

**Резниченко Валерий Анатольевич**  
**Организация баз данных и знаний**

## **Лекция 7. Реляционное исчисление**

**Национальный авиационный университет**  
**Факультет компьютерных наук**  
**Кафедра инженерии программного**  
**обеспечения**

## СОДЕРЖАНИЕ

---

- Реляционные исчисления
- Исчисление Кодда
- Язык ALPHA
- Эквивалентность и полнота
- Примеры

## Лекция 7. Реляционное исчисление

# Теория множеств и логика

Теория множеств (отношений)	Исчисление предикатов	Связь
Множество $M$ $x$ – переменная	$P(x)$ ; $P$ – предикатный символ	$x \in M \Leftrightarrow P(x) = t$
$M \theta N$ $P(x) \theta Q(y)$ $\theta = \{ \cup, \cap, \square, - \}$	$x \in M \theta N \Leftrightarrow P(x) \theta Q(y) = t$ $\theta = \{ \forall, \wedge, \neg, \rightarrow \}$	

Теория множеств



Исчисление предикатов



Реляционная алгебра



Реляционное исчисление



Пример связи ТМ и ИП:

$$T(A) = R(A) \boxtimes S(A) = \{t \mid t \in R \vee t \in S\}$$

## Реляционное исчисление

---

- Подмножество формул исчисление предикатов
- Формальное описание того, ЧТО следует получить из базы данных. Например:  
ЕЯ - "Выдать факультеты с фондом финансирования > 10000"  
РИ -  $\{t \mid (FAC(t) \ \& \ t.fund > 10000)\}$
- Имеет средства языка запросов, но не обладает возможностями манипулирования данными (как и реляционная алгебра)

Два варианта реляционного исчисления:

- **Кортежное реляционное исчисление (TRC)** – переменные представляют строки отношений
- **Доменное реляционное исчисление (DRC)** - переменные представляют домены атрибутов отношений

## Кортежное реляционное исчисление (TRC)

---

**Запрос** ( в простейшем случае) имеет вид  $\{t \mid (F(t))\}$

- $t$  - кортежная переменная;
- $F(t)$  - формула, в которой присутствует кортежная переменная  $t$

**Ответ:** Множество всех таких кортежей  $t$ , для которых формула  $F(t)$  принимает значение true.

**Формула:** Рекурсивно определяется через атомарные формулы и построением более сложных формул с помощью логических связок и кванторов.

**SQL:** Является формальной основой языка SQL.

## TRC - Базовые составляющие языка

---

- **Константы:** 7, 'john', 3.14159 и т.д.
- **Кортежные переменные:**  $t_1, t_2, \dots$  – интерпретируются кортежами отношений.
- **Срезы кортежных переменных:**  $t.a, \dots$   
а - имя атрибута, значение которого входит в состав кортежа.
- **Предикатные символы:**  $P, P_1, \dots, Q, Q_1, \dots$   
Интерпретируются отношениями
- **Операторы (предикаты) сравнения :**  
 $= \{=, <, <=, >, >=, <>\}$  и т.д.
- **Логические связки:**  $\vee$  (или),  $\wedge$  (&) (и),  $\neg$  (не),  $\rightarrow$  (влечет)
- **Кванторы:**  $\exists$  (существует),  $\forall$  (для любого)

## TRC - Правильно определенные формулы

---

Атомарные формулы:

- $P(t)$  -  $P$  - предикатный символ,  
-  $t$  - кортежная переменная.  
Если  $P$  интерпретируется отношением  $R$ , то  
 $P(t)$  означает  $t \in R$
- $t.a \theta t.b$  -  $t.a$  и  $t.b$  - срезы
- $t.a \theta c$  -  $t.a$  - срез,  $c$  - константа

Правильно построенные формулы (ппф):

- Атомарные формулы являются ппф;
- Если  $F$  ппф, то ппф также являются  $\neg F$  и  $(F)$
- Если  $F$  и  $G$  ппф, то ппф также являются  $F \vee G$ ,  $F \wedge G$ ,  $F \rightarrow G$
- Если  $F$  ппф со свободной переменной  $t$ , то ппф также являются  $\exists tF(t)$  и  $\forall tF(t)$ .

## Лекция 7. Реляционное исчисление

# TRC - Свободные и связанные переменные. Запросы.

Говорят, что наличие кванторов  $\exists tF(t)$  и  $\forall tF(t)$  связывает переменную  $t$  в формуле  $F$ . Переменная, которая не связана, называется **свободной**.

**Запрос** - в общем случае это выражение вида:

$$\{(t_1, t_2, \dots) \mid (F(t_1, t_2, \dots))\}$$

где -  $t_1, t_2, \dots$  - кортежные переменные или их срезы;

-  $F(t_1, t_2, \dots)$  - формула, в которой присутствуют свободные кортежные переменные  $t_1, t_2, \dots$

**ВНИМАНИЕ:** Переменные  $t_1, t_2, \dots$ , расположенные слева от символа '|', должны быть единственными свободными переменными в формуле  $F$ . Это значит, что если  $F$  содержит другие переменные, то они должны быть связанными с помощью кванторов.



## Лекция 7. Реляционное исчисление

# Пример БД для запросов в РИ

---

<b>FAC</b>	<b>FACULTY (FNo, Name, Dean, Bld, Fund)</b>
<b>DEP</b>	<b>DEPARTMENT (DNo, FNo, Name, Head, Bld, Fund)</b>
<b>TCH</b>	<b>TEACHER(TNo, DNo, Name, Post, Tel, Salary, Comm)</b>
<b>GRP</b>	<b>GROUP(GNo, DNo, Course, Num, Quantity, CurNo)</b>
<b>SBJ</b>	<b>SUBJECT(SNo, Name)</b>
<b>ROM</b>	<b>ROOM (RNo, Num, Building, Seats)</b>
<b>LEC</b>	<b>LECTURE (TNo, GNo, SNo, RNo, Type, Day, Week)</b>

↑  
Предикатные  
символы

↑  
Отношения БД

## Лекция 7. Реляционное исчисление

# TRC - Примеры проекции, селекции и соединения

### 1) Проекция

**Запрос.** Вывести названия факультетов и имена их деканов

```
{(f.Name, f.Dean) | FAC(f)}
```

### 2) Селекция и проекция

**Запрос.** Вывести имена и должности преподавателей, имеющих зарплату > 1000 и надбавку > 500

```
{(t.Name, t.Post) | TCH(t) & t.Salary > 1000 & t.Comm > 500}
```

### 3) Соединение, селекция и проекция

**Запрос.** Вывести имена факультетов и их кафедр, расположенных (факультетов) в корпусе 5

```
{(f.Name, d.Name) | FAC(f) & DEP(d) & f.FNo = d.FNo & f.Bld = 5}
```

## Лекция 7. Реляционное исчисление

# TRC - Примеры кванторов существования

Запрос. Вывести имена факультетов корпуса 5 и их кафедр

```
{(f.Name,d.Name) | DEP(d) & FAC(f) & f.FNo = d.FNo & f.Bld = 5}
```

Запрос. Вывести имена кафедр факультетов корпуса 5

```
{d.Name | DEP(d) & ∃ f(FAC(f) & f.FNo = d.FNo & f.Bld = 5)}
```

Запрос. Вывести имена преподавателей-профессоров, имеющих лекции в группах первого курса

```
{t.Name | TCH(t) & t.Post='professor' &  
∃ l(LEC(l) & l.TNo = t.Tno &  
∃ g(GRP(g) & l.GNo = g.GNo & g.Course =
```

1)))  
Вывести имена профессоров,  
для которых существуют такие лекции,  
для которых (лекций) существуют группы первого курса

GRP      GROUP(GNo, Course...)

TCH      TEACHER (TNo, Name, Post...)

LEC      LECTURE(GNo, TNo...)

## Лекция 7. Реляционное исчисление

# TRC - Примеры кванторов общности

**Запрос.** Вывести номера преподавателей, читающих лекции во всех группах

$$\{I.TNo \mid LEC(I) \ \& \ \forall g(GRP(g) \rightarrow g.GNo = I.GNo)\}$$

**Запрос.** Вывести номера преподавателей, которые читают лекции во всех группах первого курса

$$\{I.TNo \mid LEC(I) \ \& \ \forall g((GRP(g) \ \& \ g.Course = 1) \rightarrow g.GNo = I.GNo)\}$$

**Запрос.** Вывести имена преподавателей, которые читают лекции во всех группах первого курса

$$\{t.Name \mid TCH(t) \ \& \ \exists I(L \& I.TNo = t.TNo \ \& \ \forall g((GRP(g) \ \& \ g.Course = 1) \rightarrow g.GNo = I.GNo))\}$$

Вывести имена преподавателей,

для которых существуют такие лекции,

которые читаются всем группам 1 курса

## TRC - Безопасные формулы (запросы)

Можно формулировать запросы, содержащие ппф, но не имеющие интерпретации в БД. Или, другими словами, имеющие бесконечное к-во ответов (дающие в результате бесконечные отношения). Примеры:

$$\begin{aligned} &\{t \mid t.\text{Post} = \text{'professor'}\} \\ &\{t \mid \neg \text{TCH}(t)\} \\ &\{(t,d) \mid \text{TCH}(t) \vee \text{DEP}(d)\} \end{aligned}$$

Запрос называется **безопасным**, если он интерпретируется конечным отношением.

Для построения безопасных запросов:

- все переменные в формуле должны быть ограниченными;
- все логические связки должны быть ограниченными;
- все кванторы должны быть ограниченными.

## TRC - Ограниченные переменные

Кортежная переменная **ограничена**, если она принадлежит предикату, который интерпретируется отношением БД, или если она ограничена другой переменной или константой.

Переменная  $t$  ограничена, если:

- она появляется в предикате  $P(t)$ , где  $P$  интерпретируется отношением БД;
- она появляется в формуле  $t.a_1 = c_1 \ \& \ \dots \ \& \ t.a_n = c_n$ , где  $a_1, \dots, a_n$  - все атрибуты кортежа  $t$ , а  $c_1, \dots, c_n$  - константы;
- она появляется в формуле  $t = s$ , где  $s$  – ограниченная переменная.

Примеры:

$P(t)$  - переменная  $t$  ограничена предикатом  $P$

$P(t) \ \& \ t = x$  - переменная  $x$  ограничена ограниченной переменной  $t$   
 $t.Name='john' \ \& \ t.Post='prof' \ \& \ t.Salary = '1200'$

## Ограниченные логические связки

Если две формулы  $F$  и  $G$  имеют ограниченные переменные, то:

- $F \vee G$  будет являться формулой с ограниченными переменными, если  $F$  и  $G$  имеют одинаковые свободные переменные;
- $F \& G$  всегда является формулой с ограниченной переменной независимо от того, совпадают ли в этих формулах переменные;
- $F \& \neg G$  является формулой с ограниченной переменной, если в этих формулах совпадают свободные переменные;
- $F \rightarrow G$  никогда не является формулой с ограниченной переменной.

**Примеры:**

- $P(t) \vee Q(t)$  – формула с ограниченной переменной
- $P(t) \vee Q(q)$  – переменные в результирующей формуле не ограничены
- $P(t) \& Q(q)$  – формула с ограниченной переменной
- $\neg Q(t)$  – не является формулой с ограниченной переменной
- $P(t) \& \neg Q(q)$  – не является формулой с ограниченной переменной
- $P(t) \& \neg Q(t)$  – формула с ограниченной переменной



## Ограниченные кванторы

Если  $F(t)$  - ограниченная формула а  $G(t)$  – произвольная формула, то формула  $\exists t(F(t) \& G(t))$  называется формулой с **ограниченным квантором существования**. Сама формула интерпретируется ограниченным отношением.

Если  $F(t)$  - ограниченная формула, а  $G(t)$  – произвольная формула, то формула  $\forall t(F(t) \rightarrow G(t))$  называется формулой с **ограниченным квантором общности**. Сама формула интерпретируется ограниченным отношением

### Примеры:

$\exists x(x.\text{Fund} < 5)$  – квантор существования не ограничен

$\exists x(P(x) \& x.\text{Fund} < 5)$  – квантор существования ограничен

$\forall x(P(x) \rightarrow Q(x, y))$  – квантор общности ограничен



## Лекция 7. Реляционное исчисление

# Доменное реляционное исчисление (DRC)

---

**Запрос** имеет вид  $\{x_1, x_2, \dots, x_n \mid (F(x_1, x_2, \dots, x_n))\}$

- $x_1, x_2, \dots, x_n$  - доменные переменные;
- $F(x_1, x_2, \dots, x_n)$  - формула, в которой единственными свободными переменными являются  $x_1, x_2, \dots, x_n$

**Ответ.** Множество всех таких кортежей  $\langle c_1, c_2, \dots, c_n \rangle$ , для которых формула  $F$  принимает значение true.

**Формула.** Рекурсивно определяется через атомарные формулы и построением более сложных формул с помощью логических связок и кванторов **точно так же, как и в TRC.**

**QBE.** Является формальной основой языка QBE.

## Лекция 7. Реляционное исчисление

# Примеры запросов в DRC

1) Проекция

**Запрос.** Вывести названия факультетов и имена их деканов

$$\{(y, z) \mid \exists x \exists u \exists v \text{FAC}(x, y, z, u, v)\}$$

2) Селекция и проекция

**Запрос.** Вывести названия факультетов корпуса 5 и их деканов

$$\{(y, z) \mid \exists x \exists u \exists v \text{FAC}(x, y, z, u, v) \ \& \ u = 5\}$$

3) Соединение, селекция и проекция

**Запрос.** Вывести имена факультетов корпуса 5 и имена их кафедр

$$\{(y, n) \mid \exists x \exists f (\exists z \exists u \exists v (\text{FAC}(x, y, z, u, v) \ \& \ u=5) \ \& \ \exists d \exists h \exists b \exists u \text{DEP}(d, f, n, h, b, u) \ \& \ x=f)\}$$

**FAC (FNo, Name, Dean, Bld, Fund)**

x y z u v

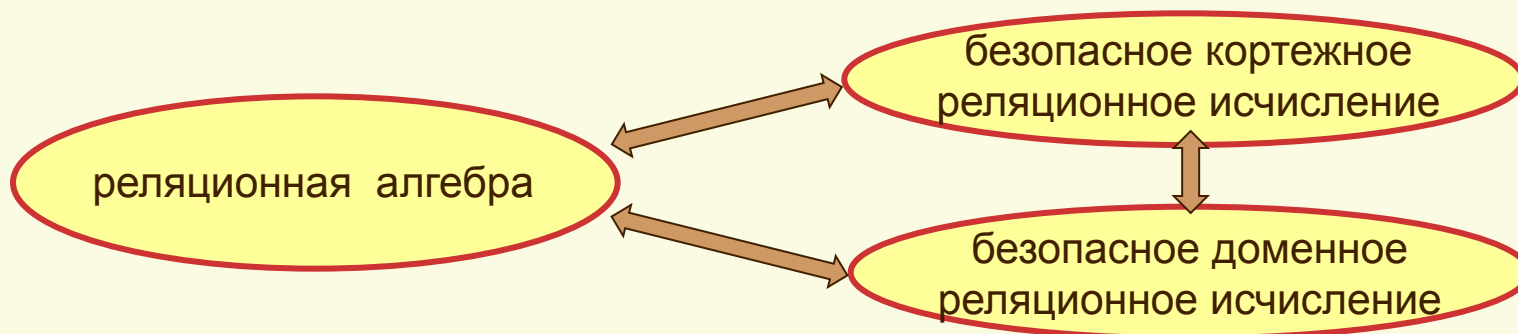
**DEP(DNo, FNo, Name, Head, Bld, Fund)**

d f n h b u

## Лекция 7. Реляционное исчисление

### Эквивалентность RA, TRC, DRC и реляционная полнота.

**Утверждение:** Реляционная алгебра, безопасное кортежное реляционное исчисление и безопасное доменное реляционное исчисление являются эквивалентными.



**Тезис** (о реляционной полноте): Язык реляционной модели является реляционно полным, если он обладает селективными возможностями реляционной алгебры.

**Утверждение:** Язык SQL является реляционно полным.

## Реляционное исчисление Кодда (CRC)

---

- Является подмножеством исчисления предикатов (1-го порядка)
- Является кортежно-ориентированным
- Решает проблему различия между ппф и безопасными формулами
- Запросы являются безопасными
- Является эквивалентным реляционное алгебре; значит обладает реляционной полнотой

## CRC – основные составляющие

---

- **Константы:** 7, 'john', 3.14159 и т.д.
- **Кортежные (строковые) переменные:**  $t_1, t_2, \dots$  – интерпретируются кортежами отношений.
- **Срезы кортежных переменных:**  $t.[i], \dots$   
 $i$  – компонента кортежной переменной.
- **Предикатные символы:**  $P, P_1, \dots, Q, Q_1, \dots$   
Интерпретируются отношениями
- **Операторы (предикаты) сравнения :**  
 $= \{=, <, <=, >, >=, <>\}$  и т.д.
- **Логические связки:**  $\vee$  (или),  $\wedge$  (&) (и),  $\neg$  (не),  $\rightarrow$  (влечет)
- **Кванторы:**  $\exists$  (существует),  $\forall$  (для любого)

## CRC - правильно определенные формулы

- Термы:
  - $P.t$  – терм значений:  $P$  - предикат,  $t$  - кортежная переменная.  
Если  $P$  интерпретируется отношением  $R$ , то  $P.t$  означает  $t \in R$
  - $t[i] \theta s[j]$ ,  $t[i] = c$  - термы соединений  
Примеры:  $t_1[3] = t_2[1]$ ;  $t_4[7] = 15$
- Формулы, правильно определенные над строковой переменной:
  - Все входящие в формулу предикатные символы интерпретируются совместимыми по объединению отношениями.
  - В формулу входят только термы значений с единственной переменной.
  - Термы значений соединяются связками  $\vee$ ,  $\&$ ,  $\neg$ . Причем, связке  $\neg$  непосредственно предшествует связка  $\&$ .

Примеры формул, правильно определенных над переменной  $t$ .

$$P_1.t \vee P_2.t \vee (P_3.t \& P_4.t); \quad (P_1.t \vee P_2.t) \& \neg P_3.t.$$

## Лекция 7. Реляционное исчисление

# Формула, правильно определенная на кванторах

Пусть  $F$  – формула, правильно определенная над строковой переменной  $t$ , а  $G$  – формула, содержащая  $t$  в качестве свободной переменной, но не содержащая термов значений вида  $P.t$ . Тогда формулы:

$$\exists t(F \ \& \ G); \quad \forall t(F \rightarrow G)$$

называются *правильно определенными на кванторах*.

По сути, это частный случай ограниченных кванторов существования и общности. Здесь формула  $F$  указывает ту область, в пределах которой изменяется переменная, используемая в кванторах.

Будем записывать эти формулы следующим образом:

$$\exists F(G) \text{ и } \forall F(G)$$

**Примеры:**

$$\exists t((P.t \vee Q.t) \ \& \ \neg S.t) \ \& \ t[2] = \text{'john'} \ \& \ t[5] = 1000)$$

$$\forall t((P.t \vee Q.t) \ \& \ \neg S.t) \rightarrow (t[2] = \text{'john'} \ \& \ t[5] = 1000))$$

## Лекция 7. Реляционное исчисление

# Формула с областью определения

Формула  $W$  называется **формулой с областью определения**, если она имеет вид:

$$W = U_1 \ \& \ U_2 \ \& \dots \ \& \ U_n \ \& \ V, \quad n \geq 1$$

и обладает следующими свойствами:

- Каждая формула  $U_i$  является правильно определенной над кортежной переменной  $t_i$ .
- Формула  $V$  либо пуста, либо обладает следующими свойствами:
  - Если в ней содержатся кванторы, то они правильно определены.
  - Каждая свободная переменная из  $V$  является одной из переменных формул  $U_1, U_2, \dots, U_n$
  - В  $V$  нет термов значений, содержащих свободные переменные.

Примеры:  $P.r \ \& \ Q.s \ \& \ R.t$  - декартово произведение

$P.r \ \& \ (r[3] = b)$  - селекция

$P.r \ \& \ Q.s \ \& \ (r[2] = s[1])$  - соединение



## Альфа-выражения

Выражение  $(t_1, t_2, \dots, t_k) : W$  называется **простым альфа-выражением**, если выполнены следующие условия:

- $W$  – формула с областью определения.
- $t_1, t_2, \dots, t_k$  – последовательность кортежных переменных и срезов. Эти переменные должны быть свободными в  $W$ .

Произвольное **альфа-выражение** определяется рекурсивно следующим образом:

- Каждое простое альфа-выражение является альфа-выражением.
- Пусть  $t = (t_1, t_2, \dots, t_k)$ . Если  $t : W_1$  и  $t : W_2$  – альфа-выражения, то выражения  $t : W_1 \vee W_2$ ,  $t : W_1 \& W_2$  и  $t : W_1 \& \neg W_2$  являются альфа-выражениями.

## Лекция 7. Реляционное исчисление

# Язык ALPHA

---

Упрощенный синтаксис:

**RANGE** <relation name> <variable name> [ **SOME** | **ALL**]

...

**GET** <workspace> (<target list> ) : <wff>

**range** – указание имени кортежной переменной <variable name>, которая принимает значения из отношения <relation name>.

**SOME** | **ALL** – указывают, следует ли интерпретировать переменную с квантором существования или общности.

**workspace** – идентификатор, который именуется временное рабочее отношение, содержащее результат выполнения команды Get.

**target list** – целевой список кортежных переменных и их срезов, указывающих столбцы, которые проецируются в результирующее отношение.

**wff** – правильно построенная формула кортежного реляционного исчисления

## Примеры запросов в ALPHA и CRC

---

**Запрос.** Вывести названия факультетов и их деканов

**CRC:** {f[2], f[3] | FAC.f}

**ALPHA:** RANGE FACULTY f

GET W(f.Name, f.Dean)

---

**Запрос.** Вывести имя декана факультета CSF

**CRC:** {f[3] | FAC.f & f[2] = 'CSF'}

**ALPHA:** RANGE FACULTY f

GET W(f.Dean) : f.Name = 'CSF'

---

**Запрос.** Вывести названия кафедр факультета CSF

**CRC:** {d[3] | DEP.d & ∃ f(FAC.f & f[2] = 'CSF' & f[1] = d[2])}

**ALPHA:** RANGE FACULTY f SOME

RANGE DEPARTMENT d

GET W(d.Name) = f.Name = 'CSF' & f.FNo = d.DNo

## Заключение

---

- Реляционное исчисление является декларативным (не процедурным) языком, в котором запросы формулируются в терминах того, ЧТО требуется, а не КАК вычислить требуемые значения.
- (Безопасное) реляционное исчисление и реляционная алгебра имеют одинаковую выразительную силу, и на их основании формулируется тезис о реляционной полноте.
- Имеется возможность формулировать правильные, но не эффективные запросы. В связи с этим необходимы оптимизаторы.
- Имеются языки непосредственно реализующие реляционное исчисление (ALPHA, QUEL), а также базирующиеся на кортежном исчислении (SQL) и доменном исчислении (QBE).