Агрегатор реляционных баз данных

Студент:
Чугунов Д. С.

Руководитель:
Коновалов А. В.

МГТУ имени Н.Э. Баумана
2020 год

Цель работы

Необходимо разработать и реализовать инструмент, позволяющий выполнять запросы SQL, которые объединяют данные из различных внешних источников в роли которых выступают реляционные базы данных.

Постановка задачи

Дано:

1. Входные данные: запрос SQL
2. Выходные данные: выборка данных

Требуется:

1. Разобрать запрос SQL
2. Получить необходимые выборки из различных баз данных
3. Объединить полученные выборки согласно исходному запросу

Входные данные

На входе программе подается запрос на выборку данных, описанный на языке SQL (SELECT).

Конструкция SELECT может содержать арифметические и логические выражения, ограничения на строки (WHERE). В табличном выражении могут использоваться различные виды соединений: CROSS JOIN, INNER JOIN, LEFT JOIN, RIGHT JOIN.

Пример исходного запроса

```
select a.name
     , b.name
     , c.name
  from psql.db.public.a as a
left join mysql.db.public.b as b
    on a.id = b.id
inner join psql.db.public.c as c
    on a.c_id = c.id;
```

Пример исходного запроса

Архитектура агрегатора баз данных

1. Лексический и синтаксический анализы запроса
2. Семантический анализ
3. Генерация запросов к различным РСУБД
4. Получение необходимых данных из внешних источников
5. Формирование результирующей выборки

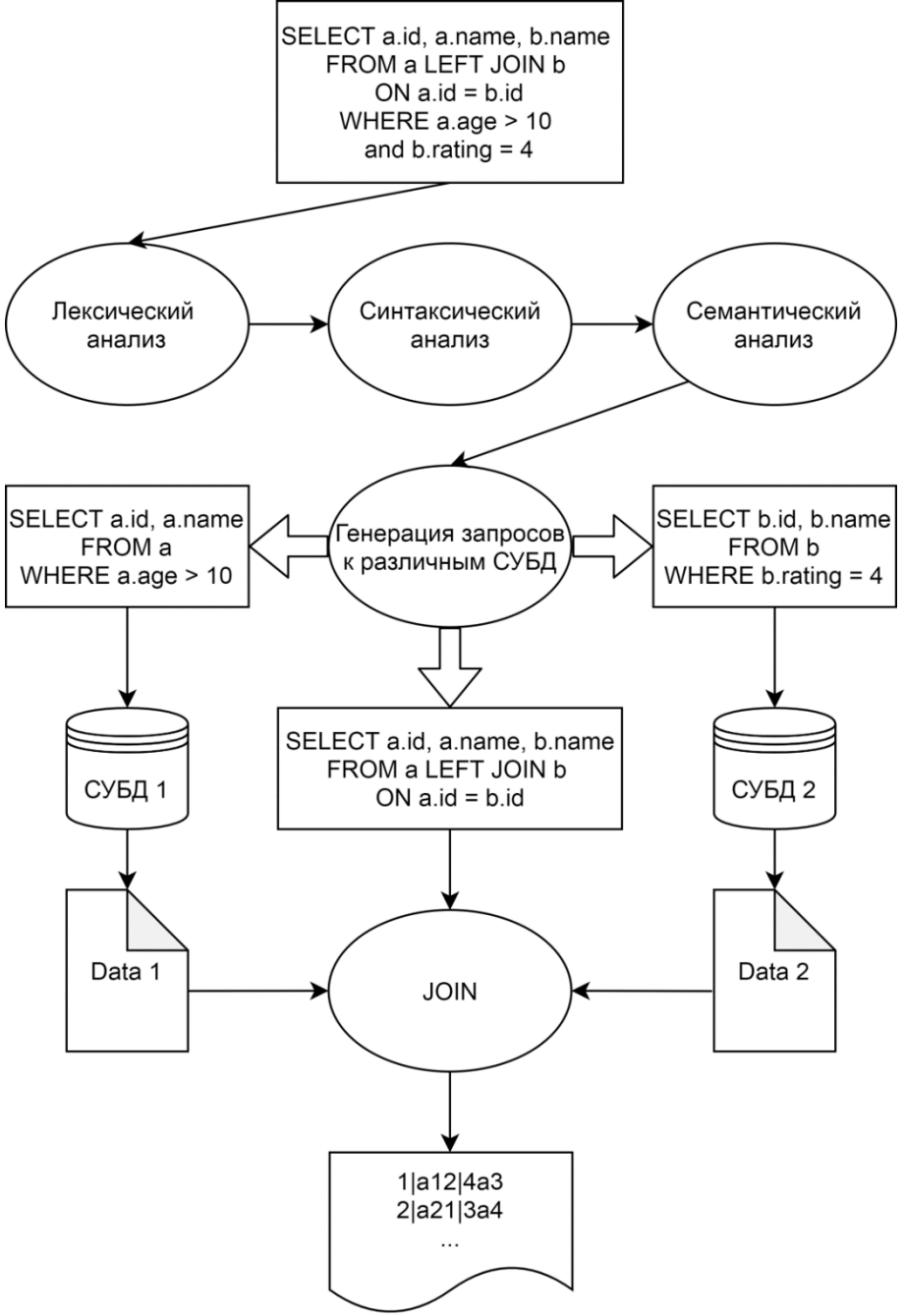


Схема работы агрегатора баз данных

Тернарная логика

- В РСУБД вместо стандартной булевой логики чаще всего используется тернарная логика, которая является расширением булевой логики.
- Помимо двух значений ИСТИНА и ЛОЖЬ добавляется еще одно — НЕИЗВЕСТНО.

A∧B		**A**		
		Л	**Н**	**И**
B	**Л**	Л	Л	Л
	Н	Л	Н	Н
	И	Л	Н	И

Конъюнкция

A∨B		**A**		
		Л	**Н**	**И**
B	**Л**	Л	Н	И
	Н	Н	Н	И
	И	И	И	И

Дизъюнкция

A	$\overline{A}$
Л	И
Н	Н
И	Л

Отрицание

СДНФ для тернарной логики

- Две эквивалентные в булевой логике формулы могут быть не эквивалентны в тернарной логике. Пример: $F_1 = A \vee \bar{B}$ и $F_2 = (\bar{A} \wedge \bar{B}) \vee (A \wedge \bar{B}) \vee (A \wedge B)$

A	**B**	F_1	F_2
Л	**Л**	И	И
Л	**Н**	Н	Н
Л	**И**	Л	Л
Н	**Л**	И	Н
Н	**Н**	Н	Н
Н	**И**	Н	Н
И	**Л**	И	И
И	**Н**	И	Н
И	**И**	И	И

Таблица истинности для F_1 и F_2

СДНФ для тернарной логики

- Операция «ЯВЛЯЕТСЯ» («IS») – бинарная операция, возвращает ИСТИНУ, когда два операнда идентичны, иначе возвращает ЛОЖЬ.
- Алгоритм:

1. Находятся все комбинации, при которых результирующая функция является ИСТИНОЙ.
2. На основе этих комбинаций составляются элементарные конъюнкции, элементами которых являются операторы $A\ is\ e$, где A – свободная переменная, а e – значение переменной в комбинации.

Упрощение СДНФ

- Из СДНФ тернарной логики можно получить форму более подходящую для дальнейшего рассмотрения.

1. $(A\ is\ \text{И} \wedge B\ is\ \text{Л}) \vee (A\ is\ \text{И} \wedge B\ is\ \text{И}) \equiv (A\ is\ \text{И}) \wedge ((B\ is\ \text{Л}) \vee (B\ is\ \text{И}))$
2. $(A\ is\ \text{Л} \wedge B\ is\ \text{И}) \vee (A\ is\ \text{И} \wedge B\ is\ \text{И}\ ) \equiv \overline{(A\ is\ \text{Н})} \wedge (B\ is\ \text{И})$
3. $(A\ is\ \text{Л} \wedge B\ is\ \text{Л}) \vee (A\ is\ \text{Н} \wedge B\ is\ \text{Л}) \vee (A\ is\ \text{И} \wedge B\ is\ \text{Л}) \equiv (B\ is\ \text{Л})$

СДНФ в WHERE

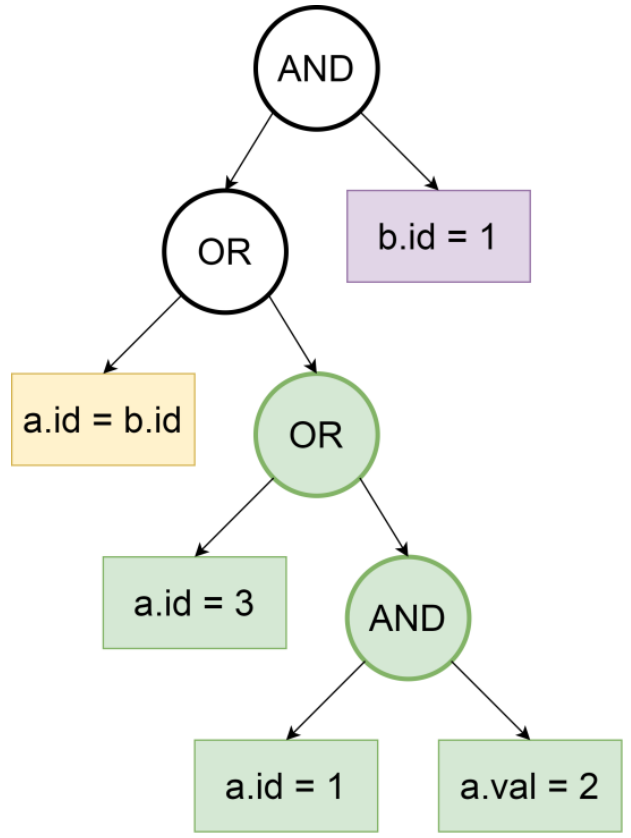
- В условии WHERE особую роль играют логические условия, которые используют только одну таблицу. Такие условия можно перенести на этап запроса данных из таблицы.
- Приведение условия в СДНФ позволяет находить вложенные условия, которые используют только одну таблицу.
- В качестве свободной переменной выступает не колонка, а максимальное поддерево дерева разбора, которое использует только одну таблицу.

Свободные переменные в условии

1. **(**a.id=b.id **or (**
   a.id=3 **or**
   (a.id=1 **and** a.val=2**)**
   )) and b.id=1

2. A ::= a.id=b.id
   B ::= a.id=3 or (a.id=1 and a.val=2)
   C ::= b.id=1

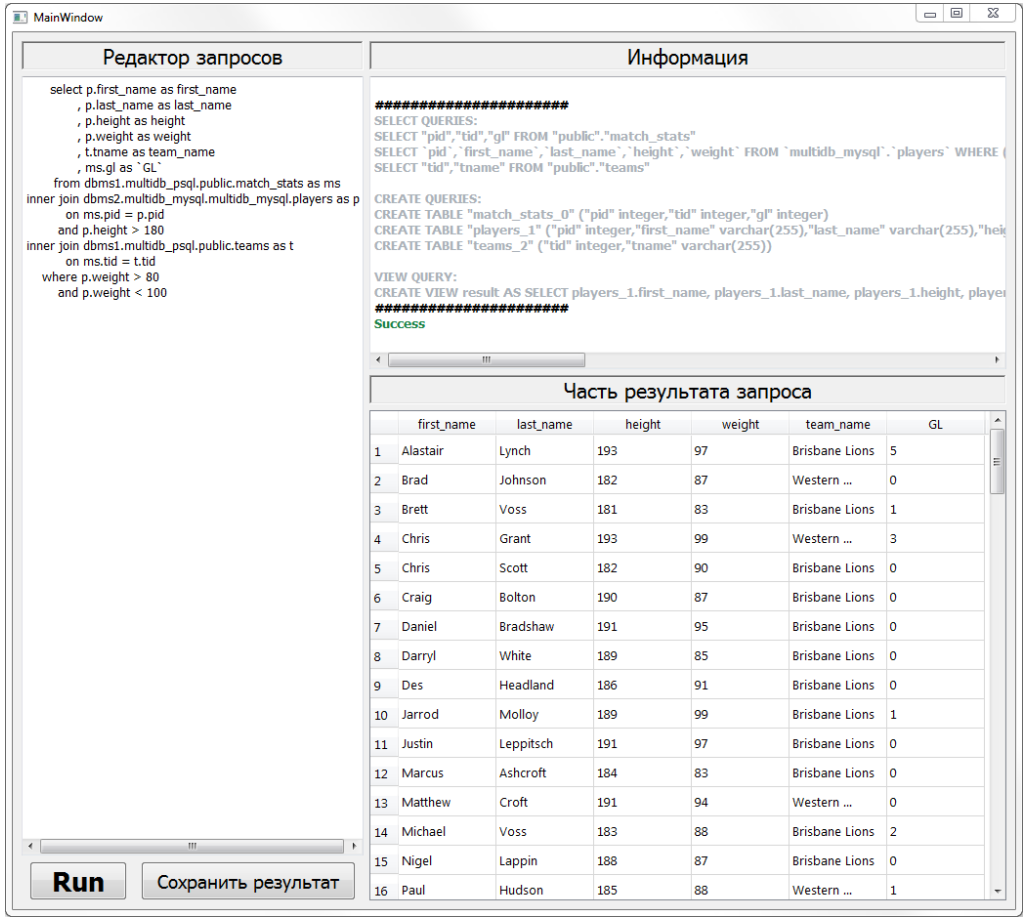
3. (A or B) and C



Дерево разбора логического выражения

Графический пользовательский интерфейс

- Левая часть – редактор запросов
- Правый верхний угол – отладочная информация: ошибки, сообщения об успехе, запросы к внешним источникам
- Правый нижний угол – часть результирующей выборки
- Кнопка «Run» – выполнение запроса
- Кнопка «Сохранить результат» – сохранение результата в файле базы данных SQLite.



Тестирование

Тестирование элементарной конъюнкции

```
-- Исходный запрос
    select a.val, b.val
      from dbms.db.public.a as a
inner join dbms.db.public.b as b
        on a.id = b.id
       and a.val > 1
       and b.val > 2

-- Запросы к внешним источникам
select a.id, a.val
  from public.a
 where (a.val > 1) is TRUE

select b.id, b.val
  from public.b
 where (b.val > 2) is TRUE
```

Тестирование отрицания элементарной дизъюнкции

```
-- Исходный запрос
    select a.val, b.val
      from dbms.db.public.a as a
inner join dbms.db.public.b as b
        on not (a.id <> b.id
        or a.val <= 1
        or b.val <= 2)

-- Запросы к внешним источникам
select a.id, a.val
  from public.a
 where (a.val <= 1) is FALSE

select b.id, b.val
  from public.b
 where (b.val <= 2) is FALSE
```

Тестирование

Тестирование вложенных конструкций

```
-- Исходный запрос
    select a.val, b.val
      from dbms.db.public.a as a
inner join dbms.db.public.b as b
        on a.id == b.id
        and (a.val > 10 or a.val / 2 < 10)

-- Запросы к внешним источникам
select a.id, a.val
  from public.a
 where (
       a.val > 10
    or a.val / 2 < 10
     ) is TRUE

select b.id, b.val
  from public.b
```

Тестирования сложного запроса с использованием различных логических конструкций

```
-- Исходный запрос
    select a.val, b.val
      from dbms.db.public.a as a
inner join dbms.db.public.b as b
        on not (
           not (
                 a.id = b.id
             and b.value > 1
          )
          or (
               a.val <= 10
            or b.val == 2
          ))

-- Запросы к внешним источникам
select a.id, a.val
  from public.a
 where (a.val <= 10) is FALSE

select b.id, b.val
  from public.b
 where (b.val > 1) is TRUE
   and (b.val == 2) is FALSE
```

Заключение

- В процессе выполнения дипломной работы был разработан и реализован инструмент, позволяющий выполнять запросы SQL, которые включают в себя обращение к нескольким внешним источникам (в роли которых выступают РСУБД)
- Помимо этого, был разработан алгоритм приведения формул тернарной логики в нормальную форму. Использование этой формы позволяет декомпозировать исходное логическое условие. В ходе тестирования была установлена корректность данного алгоритма.