ = {B, D, E, G}
3. BE → C, BE содержится в {B, D, E, G} => добавляем C => (BD)+ = {B, C, D, E, G}
4. C → A, C содержится в {B, C, D, E, G} => добавляем A => (BD)+ = {A, B, C, D, E, G}
5. Мы получили всю нашу схему, больше в closure добавить ничего нельзя, значит финально (BD)+ = {A, B, C, D, E, G}

Кроме того, видно, что, если из BD убрать либо B, либо D, этот набор мы уже не получим, значит BD еще и является candidate key для этой схемы.

Ответ: (BD)+ = {A, B, C, D, E, G}

**Exercise 2. Functional dependencies**

Look at relation Order (ProductNo, ProductName, CustomerNo, CustomerName, OrderDate, UnitPrice, Quantity, SubTotal, Tax, Total)

Tax rate depends on the Product (e.g., 20% for books or 30% for luxury items).

Only one order per product and customer is allowed per day (several orders are combined).

1. A) Determine the non-trivial functional dependencies in the relation
2. B) What are the key candidates?

**Решение:**

1. Из условий можно установить зависимости

ProductNo-> Tax

ProductNo, CustomerNo, OrderDate -> UnitPrice, Quantity, SubTotal, Tax, Total

Можно также предположить зависимости

ProductNo -> ProductName

CustomerNo -> CustomerName

UnitPrice, Quantity -> SubTotal

SubTotal, Tax -> Total

И остальные, которые можно получить из них

1. Сочетание ProductNo, CustomerNo, OrderDate у нас есть только в левой части выражения, мы не можем установить их из других параметров, если будем рассматривать наборы без них, поэтому как ключи нас интересуют этот набор и его суперсеты. Проверим closure для этого набора. По второй связи у нас сразу же собираются все столбцы. При этом набор минимальный, значит он на самом деле и есть ключ.

Ответ: ProductNo, CustomerNo, OrderDate

**Exercise 3. 3NF**

Consider relation R(A,B,C,D) with the following functional dependencies:  
F = {A**→**D, AB**→**C, AC**→**B}

1. A) What are all candidate keys?
2. B) Convert R into 3NF using synthesis algorithm from textbook.

**Решение:**

1. Определим, какие элементы не стоят ни в правой, ни в левой части отношений, какие стоят только в левой и только в правой:
   1. Элементов, которые не стоят нигде, нет
   2. Элементы, которые стоят только слева: А
   3. Элементы, которые стоят только справа: D
   4. Элементы, которые стоят и справа, и слева: B, C

Теперь сделаем closure для элементов из a и b, так как это только А, получается A+ = {A, D}, этого не хватает, поэтому добавляем к А элементы из d и строим для них closure:

(AB)+ = {A, B} + (A**→**D => D) + (AB**→**C => C) = {A, B, C, D}, мы собрали все элементы, значит AB это candidate key

Также можно проверить для AC:

(AC)+ = {A, C} + (A**→**D => D) + (AC**→**B => B) = {A, B, C, D}, мы тоже собрали все элементы, значит это тоже candidate key

Ответ: AB, AC

1. Чтобы привести R к третьей нормальной форме
   1. вычислим минимальный базис, так как мы не можем исключить ни одно отношение без потерь, он равен A**→**D, AB**→**C, AC**→**B
   2. теперь создадим объединения для элементов в базисе: A**→**D => AD, AB**→**C => ABC, AC**→**B => ACB(то же самое, что и предыдущее)
   3. добавим ключи: AB, AC
   4. теперь уберем лишние комбинации: ключи содержатся в ABC, поэтому они исключаются, остаются AD и ABC с отношениями соответственно {A->D} и {(AB -> C), (AC -> B)}

Ответ: AD, ABC

**Exercise 4. Complex example**

Consider the relation **Items(Vendor, Brand, Kind, Weight, Store)** that represent a store stocks.

Convert sentences A)-C) from English text into a functional or a multi-valued dependencies.

* 1. A Vendor holds the trademark for a brand (limited to item of a particular kind), so two different Vendors can't use the same brand name for items of the same kind.

Brand, Kind -> Vendor

* 1. For each item kind, each store sells only single brand name made by each Vendor.

Kind, Store, Vendor -> Brand

* 1. If a particular item (vendor, brand name, and kind) is available in a particular weight at a store, then that weight is available at all stores carrying that item.

Vendor, Brand, Kind, Weight ↠ Store

* 1. Now assume that all the functional and/or multi-valued dependencies you specified does hold in Items, and no other dependencies hold in Items. What are the keys for Items?

Из-за множественных связей с весом и магазинами мы не можем взять Brand, Kind как ключ, Weight и Store придется включать в ключ, так как они ничем не ограничены, тогда для вывода остальных столбцов можно взять либо Brand, Kind, либо Kind, Vendor.

Ответ: возможные ключи Brand, Kind, Weight, Store или Vendor, Kind, Weight, Store.

* 1. What normal forms does Items satisfy?

Схема удовлетворяет требованиям первой нормальной формы, так как содержит в ячейках только скалярные значения и не содержит повторов.

Схема не удовлетворяет второй нормальной форме, если брать любой из ключей из пункта D: если взять Brand, Kind, Weight, Store как ключ, то Vendor будет зависеть от подмножества ключа Brand, Kind; если взять ключ Vendor, Kind, Weight, Store, то Brand будет зависеть от Vendor, Kind, Store, тоже подмножества. Таким образом минимальность зависимости от ключа нарушена.

Следовательно, нарушены остальные формы после второй.

F) Show that this decomposition is not lossless:

Items1(Vendor, Brand, Kind, Store)

Items2(Vendor, Brand, Kind, Weight)

Распишем лемму для декомпозиции с многозначными связями:

* Items1 ∩ Items2 ↠ Items1

Или

* Items1 ∩ Items2 ↠ Items2

Items1 ∩ Items2 = Vendor, Brand, Kind

Items1 ∩ Items2 = Vendor, Brand, Kind ! ↠ Vendor, Brand, Kind, Weight Items1 ∩ Items2 = Vendor, Brand, Kind ! ↠ Vendor, Brand, Kind, Store

Следовательно, декомпозиция с потерями.

G) Modify one of the attributes in either Items1 or in Items2 to obtain a lossless decomposition. Prove that property of your decomposition.

Можно в Items1 заменить Vendor на Weight, тогда пересечение будет Brand, Kind, Weight,

Brand, Kind -> Vendor, поэтому пересечение можно представить как

Vendor, Brand, Kind, Weight, и далее

Vendor, Brand, Kind, Weight ↠ (Brand, Kind, Weight,) Store = Items1

Лемма выполняется, значит потерь нет.

H) What normal forms does the decomposition in F) satisfy?

По логике пункта D можно выбрать ключи для двух таблиц: в первой Vendor, Kind, Store, во второй Brand, Kind, Weight. Тогда вторая таблица снова не соответствует второй нормальной форме, так как Vendor зависит от части ключа. Первая же таблица соответствует второй, третьей нормальной форме, BCNF и четвертой нормальной форме, так как единственное отношение в ней (Kind, Store, Vendor -> Brand) содержит слева ключ, является нетранзитивным (потому что оно одно) и не делится.

* 1. Decompose Items into a set of relations that are in BCNF such that the decomposition is lossless. Is this decomposition dependency-preserving?

Определим ключ и зависимости, нарушающие BCNF:

Пусть ключ равен Brand, Kind, Weight, Store, тогда нарушающее отношение – Brand, Kind -> Vendor

Применим алгоритм декомпозиции, получим i1 = [Vendor, Brand, Kind] и i2 = [Brand, Kind, Weight, Store].

Проверим то, что нет потерь по лемме

I1 ∩ I2 = Brand, Kind

Brand, Kind -> Vendor,

I1 ∩ I2 = Vendor, Brand, Kind ↠ i1 (trivial MVD), декомпозиция без потерь.

Однако, кажется, здесь не обязательно сохраняются зависимости, в частности может быть нарушено правило

Vendor, Kind, Store -> Brand

так как в первую таблицу можно внести несколько брендов в одной категории от одного вендора, и во второй, поскольку вендор там не присутствует, могут появиться записи про оба бренда, соответствующие одному и тому же магазину. Значит зависимость не сохраняется.

J) Find a lossless dependency-preserving decomposition of Items into 3NF using 3NF synthesis algorithm.

* + 1. Согласно алгоритму синтеза рассчитаем базис:

Brand, Kind -> Vendor

Vendor, Kind, Store -> Brand

Теперь сделаем их объединения:

R1 = {Brand, Kind, Vendor} и

R2 = {Vendor, Brand, Kind, Store}

с зависимостями {Brand, Kind -> Vendor} и {Brand, Kind -> Vendor; Vendor, Kind, Store -> Brand} соответственно.

* + 1. Теперь выделим ключи. Как было обозначено ранее, варианты ключей это Vendor, Kind, Weight, Store или Brand, Kind, Weight, Store.

Ключи не входят в сформированные в прошлом пункте схемы, поэтому добавляем еще одну для ключа, например

R3 = {Vendor, Kind, Weight, Store}.

* + 1. R1 входит в R2, она избыточна, поэтому исключаем ее, остаются схемы R2 и R3.

Изначальная схема декомпозируется в схемы {Vendor, Brand, Kind, Store} и {Vendor, Kind, Weight, Store} с помощью алгоритма 3NF синтеза, по свойствам алгоритма декомпозиция с сохранением связей и без потерь.

Ответ: {Vendor, Brand, Kind, Store} и {Vendor, Kind, Weight, Store}