**Задание 1.**

Даны отношения А и В. Получить результирующее отношение, выполнив операцию «Θ-соединение»

A|><| (I=P) B. Записать операцию в виде формулы реляционного исчисления с переменными-кортежами. Записать операцию в виде запроса на языке SQL.

Таблица A

I J K L

a L m K

c D e F

b A d N

i K o p

Таблица B

M N O P

w z x a

i k o p

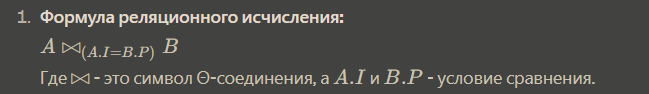
n j v c

d f e h

a l m k

**Решение**

Операция Θ-соединение (Theta Join) объединяет два отношения по определенному условию, которое задается оператором сравнения (например, =, <>, <, >, <=, >=). В данном случае условие задано как I=P.



SELECT \*

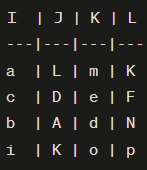
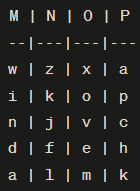
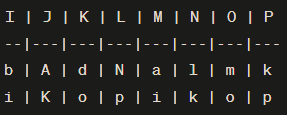
FROM A

JOIN B ON A.I = B.P;

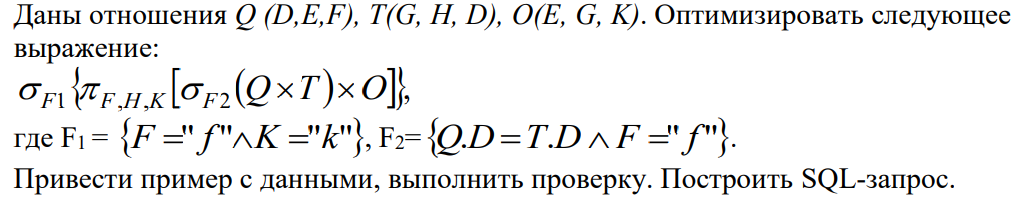
Этот SQL запрос выбирает все столбцы из таблицы A и B, объединяя строки по условию, что значение столбца I из таблицы A равно значению столбца P из таблицы B.

Применяя это к вашим таблицам A и B:

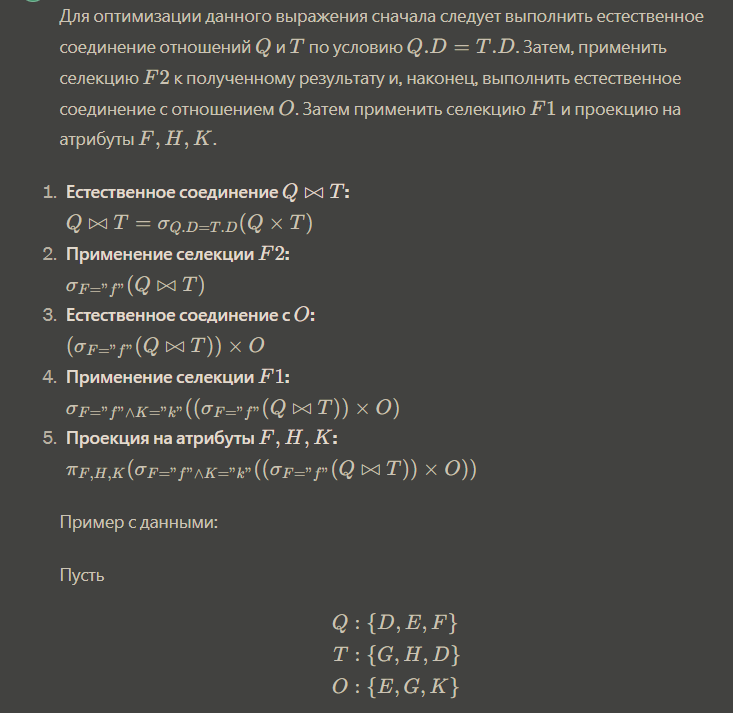
Таблица A: Таблица B: Результат Θ-соединения:

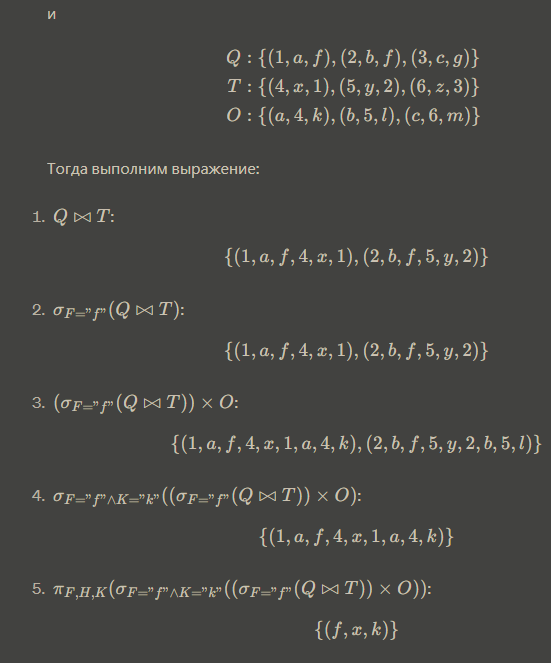
  

**Задание 2.**



**Решение**





SQL-запрос:

SELECT Q.F, T.H, O.K

FROM Q

JOIN T ON Q.D = T.D

JOIN O ON Q.E = O.E

WHERE Q.F = 'f' AND O.K = 'k';

**Задание 3**

В таблицу БД последовательно сохраняются записи со следующими уникальными значениями ключей поиска:

17, 3, 8, 19, 2, 4, 5, 9, 6, 13, 14, 20, 7, 11, 1, 40.

Организовать индекс с использованием В-дерева третьего порядка.

Показать по шагам процедуру поиска в базе данных записей по построенному

B-дереву со значением ключа 9 и значением ключа 10

**Решение**

[14]

/ \

[4, 8] [17]

/ | \ | \

[1, 2, 3] [5, 6, 7] [9, 11, 13] [19, 20, 40]

Для 9:

Начинаем с корня и идем вниз по дереву.

Сравниваем 9 с корневым узлом (14), идем влево, так как 9 меньше 14.

Сравниваем 9 с узлом (8), идем вправо, так как 9 больше 8.

Сравниваем 9 с узлом (9), находим значение.

Теперь поиск для 9 завершен. Для 10:

Начинаем с корня и идем вниз по дереву.

Сравниваем 10 с корневым узлом (14), идем влево, так как 10 меньше 14.

Сравниваем 10 с узлом (8), идем вправо, так как 10 больше 8.

Сравниваем 10 с узлом (9), не находим значение.

**Задание 4.**

Разработать диаграмму «сущность-связь» для сущностей «Поставщик», «Товар», «Поставка». Учесть, что конкретный товар может поставляться несколькими поставщиками, а каждый поставщик поставляет множество товаров. Каждая поставка может содержать несколько товаров. Описать сущности атрибутами, выделить идентифицирующие атрибуты. Определить тип(ы) связей. На основе диаграммы «сущность-связь» сформировать структуру реляционных отношений.

**Решение**

Диаграмма "сущность-связь" (ER-диаграмма) для сущностей "Поставщик", "Товар" и "Поставка" может выглядеть следующим образом:

Объяснение элементов диаграммы:

Сущность "Поставщик" (Supplier):

Атрибуты: SupplierID (идентификатор поставщика, первичный ключ), Name (название поставщика), ContactPerson (контактное лицо), Phone (телефон поставщика).

Сущность "Товар" (Product):

Атрибуты: ProductID (идентификатор товара, первичный ключ), ProductName (название товара), Category (категория товара), Price (цена товара).

Сущность "Поставка" (Supply):

Атрибуты: SupplyID (идентификатор поставки, первичный ключ), SupplierID (внешний ключ, связывающий сущность "Поставщик"), ProductID (внешний ключ, связывающий сущность "Товар"), Quantity (количество поставленного товара), DeliveryDate (дата поставки).

Типы связей:

Поставщик поставляет Товар:

Тип связи: Многие к одному (Many-to-One).

Сущность "Поставщик" связана с сущностью "Товар" по атрибуту SupplierID.

Товар может поставляться несколькими Поставщиками:

Тип связи: Многие ко многим (Many-to-Many).

Для реализации связи создана дополнительная таблица "Поставка" ("Supply"), которая связывает сущности "Поставщик" и "Товар". Поле SupplierID является внешним ключом, связывающим сущность "Поставщик", а поле ProductID является внешним ключом, связывающим сущность "Товар".

Структура реляционных отношений:

Supplier (Поставщик):

SupplierID (PK)

Name

ContactPerson

Phone

Product (Товар):

ProductID (PK)

ProductName

Category

Price

Supply (Поставка):

SupplyID (PK)

SupplierID (FK, ссылается на Supplier.SupplierID)

ProductID (FK, ссылается на Product.ProductID)

Quantity

DeliveryDate

Связи между таблицами:

Таблица Supply связана с таблицей Supplier по полю SupplierID.

Таблица Supply связана с таблицей Product по полю ProductID.

**Задание 5.**

Предметная область – сеть складов промышленных товаров. На каждом складе хранится множество товаров, при этом один и тот же товар может храниться на нескольких складах. Ведётся партионный учет товара, по дате получения и стоимости. Получить информацию о: а) товарах, поставляемых данным поставщиком; б) история об изменении цен на данный товар; в) суммарные объемы каждого товара, хранящиеся на каждом складе и всех складах. Выделить сущности, описать их атрибутами и построить диаграмму «сущность-связь». Построить реляционные отношения. Записать все запросы для получения заданной информации на языке SQL. Запрос а) записать в виде выражения реляционной алгебры.

**Решение**

Сущность "Товар" (Product):

Атрибуты: ProductID (идентификатор товара, первичный ключ), ProductName (название товара).

Сущность "Склад" (Warehouse):

Атрибуты: WarehouseID (идентификатор склада, первичный ключ), Location (местоположение склада).

Сущность "Поставщик" (Supplier):

Атрибуты: SupplierID (идентификатор поставщика, первичный ключ), SupplierName (название поставщика).

Сущность "Поставка" (Supply):

Атрибуты: SupplyID (идентификатор поставки, первичный ключ), ProductID (внешний ключ, связывающий сущность "Товар"), SupplierID (внешний ключ, связывающий сущность "Поставщик"), WarehouseID (внешний ключ, связывающий сущность "Склад"), DateReceived (дата получения), Cost (стоимость).

Типы связей:

Связь "Товар хранится на Складе" (Product is stored in Warehouse):

Тип связи: Многие ко многим (Many-to-Many).

Позволяет отразить, что один и тот же товар может храниться на нескольких складах, и на каждом складе может быть хранено множество товаров.

Связь "Поставщик поставляет Товар на Склад" (Supplier supplies Product to Warehouse):

Тип связи: Многие ко многим (Many-to-Many).

Позволяет отразить, что поставщик может поставлять разные товары на разные склады, и на каждом складе может быть несколько поставщиков.

Диаграмма "сущность-связь" может выглядеть следующим образом:

Структура реляционных отношений:

Product (Товар):

ProductID (PK)

ProductName

Warehouse (Склад):

WarehouseID (PK)

Location

Supplier (Поставщик):

SupplierID (PK)

SupplierName

Supply (Поставка):

SupplyID (PK)

ProductID (FK, ссылается на Product.ProductID)

SupplierID (FK, ссылается на Supplier.SupplierID)

WarehouseID (FK, ссылается на Warehouse.WarehouseID)

DateReceived

Cost

SQL-запросы:

a) Товары, поставляемые данным поставщиком:

SELECT Product.ProductName

FROM Product

JOIN Supply ON Product.ProductID = Supply.ProductID

JOIN Supplier ON Supply.SupplierID = Supplier.SupplierID

WHERE Supplier.SupplierName = 'Название\_поставщика';

б) История об изменении цен на данный товар:

SELECT Product.ProductName, Supply.DateReceived, Supply.Cost

FROM Product

JOIN Supply ON Product.ProductID = Supply.ProductID

WHERE Product.ProductName = 'Название\_товара';

в) Суммарные объемы каждого товара, хранящиеся на каждом складе и всех складах:

-- Суммарные объемы на каждом складе

SELECT Warehouse.Location, Product.ProductName, SUM(Supply.Cost) AS TotalCost

FROM Product

JOIN Supply ON Product.ProductID = Supply.ProductID

JOIN Warehouse ON Supply.WarehouseID = Warehouse.WarehouseID

GROUP BY Warehouse.Location, Product.ProductName;

-- Суммарные объемы на всех складах

SELECT Product.ProductName, SUM(Supply.Cost) AS TotalCost

FROM Product

JOIN Supply ON Product.ProductID = Supply.ProductID

GROUP BY Product.ProductName;