

**Язык Пролог в качестве
языка запросов к
параллельной
распределённой СУБД**

$$f(n) = 1 \cdot 2 \cdot \dots \cdot n$$

1) $k = 1, i = 1$

2) если $i < n$, то завершить

3) $k = k \cdot i$

4) $i = i + 1$

5) перейти на пункт 2

```
int f(int n){ int i, k = 1;  
  for(i = 1; i < n; i++) k = k * i;  
  return j;  
}
```

1973

- программа = множество аксиом;
- вычисление = конструктивный вывод целевого утверждения из программы.

- **predicates**

- factorial(integer, integer)

- **clauses**

- factorial(0,1):-!.

- factorial(N,R):-P=N-1,
factorial(P,Prev), R=Prev*N.

- **goal**

- factorial(3,X).

Функциональная запись

- predicates
- программист(string)
- знать_пролог(string)
- clauses
- программист("Сергей").
- программист("Оля").
- знать_пролог(X) :- программист(X).
- goal
- знать_пролог(A).
- A=Сергей
- A=Оля
- 2 Solutions

«Естественная» запись

- \$ - программист:
- \$ знает пролог:
- \$Сергей - программист.
- \$Оля - программист.
- X знает пролог, если X - программист.
- Кто знает пролог ?
- Сергей
- Оля
- 2 Solutions

Языки запросов

Программист_знает_язык

Программист	Язык
Сергей	Пролог
Сергей	Бейсик
Оля	Пролог

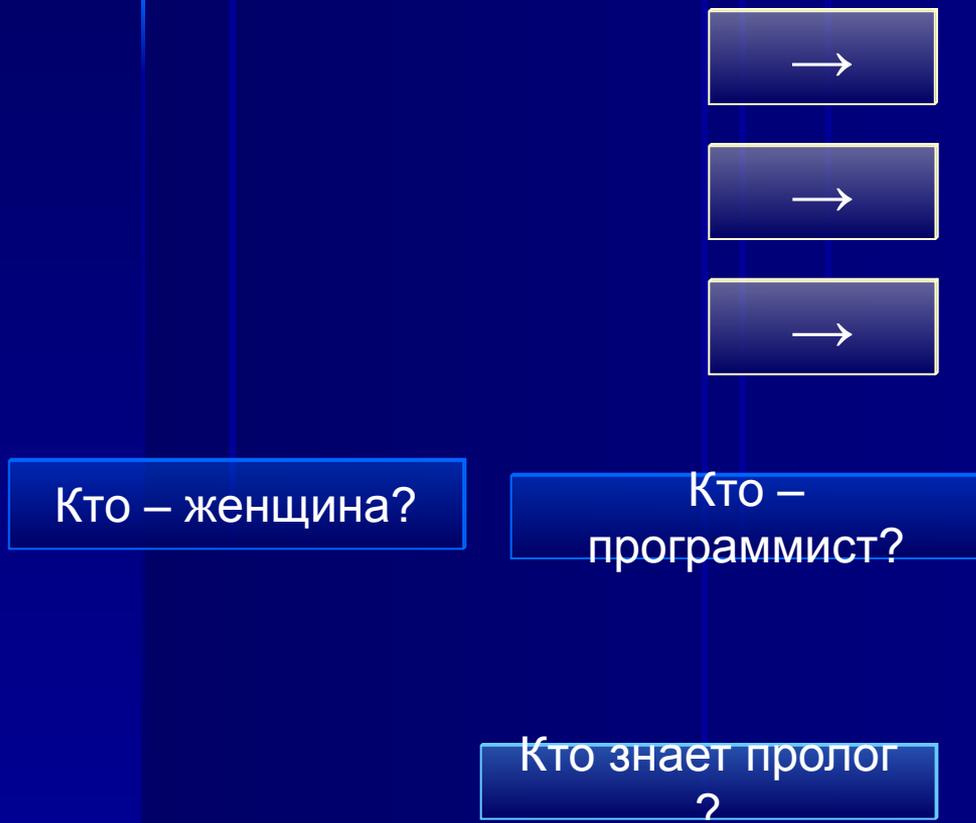
Пролог:

программист \$ знает язык \$:
программист \$Сергей знает
язык \$Пролог.
программист \$Сергей знает
язык \$Бейсик.
программист \$Оля знает
язык \$Пролог.

SQL:

```
create table "Программист_знает_язык"  
("Программист" string, "Язык" string).  
insert into "Программист_знает_язык"  
("Сергей", "Пролог").  
insert into "Программист_знает_язык"  
("Сергей", "Бейсик").  
insert into "Программист_знает_язык"  
("Оля", "Пролог").
```

Параллельное вычисление



- ❑ \$Оля – женщина.
- ❑ \$Сергей – программист.
- ❑ \$Оля – программист.
- ❑ X знает пролог, если X – программист, X - женщина.
- ❑ Кто знает пролог ?

Схема Пролог-системы

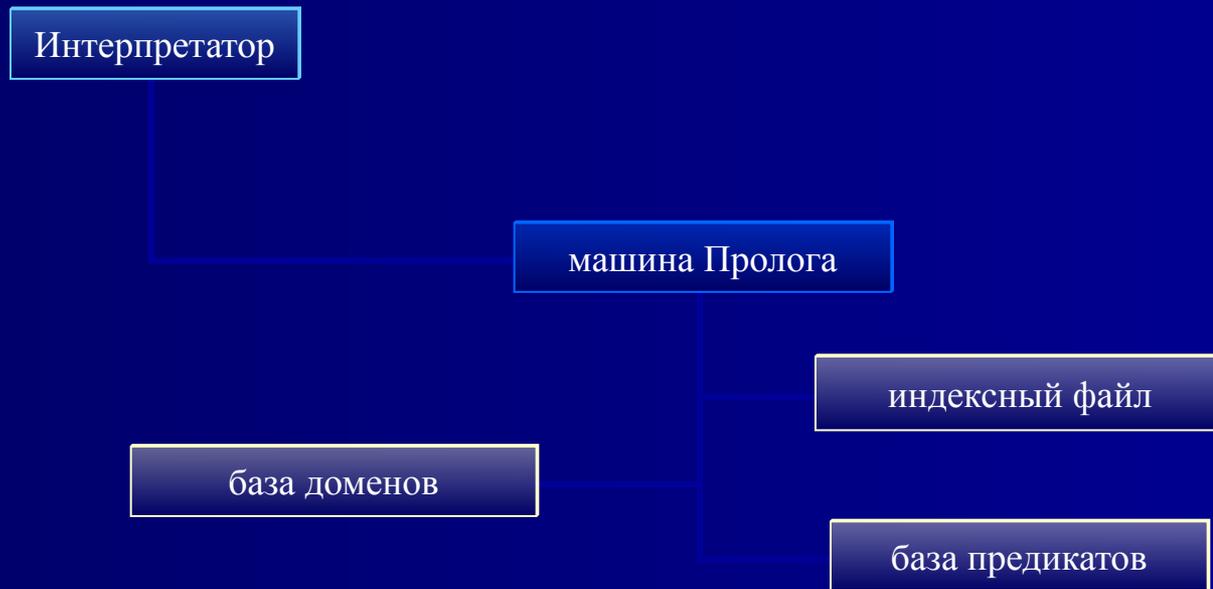


Схема распределённой Пролог-системы

