а) поле – название поля, над которым осуществляется соответствующая операция,

б) таблица - таблица в которой находится данное поле (в запросе могут использоваться несколько таблиц),

в) сортировка – она может быть по возрастанию, по убыванию, либо не использоваться,

г) вывод на экран – содержит чекбокс (если включен - поле показывается, в обратной ситуации - нет),

д) строки условий отбора (3-7 строк) – могут содержать логические операции типа =,<,>,<=,>=,<> или вложенные запросы.

Примерами могут послужить широко известные представители MS Office– MS Query и Designer MS Access.

Литература

1. Соколов С.А. Сборник тезисов студенческой научно техничекой конференции. Вертикальный QBE., Мн., ротопринт. Белорусский государственный университет информатики и радиоэлектроники,1999.
2. Бойко В.В. Организация и использование баз данных. –М.,Мир, 1991.
3. Грабер М. Введение в SQL. –М., Мир, 1996.

АЛГОРИТМИЧЕСКАЯ МУЛЬТИЯЗЫЧНОСТЬ. ПРОБЛЕМЫ. ПУТИ РЕАЛИЗАЦИИ.

Соколов С.А.

Белорусский государственный университет информатики и радиоэлектроники

г.Минск, ул.Платонова 39, к.807

Аннотация: В данной статье рассмотрены проблемы алгоритмической мультиязычности и пути реализации. Приведены основные причины тормозящие развитие мультиязычности.

Ключевые слова: язык программирования, компилятор, интерпритатор, мультиязычность.

Алгоритмическая мультиязычность – это свойство компилятора или интерпретатора, заключающееся в поддержке процесса разработки программного обеспечения (ПО) с применением более одного языка программирования.

Причина возникновения нового интереса к проблеме алгоритмической мультиязычности обусловлена накоплением большого объема профессиональных библиотек к столь не популярных сейчас COBOL, FORTRAN, ALGOL.

Существует масса других умирающих языков программирования. Часть ПО разрабатывалась для «больших» машин, которые уже исчезли из компьютерных центров.

Для Windows наиболее популярными языками программирования можно считать C/C++, PASCAL, BASIC и JAVA. Для Unix в наибольшей степени характерны C/C++, PERL, FORTRAN, JAVA. Медленно стал появляться интерес к языку COBOL.

Подобная идея по созданию алгоритмической мультиязычности возникала в 60-х годах, но дальше конвертаторов типа C в PASCAL не пошла. Сказались следующие основные причины:

1. Не появилось достаточно удачного коммерческого продукта, который бы агрессивно внедрялся и рекламировался на рынке программного обеспечения так, как Visual C++ от Microsoft или Java от Sun. В качестве языкового стандарта со временем стал выступать язык C/C++. Если подробно изучать синтаксис большинства языков программирования, созданных после создания C/C++, то очень четко просматривается попытка создания очередного клона C/C++.
2. Нежелание крупных фирм терять возможность поставки пользователям нескольких продуктов вместо одного универсального. Это бы резко снизило прибыль от продаж и еще более усилило конкуренцию на рынке программного обеспечения.
3. Сложность реализации эффективного алгоритма и наличие большого количества клонов с несовместимыми языковыми расширениями. Обычно фирмы стремятся поддерживать только синтаксис самого языка. В остальном фирмы руководствуются собственными представлениями о программировании.
4. Сложность совместимости типизируемых языков программирования с не типизированными. Например, BASIC был изначально создан как не типизируемый язык (Visual Basic имеет типизируемые переменные), а PASCAL исключительно как типизируемый.
5. Появление новой модели развития коммерческих продуктов. Вместо поддержки нескольких языков программирования на одной платформе в одном средстве предлагается вариант одного языка программирования для разных платформ. Эту модель используют для продвижения на рынке языков C/C++ и JAVA.

После рассмотрения причин, попробуем сформулировать основные реомендации к будущему универсальному языковому транслятору:

1. поддержка максимально возможного количества алгоритмических языков;
2. полная поддержка объектно-ориентированного программирования;
3. возможность наращивания типов конвертируемых программных конструкций.

Реализация алгоритмической мультиязычности фактически сводится к созданию промежуточного языка. Применение этого промежуточного кода резко упростило бы поддержку нескольких языков программирования, т.к. текст программ конвертировался в промежуточных язык или код. После конвертации имелась бы возможность обработки только этого кода. Подобный подход использовался разработчиками JAVA.

Возможны несколько основных способов поддержки мультиязычности:

1. автоматическое распознавание лексем по ходу анализа кода программы
2. обработка опций языкового блока (команды – старт/финиш нового языка)
3. определение принадлежности лексемы к языку определяется расширением файла

При реализации конвертаторов необходимо учитывать особенности конкретного языка программирования. С проблемой несовместимости правил различных языков программирования сталкиваешься при попытке совмещения разноязычных фрагментов программ. Некоторые основополагающие правила одного языка оказываются в противоречии с правилами другого.

Наиболее типичные различия языков программирования:

1. возможности совпадения названий файлов и классов в них;
2. поиск файлов библиотек;
3. распознавание больших и строчных символов;
4. реализация обработки ошибок;
5. особенности отладчика и его опции.

Сложности возникают с декларациями переменных и комментариями, с их выравниванием по отношению к коду. В ряде случаев при конвертации происходит искажение их расположения по отношению к коду.

Литература

1. Грищук И.П. Компиляторы. –М., Высшая школа, 1992.
2. Грис Д. Конструирование компиляторов для цифровых вычислительных машин. –М., Издательство «МИР», 1975.

ОРГАНИЗАЦИЯ ГРУППОВЫХ ЗАПРОСОВ К МНОГОТАБЛИЧНЫМ ДАННЫМ

Галерштейн В.Е., Ревотюк М.П.
Белорусский государственный университет информатики и радиоэлектроники
Минск, ул. П.Бровки, 6, кафедра ИТАС

Предлагается метод оптимизации групповых запросов к многотабличным данным, обсуждаются условия его применимости и оценка выигрыша в скорости доступа по сравнению с традиционными подходами. Рассматриваются способы его реализации при разработке новых СУБД.

Ключевые слова: базы данных, методы доступа к данным, оптимизация запросов.

В реальных системах обработки информации часто возникает задача выборки данных из нескольких таблиц данных, связанных по некоторому ключу. Например, в системах семейства xBase подобная связь устанавливается оператором SET RELATION, а в SQL - системах - предикатом выражения WHERE оператора SELECT [8,10]. Организация выборки записей из группы таблиц наиболее эффективно строится, как правило, на основе индексных файлов. Каждой таблице ставится в соответствие один индексный файл.

Известно, что наиболее распространенная организация индексных файлов - деревья Байера (B+ деревья) [2,3,6]. Среднее время доступа T к искомой записи в B+ дереве [3,4]:

$$T \sim = \log_a n, \quad (1)$$

где n - число записей таблицы, а - число индексов на одной странице B+ дерева.

Если применять традиционную схему последовательной обработки всех M связанных таблиц, то среднее время обработки T(M) составит:

$$T(M) = \Sigma \log_a n_i, \; i = 1 \ldots M. \quad (2)$$

Логарифмический закон времени доступа к индексированной таблице (1) побуждает предложить следующую схему организации групповых запросов. Построим индексный файл, где выражения ключа включают ссылку на идентификатор таблицы, а также флаг уникальности ключа для данной таблицы: