Capítulo 7: Outras linguagens

- Query-by-Example (QBE)
- Datalog

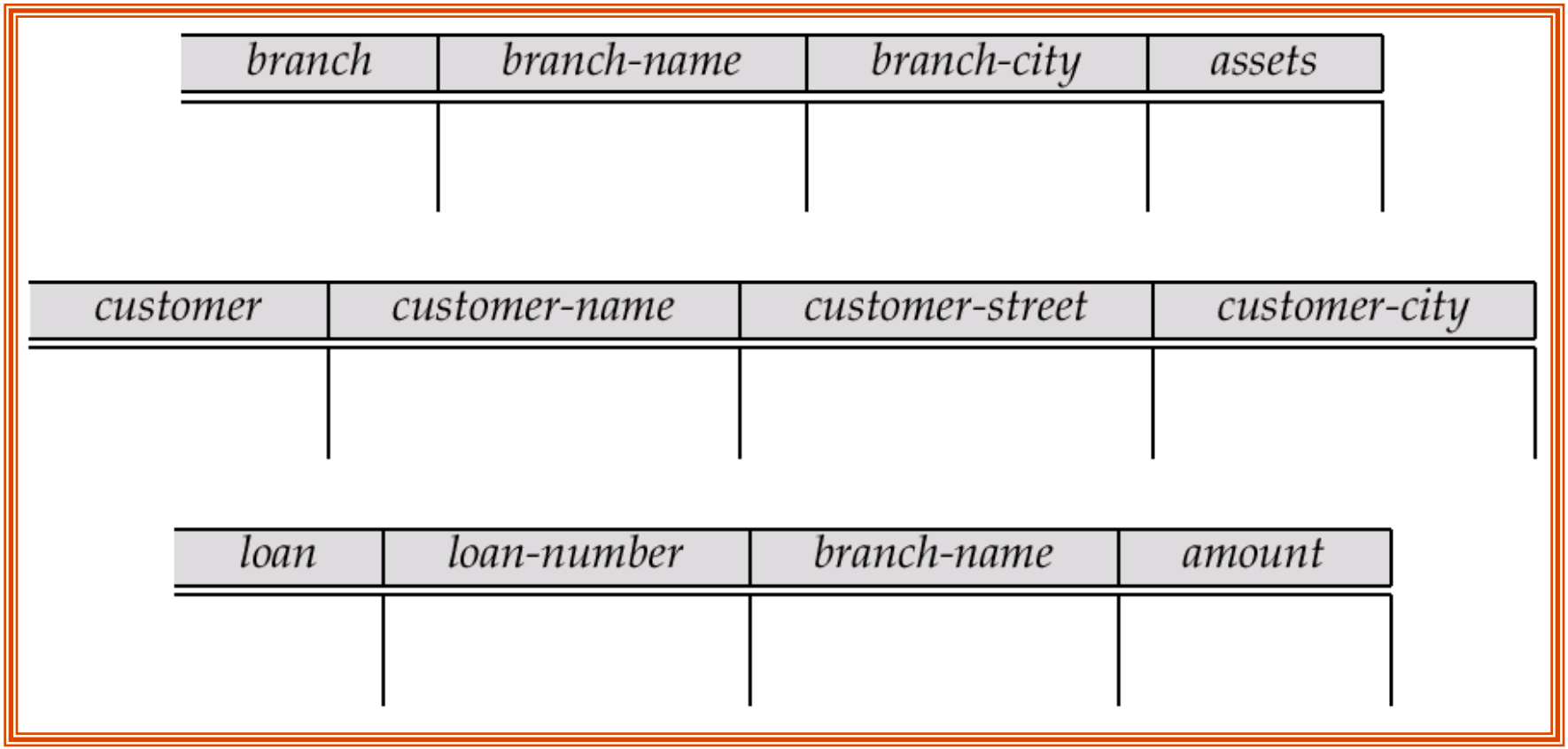
Query-by-Example (QBE)

- Estrutura básica
- Perguntas apenas numa relação
- Perguntas em várias relações
- A *Condition Box*
- Exemplo do Access

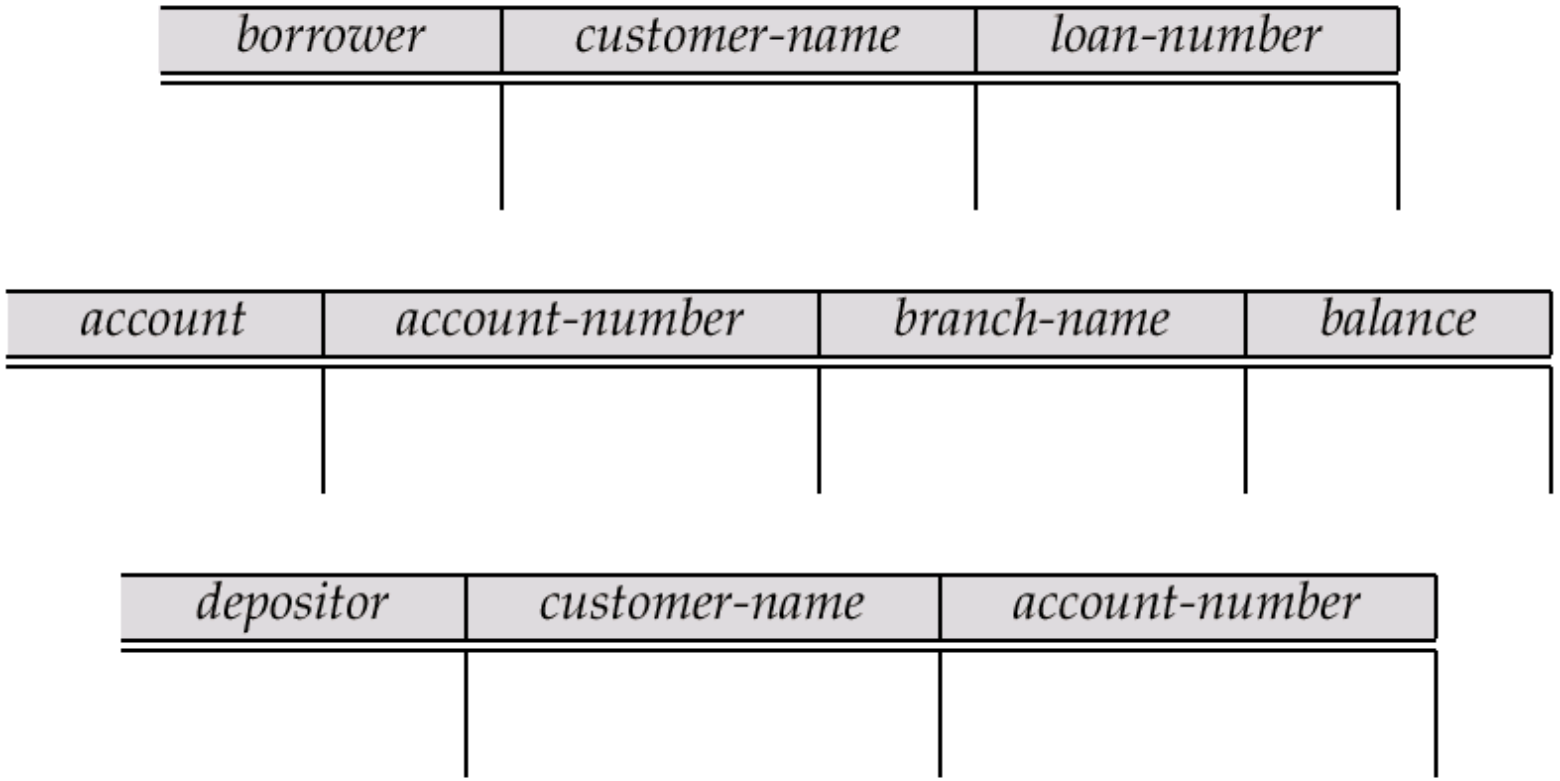
QBE — Estrutura de base

- Linguagem gráfica de perguntas a bases de dados relacionais
- O sistema cria templates de relações solicitadas por utilizadores
- As perguntas são feitas com padrões, “por exemplo”

Tabelas QBE para o exemplo do banco



Tabelas QBE (Cont.)



Perguntas numa só relação

- Encontrar todos os números de empréstimos da agência Perryridge

loan	*loan-number*	*branch-name*	*amount*
	P._x	Perryridge	

- _x é uma variável
- P. significa print (display)
- os tuplos duplicados são removidos por default
- Para reter os duplicados usar P.ALL

loan	*loan-number*	*branch-name*	*amount*
	P.ALL.	Perryridge	

Perguntas numa só relação (Cont.)

- Encontrar os números dos empréstimos de valor superior a 700

loan	*loan-number*	*branch-name*	*amount*
	P.		>700

- Nome de agências que não sejam em Brooklyn

branch	*branch-name*	*branch-city*	*assets*
	P.	¬ Brooklyn	

Perguntas numa só relação (Cont.)

- Números de empréstimos feitos conjuntamente pelo Smith e o Jones.

borrower	*customer-name*	*loan-number*
	"Smith"	P._*x*
	"Jones"	_*x*

- Todos os clientes que vivem na mesma localidade que o Jones

customer	*customer-name*	*customer-street*	*customer-city*
	P._*x*		_*y*
	Jones		_*y*

Perguntas em várias relações

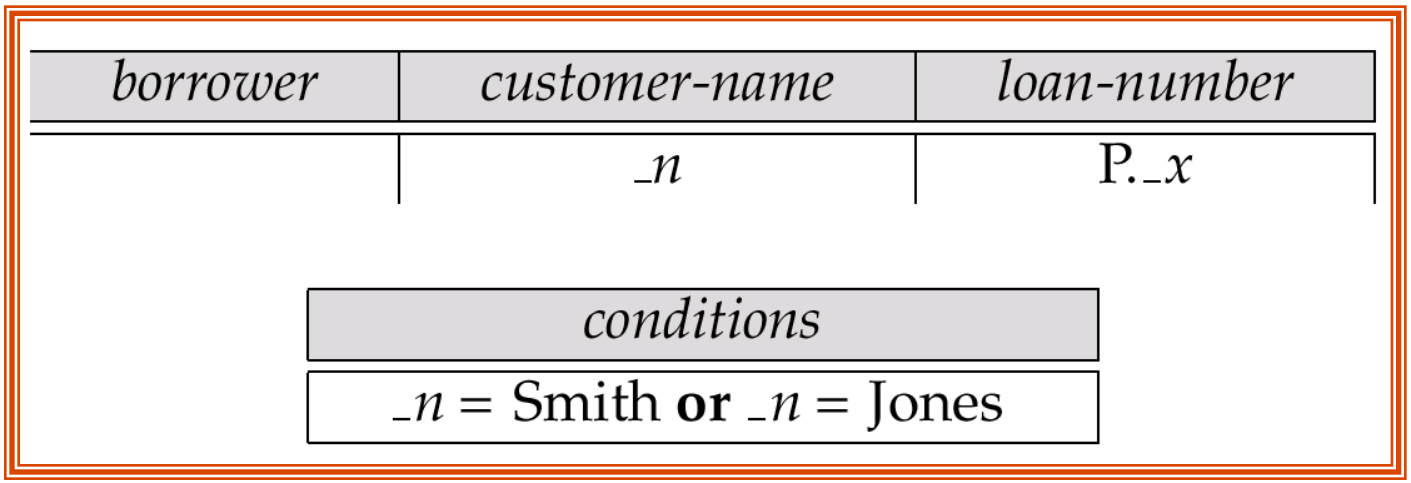
- Nomes de todos os clientes com empréstimos na agência Perryridge

loan	*loan-number*	*branch-name*	*amount*
	_x	Perryridge	

borrower	*customer-name*	*loan-number*
	P._y	_x

Condition Box

- Permite impor condições extra sobre atributos.
- E.g. Números de empréstimos feitos pelo Smith, pelo Jones, ou por ambos



Condition Box (Cont.)

- Números de contas com saldo entre $1,300 e $1,500

account	*account-number*	*branch-name*	*balance*
	P.		_x

conditions
_x ≥ 1300 _x ≤ 1500

- Agências com activos maiores que pelo menos os de uma agência em Brooklyn

branch	*branch-name*	*branch-city*	*assets*
	P._x		_y
		Brooklyn	_z

conditions
_y > _z

Outras características

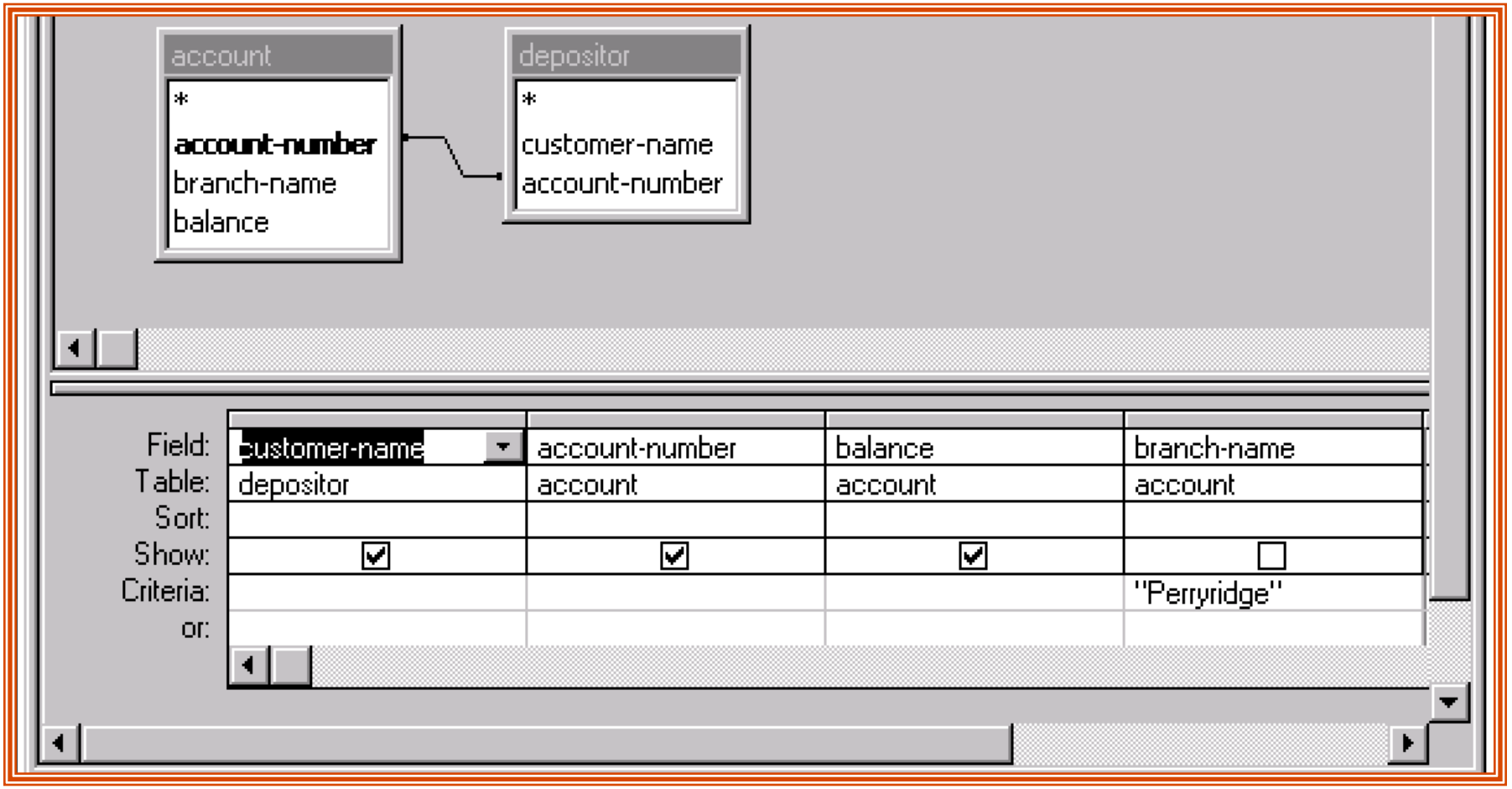
- Com o QBE é ainda possível executar ordenação de tuplos, algumas formas de agregação, e modificação de bases de dados.
- Tudo o que é possível exprimir com o QBE, também o é com SQL
- ... mas não vice-versa!!

Microsoft Access QBE

- O Microsoft Access suporta uma variante do QBE - Graphical Query By Example (GQBE)
- É diferente do QBE em:
  - Em vez de usar variáveis para dizer que os valores de dois atributos têm que ser iguais, usa linhas a ligar os atributos.
    - Por default, as linhas especificam inner joins. Mas é possível alterar para outer joins.
  - As condições, valores a afixar, etc, são especificados na **design grid**

Exemplo em Access QBE

- Encontrar os nomes de clientes, saldo e número de conta de todos as contas da agência de Perryridge



Datalog

- Estrutura de Base
- Sintaxe de regras Datalog
- Operações relacionais em Datalog

Estrutura de Base

- Um programa Datalog é um conjunto de regras, que definem relações (vistas).
- Exemplo: definir relação (*vista*) *v1* com todos os números de conta e respectivos saldos, para as contas de Perryridge com saldo superior a 700.

  *v1(A, B) :– account(A, "*Perryridge", *B), B* > 700.

- Qual o saldo da conta "A-217" na *vista v1?*

  *?- v1("*A-217", *B).*

- Quais as contas com saldo superior a 800?

  ?- account(A,_,B), B > 800

- Permite recursão.

Exemplos de perguntas

- Cada regra define um conjunto de tuplos que pertencem à *vista*
  - E.g. *v1(A, B) :– account(A, "*Perryridge", *B), B* > 700
    lê-se como

    para todo *A, B*

    se (*A*, "Perryridge", *B*) ∈ *account* **e** *B* > 700

    então (*A, B*) ∈ *v1*
- O conjunto de tuplos duma *view* é definido como a união dos conjuntos de tuplos de cada uma das suas regras.
- Exemplo:

  interest-rate(N, 5*) :– account(N,A, B), B* < 10000
  interest-rate(N, 6*) :– account(N,A, B), B* >= 10000

Negação em Datalog

- Definir a *view c* contendo os nomes de todos os clientes que têm pelo menos um depósito mas não têm empréstimos:

  c(N) :– *depositor(N, A),* **not** *is-borrower(N).*
  is-borrower(N) :–*borrower (N,L).*

- NOTA: usando directamente **not** *borrower (N, L)* na 1ª regra dava resultado diferente: clientes *N* que têm pelo menos um depósito e existe algum empréstimo *L* que não é de *N*
  - Para evitar confusão (e problemas de implementação), é usual exigir-se que as variáveis em literais negados apareçam na cabeça da regra ou em algum literal positivo do corpo da regra.

Regras por camadas (views sobre views)

- Quais os juros de cada uma das contas de Perryridge

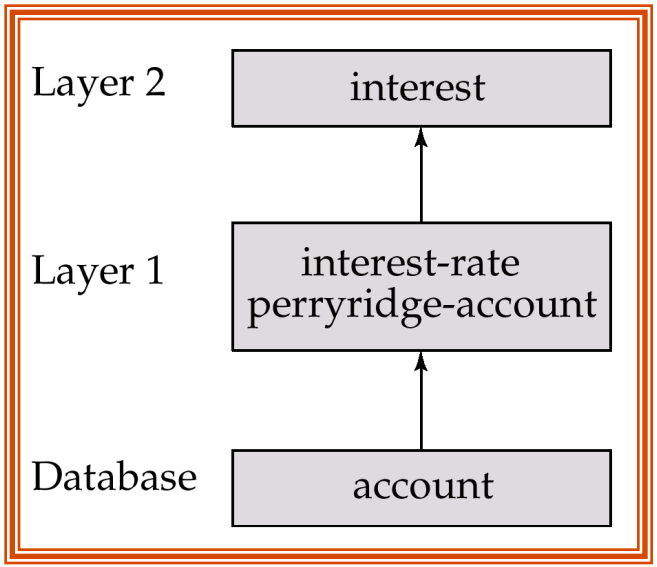
  interest(A, I) :– *perryridge-account*(*A,B*),
  *interest-rate(A,R), I = B * R*/100.

  perryridge-account(A,B) :– *account*(*A,* “Perryridge”, *B).*

  *interest-rate(N,*0) :– account(*N, A, B), B* < 2000.

  interest-rate(N,5) :– account(N, A, B), *B* >= 2000.

- Camadas das várias views:



Recursão em Datalog

- Considere a relação:

  precedencia(X, Y)

  contendo pares X, Y de cadeiras onde X dá precedência directa para Y.

- Cada cadeira pode dar precedência para outras, directa ou indirectamente
  - Uma cadeira X dá precedência indirectamente a Y, se dá precedência directa a uma cadeira Z, que por sua vez dá precedência para Y
- Encontrar todas as cadeiras que dão precedência (directa ou indirecta) a Bases de Dados. Podemos escrever o seguinte programa Datalog **recursivo**:

  precBD(X) :- precedencia(X, 'BD').

  precBD(X) :- precedencia(X, Y), precBD(Y).

Recursão em Datalog (Cont.)

- De forma mais geral: criar view *precs* com todos os pares *(X, Y)* onde *X* dá precedência directa ou indirecta para *Y:*

  precs(X, Y) :– *precedencia(X, Y).*
  precs(X, Z) :– *precs(X,Y), precs(Y,Z).*

- Quais as cadeiras que dão precedência a BD

  ?- precs(X, 'BD').

Operações Relacionais em Datalog

- Projecção do atributo *account-name* de *account*:

  query(Account-name) :– *account(Account-name, N, B).*

- Produto cartesiano das relações r_1 e r_2.

  $query(X_1, X_2, ..., X_n, Y_1, Y_2, ..., Y_m)$:–
  $r_1(X_1, X_2, ..., X_n), r_2(Y_1, Y_2, ..., Y_m).$

- União das relações r_1 e r_2.

  $query(X_1, X_2, ..., X_n)$:– $r_1(X_1, X_2, ..., X_n)$.
  $query(X_1, X_2, ..., X_n)$:– $r_2(X_1, X_2, ..., X_n)$.

- Selecção de tuplos de r_1 que obedecem a condição *cond*.

  $query(X_1, ..., X_n)$:– $r_1(X_1, ..., X_n)$, $cond(X_1,..., X_n)$.

- Diferença entre r_1 e r_2.

  $query(X_1, X_2, ..., X_n)$:– $r_1(X_1, X_2, ..., X_n)$,
  not $r_2(X_1, X_2, ..., X_n)$.