Определения и обозначения

План выполнения SQL запроса – конкретный набор операций, которые необходимо выполнить для получения результата запроса.

Планировщик SQL запросов в реляционной СУБД – часть СУБД, которая отвечает за создание итогового плана выполнения запроса.

Содержание

1 Аналитический раздел

1.1 Выбор СУБД

В современном мире при нынешних обстоятельствах на фоне политических конфликтов необходимо ПО с открытым исходным кодом. К таким относятся: PostgreSQL, MySql. Сравнение основных характеристик приводится в таблице 1.1 [2].

Таблица 1.1 – Сравнение основных характеристик СУБД

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

PostgreSQL является стабильной СУБД, имеет хорошо структурированные данные, поэтому именно она и выбирается в качестве основной.

1.2 Этапы обработки SQL-запроса

Подход к выполнению запроса СУБД на примере PostgreSQL представлен на рисунке 1.1.

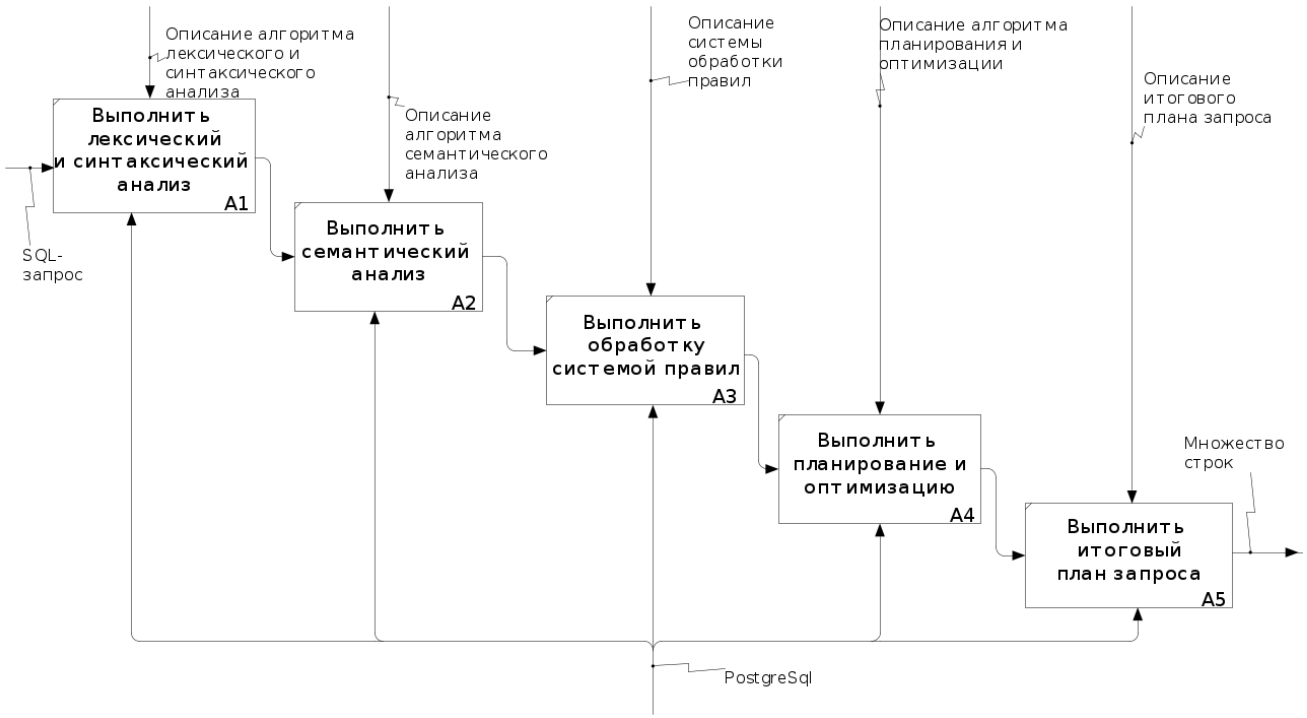

Рисунок 1.1 – Выполнение запроса в PostgreSQL.

Основными шагами, которые выполняются при выполнении SQL-запроса в реляционных СУБД являются следующие [4].

1. Лексический и синтаксический анализ. На данном этапе входная строка пользователя обрабатывается лексическим и синтаксическим анализаторами; в результате строится дерево запроса.

2. Семантический анализ. Полученное на предыдущей стадии дерево запроса дополняется различного рода метаинформацией: системными идентификаторами таблиц, типами и порядковыми номерами запрашиваемых полей, перечнем соединяемых таблиц и т.д.

3. Обработка системой правил. Далее выполняется поиск в системных каталогах правил, применимых к дереву запроса, и при обнаружении подходящих выполняются преобразования, которые описаны в теле найденного правила. Примером такого преобразования является замена представлений (*виртуальных таблиц*) на обращение к базовым таблицам из определения представления.

4. Планирование и оптимизация. На вход планировщику поступает структура с деревом запроса. Осуществляется выбор наиболее эф-

фективного пути выполнения этого запроса с точки зрения имеющихся оценок затрат и статической информации на момент выполнения. После выбора оптимального метода доступа к данным, конечный вариант преобразуется в полноценный план запроса и передается исполнителю.

5. Выполнение итогового плана запроса. Исполнителем осуществляется рекурсивный обход по дереву плана: *сканируются отношения*, выполняется *сортировка* и *соединения*, вычисляет *условия фильтра* и др. После выполненных этапов возвращается результирующее множество строк.

1.3 План выполнения запроса

План выполнения запроса действует как дерево инструкций, которым должен следовать механизм выполнения запроса для получения результатов. Он показывает, как будут сканироваться таблицы; если необходимо связывание нескольких таблиц, то какой алгоритм будет выбран для объединения считанных строк.

На примере PostgreSQL работа планировщика запросов для одной таблицы выглядит следующим образом [3].

1. Выполнение предварительной обработки.

2. Оценка всевозможных путей доступа к данным. Оцениваются затраты на последовательный доступ (*seq scan*), сканирование индексов (*index scan*), *bitmap scan*; затем выполняется сортировка (*sort*) или присоединение данных (*join*).

3. Выбор кратчайшего пути по затратам.

При увеличении числа таблиц, участвующих в запросе, к основному алгоритму добавляются шаги.

- Уровень 1. Найти кратчайший путь выполнения запроса для каждой таблицы.

- Уровень 2. Получить самый кратчайший путь для каждой комбинации, в которой которой выбирается две таблицы из всех.
- Уровень 3. Продолжить ту же обработку, пока в результате число таблиц не станет равным текущему уровню.

Процесс получения «дешевого» доступа к данным приведен на рисунке 1.2.

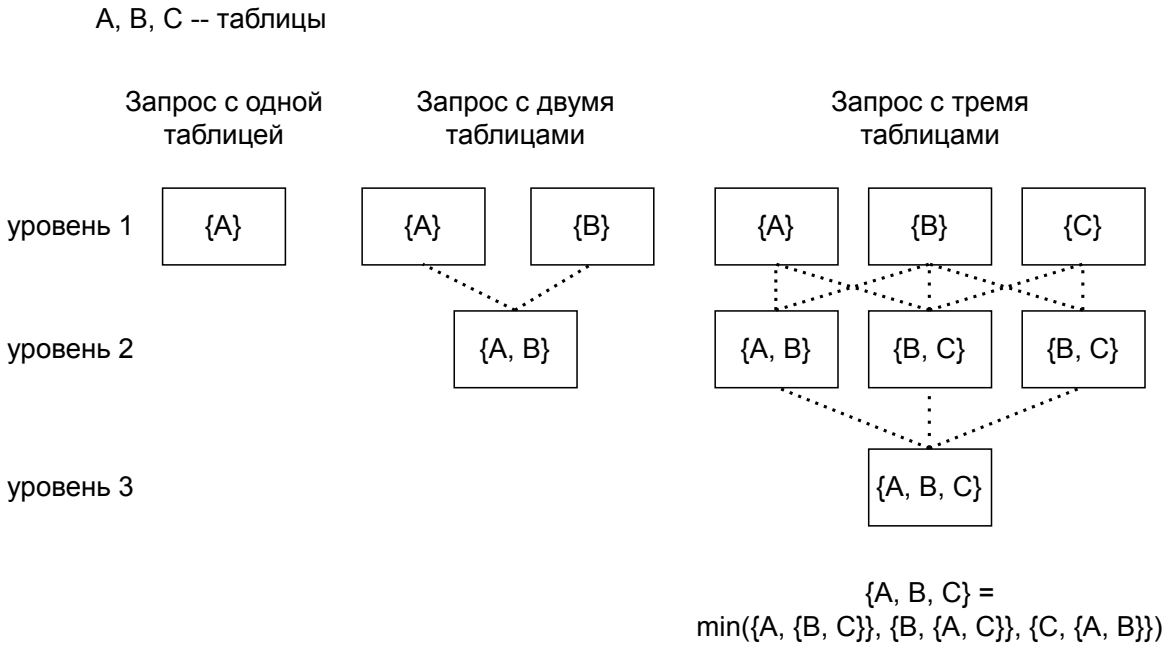

Рисунок 1.2 – Получение «дешевого» доступа к данным.

Для получения оптимального дерева плана запроса, планировщик должен рассмотреть комбинации всех индексов и возможности методов объединения. При увеличении числа используемых таблиц может наступить «комбинаторный взрыв». Таким образом, при количестве таблиц, большем 12, используются *генетические алгоритмы.*

1.4 Выбор БД

1.4.1 Northwind

1.4.2 AdventureWorks

В качестве базы данных была выбрана «Adventure Works Cycles» – фиктивная компания, разработанная Microsoft для моделирования бизнес-процесса [5]. Данная фирма «является» крупной и многонациональной, которая производит и продает металлические и композитные велосипеды на коммерческих рынках Северной Америки, Европы, Азии.

По своей структуре база данных «AdventureWorks» является сложной: состоит из множества схем отношений, соединенных друг с другом различными связями. Диаграмма базы данных этой компании приведена в приложении А.

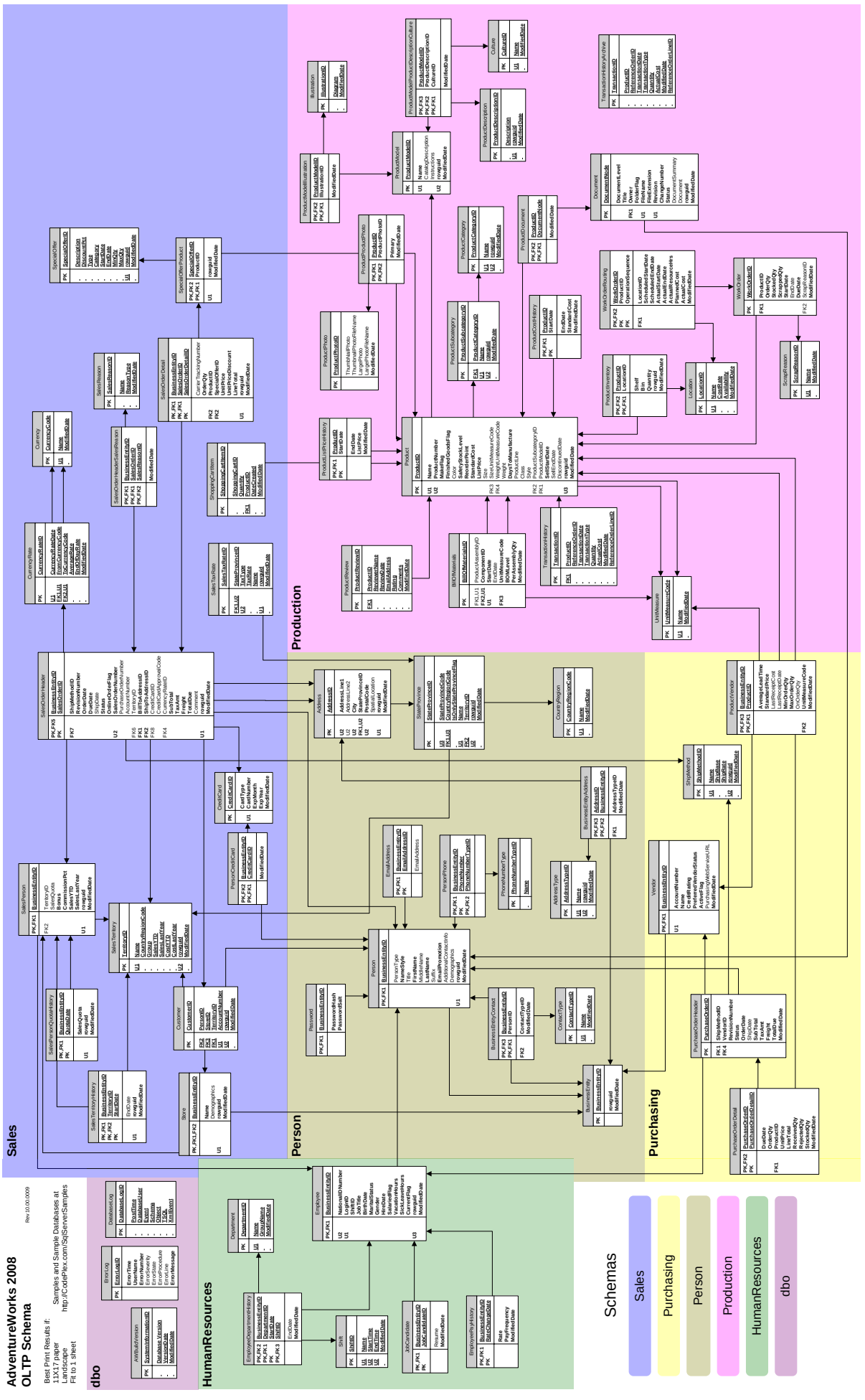

Рисунок 1.3 – Диаграмма БД «AdventureWorks».

Список литературы

[1] PostgreSQL 14.2 Documentation [Электронный ресурс]. URL: https://www.postgresql.org/docs/14/index.html (дата обращения: 20.03.2022)

[2] DB-Engines Ranking [Электронный ресурс]. URL: https://db-engines.com/en/ranking (дата обращения: 21.03.2022)

[3] The Internals of PostgreSQL : Chapter 3 Query Processing [Электронный ресурс]. URL: https://www.interdb.jp/pg/pgsql03.html

[4] Пантилимонов М. В., Бучацкий Р. А., Жуйков Р. А. Кэширование машинного кода в динамическом компиляторе SQL-запросов для СУБД PostgreSQL //Труды Института системного программирования РАН. – 2020. – Т. 32. – №. 1. – С. 205-220.

[5] Adventure Works Cycles Business Scenarios [Электронный ресурс]. URL: https://docs.microsoft.com/en-us/previous-versions/sql/sql-server-2008/ms124825(v=sql.100)