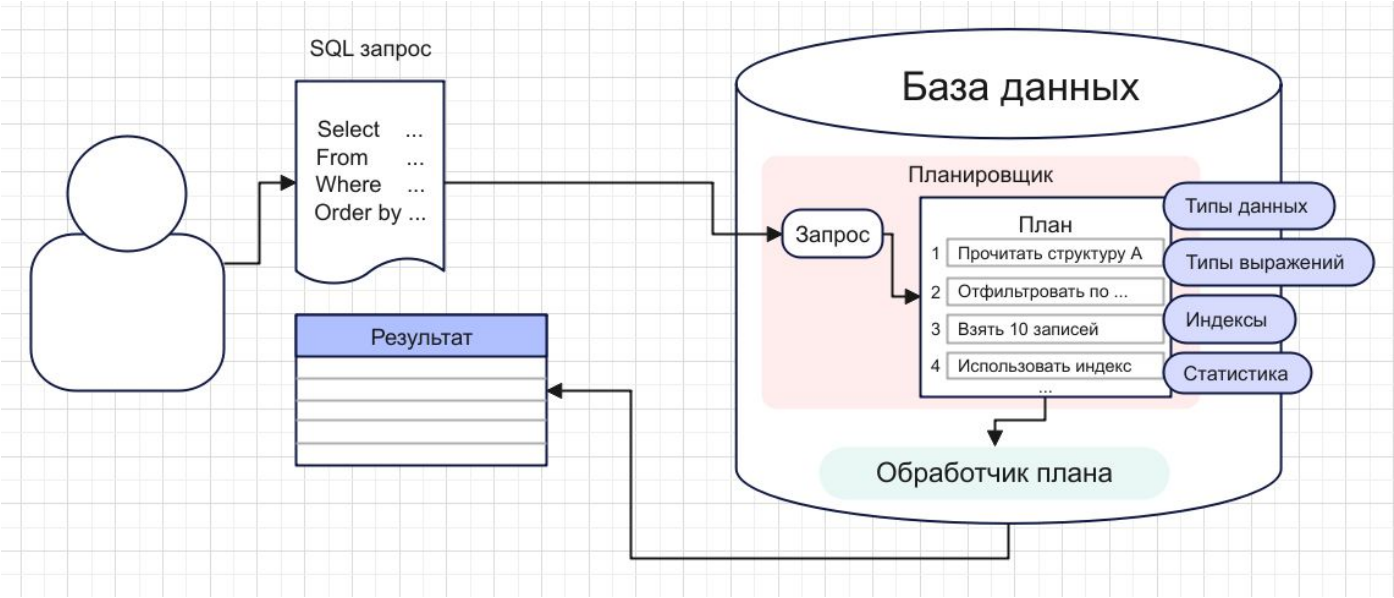
Новосибирский государственный университет, НГУ
09.03.01 Информатика и вычислительная техника
Компьютерные науки и системотехника

Разработка учебной системы оценки сложности планов SQL запросов

Выполнил: Савченко Егор Владимирович, студент гр. 20215, каф. СИ
Руководитель: Мигинский Денис Сергеевич, к.ф-м.н, доцент каф. СИ

Новосибирск 2024

Обработка запроса



Запрос и план запроса

Запрос - описание множества необходимых данных.

План запроса - последовательность операций по выборке, фильтрации, агрегации данных, приводящая к множеству необходимых данных.

Запрос и план запроса

```
from content c
where c.name like 'Tutorial #%'
join content_descriptor cd on true
where cd.status = 'published' and cd.id = c.descriptor_id
select *;
```

План №1

```
select *
from content c
join content_descriptor cd on cd.id = c.descriptor_id
where cd.status = 'published' and c.name like 'Tutorial #%';
```

SQL запрос

```
from content c
join content_descriptor cd on
    cd.id = c.descriptor_id and
    cd.status = 'published' and
    c.name like 'Tutorial #%'
select *;
```

План №2

План запроса

	QUERY PLAN
1	Hash Join (cost=9.55..21.07 rows=6 width=222)
2	Hash Cond: (cd.id = c.descriptor_id)
3	-> Seq Scan on content_descriptor cd (cost=0.00..11.14 rows=64 width=140)
4	Filter: ((status)::text = 'published'::text)
5	-> Hash (cost=9.14..9.14 rows=33 width=82)
6	-> Seq Scan on content c (cost=0.00..9.14 rows=33 width=82)
7	Filter: (name ~~ 'Tutorial #%'::text)

Цель

Реализовать **учебный** инструмент построения, исполнения и оценивания планов SQL запросов, поддерживающий основные операции по выборке/агрегации данных.

*В данном контексте “учебный “ означает, что с помощью полученной системы не планируется обрабатывать большие объемы данных. От инструмента не требуется высокая производительность и эффективность в организации/обработке данных, а также не требуется поддержка некоторых типов данных и операций над ними.

Для чего?

Инструмент может быть полезен для студентов начальных курсов, людей только начинающих изучать SQL и программирование в целом:

- Поможет понять концепцию декларативных/императивных ЯП
- Поможет в осознании принципов работы движков промышленных реализаций стандарта SQL
- Предоставит базовое понимание принципов обработки данных

Задачи

- Исследовать методы оценивания сложности планов SQL запросов
- Исследовать принципы обработки/организации данных промышленных реализаций стандарта SQL
- Определить множество необходимых типов данных и множество операций над ними
- Определить набор инструментов реализации
- Разработать архитектуру решения
- Реализовать учебную систему

Требования

- Наличие основных операций обработки: **Join** (Full, Inner, Left, Right + стратегии соединения), **Filter**, **Group By**, **Order By**
- Поддержка подзапросов
- Поддержка ленивости, кеширования, индексов
- Детерминированность результата подсчета сложности
- Реализация клиентской части системы не должна быть привязана к конкретному ЯП
- Реализация не должна быть привязана к реализации конкретной СУБД

Пример использования

*каждая из таблиц содержит по 331 записи

План №1

```
Query.from("content").as("c")
    .innerJoin(
            Query.from("content_descriptor").as("cd"),
            Q.op(EQ, Q.column("cd", "id"), Q.column("c", "descriptor_id")),
            JoinStrategy.LOOP
    ).as("res")
     .where(Q.op(LIKE, Q.column("res", "name"), new StringValue("Tutorial #%")))
     .where(Q.op(EQ, Q.column("res", "status"), new StringValue("published")));
```

329409 *ops*

План №2

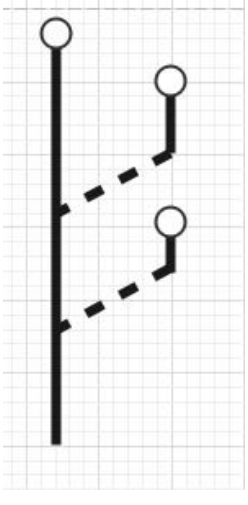
```
Query.from("content").as("c")
    .where(Q.op(LIKE, Q.column("c", "name"), new StringValue("Tutorial #%")))
    .innerJoin(
            Query.from("content_descriptor").as("cd")
                 .where(Q.op(EQ, Q.column("cd", "status"), new StringValue("published"))),
            Q.op(EQ, Q.column("cd", "id"), Q.column("c", "descriptor_id")),
            JoinStrategy.HASH
    );
```

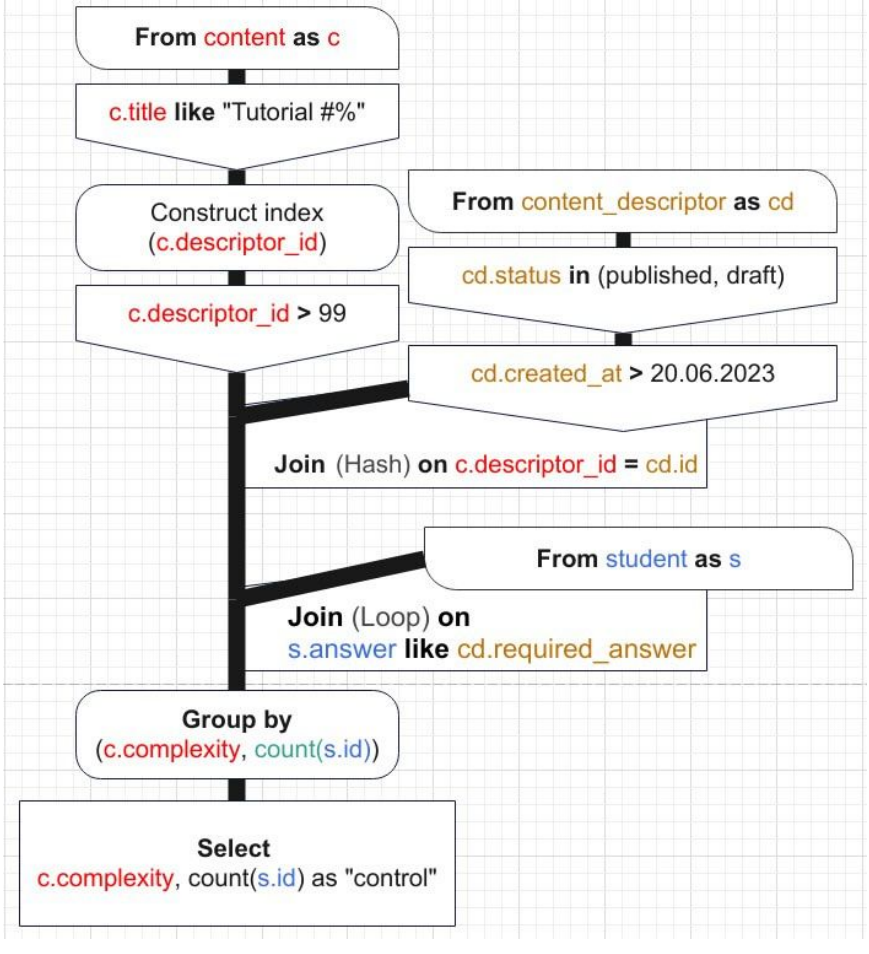
6806 *ops*

Общее представление

Инструменты реализации

Поток данных





Команды обработки данных

Java Stream Api

```
Query = [Command]

Command = Select [Column]
        | Distinct
        | From String
        | Join Query Expression JoinStrategy
        | Where Expression
        | OrderBy [(Column, OrderDirection)]
        | GroupBy [(Column, AggregationFunction)]
        | Limit Int
        | Offset Int
        | ConstructIndex (IndexName, [Column])
        | Alias String
```

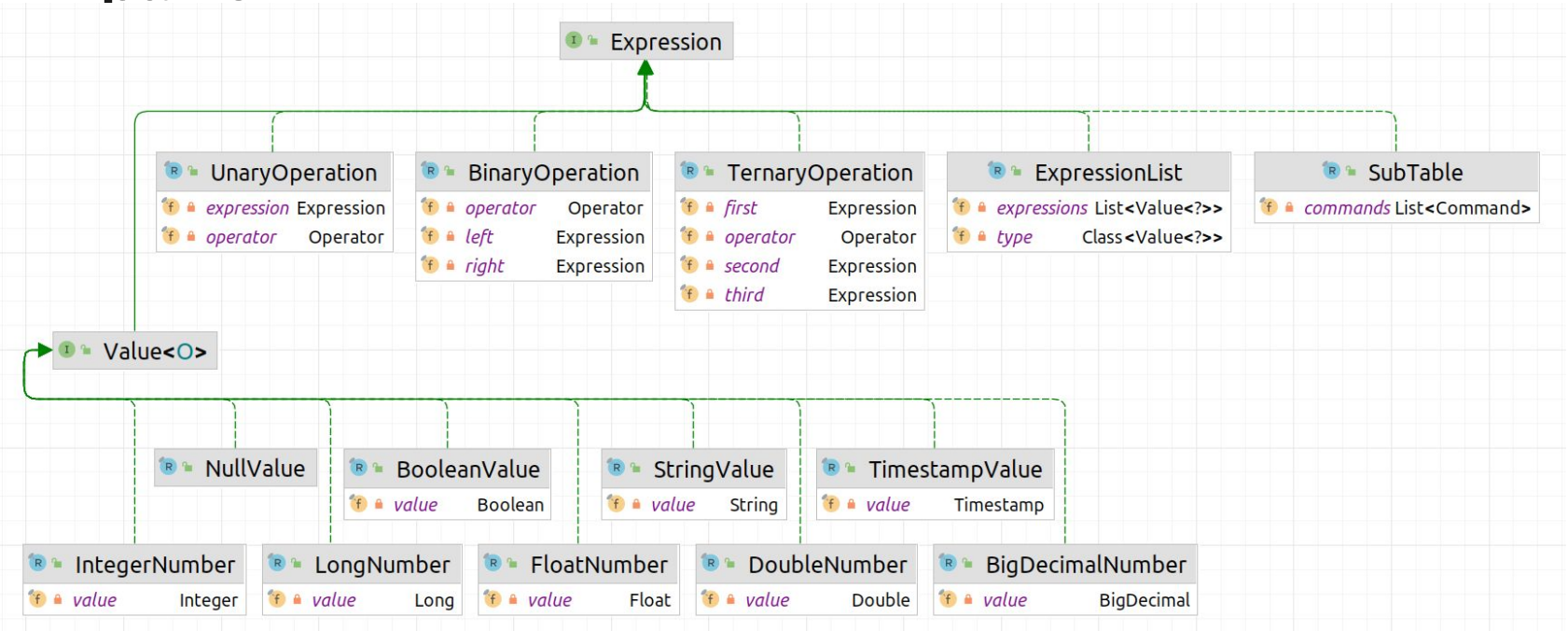
Выражения

```
var expression = Q.op(AND,
        Q.op(EQ,Q.column("c", "descriptor_id"), Q.column("cd", "id")),
        Q.op(EXISTS, Query.from("user").where(Q.op(GREATER, Q.column("user", "age"), new IntegerNumber(12))))
);
```

```
Operator =
  AND | BETWEEN | EXISTS | NOT | IN | OR | IS_NULL | LIKE |
  EQ | NOT_EQ | GREATER_OR_EQ | LESS_OR_EQ | GREATER | LESS |
  PLUS | MINUS | MULTIPLY | DIVISION | MOD
```

```
Expression = Value
           | UnaryOperation (Operator, Expression)
           | BinaryOperation (Operator, Expression, Expression)
           | TernaryOperation (Operator, Expression, Expression, Expression)
           | SubTable Query
           | ValueList [Value]
```

Выражения


Expression
UnaryOperation
expression Expression
operator Operator
BinaryOperation
operator Operator
left Expression
right Expression
TernaryOperation
first Expression
operator Operator
second Expression
third Expression
ExpressionList
expressions List<Value<?>>
type Class<Value<?>>
SubTable
commands List<Command>
Value<O>
NullValue
BooleanValue
value Boolean
StringValue
value String
TimestampValue
value Timestamp
IntegerNumber
value Integer
LongNumber
value Long
FloatNumber
value Float
DoubleNumber
value Double
BigDecimalNumber
value BigDecimal

Ленивость

```
Query.from("content").as("c")
    .where(Q.op(LESS, Q.column("c", "id"), new LongNumber(300L)))
    //.orderBy(List.of(Pair.of(Q.column("c", "id"), false)))
    //.limit(10)
```

```
| ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
| TOTAL COMPLEXITY: 662
| ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
| FROM[content] -
| ALIAS[c] -
| WHERE 662
|     Expression: COLUMN[c.id] < 300L
|     2 (expression complexity) * 331 (number of calculations) = 662 (total)
| ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

```
Query.from("content").as("c")
    .where(Q.op(LESS, Q.column("c", "id"), new LongNumber(300L)))
    //.orderBy(List.of(Pair.of(Q.column("c", "id"), false)))
    .limit(10)
```

```
| ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
| TOTAL COMPLEXITY: 22
| ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
| FROM[content] -
| ALIAS[c] -
| WHERE 22
|     Expression: COLUMN[c.id] < 300L
|     2 (expression complexity) * 11 (number of calculations) = 22 (total)
| LIMIT -
| ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

Лениво

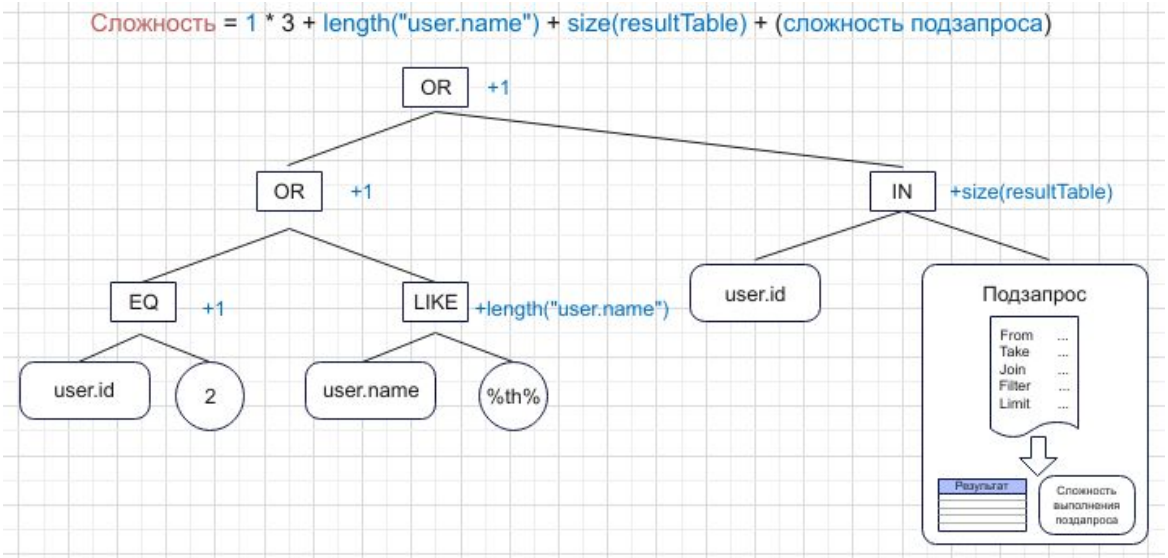
```
Query.from("content").as("c")
    .where(Q.op(LESS, Q.column("c", "id"), new LongNumber(300L)))
    //.orderBy(List.of(Pair.of(Q.column("c", "id"), false)))
    .limit(10)
```

```
| ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
| TOTAL COMPLEXITY: 22
| ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
| FROM[content] -
| ALIAS[c] -
| WHERE 22
|     Expression: COLUMN[c.id] < 300L
|     2 (expression complexity) * 11 (number of calculations) = 22 (total)
| LIMIT -
| ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

```
Query.from("content").as("c")
    .where(Q.op(LESS, Q.column("c", "id"), new LongNumber(300L)))
    .orderBy(List.of(Pair.of(Q.column("c", "id"), false)))
    .limit(10)
```

```
| ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
| TOTAL COMPLEXITY: 2582
| ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
| FROM[content] -
| ALIAS[c] -
| WHERE 662
|     Expression: COLUMN[c.id] < 300L
|     2 (expression complexity) * 331 (number of calculations) = 662 (total)
| ORDERBY 1920
| LIMIT -
| ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

Подсчет сложности



Пример работы. Результаты

```
var commands : List<Command> = new Query()
        .from("courses")
        .as("c1")
        .where(Q.op(EQ, Q.column("c", "id"), Q.op(MINUS, Q.column("c1", "id"), new LongNumber(4L))))
        .build();

var query : Query = new Query()
        .from("courses")
        .as("c")
        .where(
                Q.op(OR,
                        Q.op(EXISTS, new SubTable(commands)),
                        Q.op(NOT, Q.op(EXISTS, new SubTable(commands)))
                )
        );
```

```
| ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
| TOTAL COMPLEXITY: 840    CACHE INFLUENCE: 784 (1624 -> 840)
| ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
| FROM[courses] -
| ALIAS -
| WHERE 840
|     Expression: (EXISTS(SUB_TABLE[?])) OR (NOT(EXISTS(SUB_TABLE[?])))
|     60 (expression complexity) * 14 (number of calculations) = 840 (total)
|
|     | ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
|     | SUB TABLE COMPLEXITY: 56
|     | ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
|     | FROM[courses] -
|     | ALIAS -
|     | WHERE 56
|     |     Expression: COLUMN[c.id] = (COLUMN[c1.id] - 4L)
|     |     4 (expression complexity) * 14 (number of calculations) = 56 (total)
|     | ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
|
|     | ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
|     | SUB TABLE COMPLEXITY: 0    CACHE INFLUENCE: 56 (56 -> 0)
|     | ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
|     | FROM CACHED(0 -> 0)
|     | ALIAS CACHED(0 -> 0)
|     | WHERE CACHED(56 -> 0)
|     | ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
| ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

Результаты

Реализована учебная система, позволяющая задавать план запроса посредством “клиента системы”. Она позволяет исполнить план запроса и возвращает результат в виде реляционных данных и объекта, характеризующего сложность исполнения.

- Исследованы методы оценки сложности планов запросов и принципы обработки/организации данных промышленных реализаций стандарта SQL
- Определено множество команд над данными, позволяющее составлять планы запросов и специфицировать ход выборки/агрегации данных
- Реализован обработчик команд над данными, поддерживающий ленивость, кеширование, “простые” типы данных и операции над ними

Спасибо за внимание!