Метод ускорения запросов в базе данных

Студент: Тартыков Лев Евгеньевич

Группа: ИУ7-64Б

Научный руководитель: Строганов Юрий Владимирович

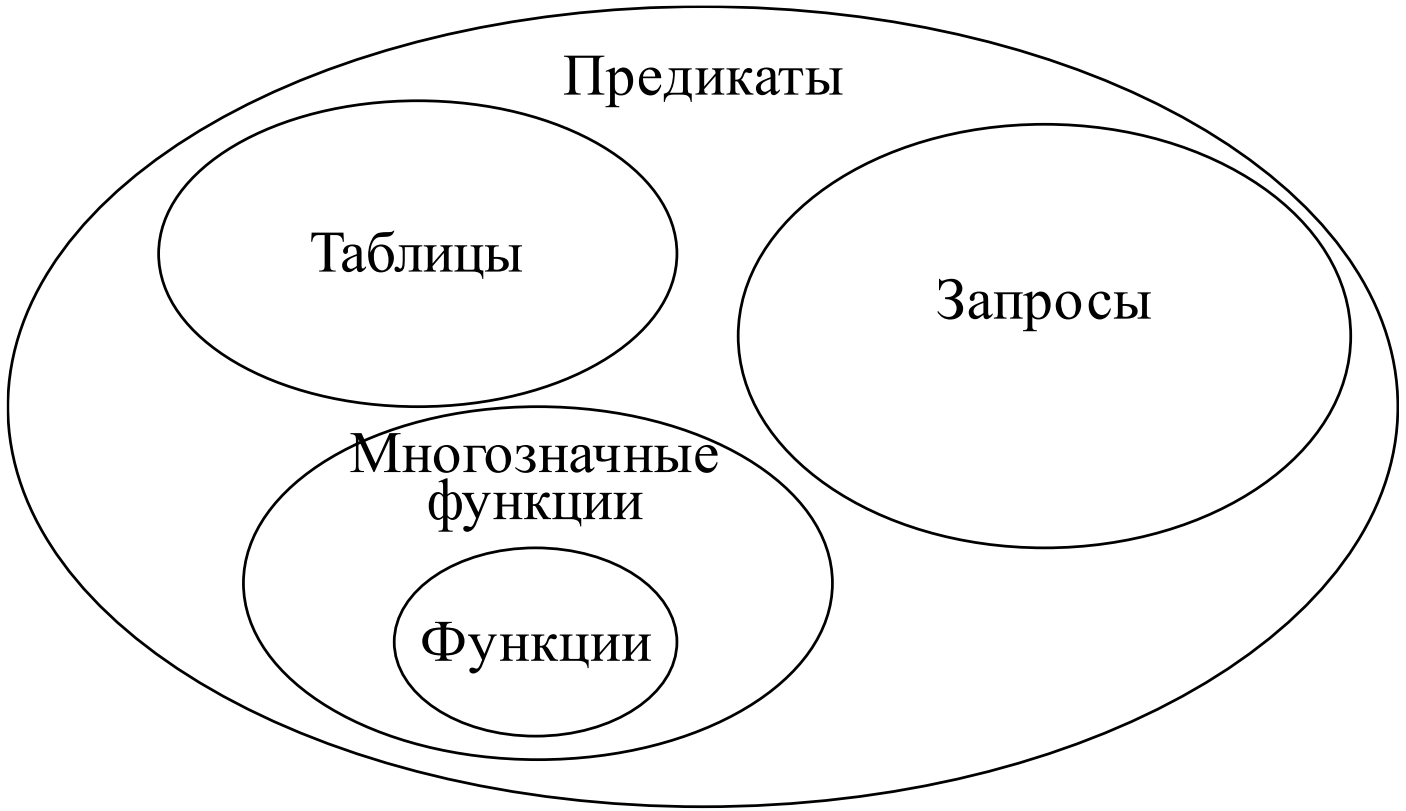
Москва, 2022 г.

Цель и задачи

Цель – нахождение способа ускорения выполнения запросов в базе данных на основе распараллеливания с использованием логического языка программирования.
Задачи:

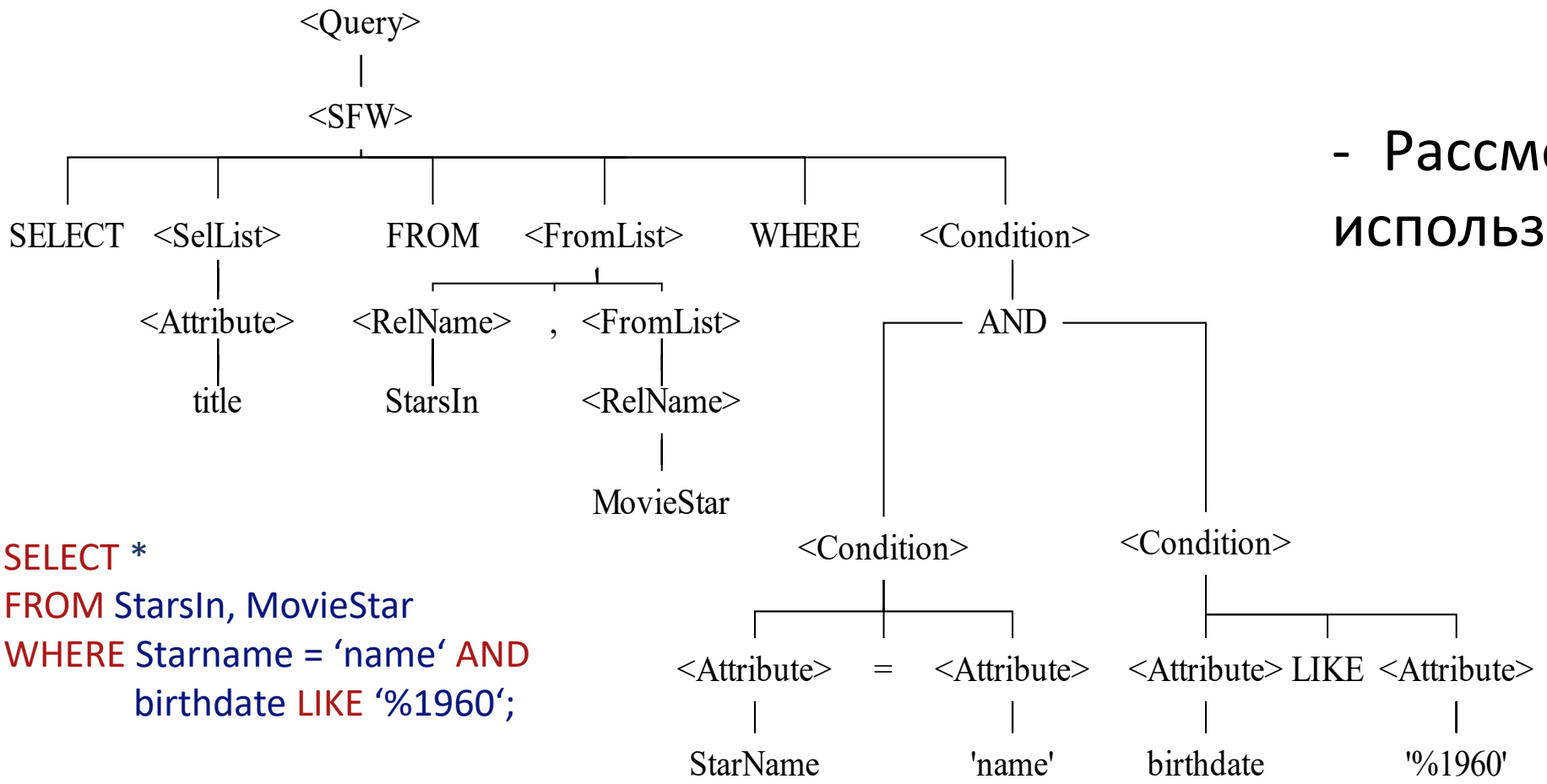
- рассмотреть подходы к представлению реляционных баз данных;
- описать методы параллелизма плана запроса и логического запроса;
- выполнить анализ существующих СУБД и реализаций Prolog, способных к распараллеливанию;
- реализовать метод ускорения выполнения запросов.

Организация хранения данных в БД



– Предикат как универсум представления данных.
– Использование теории множеств.
– Лингвистическое упрощение описания запросов.

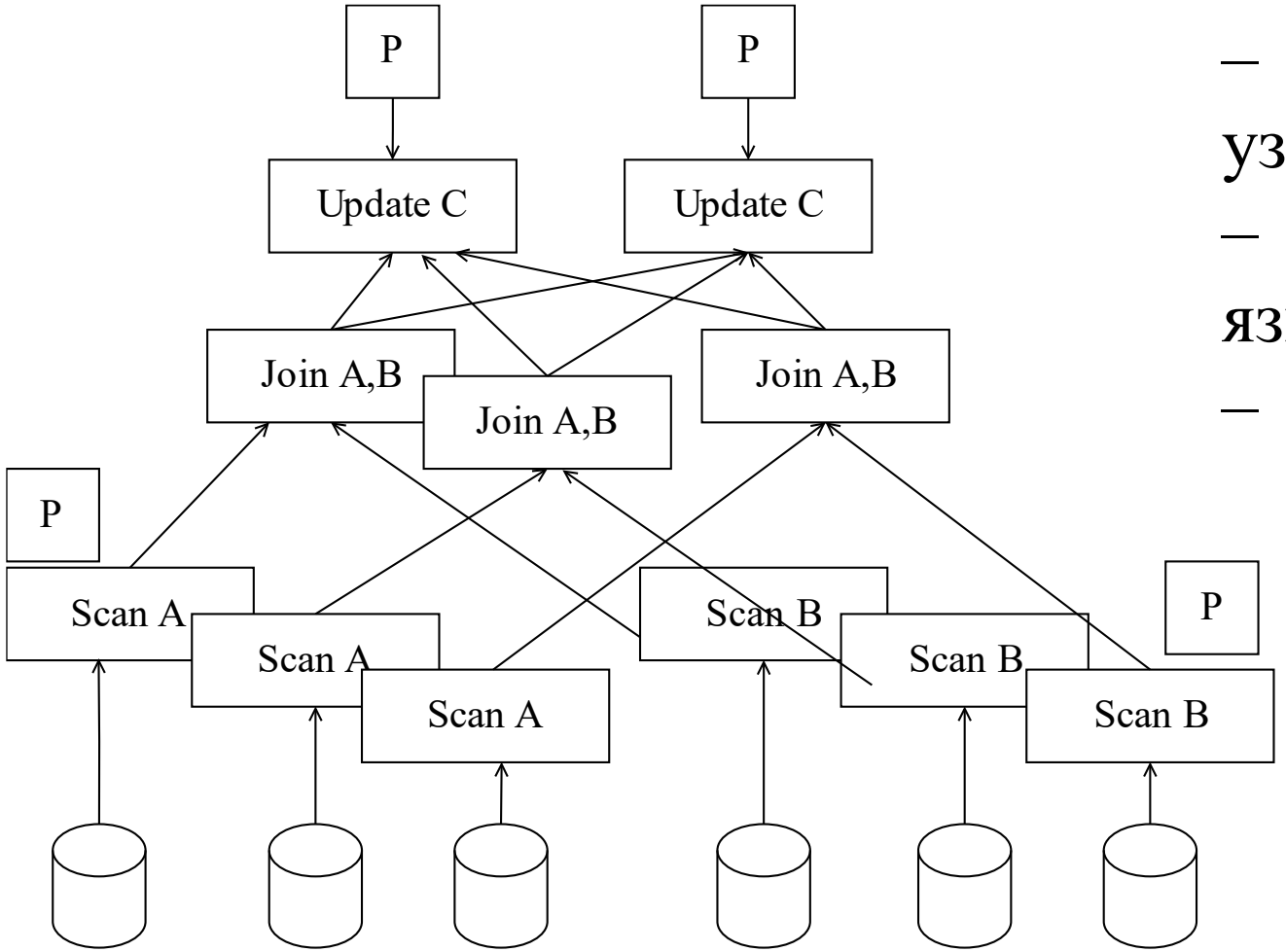
Ограничение SQL-грамматики



- Рассмотрение использования JOIN.

```
SELECT *
FROM StarsIn, MovieStar
WHERE Starname = ‘name‘ AND
        birthdate LIKE ‘%1960‘;
```

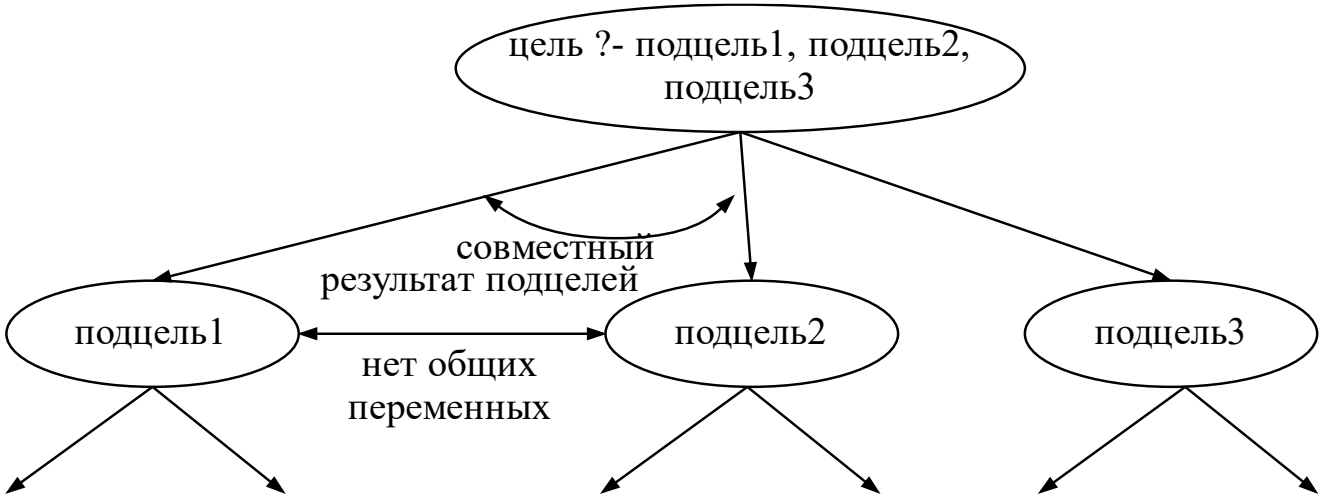
Узлы планировщика



– Выделение независимых узлов планировщика.
– Применение логического языка.
– Использование параллелизма.

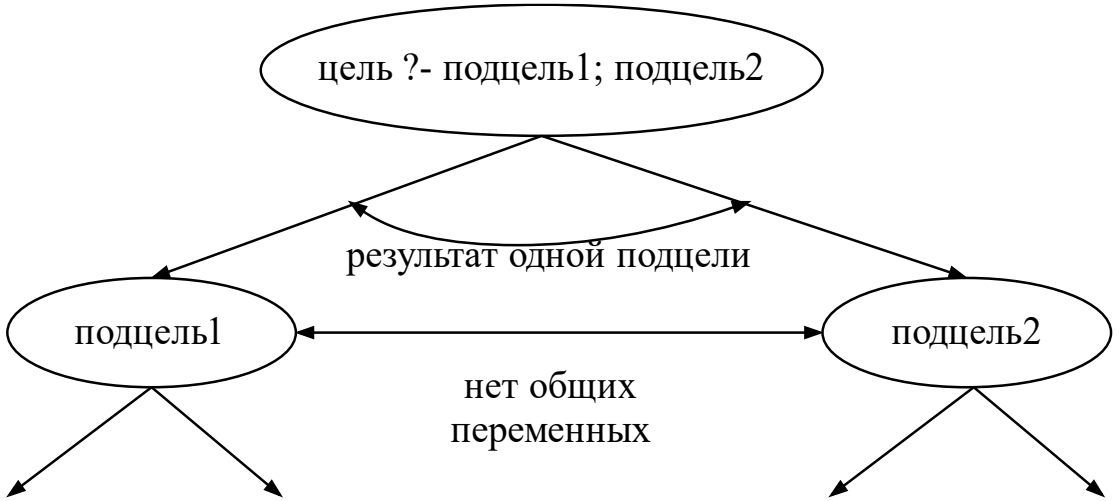
Способы ускорения

AND-параллелизм



AND-дерево

OR-параллелизм



OR-дерево

Обзор СУБД

СУБД	Open source	ACID	RDBMS	In-memory	SQL
★ PostgreSQL	+	+	+	-	+
Oracle	-	+	+	-	+
MySQL	+	+	+	-	+
MariaDB	+	+	+	-	+
MongoDB	+	+	-	-	-
SQLite	+	+	+	-	+
Cassandra	+	-	-	-	-
Redis	+	-	-	+	-

Параллельные реализации Prolog

Реализация	Крит. (1)	Крит. (2)	Крит. (3)	Крит. (4)	Крит. (5)
Datalog	+	-	+	+	+
Eclipse	+	+	-	+	-
Visual Prolog	-	?	?	?	?
CIAO	+	-	+	+	+
★ SWI-Prolog	+	+	+	+	+
Parlog	+	+	+	-	-
SICStus	-	?	?	?	?

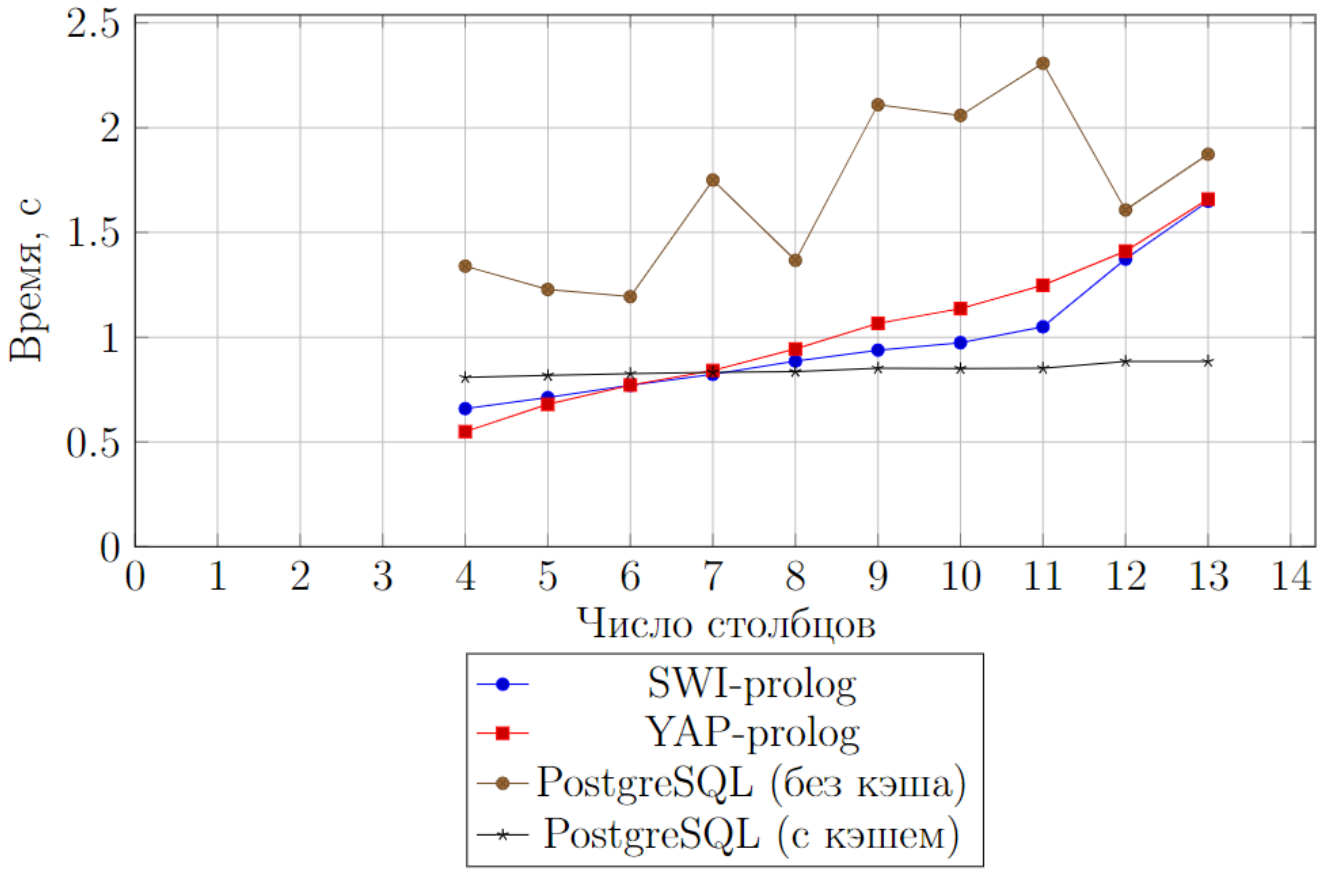
(1) Открытость.
(2) τ загрузки < 10 мин.
(3) Объем ОП < 5ГБ.

(4) Больше одного столбца результата.
(5) Больше 10^6 записей.

Технические характеристики устройства

– Процессор: Intel(R) Core(TM) i5-8265U CPU @ 1.60ГГц 1.80ГГц.
– Память: 8Гб.
– Число ядер: 4.
– ОС Ubuntu 22.04.

Эксперимент: SELECT-запрос без потоков



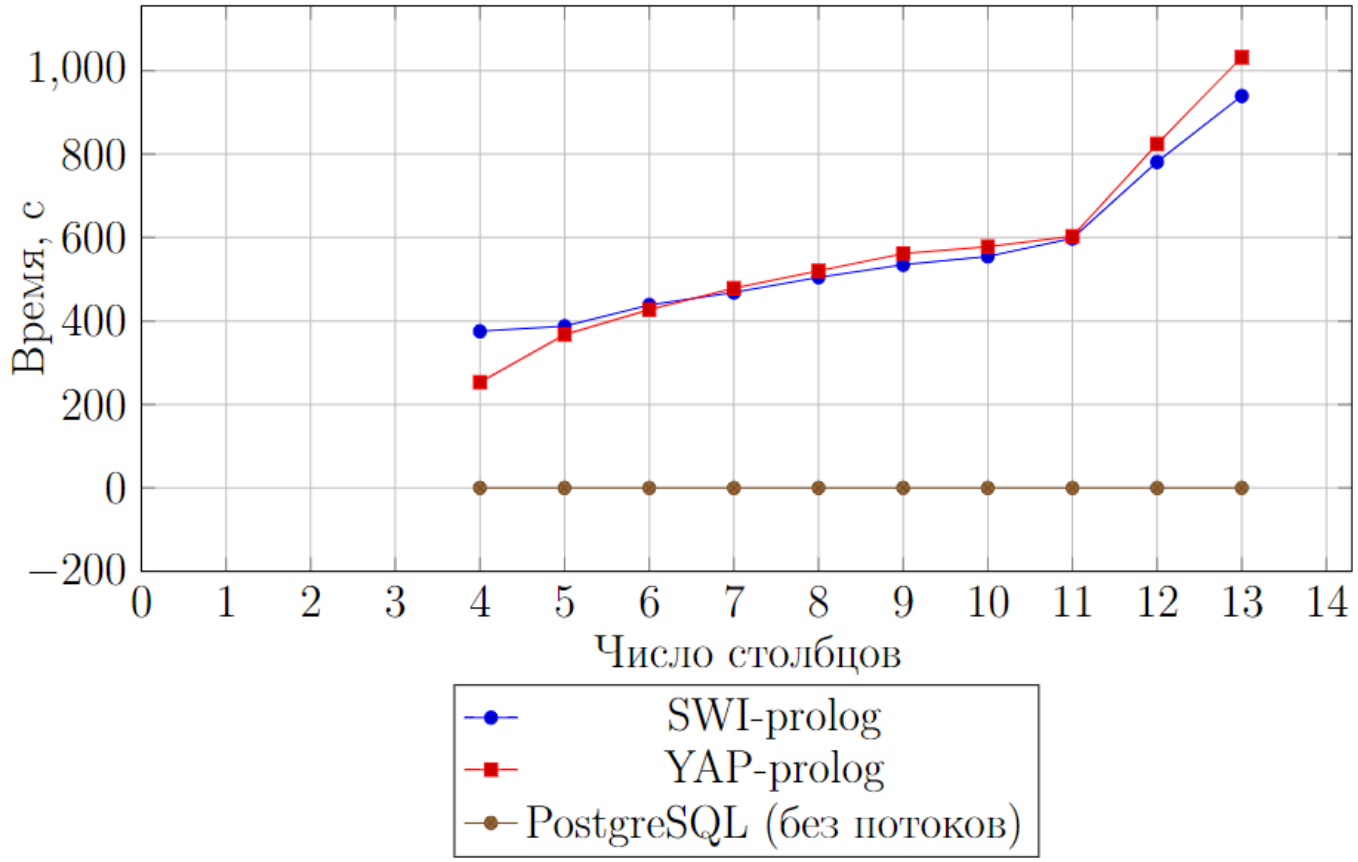
– Преимущество Prolog над SQL.
– 4-7 столбцов.

Число столбцов	Число записей	Программное обеспечение			
		SWI Prolog	YAP Prolog	PostgreSQL (без кэша)	PostgreSQL (с кэшем)
4	10^7	0.659228	0.549383	1.339074	0.807485
5		0.711844	0.679519	1.227935	0.818097
6		0.770522	0.771545	1.193835	0.825795
7		0.823258	0.841355	1.750495	0.832614
8		0.886037	0.94406	1.366828	0.836362
9		0.938527	1.066499	2.109747	0.851722
10		0.974033	1.136439	2.058274	0.850416
11		1.04980	1.248144	2.307539	0.852188
12		1.37296	1.411219	1.607229	0.884879
13		1.64924	1.659166	1.873321	0.884116

Эксперимент: JOIN-запрос без потоков

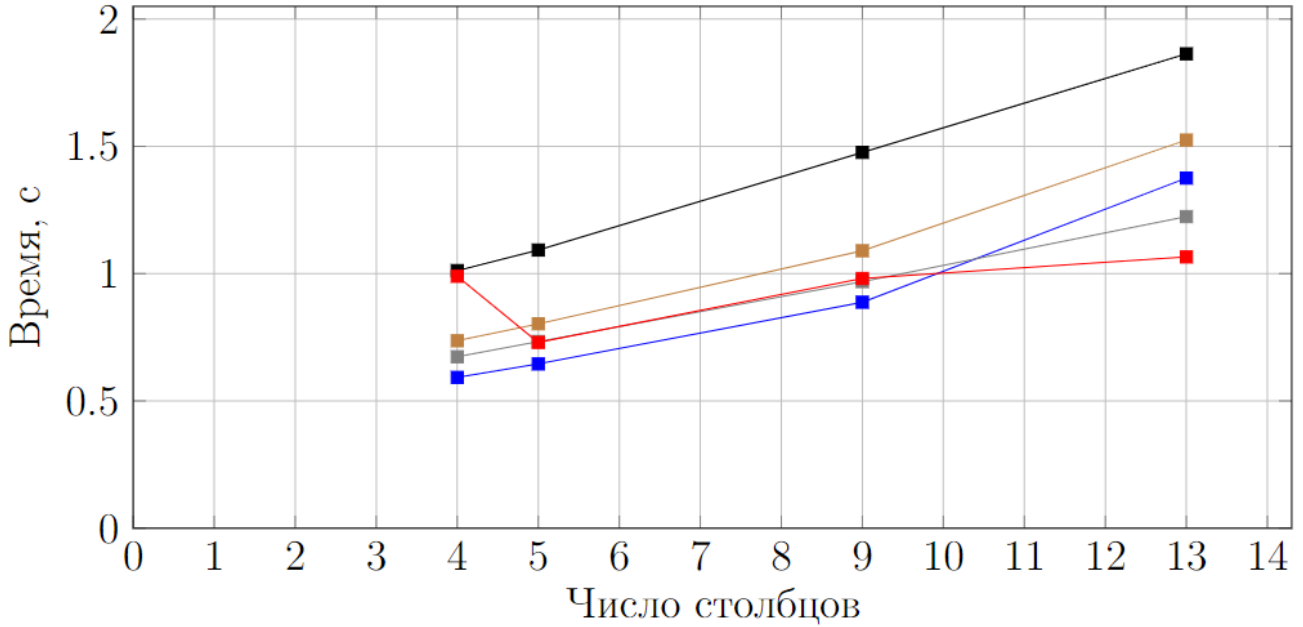
– Различие в производительности в 10000 раз!
– 10^5 записей

Число столбцов	Число записей	Программное обеспечение		
		SWI Prolog	YAP Prolog	PostgreSQL (без потоков)
4	10^5	375.436966	253.256186	0.031547
5		387.485804	367.456964	0.033847
6		438.366495	427.122528	0.038549
7		468.279248	478.549645	0.043654
8		504.606969	520.154154	0.048515
9		534.500522	561.591546	0.052371
10		554.721574	578.218155	0.055763
11		597.871645	603.125187	0.059163
12		780.914253	824.519815	0.063312
13		939.258741	1031.98327	0.069855



Эксперимент: SELECT-запрос с потоками

– Использование 2-4 SWI-потоков.



Число столбцов	Число записей	SWI Prolog(2)	SWI Prolog(3)	SWI Prolog(4)	SWI Prolog(5)	Postgre SQL(5)
4	10^7	0.592607	0.673686	0.736398	1.012103	0.989799
5		0.645411	0.733633	0.803135	1.092881	0.730035
9		0.887928	0.969058	1.090071	1.476527	0.981088
13		1.375427	1.224211	1.524814	1.863615	1.065807

Заключение

- рассмотрены подходы к представлению реляционных баз данных;
- описаны методы параллелизма плана запроса и логического запроса;
- выполнен анализ существующих СУБД и реализаций Prolog, способных к распараллеливанию;
- реализован метод ускорения выполнения запросов.

Направления развития

– создать concurrent_findall();

– использовать GPU для параллелизма логического вывода;

– рассмотреть параллельное выполнение инструкций WAM.