Ира <https://docs.google.com/document/d/1m1kSZckCkdzBroYgbjkX0KShICw0DU_qAEwsp9tIutQ/edit#>

Все <https://docs.google.com/document/d/1TWgGDnVutqhC9tBcDqaePj49vDydggiGMT1d6bcRUFk/edit#>

WHERE Description LIKE '%pasta%'

% - Знак процента представляет нулевой, один или несколько символов

\_ - Подчеркнутый символ представляет собой один символ

РА

8 операторов и 2 группы

1. Традиционные операции над множествами
   1. Объединение **UNION** (БК)
   2. Пересечение **INTERSECT** (БК)
   3. Разность **MINUS** (EXCEPT) (Б)

(Для первых 3 – принцип одинаковых атрибутов)

* 1. Декартово произведение **TIMES** (Б)

1. Специальные операции
   1. Проекция **PROJECT** (уменьшенный select \*) (У)
   2. Фильтрация **WHERE** (У)
   3. Соединение **JOIN** расширение вширь (по сути-декартово произведение с уменьшением) (inner, а outer-декартово произведение с условием) (БК)
   4. Деление **DIVIDE BY** (Б)

**RENAME** x **AS** y

Join строго по совпадающим

переименования таков: (терм RENAМE список- переименований ). Здесь каждый из элементов списка переименований представляет собой выражение имя\_атрибута AS имя\_атрибута.

Count()

=, <>, IS\_EMPTY (реляционное выражение )

S(Sno: integer, Sname: string, Status: integer, City: string)

P(Pno: integer, Pname: string, Color: string, Weight: real, City: string)

SP(Sno: integer, Pno: integer, Qty: integer)

Объединение 2 и выполнение условия хоть раз

Получить имена поставщиков, которые поставляют деталь под номером 2

( ( SP JOIN S ) WНEPE Рno = 2 ) [ Sname ]

Объединение 3 и выполнение условия хоть раз

Получить имена поставщиков, которые поставляют по крайней мере одну красную деталь

( ( ( Р WНERE Color = 'Красный' ) JOIN SP) [ Sno ] JOIN S ) [ Sname ]

Другая формулировка того же запроса:

( ( ( Р WНERE Color = 'Красный' ) [ Рno ] JOIN SP ) JOIN S ) [ Sname ]

Деление

Получить имена поставщиков, которые поставляют все детали

( ( SP [ Sno, Рno] DIVIDE BY Р [ Рno ] JOIN S ) [ Sname ]

Условие Деление по другому участнику

Получить номера поставщиков, поставляющих по крайней мере все те детали, которые поставляет поставщик под номером 2

SP [ Sno, Рno ] DIVIDE ВY ( SP WНEPE Sno = 2 ) [ Рno ]

Двойное по 1 таблице

Получить все пары номеров поставщиков, размещенных в одном городе

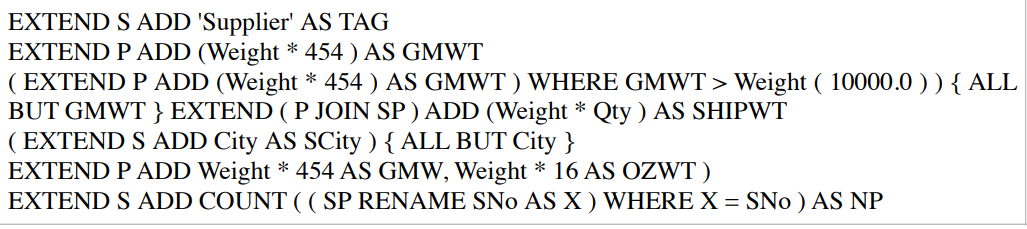
( ( ( S RENAМE Sno AS FirstSno ) [ FirstSno, City ] JOIN ( S RENAМE Sno AS SecondSno ) [ SecondSno , City ] ) WНEPE FirstSno < SecondSno ) [ FirstSno, SecondSno ]

Отрицание условия

Получить имена поставщиков, которые не поставляют деталь под номером 2

((S[ Sno ] MINUS (SP WНEPE Рno = 2 ) [ Sno ] ) JOIN S ) [Sname]

Операция расширения

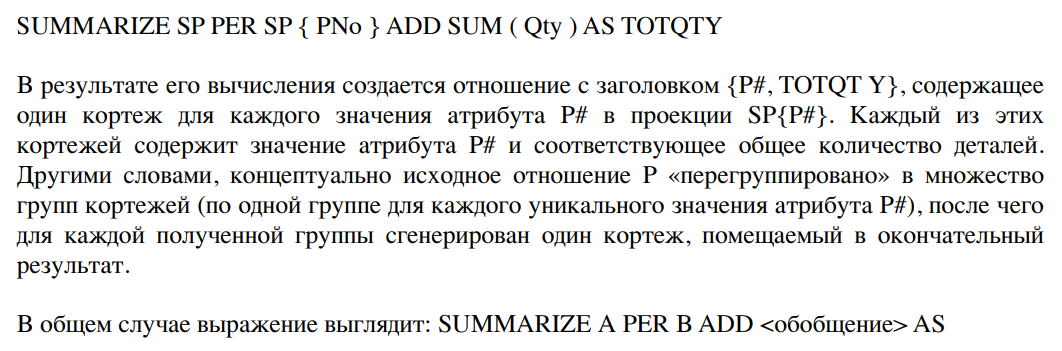


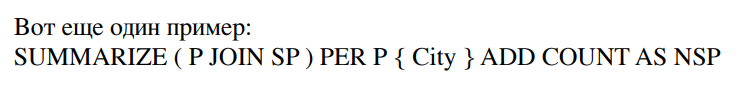
Обобщение

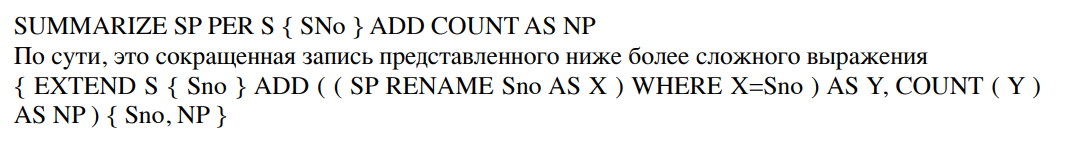


Если параметр имеет значение COUNT, то параметр недопустим и должен быть опущен. В остальных случаях параметр может быть опущен тогда и только тогда, когда параметр задает отношение со степенью, равной единице.

Summarize







1. Найти все тройки вида водительские права, тип нарушения, дата нарушения

((Drivers join DF)[FineID] join Fines)[DriverLicense, FineType, FineDate]

DriverID подразумевается

2.

- Найти все штрафы водителей, автомобили которых были зарегистрированы в 2020 году

((Cars where RegistrationDate >=’01.01.2020’ and RegistrationDate < ‘01.01.2021’) join DC)[DriverID] join Fines

Пример с лекции:

Получить имена поставщиков, которые не поставляют деталь под номером 2

((S[ Sno ] MINUS (SP WНEPE Рno = 2 ) [ Sno ] ) JOIN S ) [Sname]

Наш пример

Найти автомобили, водители которых не получили ни одного штрафа

((Drivers[DriverID] minus Fines[DriverID]) join DC)[CarId] join Cars

# ИК

1. Общее

RANGE OF переменная IS список-областей

переменная | переменная.атрибут [AS атрибут]

( РХ.Рno, PX.Weight AS GМWТ ) или ( SX, 'Поставщик' AS TAG )

(список-целевых-элементов)[WHERE wff]

wff ::= условие | NOT wff | условие AND wff | условие OR wff | IF условие THEN wff **| EXISTS** переменная (wff) | **FORALL** переменная (wff) | (wff)

DISTINCT для исключения дублирующих кортежей – не надо. Вычисление аргумента expression дает отношение, из которого повторяющиеся кортежи всегда исключаются по определению. Аргумент attribute обозначает атрибут такого отношения, по которому выполняются итоговые вычисления, а дублирующие значения перед подсчетом итога из такого атрибута не удаляются.

Осторожно с агрегатными

( РХ.Рno, SUМ ( SPX WНERE SPX.Pno = РХ.Рno, Qty ) AS TOТQТY )

1. Объявление

SY может принимать значения из множества кортежей S для поставщиков, которые или размещены в Смоленске, или поставляют деталь под номером 1, или и то и другое.

RANGE OF SX IS S

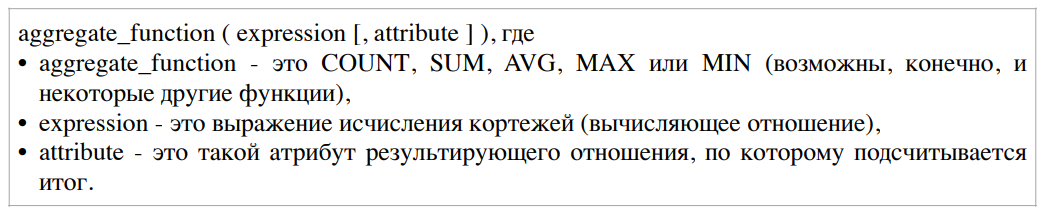
RANGE OF SPX IS SP

RANGE OF SY IS

(SX) WHERE SX.City = 'Смоленск'

(SX) WHERE EXISTS SPX (SPX.Sno = SX.Sno AND SPX.Pno = 1)

1. Агрегатные функции



Для функции СOUNT аргумент attribute не нужен и опускается; для других итоговых функций его можно опустить. Обратите внимание, что ссылка на итоговую функцию возвращает скалярное значение и поэтому допустима в качестве операнда скалярного выражения.

aggregate\_function ( ( tic ) WHERE f [, аttributе ] )

tic- целевой элемент списка (target\_item\_commalist), а f - это формула WFF,

1. **Запросы**
   1. **Общеее**

Просто выполнение условий - **WHERE**

Получить номера поставщиков из Смоленска со статусом больше 20

( SХ.Sno ) WHERE SX.City = 'Смоленск' AND SX.Status > 20

Дважды по одной таблице - **WHERE**

Получить все такие пары номеров поставщиков, что два поставщика размещаются в одном городе ( SX.Sno AS FirstSno, SY.Sno AS SecondSno ) WНERE SX.City = SY.City AND SX.Sno < SY.Sno

Спецификации «AS FirstSno» и «AS SecondSno» дают имена атрибутам результата; следовательно, такие имена недоступны для использования во фразе WНERE и потому второе сравнение во фразе WНERE «SX.Sno < SY.Sno», а не «FirstSno < SecondSno».

Объединение – **WHERE EXISTS A.s1=B.s1** - 2 и выполнение условия

Получить имена поставщиков, которые поставляют деталь c номером 2

SX.Sname WНERE EXISTS SPX ( SPX.Sno = SX.Sno AND SPX.Pno = 2 )

Объединение 3 - **WHERE EXISTS A.s1=B.s1 AND EXISTS -**  и выполнение условия

Получить имена поставщиков, которые поставляют по крайней мере одну красную деталь

SX.Sname WНERE EXISTS SPX (SX.Sno = SPX.Sno AND EXISTS РХ ( РХ.Рno = SPX.Рno AND PX.Color = 'Красный' ) )

Получить имена поставщиков, которые поставляют по крайней мере одну деталь, поставляемую поставщиком под номером 2 - значит надо **дважды по объединяющей**

SX.Sname WHERE EXISTS SPX ( EXISTS SPY (SX.Sno = SPX.Sno AND SPX.Pno = SPY.Pno AND SPY.sno = 2 ) )

Получить номера поставщиков, которые поставляют по крайней мере все детали, поставляемые поставщиком с номером 2

SX.Sno WНERE FORALL SPY ( SPY.Sno <> 2 OR EXISTS SPZ ( SPZ.Sno = SX.Sno AND SPZ.Pno = SPY.Pno ) )

Деление – значит **FORALL или отрицание усолвия**

Получить имена поставщиков, которые поставляют все детали

SX.SNAМE WНERE FORALL РХ ( EXISTS SPX ( SPX.Sno = SX.Sno AND SPX.Pno = РХ.Рno ))

SX. SNAМE WНERE NOT EXISTS РХ ( NOT EXISTS SPX ( SPX.Sno = SX.Sno AND SPX.Pno = РХ.Рno ) )

Отрицаниие условия – **NOT EXISTS**

Получить имена поставщиков, которые не поставляют деталь под номером 2

SX.Sname WНERE NOT EXISTS SPX ( SPX.Sno = SX.Sno AND SPX.Pno = 2 )

* 1. **Агрегатные**

Получить номера деталей и их вес всех типов деталей, вес которых превышает 10 ед.

( РХ.Рno, PX.Weight AS GМWТ ) WHERE РХ.Weight > 10

Добавление столбца

Получить всех поставщиков, добавив для каждого литеральное значение "Поставщик"

( SX, 'Поставщик' AS TAG )

Умножение столбцов между таблицами

Получить каждую поставку с полными данными о входящих в нее деталях и общим весом поставки

( SPX.Sno, SPX.Qty, РХ, PX.Weight \* SPX.Qty AS SНIPWT ) WНERE РХ.Рno = SPX.Pno

**Сумма по группе**

Для каждой детали получить ее номер и общее поставляемое количество

( РХ.Рno, SUМ ( SPX WНERE SPX.Pno = РХ.Рno, Qty ) AS TOТQТY )

Получить города, в которых хранится а) более пяти красных деталей;

PX.City WНERE COUNТ ( РY WНERE PY.City = PX.City AND РY.Сolor = 'Красный' ) > 5

Получить общее количество поставляемых деталей

( SUМ (SPX, Qty ) AS GRANDТOТAL )

Для каждого поставщика получить его номер и общее количество поставляемых им деталей

( SX.Sno, COUNТ ( SPX WНERE SPX.Sno = SX.Sno ) AS NUMBER\_OF\_PARTS )

* 1. **С РК**

1. Найти все тройки вида водительские права, тип нарушения, дата нарушения

(DX.driverlicense, FX.type, FX.date) where FX.driverID=DX.DriverID

Права, цвет авто, модель авто

(Dx.DriverLicense, Cx.color, Cx.model) where exists DCx(DCx.Did = Dx.Did and exists Cx(DCx.Cid = Cx.Cid))

2. Найти все штрафы водителей, автомобили которых были зарегистрированы в 2020 году

RANGE OF CX IS C where C.RD >= ‘2020-01-01’ and < ‘2021-01-01’

RANGE OF DX IS D

RANGE OF DCS IS DC

RANGE OF FX IS F

Fx.type where exists DCx (exists Cx (Fx.DID = DCx.DID and DC.CID = Cx.CID)

Fx.Fid where exists DX(Dx.Did = Fx.Did and exists DCx(Dcx.Did = Dx.Did and exists Cx(Cx.Cid = Dcx.Cid)))-без промежутка

3. Найти общую сумму выписанных штрафов в 2018 году

RANGE OF FX IS F where date like ‘%2018%’

Sum(FX where where date like ‘%2018%’, amount)

Найти водителей, получивших более 3х штрафов

Dx.FIO where count(Fx where Fx.Did = D.Did) > 3

# SQL

Select 5 обязательно

From 1 обычно необходимо

Where 2

Group by 3

Having 4

Order by 6 отступление от реляционной модели

1. Имена поставщиков, которые посставляют деталь номер 3

(первый вариант лучше)

Select S.Snam

From S join SP on

S.Sno=SP.Sno

Where SP.Pno=2

Или

Selet S.Sname

From S join (selet \*

From SP

Where Pno=2) t

On S.Sno=t.Sno

1. Имена посставщиков, поставляющих хотя бы 1 красную деталь

Select Sname

From S

where S.Sno in (select S.Sno

from SP inner join P

on P.Pno=SP.Pno

where P.color=’K’) t

или

Select S.Sname

From S join SP on S.Sno=SP.Sno

Join P on SP.Pno=P.Pno

Where P.Color=’K’

1. Пары поставщиков из одного города

Select fs.Sno

From S fs join S ss

On fs.City=ss.City

Where fs.Sno<ss.Sno

1. Имена поставщиков, не поставляющих деталь 2

Из s вычесть S c Pno=2

Select S.Sno, S.Sname

From S except

Select S.Sno, S.Sname

From S join SP on S.Sno=SP.Sno

Where SP.Pno=2

1. Все поставщики, поставляющие все детали

Select S.Sno, S.Sname

From SP join S on SP.Sno=S.Sno

Group by S.Sno, S.Sname

Having Count(distinct (Pno))=(select count(\*)

From P)

1. Поставщик, поставляющий больше всех деталей

Select S.Sno, S.Sname

From SP join S on SP.Sno=S.Sno

Groupby S.Sno, S.Sname

Having count(distinct(Pno))=(select max(cnt)

From (

Select Sno, count(distinct(Pno) as cnt

From SP

Group by Sno) t

)

Или используюя ОТВ-обобщенное табличное выражение

With groupSP(Sno, cnt) as

(select Snno, Count(distinct(Pno))

From SP

Group by Sno)

Select S.Snno, S.Sname

From groupSP join S on groupSP.Sno=S.Sno

Where cnt=(select max(cnt)

From groupSP)

# ФЗ

<https://github.com/Panda-Lewandowski/DataBase/wiki/Теория-проектирования-реляционных-баз-данных>

<https://github.com/Sunshine-ki/BMSTU5_DB/wiki>

Правила Армстронга

1. Правило рефлексивности: (B принадлежит A) следовательно (А -> В)
2. Правило дополнения: (A -> B) следовательно АС -> BC
3. Правило транзитивности: (А -> В) и (В -> С) следовательно (А -> C)
4. Правило самоопределения: А -> А
5. Правило декомпозиции: (A -> BC) следовательно (А -> В) и (А -> С)
6. Правило объединения: (А -> В) и (А -> С) следовательно (А -> BC)
7. Правило композиции: (А -> В) и (С -> D) следовательно (АС -> BD)
8. теоремой объединения: (А->В) и (С->D) следовательно (А(С–B)->BD).

Суперключ –покрывает все, но избыточен

Потенциальный- супер - избыточные

Первичный – потенциальный выбранный

Множество ФЗ является **неприводимым** тогда и только тогда, когда оно обладает всеми перечисленными ниже свойствами.

1. Каждая ФЗ этого множества имеет одноэлементную правую часть.

2. Ни одна ФЗ множества не может быть устранена без изменения замыкания этого множества.

3. Ни один атрибут не может быть устранен из левой части любой ФЗ данного множества без изменения замыкания множества.

1. Найти минимальное покрытие

Упражнение. Пусть дана переменная-отношение R с атрибутами A, B, С, D и следующими функциональными зависимостями. А -> BC В -> С А -> В АВ -> С АС -> D Найти неприводимое множество функциональных зависимостей эквивалентное данному множеству.

1. Прежде всего, следует переписать заданные ФЗ таким образом, чтобы каждая из них имела одноэлементную правую часть. A -> B A -> C В -> С A -> B АВ -> C АС -> D

Нетрудно заметить, что зависимость A -> В записана дважды, так что одну из них можно удалить.

2. Затем в левой части зависимости АС -> D может быть опущен атрибут С, поскольку дана зависимость А -> С, из которой по правилу дополнения можно получить зависимость А -> AС. Кроме того, дана зависимость АС -> D, из которой по правилу транзитивности можно получить зависимость A -> D. Таким образом, атрибут C в левой части исходной зависимости АС -> D является избыточным.

3. Далее можно заметить, что зависимость АВ -> С может быть исключена, поскольку дана зависимость А -> С, из которой по правилу дополнения можно получить зависимость АВ -> СВ, а затем по правилу декомпозиции - зависимость АВ -> С.

4. Наконец зависимость A -> С подразумевается зависимостями A -> В и B -> С, так что она также может быть отброшена. В результате получается неприводимое множество зависимостей.

A -> B В -> С A -> D

2. Дана переменная-отношение R(A, B, C, D, E), для которой выполняется множество функциональных зависимостей S={AB–>DE, C–>E, D–>C, E–>A}. В результате декомпозиции получена переменная-отношение R1(A, B, C). Какие функциональные зависимости из S будут выполняться для R1?

Необходимо вычислить замыкания всех подмножеств множества {A, B, C}, кроме пустого множества и ABC. Затем, не учитывая функциональные зависимости, которые являются тривиальными и те, которые имеют D или E в правой части, получим искомое множество.

S={AB–>DE, C–>E, D–>C, E–>A}

A+ = A

B+ = B

C+ = CEA {C->E, E->A}

AB+ = ABDEC {AB->DE, D->C}

AC+ = ACE {C->E}

BC+ = BCEAD {C->E, E->A, AB->DE}

Не учитываем D и E.

Искомое множество функциональных зависимостей: {C->A, AB->C, BC->A}

(замечание: BC->A можно не учитывать, так как эта функциональная зависимость логически следует из C->A)