МИНОБРНАУКИ
РОССИИ

Передовые
инженерные
школы

СИСТЕМЫ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ

Лекция 3

ПЛАН ЛЕКЦИИ

- Специальные реляционные операции
- Алгебра А Дейта и Дарвена
- Реляционное исчисление
- Функциональные зависимости
- Декомпозиция без потерь

СПЕЦИАЛЬНЫЕ РЕЛЯЦИОННЫЕ ОПЕРАЦИИ

СПЕЦИАЛЬНЫЕ РЕЛЯЦИОНЫЕ ОПЕРАЦИИ (1/4)

Операция ограничения WHERE требует наличия двух операндов: ограничиваемого отношения и простого условия ограничения. Простое условие ограничения может иметь:

- вид (*a comp-op b*), где *a* и *b* – имена атрибутов ограничиваемого отношения; атрибуты *a* и *b* должны быть определены на одном и том же домене, для значений базового типа данных которого поддерживается операция сравнения *comp-op*, или на базовых типах данных, над значениями которых можно выполнять эту операцию сравнения;
- или вид (*a comp-op* const), где *a* – имя атрибута ограничиваемого отношения, а const – литерально заданная константа; атрибут *a* должен быть определен на домене или базовом типе, для значений которого поддерживается операция сравнения *comp-op*.

Операцией сравнения *comp-op* могут быть «=», «≠», «>», «≥», «<», «≤».

Для **обозначения вызова операции ограничения** будем использовать конструкцию A WHERE *comp*, где A – ограничиваемое отношение, а *comp* – простое условие сравнения. Пусть *comp1* и *comp2* – два простых условия ограничения. Тогда по определению:

- A WHERE (*comp1* AND *comp2*) равносильно (A WHERE *comp1*) INTERSECT (A WHERE *comp2*);
- A WHERE (*comp1* OR *comp2*) равносильно (A WHERE *comp1*) UNION (A WHERE *comp2*);
- A WHERE NOT *comp1* равносильно A MINUS (A WHERE *comp1*).

СПЕЦИАЛЬНЫЕ РЕЛЯЦИОНЫЕ ОПЕРАЦИИ (2/4)

Операция взятия проекции также требует наличия двух операндов – проецируемого отношения A и подмножества множества имен атрибутов, входящих в заголовок отношения A.

Результатом проекции отношения A на множество атрибутов $\{a_1, a_2, ..., a_n\}$ (PROJECT A $\{a_1, a_2, ..., a_n\}$) является отношение с заголовком, определяемым множеством атрибутов $\{a_1, a_2, ..., a_n\}$, и с телом, состоящим из кортежей вида $<a_1:v_1, a_2:v_2, ..., a_n:v_n>$ таких, что в отношении A имеется кортеж, атрибут a_1 которого имеет значение v_1, атрибут a_2 имеет значение v_2, ..., атрибут a_n имеет значение v_n. Тем самым, при выполнении операции проекции выделяется «вертикальная» вырезка отношения-операнда с естественным уничтожением потенциально возникающих кортежей-дубликатов.

ПРИМЕРЫ

СЛУЖАЩИЕ_В_ПРОЕКТЕ_1

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	СЛУ_ОТД_НОМЕР
2934	Иванов	22000.00	310
2935	Петров	30000.00	310
2936	Сидоров	18000.00	313
2937	Федоров	20000.00	310
2938	Иванова	22000.00	315

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	СЛУ_ОТД_НОМЕР
2934	Иванов	22000.00	310
2935	Петров	30000.00	310
2938	Иванова	22000.00	315

СЛУЖАЩИЕ_В_ПРОЕКТЕ_1 **WHERE** (СЛУ_ЗАРП 200 > 20000.00 AND (СЛУ_ОТД_НОМ = 310 OR СЛУ_ОТД_НОМ = 315))

СЛУ_ОТД_НОМЕР
310
313
315

PROJECT
СЛУЖАЩИЕ_В_ПРОЕКТЕ_1
{СЛУ_ОТД_НОМ}

Уничтожение дублей требует дорогостоящей (по числу операций) внешней сортировки.

СПЕЦИАЛЬНЫЕ РЕЛЯЦИОНЫЕ ОПЕРАЦИИ (3/5)

Результатом ***операции соединения*** A **JOIN** B WHERE *comp* совместимых по взятию расширенного декартова произведения отношений A и B является отношение, получаемое путем выполнения операции ограничения по условию comp расширенного декартова произведения отношений A и B.

A JOIN B WHERE *comp* равносильно **(A TIMES B) WHERE *comp***

Операцию соединения с условием *comp* разумно считать операцией ***действительного*** соединения, если оно имеет вид (или может быть преобразовано к виду) *comp1* AND (*a comp-op b*), где *a* и *b* – имена атрибутов **разных отношений-операндов**.

СЛУЖАЩИЕ

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ
2934	Иванов	22400.00	1
2935	Петров	29600.00	1
2936	Сидоров	18000.00	1
2937	Федоров	20000.00	1
2938	Иванова	22000.00	1
2934	Иванов	22400.00	2
2935	Петров	29600.00	2
2939	Сидоренко	18000.00	2
2940	Федоренко	20000.00	2
2941	Иваненко	22000.00	2

ПРОЕКТЫ

ПРО_НОМ	ПРОЕКТ_РУК	ПРО_ЗАРП
1	Иванов	22400.00
2	Иваненко	22400.00

СЛУЖАЩИЕ JOIN ПРОЕКТЫ WHERE (СЛУ_ЗАРП > ПРО_ЗАРП)

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ	ПРО_НОМ1	ПРОЕКТ_РУК	ПРО_ЗАРП
2935	Петров	29600.00	1	1	Иванов	22400.00
2935	Петров	29600.00	2	2	Иваненко	22400.00

Действительное соединение реально уменьшает мощность промежуточного декартова произведения отношений-операндов в отличие от простого условия лишь на атрибут одного операнда.

СПЕЦИАЛЬНЫЕ РЕЛЯЦИОНЫЕ ОПЕРАЦИИ (4/5)

ЧАСТНЫЕ СЛУЧАИ СОЕДИНЕНИЯ

EQUIJOIN

Операция соединения называется *операцией эквисоединения*, если условие соединения имеет вид (*a* = *b*), где *a* и *b* – атрибуты разных операндов соединения. Этот случай важен потому, что он чаще всего встречается на практике, и для него существуют наиболее эффективные алгоритмы реализации.

NATURAL JOIN

Операция естественного соединения применяется к паре отношений A и B, обладающих (возможно, составным) общим атрибутом *c* (т. е. атрибутом с одним и тем же именем и определенным на одном и том же домене).
Пусть AB обозначает объединение заголовков отношений A и B. Тогда *естественное соединение* A и B – это спроецированный на AB результат эквисоединения A и B по условию A.c = B.c .

Результат операции **СЛУЖАЩИЕ** JOIN (**ПРОЕКТЫ** RENAME (ПРО_НОМ, ПРО_НОМ1)) WHERE (СЛУ_ЗАРП = ПРО_ЗАРП)

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ	ПРО_НОМ1	ПРОЕКТ_РУК	ПРО_ЗАРП
2934	Иванов	22400.00	1	1	Иванов	22400.00
2934	Иванов	22400.00	2	2	Иваненко	22400.00

Результат операции **СЛУЖАЩИЕ** NATURAL JOIN **ПРОЕКТЫ**

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ	ПРОЕКТ_РУК	ПРО_ЗАРП
2934	Иванов	22400.00	1	Иванов	22400.00
2935	Петров	29600.00	1	Иванов	22400.00
2936	Сидоров	18000.00	1	Иванов	22400.00
2937	Федоров	20000.00	1	Иванов	22400.00
2938	Иванова	22000.00	1	Иванов	22400.00
2934	Иванов	22400.00	2	Иваненко	22400.00
2935	Петров	29600.00	2	Иваненко	22400.00
2939	Сидоренко	18000.00	2	Иваненко	22400.00
2940	Федоренко	20000.00	2	Иваненко	22400.00
2941	Иваненко	22000.00	2	Иваненко	22400.00

Естественное соединение часто используется для восстановления после декомпозиции в 1NF сложных структур.

СПЕЦИАЛЬНЫЕ РЕЛЯЦИОНЫЕ ОПЕРАЦИИ (5/5)

ДЕЛЕНИЕ ОТНОШЕНИЙ

Пусть заданы два отношения – A с заголовком $\{a_1, a_2, ..., a_n, b_1, b_2, ..., b_m\}$ и B с заголовком $\{b_1, b_2, ..., b_m\}$. Будем считать, что атрибут b_i отношения A и атрибут b_i отношения B (i = 1, 2, …, m) не только обладают одним и тем же именем, но и определены на одном и том же домене. Назовем множество атрибутов $\{a_j\}$ составным атрибутом *a*, а множество атрибутов $\{b_j\}$ – составным атрибутом *b*. После этого будем говорить о реляционном делении «бинарного» отношения A{*a*, *b*} на унарное отношение B{b}.

Результатом **деления** A на B (A **DIVIDE BY** B) является «унарное» отношение C{*a*}, тело которого состоит из кортежей v таких, что в теле отношения A содержатся кортежи v UNION w такие, что множество {w} включает тело отношения B.

СЛУЖАЩИЕ

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ
2934	Иванов	22400.00	1
2935	Петров	29600.00	1
2936	Сидоров	18000.00	1
2937	Федоров	20000.00	1
2938	Иванова	22000.00	1
2934	Иванов	22400.00	2
2935	Петров	29600.00	2
2939	Сидоренко	18000.00	2
2940	Федоренко	20000.00	2
2941	Иваненко	22000.00	2

Отношение НОМЕРА_ПРОЕКТОВ

ПРО_НОМ
1
2

Результат операции СЛУЖАЩИЕ DIVIDE BY НОМЕРА_ПРОЕКТОВ

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП
2934	Иванов	22400.00
2935	Петров	29600.00

АЛГЕБРА А
ДЕЙТА И ДАРВЕНА

ВВЕДЕНИЕ

Сравнение алгебры Кодда и алгебры А Дейта и Дарвена

Алгебра Кодда

Опирается на теорию множеств.

Базовые операции:
- переименование атрибутов,
- объединение,
- пересечение,
- взятие разности,
- декартово произведение,
- проекция,
- ограничение

Соединение можно выразить через другие операции.
Проще определяются алгебраические черты SQL.

Алгебра А Дейта и Дарвена

Опирается на логические операции.

Базовые операции:
- реляционного отрицания (дополнения),
- реляционной конъюнкции (или дизъюнкции),
- проекции (удаления атрибута).

Через базовые операции выражаются:
- переименования атрибутов,
- соединения общего вида,
- взятия разности отношений.

Менее практичная. Более теоретизированная.

БАЗОВЫЕ ОПЕРАЦИИ АЛГЕБРЫ A (1/5)

Обозначения

r – отношение,
A – имя атрибута отношения *r*,
T – имя соответствующего типа (т. е. типа или домена атрибута A),
v – значение типа T.

Описание операции состоит из:

1. Формальная спецификация ограничений,
2. Формальная спецификация заголовка результата,
3. Пояснения.

- заголовком H_r отношения *r* называется множество атрибутов, т.е. упорядоченных пар вида <A, T>. По определению никакие два атрибута в этом множестве не могут содержать одно и то же имя атрибута A;
- кортеж t_r, соответствующий заголовку H_r, – это множество упорядоченных триплетов вида <A, T, v>, по одному такому триплету для каждого атрибута в H_r;
- тело B_r отношения *r* – это множество кортежей t_r. Заметим, что (в общем случае) могут существовать такие кортежи t_r, которые соответствуют H_r, но не входят в B_r.

Поскольку некоторые базовые операции Алгебры A имеют названия обычных логических операций, чтобы избежать путаницы, имена реляционных операций берутся в угловые скобки: <NOT>, <AND>, <OR> и т. д. В исходный базовый набор операций входят операции реляционного дополнения <NOT>, удаления атрибута <REMOVE>, переименования атрибута <RENAME>, реляционной конъюнкции <AND> и реляционной дизъюнкции <OR>.

БАЗОВЫЕ ОПЕРАЦИИ АЛГЕБРЫ A (2/5)

Операция реляционного дополнения

Пусть `s` обозначает результат операции `<NOT> r`. Тогда:

- `Hs = Hr` (заголовок результата совпадает с заголовком операнда);
- `Bs = {ts : exists tr (tr ∉ Br and ts = tr) }` (в тело результата входят все кортежи, соответствующие заголовку и не входящие в тело операнда).

Операция `<NOT>` производит дополнение `s` заданного отношения `r`. Заголовком `s` является заголовок `r`. Тело `s` включает все кортежи, соответствующие этому заголовку и не входящие в тело `r`.

Почему дополнение? Результат дополняет тело отношения до полного множества возможных кортежей, определенных на заголовке, определенном на доменах атрибутов.

r =

дни недели	чет-нечет
понедельник	1
вторник	0
четверг	0
суббота	1
вторник	1
четверг	1
пятница	1

<NOT>r =

дни недели	чет-нечет
среда	1
пятница	0
воскресенье	1
понедельник	0
среда	0
суббота	0
воскресенье	0

БАЗОВЫЕ ОПЕРАЦИИ АЛГЕБРЫ A (3/5)

Операция удаления атрибута

Пусть *s* обозначает результат операции *r* `<REMOVE> A`. Для обеспечения возможности выполнения операции требуется, чтобы существовал некоторый тип (или домен) `T` такой, что `<A, T> ∈ Hr` (т. е. в состав заголовка отношения *r* должен входить атрибут `A`). Тогда:

- `Hs = Hr minus {<A, T>}`, т. е. заголовок результата получается из заголовка операнда изъятием атрибута `A`;
- `Bs = {ts : exists tr exists v (tr ∈ Br and v ∈ T and <A,T,v> ∈ tr and ts = tr minus {<A,T,v>})}`, т. е. в тело результата входят все кортежи операнда, из которых удалено значение атрибута `A`.

Операция переименования атрибута

Пусть *s* обозначает результат операции *r* `<RENAME> (A, B)`. Для обеспечения возможности выполнения операции требуется, чтобы существовал некоторый тип `T`, такой, что `<A, T> ∈ Hr`, и чтобы не существовал такой тип `T`, что `<B, T> ∈ Hr`. (Другими словами, в схеме отношения *r* должен присутствовать атрибут `A` и не должен присутствовать атрибут `B`.) Тогда:

- `Hs = (Hr minus {<A, T>}) union {<B, T>}`, т. е. в схеме результата `B` заменяет `A`;
- `Bs = {ts : exists tr exists v (tr ∈ Br and v ∈ T and <A, T, v> ∈ tr and ts = (tr minus {<A, T, v>}) union {<B, T, v>})}`, т. е. в кортежах тела результата имя значений атрибута `A` меняется на `B`.

БАЗОВЫЕ ОПЕРАЦИИ АЛГЕБРЫ A (4/5)

Операция реляционной конъюнкции

Пусть s обозначает результат операции r_1 <AND> r_2. Для обеспечения возможности выполнения операции требуется, чтобы если <A, T1>$\in Hr_1$ и <A, T2>$\in Hr_2$, то T1=T2. (Другими словами, если в двух отношениях-операндах имеются одноименные атрибуты, то они должны быть определены на одном и том же типе (домене).) Тогда:

- $Hs = Hr_1$ union Hr_2, т. е. заголовок результата получается путем объединения заголовков отношений-операндов, как в операциях TIMES и JOIN
- Bs = { ts : exists tr_1 exists tr_2 (($tr_1 \in Br_1$ and $tr_2 \in Br_2$) and $ts = tr_1$ union tr_2)}; обратите внимание на то, что кортеж результата определяется как *объединение кортежей операндов*; поэтому:
    - если схемы отношений-операндов имеют непустое пересечение, то операция <AND> работает как естественное соединение;
    - если пересечение схем операндов пусто, то <AND> работает как расширенное декартово произведение;
    - если схемы отношений полностью совпадают, то результатом операции является пересечение двух отношений-операндов.

Операция <AND> является реляционной конъюнкцией, в некоторых случаях выдающей в результате отношение rs, ранее называвшееся естественным соединением двух заданных отношений r_1 и r_2. Заголовок rs является объединением заголовков r_1 и r_2.

БАЗОВЫЕ ОПЕРАЦИИ АЛГЕБРЫ A (5/5)

Операция реляционной дизъюнкции

Пусть s обозначает результат операции r_1 <OR> r_2. Для обеспечения возможности выполнения операции требуется, чтобы если <A, T1>∈Hr_1 и <A, T2>∈Hr_2, то должно быть T1 = T2 (одноименные атрибуты должны быть определены на одном и том же типе). Тогда:

- Hs = Hr_1 union Hr_2 (из схемы результата удаляются атрибуты-дубликаты);
- Bs = { ts : exists tr_1 exists tr_2 ((tr_1∈Br_1 or tr_2∈Br_2) and ts = tr_1 union tr_2)}; очевидно, что при этом:
  - если у операндов нет общих атрибутов, то в тело результирующего отношения входят все такие кортежи ts, которые являются объединением кортежей tr_1 и tr_2, соответствующих заголовкам отношений-операндов, и хотя бы один из этих кортежей принадлежит телу одного из операндов;
  - если у операндов имеются общие атрибуты, то в тело результирующего отношения входят все такие кортежи ts, которые являются объединением кортежей tr_1 и tr_2, соответствующих заголовкам отношений-операндов, если хотя бы один из этих кортежей принадлежит телу одного из операндов, и значения общих атрибутов tr_1 и tr_2 совпадают;
  - если же схемы отношений-операндов совпадают, то тело отношения-результата является объединением тел операндов.

Операция <OR> является реляционной дизъюнкцией и обобщением того, что ранее называлось объединением.

ПОЛНОТА АЛГЕБРЫ А

Алгебра А является полной, т. е. на основе введенных операций выражаются все операции алгебры Кодда.

Операция взятия разности. Если отношения r_1 и r_2 совместимы по объединению, то

r_1 MINUS r_2 = r_1 <AND> <NOT> r_2.

Операция ограничения (WHERE) выводится путем наложения ограничения на операнд, а потом применением операции <AND> к получившемуся отношению.

Чтобы получить результат соединения общего вида произвольных отношений A и B, нужно:

- выполнить над одним из отношений одну или несколько операций <RENAME>, чтобы избавиться от общих имен атрибутов;
- выполнить над полученными отношениями операцию <AND>, производящую расширенное декартово произведение;
- и для полученного отношения выполнить одну или несколько операций <AND> с отношениями-константами, чтобы должным образом ограничить его.

Операция реляционного деления. r_1 DIVIDE BY r_2 совпадает с результатом выражения (r_1 PROJECT A) MINUS (((r_2 TIMES (r_1 PROJECT A)) MINUS r_1) PROJECT A) в терминах операций реляционной алгебры Кодда или

(r_1 <REMOVE> B) <AND> <NOT> (((r_2 <AND> (r_1 <REMOVE> B)) <AND> <NOT> r_1) <REMOVE> B) в терминах операций Алгебры А.

ИЗБЫТОЧНОСТЬ АЛГЕБРЫ А (1/2)

Избыточность – это когда одни операции можно выразить через другие. То есть операций с избытком.

Булевские функции : отрицание, конъюнкция, дизъюнкция.

NOT	1	0
	0	1

AND	1	0
1	1	0
0	0	0

OR	1	0
1	1	1
0	1	0

Этот набор булевских функций избыточен. Существуют функции «стрелка Пирса» и «штрих Шеффера», через **каждую** из которых можно выразить эти три булевских.

```
sh (A, A) ≡ NOT A;
sh (NOT A, NOT B) ≡ A OR B
NOT sh (A, B) ≡ A AND B.
```

```
pi (A, A) ≡ NOT A;
pi (NOT A, NOT B) ≡ A AND B
NOT pi (A, B) ≡ A OR B.
```

Реляционные аналоги

```
<sh> (r1, r2) ≡ <NOT> r1 <OR> <NOT> r2
<pi> (r1, r2) ≡ <NOT> r1 <AND> <NOT> r2
```

Можно свести набор операций Алгебры А к трем операциям: <sh> (или <pi>), <RENAME> и <REMOVE>.

ИЗБЫТОЧНОСТЬ АЛГЕБРЫ А (2/2)

Операция <RENAME> тоже избыточна.

Пусть мы хотим в отношении r заменить имя атрибута А на А1.
Возьмем отношение r1 такое, которое состоит из кортежей {A1, T, v}, где T и v соответствуют кортежам {A, T, v} исходного отношения, отличаясь только названием атрибута.
Тогда операция r <RENAME> (A, A1) эквивалентна по результату выражению (r <AND> r1) <REMOVE> A.

!

Базисом Алгебры А являются операции реляционного отрицания (дополнения), реляционной конъюнкции (или дизъюнкции) и проекции (удаления атрибута). Реляционные аналоги логических операций определяются в терминах отношений на основе обычных теоретико-множественных операций и позволяют выражать напрямую операции пересечения, декартова произведения, естественного соединения и объединения отношений. Путем комбинирования базовых операций выражаются операции переименования атрибутов, соединения общего вида, взятия разности отношений. Алгебра А позволяет лучше осознать логические основы реляционной модели, хотя, безусловно, является в меньшей степени ориентированной на практическое применение, чем алгебра Кодда.

РЕЛЯЦИОННОЕ ИСЧИСЛЕНИЕ

ВВЕДЕНИЕ

РЕЛЯЦИОННАЯ АЛГЕБРА VS **РЕЛЯЦИОННОЕ ИСЧИСЛЕНИЕ**

РЕЛЯЦИОННАЯ АЛГЕБРА	РЕЛЯЦИОННОЕ ИСЧИСЛЕНИЕ
ЗАПРОС	ПРОГРАММА
↓	↓
ПОСЛЕДОВАЕЛЬНОСТЬ ДЕЙСТВИЙ	ПРОЧТЕНИЕ ФОРМУЛЫ
↑	↑
Процедурная формулировка	Декларативная формулировка

Предикат – это символ, который представляет свойство или отношение. Пример: P(a), P – предикат, применимый к индивидуальной константе a. Предикат первого порядка принимает в качестве переменной только одиночные переменные или константы.

Реляционное исчисление является прикладной ветвью формального механизма исчисления предикатов первого порядка. В основе исчисления лежит понятие переменной с определенной для нее областью допустимых значений и понятие правильно построенной формулы, опирающейся на переменные, предикаты и кванторы.

В качестве синтаксиса будем использовать нечто похожее на QUEL, используемый в СУБД Ingres.

ИСЧИСЛЕНИЕ КОРТЕЖЕЙ (1/3)

`RANGE СЛУЖАЩИЙ IS СЛУЖАЩИЕ` - Определение кортежной переменной. СЛУЖАЩИЙ.СЛУ_ИМЯ – значение атрибута.

Правильно построенная формула (Well-Formed Formula, WFF) служит для выражения условий, накладываемых на кортежные переменные.

Простые условия

Основой WFF являются простые условия, представляющие собой операции сравнения скалярных значений (значений атрибутов переменных или литерально заданных констант). Например, конструкции

`СЛУЖАЩИЙ.СЛУ_НОМ = 2934` и

`СЛУЖАЩИЙ.СЛУ_НОМ = ПРОЕКТ.ПРОЕКТ_РУК`

По определению, простое сравнение является WFF, а WFF, заключенная в круглые скобки, представляет собой простое сравнение.

Обозначим:
comp – простое сравнение
form - WFF

WFF:
NOT ***form***
comp AND ***form***
comp OR ***form***
IF ***comp*** THEN ***form***

WFF выдает либо *true* либо *false*. Результатом WFF будет отношение, для которого *true*.

ИСЧИСЛЕНИЕ КОРТЕЖЕЙ (2/3)

Кванторы EXISTS и FORALL

EXISTS var (form)
FORALL var (form)

(WFF)

var – какое-то отношение.

EXISTS равносильно «СУЩЕСТВУЕТ кортеж, для которого form принимает значение *true*».
FORALL равносильно «ЛЮБОЙ кортеж, для которого form принимает значение *true*».

Свободные и связанные переменные

По определению, все переменные, входящие в WFF, при построении которой не использовались кванторы, являются ***свободными***. Фактически, это означает, что если для какого-то набора значений свободных кортежных переменных при вычислении WFF получено значение *true*, то эти значения кортежных переменных могут входить в результирующее отношение.

Если же имя переменной использовано сразу после квантора при построении WFF вида EXISTS *var* (*form*) или FORALL *var* (*form*), то в этой WFF и во всех WFF, построенных с ее участием, var является ***связанной переменной***. Это означает, что такая переменная не видна за пределами минимальной WFF, связавшей эту переменную. При вычислении значения такой WFF используется не одно значение связанной переменной, а вся область ее определения.

Каждая новая связанная переменная – плюс один цикл при выполнении запроса.

ИСЧИСЛЕНИЕ КОРТЕЖЕЙ (3/3)

Целевые списки

Целевой список строится из целевых элементов, каждый из которых может иметь следующий вид:

- `var.attr`, где `var` – имя свободной переменной соответствующей WFF, а `attr` – имя атрибута отношения, на котором определена переменная `var`;
- `var`, что эквивалентно наличию подсписка `var.attr`$_1$`, var.attr`$_2$`, ..., var.attr`$_n$, где {`attr`$_1$`, attr`$_2$`, ..., attr`$_n$} включает имена всех атрибутов определяющего отношения;
- `new_name = var.attr`; `new_name` – новое имя соответствующего атрибута результирующего отношения.

Последний вариант нужен для решения конфликтов, когда требуется переименование.

Выражение реляционного исчисления кортежей имеет вид:

```
target_list WHERE WFF
```

ПРИМЕР СЛУЖАЩИЕ DIVIDE BY НОМЕРА_ПРОЕКТОВ

```
СЛУ1, СЛУ2 RANGE IS СЛУЖАЩИЕ
НОМЕР_ПРОЕКТА RANGE IS НОМЕРА_ПРОЕКТОВ
СЛУ1.СЛУ_НОМЕР, СЛУ1.СЛУ_ИМЯ, СЛУ1.СЛУ_ЗАРП
WHERE FORALL НОМЕР_ПРОЕКТА EXISTS СЛУ2
  (СЛУ1.СЛУ_НОМЕР = СЛУ2.СЛУ_НОМЕР AND
  СЛУ1.ПРО_НОМ = НОМЕРА_ПРОЕКТОВ.ПРО_НОМ)
```

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП
2934	Иванов	22400.00
2935	Петров	29600.00

ИСЧИСЛЕНИЕ ДОМЕНОВ

В исчислении доменов областью определения переменных являются домены.
Главное понятие исчисления доменов - условия членства.

Условия членства

Если R – это n-арное отношение с атрибутами a_1, a_2, ..., a_n, то условие членства имеет вид
R (a_{i1} : v_{i1}, a_{i2} : v_{i2}, ..., a_{im} : v_{im}) ($m \leq n$),
где v_{ij} – это либо литерально задаваемая константа, либо имя доменной переменной.
Условие членства принимает значение true в том и только в том случае, если в отношении R существует кортеж, содержащий указанные значения указанных атрибутов.
Если v_{ij} – константа, то на атрибут a_{ij} накладывается жесткое условие, не зависящее от текущих значений доменных переменных; если же v_{ij} – имя доменной переменной, то условие членства может принимать разные значения при разных значениях этой переменной.

СЛУЖАЩИЕ (СЛУ_НОМ:2934, СЛУ_ИМЯ:'Иванов', СЛУ_ЗАРП:22400.00, ПРО_НОМ:ПРО_НОМ) →

<2934, 'Иванов', 22400.00, 1>
<2934, 'Иванов', 22400.00, 2>

Во всех остальных отношениях формулы и выражения исчисления доменов выглядят похожими на формулы и выражения исчисления кортежей. В частности, формулы могут включать кванторы, и различаются свободные и связанные вхождения доменных переменных.

ФУНКЦИОНАЛЬНЫЕ ЗАВИСИМОСТИ

ОБЩИЕ ОПРЕДЕЛЕНИЯ

Понятие функциональной зависимости лежит в основе базовых свойств реляционных баз данных и используется при проектировании на основе нормализации.

R – отношение,
X, Y – подмножества заголовка R
(составные атрибуты тоже могут входить).

В значении переменной отношения **R** *атрибут* **Y** *функционально зависит от атрибута* **X** в том и только в том случае, если каждому значению **X** соответствует в точности одно значение **Y**.

В этом случае говорят также, что атрибут X *функционально определяет* атрибут Y (X является детерминантом (*определителем*) для Y, а Y является зависимым от X). Будем обозначать это как *R*.X → *R*.Y.

Функциональная зависимость (functional dependency FD) X → Y

Детерминант

Для примера будем использовать отношение СЛУЖАЩИЕ_ПРОЕКТЫ {СЛУ_НОМ, СЛУ_ИМЯ, СЛУ_ЗАРП, ПРО_НОМ, ПРОЕКТ_РУК}. Очевидно, что если СЛУ_НОМ является первичным ключом отношения СЛУЖАЩИЕ, то для этого отношения справедлива функциональная зависимость (Functional Dependency – FD) СЛУ_НОМ → СЛУ_ИМЯ.

ПРИМЕРЫ ФУНКЦИОНАЛЬНЫХ ЗАВИСИМОСТЕЙ

СЛУ_НОМ	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ	ПРОЕКТ_РУК
2934	Иванов	22400.00	1	Иванов
2935	Петров	29600.00	1	Иванов
2936	Сидоров	18000.00	1	Иванов
2937	Федоров	20000.00	1	Иванов
2938	Иванова	22000.00	1	Иванов
2939	Сидоренко	18400.00	2	Иваненко
2940	Федоренко	20400.00	2	Иваненко
2941	Иваненко	22600.00	2	Иваненко

СЛУ_НОМ→СЛУ_ИМЯ

СЛУ_НОМ→СЛУ_ЗАРП

СЛУ_НОМ→ПРО_НОМ

СЛУ_НОМ→ПРОЕКТ_РУК

{СЛУ_НОМ, СЛУ_ИМЯ}→СЛУ_ЗАРП

{СЛУ_НОМ, СЛУ_ИМЯ}→ПРО_НОМ

{СЛУ_НОМ, СЛУ_ИМЯ}→{СЛУ_ЗАРП, ПРО_НОМ}

…

ПРО_НОМ→ПРОЕКТ_РУК и т.д.

СЛУ_ИМЯ→СЛУ_НОМ

СЛУ_ИМЯ→СЛУ_ЗАРП

СЛУ_ИМЯ→ПРО_НОМ и т.д.

Поскольку имена всех служащих различны

Функциональных зависимостей может быть много, и они трактуются как ограничения целостности, поэтому нужно свести к минимуму множество зависимостей отношения, из которых следуют остальные.

СЛУ_ЗАРП→ПРО_НОМ

Дичь

FD A → B называется *тривиальной*, если B ⊆ A (т. е. множество атрибутов A включает множество B или совпадает с множеством B). Не являются ограничениями целостности.

ЗАМЫКАНИЕ МНОЖЕСТВА FD

Замыканием множества FD S является множество FD S^+, включающее все FD, логически выводимые из FD множества S.

СЛУ_НОМ → {СЛУ_ЗАРП, ОТД_НОМ}

СЛУ_НОМ → СЛУ_ЗАРП

СЛУ_НОМ → ОТД_НОМ

Это пример вывода из FD других FD

FD A → C называется ***транзитивной***, если существует такой атрибут B, что имеются функциональные зависимости A → B и B → C и отсутствует функциональная зависимость C → A.

В отношении СЛУЖАЩИЕ_ПРОЕКТЫ имеется также пара FD СЛУ_НОМ → ОТД_НОМ и ОТД_НОМ → ПРОЕКТ_РУК. Из них выводится FD СЛУ_НОМ → ПРОЕКТ_РУК. Заметим, что FD вида СЛУ_НОМ → ПРОЕКТ_РУК называются транзитивными, поскольку ПРОЕКТ_РУК зависит от СЛУ_НОМ «транзитивно», через ПРО_НОМ.

Подход к решению проблемы поиска замыкания S^+ множества FD S впервые предложил **Вильям Армстронг**.
Пусть A, B и C являются (в общем случае, составными) атрибутами отношения *R*. Множества A, B и C могут иметь непустое пересечение. Для краткости будем обозначать через AB A UNION B. Тогда:

Аксиомы Армстронга (не аксиомы, потому что выводятся).

1. если B⊆A, то A→B (рефлексивность);
2. если A→B, то AC→BC (пополнение);
3. если A→B и B→C, то A→C (транзитивность).

Истинность первой аксиомы Армстронга следует из того, что при B ⊆ A
FD A → B является тривиальной.

ВЫВОД II И III АКСИОМ АРМСТРОНГА

1. если $B \subseteq A$, то $A \to B$ (рефлексивность);
2. если $A \to B$, то $AC \to BC$ (пополнение);
3. если $A \to B$ и $B \to C$, то $A \to C$ (транзитивность).

Справедливость второй аксиомы докажем от противного. Предположим, что FD $AC \to BC$ не соблюдается. Это означает, что в некотором допустимом теле отношения найдутся два кортежа t_1 и t_2, такие, что $t_1\ \{AC\} = t_2\ \{AC\}$ (a), но $t_1\ \{BC\} \neq t_2\ \{BC\}$ (b) (здесь $t\ \{A\}$ обозначает проекцию кортежа t на множество атрибутов A). По аксиоме рефлексивности из равенства (a) следует, что $t_1\ \{A\} = t_2\ \{A\}$. Поскольку имеется FD $A \to B$, должно соблюдаться равенство $t_1\ \{B\} = t_2\ \{B\}$. Тогда из неравенства (b) следует, что $t_1\ \{C\} \neq t_2\ \{C\}$, что противоречит наличию тривиальной FD $AC \to C$. Следовательно, предположение об отсутствии FD $AC \to BC$ не является верным, и справедливость второй аксиомы доказана.

Аналогично докажем истинность третьей аксиомы Армстронга. Предположим, что FD $A \to C$ не соблюдается. Это означает, что в некотором допустимом теле отношения найдутся два кортежа t_1 и t_2, такие, что $t_1\ \{A\} = t_2\ \{A\}$, но $t_1\ \{C\} \neq t_2\ \{C\}$. Но из наличия FD $A \to B$ следует, что $t_1\ \{B\} = t_2\ \{B\}$, а потому из наличия FD $B \to C$ следует, что $t_1\ \{C\} = t_2\ \{C\}$. Следовательно, предположение об отсутствии FD $A \to C$ не является верным, и справедливость третьей аксиомы доказана.

5 ПРАВИЛ ДЕЙТА

Можно доказать, что система правил вывода Армстронга *полна* и *совершенна (sound and complete)* в том смысле, что для данного множества FD S любая FD, потенциально выводимая из S, может быть выведена на основе аксиом Армстронга, и применение этих аксиом не может привести к выводу лишней FD.

Дейт по практическим соображениям предложил расширение аксиом Армстронга 5ю правилами:

4. $A \to A$ (самодетерминированность) – прямо следует из правила (1);
5. если $A \to BC$, то $A \to B$ и $A \to C$ (декомпозиция) – из правила (1) следует, что $BC \to B$; по правилу (3) $A \to B$; аналогично, из $BC \to C$ и правила (3) следует $A \to C$;
6. если $A \to B$ и $A \to C$, то $A \to BC$ (объединение) – из правила (2) следует, что $A \to AB$ и $AB \to BC$; из правила (3) следует, что $A \to BC$;
7. если $A \to B$ и $C \to D$, то $AC \to BD$ (композиция) – из правила (2) следует, что $AC \to BC$ и $BC \to BD$; из правила (3) следует, что $AC \to BD$;
8. если $A \to BC$ и $B \to D$, то $A \to BCD$ (накопление) – из правила (2) следует, что $BC \to BCD$; из правила (3) следует, что $A \to BCD$.

ЗАМЫКАНИЕ АТРИБУТОВ НАД S (1/2)

Определение

Пусть заданы отношение R, множество Z атрибутов этого отношения (подмножество заголовка R, или составной атрибут R) и некоторое множество FD S, выполняемых для R. Тогда *замыканием Z над S* называется наибольшее множество Z^+ таких атрибутов Y отношения R, что FD $Z \rightarrow Y$ входит в S^+.

Алгоритм вычисления замыкания атрибутов над S

```
K := 0; Z[0] := Z;
DO
   K := K+1;
   Z[K] := Z[K-1];
   FOR EACH FD A→B IN S DO
      IF A⊆Z[K] THEN Z[K] := (Z[K] UNION B) END DO;
UNTIL Z[K] = Z[K-1];
Z+ := Z[K];
```

Докажем корректность алгоритма по индукции. На нулевом шаге $Z[0] = Z$, FD $Z \rightarrow Z[I]$, очевидно, принадлежит S^+ (тривиальная FD «выводится» из любого множества FD). Пусть для некоторого K выполняется FD $Z \rightarrow Z[K]$, и пусть мы нашли в S такую FD $A \rightarrow B$, что $A \subseteq Z[K]$. Тогда можно представить $Z[K]$ в виде AC, и, следовательно, выполняется FD $Z \rightarrow AC$. Но по правилу (8) мы имеем FD $Z \rightarrow ACB$, т.е. FD $Z \rightarrow (Z[K]\ \mathrm{UNION}\ B)$ входит во множество S^+, что переводит нас на следующий шаг индукции.

ЗАМЫКАНИЕ АТРИБУТОВ НАД S (2/2)

Пример построения замыкания атрибутов над S

Пусть для примера имеется отношение с заголовком {A, B, C, D, E, F} и заданным множеством FD S = {A⟶D, AB⟶E, BF⟶E, CD⟶F, E⟶C}. Пусть требуется найти $\{AE\}^+$ над S. На первом проходе тела цикла DO Z[1] равно AE. В теле цикла FOR EACH будут найдены FD A⟶D и E⟶C, и в конце цикла Z[1] станет равным ACDE. На втором проходе тела цикла DO при Z[2], равном ACDE, в теле цикла FOR EACH будет найдена FD CD⟶F, и в конце цикла Z[2] станет равным ACDEF. Следующий проход тела цикла DO не изменит Z[3], и Z^+ ($\{AE\}^+$) будет равно ACDEF.

Алгоритм построения замыкания множества атрибутов Z над заданным множеством FD S помогает легко установить, входит ли заданная FD Z → B в замыкание S^+. Очевидно, что необходимым и достаточным условием для этого является $B \subseteq Z^+$, т. е. вхождение составного атрибута B в замыкание Z

Суперключ

Суперключом отношения R называется любое подмножество K заголовка R, включающее, по меньшей мере, хотя бы один возможный ключ R.

Одно из следствий этого определения состоит в том, что подмножество K заголовка отношения R является суперключом тогда и только тогда, когда для любого атрибута A (возможно, составного) заголовка отношения R выполняется FD K⟶A. В терминах замыкания множества атрибутов K является суперключом тогда и только тогда, когда K^+ совпадает с заголовком R.

МИНИМАЛЬНОЕ ПОКРЫТИЕ МНОЖЕСТВА ФУНКЦИОНАЛЬНЫХ ЗАВИСИМОСТЕЙ

РПИ МФТИ

Определение

Множество FD S2 называется *покрытием множества FD* S1, если любая FD, выводимая из S1, выводится также из S2.

Легко заметить, что S2 является покрытием S1 тогда и только тогда, когда $S1^+ \subseteq S2^+$. Два множества FD S1 и S2 называются *эквивалентными*, если каждое из них является покрытием другого, т. е. $S1^+ = S2^+$.

Множество FD S называется минимальным в том и только в том случае, когда удовлетворяет следующим свойствам:

1. правая часть любой FD из S является множеством из одного атрибута (простым атрибутом);
2. детерминант каждой FD из S обладает свойством *минимальности*; это означает, что удаление любого атрибута из детерминанта приводит к изменению замыкания S^+, т. е. порождению множества FD, не эквивалентного S;
3. удаление любой FD из S приводит к изменению S^+, т. е. порождению множества FD, не эквивалентного S.

СЛУ_НОМ	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ	ПРОЕКТ_РУК
2934	Иванов	22400.00	1	Иванов
2935	Петров	29600.00	1	Иванов
2936	Сидоров	18000.00	1	Иванов
2937	Федоров	20000.00	1	Иванов
2938	Иванова	22000.00	1	Иванов
2939	Сидоренко	18400.00	2	Иваненко
2940	Федоренко	20400.00	2	Иваненко
2941	Иваненко	22600.00	2	Иваненко

Если считать, что единственным возможным ключом этого отношения является атрибут СЛУ_НОМ, то множество FD {СЛУ_НОМ → СЛУ_ИМЯ, СЛУ_НОМ → СЛУ_ЗАРП, СЛУ_НОМ → ПРО_НОМ, ПРО_НОМ → ПРОЕКТ_РУК} будет минимальным.

ЭКВИВАЛЕНТНОЕ МИНИМАЛЬНОЕ МНОЖЕСТВО FD

Для любого множества FD S существует (и даже может быть построено) эквивалентное ему минимальное множество S^-.

Приведем общую схему построения S^- по заданному множеству FD S. Во-первых, используя правило (5) (декомпозиции), мы можем привести множество S к эквивалентному множеству FD $S1$, правые части FD которого содержат только одноэлементные множества (простые атрибуты). Далее, для каждой FD из $S1$, детерминант D $\{D_1, D_2, …, D_n\}$ которой содержит более одного атрибута, будем пытаться удалять атрибуты D_i, получая множество FD $S2$. Если после удаления атрибута D_i $S2$ эквивалентно $S1$, то этот атрибут удаляется, и пробуется следующий атрибут. Назовем $S3$ множество FD, полученное путем допустимого удаления атрибутов из всех детерминантов FD множества $S1$. Наконец, для каждой FD f из множества $S3$ будем проверять эквивалентность множеств $S3$ и $S3$ MINUS $\{f\}$. Если эти множества эквивалентны, удалим f из множества $S3$, и в заключение получим множество $S4$, которое минимально и эквивалентно исходному множеству FD S.

Минимальным покрытием множества FD S называется любое минимальное множество FD S1, эквивалентное S.

Поскольку для каждого множества FD существует эквивалентное минимальное множество FD, у каждого множества FD имеется хотя бы одно минимальное покрытие, причем для его нахождения не обязательно находить замыкание исходного множества.

С целью минимизации ограничений целостности необходимо уметь выделять минимальный набор FD, для этого все вышеуказанное и затевалось.

ДЕКОМПОЗИЦИЯ БЕЗ ПОТЕРЬ

ДЕКОМПОЗИЦИЯ

Считаются правильными такие декомпозиции отношения, которые обратимы, т. е. имеется возможность собрать исходное отношение из декомпозированных отношений без потери информации. Такие декомпозиции называются ***декомпозициями без потерь***.

СЛУЖАЩИЕ_ПРОЕКТЫ

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ	ПРОЕКТ_РУК
2934	Иванов	22000.00	1	Иванов
2941	Иваненко	22000.00	2	Иваненко

Декомпозиция (1). Отношения СЛУЖ и СЛУ_ПРО

СЛУ_НОМ	СЛУ_ИМЯ	СЛУ_ЗАРП
2934	Иванов	22000.00
2941	Иваненко	22000.00

СЛУ_НОМ	ПРО_НОМ	ПРОЕКТ_РУК
2934	1	Иванов
2941	2	Иваненко

Ничего не потеряли, обратно все отлично собирается естественным соединением

Декомпозиция (2). Отношения СЛУЖ и ЗАРП_ПРО

СЛУ_НОМ	СЛУ_ИМЯ	СЛУ_ЗАРП
2934	Иванов	22000.00
2941	Иваненко	22000.00

СЛУ_ЗАРП	ПРО_НОМ	ПРОЕКТ_РУК
22000.00	1	Иванов
22000.00	2	Иваненко

Потеряли, кто в каком проекте работает (а не только руководит), в итоге при обратном процессе – лишние кортежи

СЛУ_НОМ	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ	ПРОЕКТ_РУК
2934	Иванов	22000.00	1	Иванов
2941	Иваненко	22000.00	2	Иваненко
2934	Иванов	22000.00	2	Иваненко
2941	Иваненко	22000.00	1	Иванов

ТЕОРЕМА ХИТА

Корректность декомпозиции часто следует из теоремы Хита.

Теорема Хита

Пусть задано отношение *r* {A, B, C} (A, B и C, в общем случае, являются составными атрибутами) и выполняется FD A → B. Тогда *r* = (*r* PROJECT {A, B}) NATURAL JOIN (*r* PROJECT {A, C}).

Доказательство. Прежде всего, докажем, что в теле результата естественного соединения (обозначим этот результат через r_1) содержатся все кортежи тела отношения r. Действительно, пусть кортеж {a, b, c} $\in$ r. Тогда по определению операции взятия проекции {a, b} $\in$ (r PROJECT {A, B}) и {a, c} $\in$ (r PROJECT {A, C}). Следовательно, {a, b, c} $\in$ r_1. Теперь докажем, что в теле результата естественного соединения нет лишних кортежей, т. е. что если кортеж {a, b, c} $\in$ $r1$, то {a, b, c} $\in$ r. Если {a, b, c} $\in$ r_1, то существуют {a, b} $\in$ (r PROJECT {A, B}) и {a, c} $\in$ (r PROJECT {A, C}). Последнее условие может выполняться в том и только в том случае, когда существует кортеж {a, b*, c} $\in$ r. Но поскольку выполняется FD A→B, то b = b* и, следовательно, {a, b, c} = {a, b*, c}. *Конец доказательства.*

ИЛЛЮСТРАЦИЯ ПРИМЕНЕНИЯ ТЕОРЕМЫ ХИТА

СЛУЖАЩИЕ_ОТДЕЛЫ_ПРОЕКТЫ

СЛУ_НОМ	СЛУ_ОТД	ПРО_НОМ
2934	630	1
2941	631	1
2934	630	2
2941	631	2

Декомпозиция.
Отношения СЛУЖ_ОТДЕЛЫ и СЛУЖ_ПРОЕКТЫ

СЛУ_НОМ	СЛУ_ОТД
2934	630
2941	631

СЛУ_НОМ	ПРО_НОМ
2934	1
2941	1
2934	2
2941	2

Каждый служащий работает только в одном отделе, т. е. имеется FD СЛУ_НОМ → СЛУ_ОТД, но один служащий может участвовать в нескольких проектах.
В отношении СЛУЖАЩИЕ_ОТДЕЛЫ_ПРОЕКТЫ атрибут СЛУ_НОМ не является возможным ключом, но наличия FD СЛУ_НОМ → СЛУ_ОТД оказывается достаточно для декомпозиции этого отношения без потерь.

Атрибут B минимально зависит от атрибута A, если выполняется минимальная слева FD A → B.

Пример

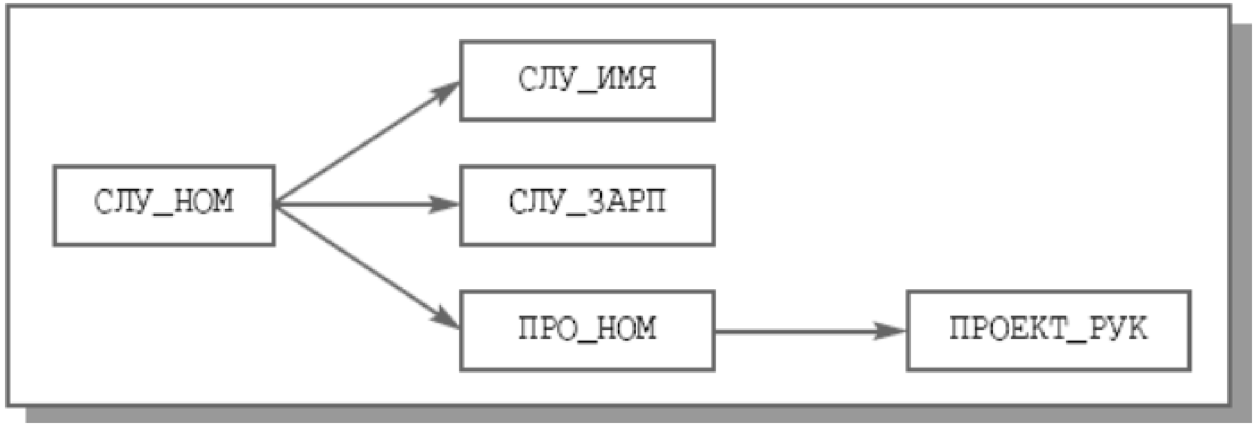
в отношении СЛУЖАЩИЕ_ПРОЕКТЫ выполняются FD СЛУ_НОМ → СЛУ_ЗАРП и {СЛУ_НОМ, СЛУ_ИМЯ} → СЛУ_ЗАРП. Первая FD является минимальной слева, а вторая – нет.

ДИАГРАММЫ ФУНКЦИОНАЛЬНЫХ ЗАВИСИМОСТЕЙ

Диаграмма минимального множества FD отношения СЛУЖАЩИЕ_ПРОЕКТЫ.

СЛУЖАЩИЕ_ПРОЕКТЫ

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ	ПРОЕКТ_РУК
2934	Иванов	22000.00	1	Иванов
2941	Иваненко	22000.00	2	Иваненко



В левой части диаграммы все стрелки начинаются с атрибута СЛУ_НОМ, который является единственным возможным (и, следовательно, первичным) ключом отношения СЛУЖАЩИЕ_ПРОЕКТЫ. Обратите внимание на отсутствие стрелки от СЛУ_НОМ к ПРОЕКТ_РУК. Конечно, поскольку СЛУ_НОМ является возможным ключом, должна выполняться и FD СЛУ_НОМ→ПРОЕКТ_РУК. Но эта FD является транзитивной (через ПРО_НОМ) и поэтому не входит в минимальное множество FD. Заметим, что в процессе нормализации, к рассмотрению которого мы приступим в следующей лекции, из диаграмм множества FD удаляются стрелки, начинающиеся не от возможных ключей.

ВСЁ!

СЕМИНАР

ПРИМЕР ОПЕРАЦИИ УДАЛЕНИЯ

<u>Результат операции </u>СЛУЖАЩИЕ <REMOVE> ПРО_НОМ

СЛУЖАЩИЕ

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ
2934	Иванов	22400.00	1
2935	Петров	29600.00	1
2936	Сидоров	18000.00	1
2937	Федоров	20000.00	1
2938	Иванова	22000.00	1
2934	Иванов	22400.00	2
2935	Петров	29600.00	2
2939	Сидоренко	18000.00	2
2940	Федоренко	20000.00	2
2941	Иваненко	22000.00	2

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП
2934	Иванов	22400.00
2935	Петров	29600.00
2936	Сидоров	18000.00
2937	Федоров	20000.00
2938	Иванова	22000.00
2939	Сидоренко	18000.00
2940	Федоренко	20000.00
2941	Иваненко	22000.00

ПРИМЕР ОПЕРАЦИИ КОНЪЮНКЦИИ (<AND>)

СЛУЖАЩИЕ_В_ПРОЕКТЕ_1

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	СЛУ_ОТД_НОМЕР
2934	Иванов	22000.00	310
2935	Петров	30000.00	310
2936	Сидоров	18000.00	313
2937	Федоров	20000.00	310
2938	Иванова	22000.00	315

СЛУЖАЩИЕ_В_ПРОЕКТЕ_2

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	СЛУ_ОТД_НОМЕР
2934	Иванов	22000.00	310
2935	Петров	30000.00	310
2939	Сидоренко	18000.00	313
2940	Федоренко	20000.00	310
2941	Иваненко	22000.00	315

ПРОЕКТЫ

ПРО_НОМ	ПРОЕКТ_РУК
1	Иванов
2	Иваненко

Результат операции СЛУЖАЩИЕ_В_ПРОЕКТЕ_1 <AND> ПРОЕКТЫ

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	СЛУ_ОТД_НОМЕР	ПРО_НОМ	ПРОЕКТ_РУК
2934	Иванов	22000.00	310	1	Иванов
2935	Петров	30000.00	310	1	Иванов
2936	Сидоров	18000.00	313	1	Иванов
2937	Федоров	20000.00	310	1	Иванов
2938	Иванова	22000.00	315	1	Иванов
2934	Иванов	22000.00	310	2	Иваненко
2935	Петров	30000.00	310	2	Иваненко
2936	Сидоров	18000.00	313	2	Иваненко
2937	Федоров	20000.00	310	2	Иваненко
2938	Иванова	22000.00	315	2	Иваненко

Результат операции СЛУЖАЩИЕ_В_ПРОЕКТЕ_1 <AND> СЛУЖАЩИЕ_В_ПРОЕКТЕ_2

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	СЛУ_ОТД_НОМЕР
2934	Иванов	22000.00	310
2935	Петров	30000.00	310

ПРИМЕР ОПЕРАЦИИ ДИЗЪЮНКЦИИ (<OR>)

Операция <OR> является реляционной дизъюнкцией и обобщением того, что ранее называлось объединением. Заголовок s есть объединение заголовков r_1 и r_2. Тело s состоит из всех кортежей, соответствующих заголовку s и являющихся надмножеством *либо* некоторого кортежа из тела r_1, *либо* некоторого кортежа из тела r_2.

Предположим, у нас имеются отношения ПРОЕКТЫ_1 {ПРОЕКТ_НАЗВ, ПРОЕКТ_РУК} и НОМЕРА_ПРОЕКТОВ {ПРО_НОМ} . Предположим также, что домен атрибута ПРОЕКТ_НАЗВ включает значения ПРОЕКТ_1, ПРОЕКТ_2, ПРОЕКТ_3, домен атрибута ПРОЕКТ_РУК ограничен значениями Иванов, Иваненко, а доменом атрибута ПРО_НОМ является множество {1, 2, 3}.

ПРОЕКТЫ_1

ПРОЕКТ_НАЗВ	ПРОЕКТ_РУК
ПРОЕКТ 1	Иванов
ПРОЕКТ 2	Иваненко

НОМЕРА_ПРОЕКТОВ

ПРО_НОМ
1
2

Результат операции ПРОЕКТЫ <OR> НОМЕРА_ПРОЕКТОВ

ПРОЕКТ_НАЗВ	ПРОЕКТ_РУК	ПРО_НОМ
ПРОЕКТ 1	Иванов	1
ПРОЕКТ 2	Иванов	1
ПРОЕКТ 3	Иванов	1
ПРОЕКТ 1	Иваненко	1
ПРОЕКТ 2	Иваненко	1
ПРОЕКТ 3	Иваненко	1
ПРОЕКТ 1	Иванов	2
ПРОЕКТ 2	Иванов	2
ПРОЕКТ 3	Иванов	2
ПРОЕКТ 1	Иваненко	2
ПРОЕКТ 2	Иваненко	2
ПРОЕКТ 3	Иваненко	2
ПРОЕКТ 1	Иванов	3
ПРОЕКТ 2	Иваненко	3

ИНТЕРПРЕТАЦИЯ ОГРАНИЧЕНИЯ WHERE (1/2)

Результат операции СЛУЖАЩИЕ_1 WHERE СЛУ_ЗАРП = 20000.00

СЛУЖАЩИЕ_1

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	РУК_НОМ
2934	Иванов	22000.00	2934
2935	Петров	30000.00	2934
2936	Сидоров	18000.00	2934
2937	Федоров	20000.00	2934
2938	Иванова	22000.00	2941
2939	Сидоренко	18000.00	2941
2940	Федоренко	20000.00	2941
2941	Иваненко	22000.00	2941

1 Создаем отношение

ЗАРП_20000

СЛУ_ЗАРП
20000.00

2

СЛУЖАЩИЕ_1 <AND> **ЗАРП_20000**

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	РУК_НОМ
2937	Федоров	20000.00	2934
2940	Федоренко	20000.00	2941

ИНТЕРПРЕТАЦИЯ ОГРАНИЧЕНИЯ WHERE (2/2)

<u>Результат операции</u> СЛУЖАЩИЕ_1 WHERE СЛУ_НОМЕР = РУК_НОМ

СЛУЖАЩИЕ_1

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	РУК_НОМ
2934	Иванов	22000.00	2934
2935	Петров	30000.00	2934
2936	Сидоров	18000.00	2934
2937	Федоров	20000.00	2934
2938	Иванова	22000.00	2941
2939	Сидоренко	18000.00	2941
2940	Федоренко	20000.00	2941
2941	Иваненко	22000.00	2941

(((**СЛУЖАЩИЕ_1** <REMOVE> СЛУ_НОМЕР) <REMOVE> СЛУ_ИМЯ) <REMOVE> СЛУ_ЗАРП) <RENAME> (РУК_НОМ, СЛУ_НОМЕР)

СЛУ_НОМЕР
2934
2941

СЛУЖАЩИЕ_1 <AND> ((((**СЛУЖАЩИЕ_1** <REMOVE> СЛУ_НОМЕР) <REMOVE> СЛУ_ИМЯ) <REMOVE> СЛУ_ЗАРП) <RENAME> (РУК_НОМ, СЛУ_НОМЕР))

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	РУК_НОМ
2934	Иванов	22000.00	2934
2941	Иваненко	22000.00	2941

ИНТЕРПРЕТАЦИЯ ДЕЛЕНИЯ DIVIDE BY (1/3)

Результат операции СЛУЖАЩИЕ **DIVIDE BY** НОМЕРА_ПРОЕКТОВ

СЛУЖАЩИЕ

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ
2934	Иванов	22400.00	1
2935	Петров	29600.00	1
2936	Сидоров	18000.00	1
2937	Федоров	20000.00	1
2938	Иванова	22000.00	1
2934	Иванов	22400.00	2
2935	Петров	29600.00	2
2939	Сидоренко	18000.00	2
2940	Федоренко	20000.00	2
2941	Иваненко	22000.00	2

НОМЕРА_ПРОЕКТОВ

ПРО_НОМ
1
2

1

СЛУЖАЩИЕ **PROJECT** (СЛУ_НОМЕР, СЛУ_ИМЯ, СЛУ_ЗАРП)

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП
2934	Иванов	22400.00
2935	Петров	29600.00
2936	Сидоров	18000.00
2937	Федоров	20000.00
2938	Иванова	22000.00
2939	Сидоренко	18000.00
2940	Федоренко	20000.00
2941	Иваненко	22000.00

ИНТЕРПРЕТАЦИЯ ДЕЛЕНИЯ DIVIDE BY (2/3)

СЛУЖАЩИЕ **PROJECT** (СЛУ_НОМЕР, СЛУ_ИМЯ, СЛУ_ЗАРП)

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП
2934	Иванов	22400.00
2935	Петров	29600.00
2936	Сидоров	18000.00
2937	Федоров	20000.00
2938	Иванова	22000.00
2939	Сидоренко	18000.00
2940	Федоренко	20000.00
2941	Иваненко	22000.00

2

(**СЛУЖАЩИЕ** PROJECT (СЛУ_НОМЕР, СЛУ_ИМЯ, СЛУ_ЗАРП))
TIMES **НОМЕРА_ПРОЕКТОВ**

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ
2934	Иванов	22400.00	1
2934	Иванов	22400.00	2
2935	Петров	29600.00	1
2935	Петров	29600.00	2
2936	Сидоров	18000.00	1
2936	Сидоров	18000.00	2
2937	Федоров	20000.00	1
2937	Федоров	20000.00	2
2938	Иванова	22000.00	1
2938	Иванова	22000.00	2
2939	Сидоренко	18000.00	1
2939	Сидоренко	18000.00	2
2940	Федоренко	20000.00	1
2940	Федоренко	20000.00	2
2941	Иваненко	22000.00	1
2941	Иваненко	22000.00	2

ИНТЕРПРЕТАЦИЯ ДЕЛЕНИЯ DIVIDE BY (3/3)

(**СЛУЖАЩИЕ** PROJECT (СЛУ_НОМЕР, СЛУ_ИМЯ, СЛУ_ЗАРП))
TIMES **НОМЕРА_ПРОЕКТОВ**

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ
2934	Иванов	22400.00	1
2934	Иванов	22400.00	2
2935	Петров	29600.00	1
2935	Петров	29600.00	2
2936	Сидоров	18000.00	1
2936	Сидоров	18000.00	2
2937	Федоров	20000.00	1
2937	Федоров	20000.00	2
2938	Иванова	22000.00	1
2938	Иванова	22000.00	2
2939	Сидоренко	18000.00	1
2939	Сидоренко	18000.00	2
2940	Федоренко	20000.00	1
2940	Федоренко	20000.00	2
2941	Иваненко	22000.00	1
2941	Иваненко	22000.00	2

3

((**СЛУЖАЩИЕ** PROJECT (СЛУ_НОМЕР, СЛУ_ИМЯ, СЛУ_ЗАРП))
TIMES **НОМЕРА_ПРОЕКТОВ**) MINUS **СЛУЖАЩИЕ**

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ
2936	Сидоров	18000.00	2
2937	Федоров	20000.00	2
2938	Иванова	22000.00	2
2939	Сидоренко	18000.00	1
2940	Федоренко	20000.00	1
2941	Иваненко	22000.00	1

4

(**СЛУЖАЩИЕ** PROJECT (СЛУ_НОМЕР, СЛУ_ИМЯ, СЛУ_ЗАРП))
MINUS ((((**СЛУЖАЩИЕ** PROJECT (СЛУ_НОМЕР, СЛУ_ИМЯ, СЛУ_ЗАРП)) TIMES
НОМЕРА_ПРОЕКТОВ) MINUS **СЛУЖАЩИЕ**)
PROJECT (СЛУ_НОМЕР, СЛУ_ИМЯ, СЛУ_ЗАРП))

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП
2934	Иванов	22400.00
2935	Петров	29600.00

ПРИМЕР WFF

СЛУЖАЩИЕ

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ
2934	Иванов	22400.00	1
2935	Петров	29600.00	1
2936	Сидоров	18000.00	1
2937	Федоров	20000.00	1
2938	Иванова	22000.00	1
2934	Иванов	22400.00	2
2935	Петров	29600.00	2
2939	Сидоренко	18000.00	2
2940	Федоренко	20000.00	2
2941	Иваненко	22000.00	2

ПРОЕКТЫ

ПРО_НОМ	ПРОЕКТ_РУК
1	Иванов
2	Иваненко

НОМЕРА_ПРОЕКТОВ

ПРО_НОМ
1
2

```
IF СЛУЖАЩИЙ.СЛУ_ИМЯ = 'Иванов'
THEN (СЛУЖАЩИЙ.СЛУ_ЗАРП >= 22400.00 AND СЛУЖАЩИЙ.ПРО_НОМ = 1)
```

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ
2934	Иванов	22400.00	1
2935	Петров	29600.00	1
2936	Сидоров	18000.00	1
2937	Федоров	20000.00	1
2938	Иванова	22000.00	1
2935	Петров	29600.00	2
2939	Сидоренко	18000.00	2
2940	Федоренко	20000.00	2
2941	Иваненко	22000.00	2

Формула неверна только для записи про Иванова на 2м проекте, остальные записи подходят.

ПРИМЕР ДЛЯ КВАНТОРОВ EXISTS И FORALL

Пусть СЛУ1 и СЛУ2 представляют собой две кортежные переменные, определенные на отношении СЛУЖАЩИЕ.

(a) Область истинности WFF

EXISTS СЛУ2 (СЛУ1.СЛУ_ЗАРП > СЛУ2.СЛУ_ЗАРП)

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ
2934	Иванов	22400.00	1
2935	Петров	29600.00	1
2937	Федоров	20000.00	1
2938	Иванова	22000.00	1
2934	Иванов	22400.00	2
2935	Петров	29600.00	2
2940	Федоренко	20000.00	2
2941	Иваненко	22000.00	2

(b) Область истинности WFF

FORALL СЛУ2 (СЛУ1.СЛУ_ЗАРП ≥ СЛУ2.СЛУ_ЗАРП)

СЛУ_НОМЕР	СЛУ_ИМЯ	СЛУ_ЗАРП	ПРО_НОМ
2935	Петров	29600.00	1
2935	Петров	29600.00	2

ПРИМЕР НЕМИНИМАЛЬНОГО ПОКРЫТИЯ FD

1. {СЛУ_НОМ→{СЛУ_ИМЯ, СЛУ_ЗАРП}, СЛУ_НОМ→ПРО_НОМ, СЛУ_НОМ→ПРОЕКТ_РУК, ПРО_НОМ→ПРОЕКТ_РУК},
2. {СЛУ_НОМ→СЛУ_ИМЯ, {СЛУ_НОМ, СЛУ_ИМЯ}→СЛУ_ЗАРП, СЛУ_НОМ→ПРО_НОМ, СЛУ_НОМ→ПРОЕКТ_РУК, ПРО_НОМ→ПРОЕКТ_РУК} и
3. {СЛУ_НОМ→СЛУ_НОМ, СЛУ_НОМ→СЛУ_ИМЯ, СЛУ_НОМ→СЛУ_ЗАРП, СЛУ_НОМ→ПРО_НОМ, СЛУ_НОМ→ПРОЕКТ_РУК, ПРО_НОМ→ПРОЕКТ_РУК}

не являются минимальными. Для множества (1) в правой части первой FD присутствует множество из двух элементов. Для множества (2) удаление атрибута СЛУ_ИМЯ из детерминанта второй FD не меняет замыкание множества FD. Для множества (3) удаление первой FD не приводит к изменению замыкания. Эти примеры показывают, что для определения минимальности множества FD не всегда требуется явное построение замыкания данного множества.