ЧЕРЕПОВЕЦКИЙ
ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ

Лекция 9. Оптимизация запросов

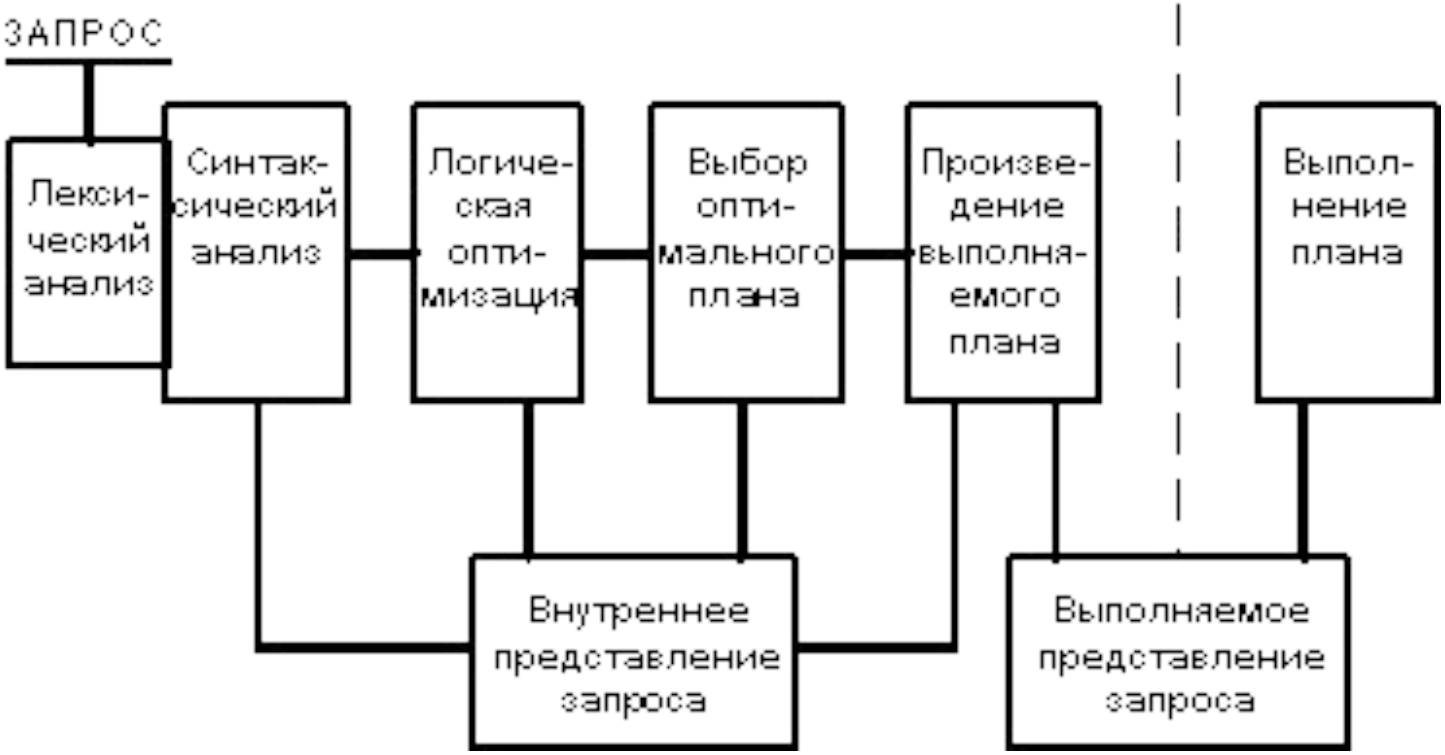
Под ***оптимизацией запросов в реляционных СУБД***, имеют в виду такой способ обработки запросов, когда по *начальному представлению* запроса путем его преобразований *вырабатывается процедурный план его выполнения*, наиболее оптимальный при существующих в базе данных управляющих структурах.

Соответствующие преобразования начального представления запроса выполняются специальным компонентом СУБД – оптимизатором.

Оптимальность производимого им плана запроса носит условный характер: план оптимален в соответствии с критериями, заложенными в оптимизатор.

Целью выполнения процедуры обработки запросов является преобразование запроса, оформленного на языке высокого уровня (обычно это язык SQL), в правильную и эффективную последовательность операций, выраженную на языке низкого уровня, реализующем операции реляционной алгебры.
Затем подготовленный план обработки запроса выполняется с целью выборки требуемых данных.

Вопрос 1. Общая схема обработки запроса



Первая фаза: запрос, заданный на языке запросов, подвергается *лексическому и синтаксическому анализу*. При этом вырабатывается его внутреннее представление. Внутреннее представление запроса используется и преобразуется на следующих стадиях обработки запроса.

Вторая фаза: запрос во внутреннем представлении подвергается логической оптимизации. Применяются различные преобразования, "улучшающие" начальное представление запроса:

- ***эквивалентные***, после проведения которых получается внутреннее представление, семантически эквивалентное начальному;
- ***семантические,*** получаемое представление не является семантически эквивалентным начальному.

После выполнения второй фазы обработки запроса его внутреннее представление остается непроцедурным, хотя и является в некотором смысле более эффективным, чем начальное.

Третья фаза:

- генерация на основе информации, которой располагает оптимизатор, набора альтернативных процедурных планов выполнения данного запроса в соответствии с его внутреннем представлением, полученным на второй фазе.
- оценка стоимости выполнения запроса. При оценках используется статистическая информация о состоянии базы данных, доступная оптимизатору.
- из полученных альтернативных планов выбирается наиболее дешевый, и его внутреннее представление теперь соответствует обрабатываемому запросу.

Четвертая фаза:
формируется выполняемое представление плана. Выполняемое представление плана может быть:

-программой в машинных кодах,если система ориентирована на компиляцию запросов в машинные коды (System R),

- машинно-независимым, если система ориентирована на интерпретацию запросов (INGRES).

Пятая фаза: происходит реальное выполнение запроса.

Вопрос 2. Синтаксическая оптимизация запросов

На этапе логической оптимизации производятся эквивалентные преобразования внутреннего представления запроса, которые "улучшают" начальное внутреннее представление. Характер "улучшений" связан со спецификой общей организации оптимизатора.

Поэтому трудно привести полную характеристику и классификацию методов логической оптимизации. Мы ограничимся несколькими примерами, а также рассмотрим один частный, но важный класс логических преобразований, касающихся сложных запросов, выраженных на языке SQL.

2.1. Простые логические преобразования запросов

преобразования предикатов к каноническому представлению.
Предикат имеет вид *"арифметическое выражение op арифметическое выражение",* где "op" - операция сравнения, а арифметические выражения левой и правой частей в общем случае содержат имена полей отношений и константы.

Канонические представления могут быть различными для предикатов разных типов:
- *предикат включает только одно имя поля*, то его каноническое представление может, например, иметь вид *"имя поля op константное арифметическое выражение"*.
Пример: для предиката имеет вида (a+5)(A op 36)

каноническим представлением такого предиката может быть A op 36/(a+5).

- *предикат включает в точности два имени поля разных отношений* (или двух разных вхождений одного отношения).

Его каноническое представление может иметь вид "*имя поля op арифметическое выражение*", где арифметическое выражение в правой части включает только константы и второе имя поля (это тоже форма, полезная для выполнения следующего шага оптимизации, - предикат соединения; особенно важен случай эквисоединения, когда op - это равенство).

Пример: предикат имеет вид 5(A-a(B)) op b

его каноническое представление - A op (b+a(B)/5.

- *предикаты более общего вида.*

Цель преобразования - каноническое представление вида *"арифметическое выражение op константное арифметическое выражение",* где выражения правой и левой частей также приведены к каноническому представлению;

- приведение к каноническому виду логического выражения, задающего условие выборки запроса – дизъюнктивной или конъюнктивной нормальной форме.

При этом можно производить поиск общих предикатов и упрощать логическое выражение за счет, например, выявления конъюнкции взаимно противоречащих предикатов.

Пример: фрагмент логического выражения 1/4(A>5)AND(A<5)1/4 можно заменить на 1/4FALSE1/4.

Пример, фрагмент логического выражения 1/4(A>B)AND(B=5)1/4 можно заменить на 1/4(A>5) 1/4.

?: в чем преимущество полученного представления.

2.2 Преобразования запросов с изменением порядка реляционных операций

В оптимизаторах распространены логические преобразования, связанные с изменением порядка выполнения реляционных операций.
Примером соответствующего правила преобразования в терминах реляционной алгебры может быть следующее (A и B - имена отношений):
(A JOIN B) WHERE restriction-on-A AND restriction-on-B

эквивалентно выражению
(A WHERE restriction-on-A) JOIN (B WHERE restriction-on-B).

Замечание: часто реляционная алгебра используется в качестве основы внутреннего представления запроса.
Причины:
- использование РА в качестве универсального внутреннего языка;
- РА более проста языка SQL.

2.3 Приведение запросов со вложенными подзапросами к запросам с соединениями

Возможность использовать в логическом условии выборки предикаты, содержащие вложенные подзапросы – основное отличие SQL от РА.
Особенности:
- глубина вложенности может быть произвольной;
- предикаты могут обладать различной семантикой;
- общим алгоритмом выполнения запроса является вычисление вложенного подзапроса всякий раз при вычислении значения предиката.

Поэтому естественно стремиться к такому преобразованию запроса, содержащего предикаты со вложенными подзапросами, которое сделает семантику подзапроса более явной.

Предикаты можно разбить на следующие четыре группы:

- *простые предикаты*.

Это предикаты вида Ri.Ck op X, где X - константа или список констант, и op - оператор скалярного сравнения (=, !=, >, >=, <, <=) или оператор проверки вхождения во множество (IS IN, IS NOT IN);

- *предикаты со вложенными подзапросами*.

Это предикаты вида Ri.Ck op Q, где Q - блок запроса, а op может быть таким же, как для простых предикатов. Предикат может также иметь вид Q op Ri.Ck. В этом случае оператор принадлежности ко множеству заменяется на CONTAINS или DOES NOT CONTAIN.

- *предикаты соединения*.

Это предикаты вида Ri.Ck op Rj.Cn, где Ri != Rj и op - оператор скалярного сравнения.

- *предикаты деления*.

Это предикаты вида Qi op Qj, где Qi и Qj - блоки запросов, а op может быть оператором скалярного сравнения или оператором проверки вхождения в множество.

Приведенная классификация является упрощением реальной ситуации в SQL.

Каноническим представлением запроса на n отношениях называется запрос, содержащий n-1 предикат соединения и не содержащий предикатов со вложенными подзапросами. Фактически, каноническая форма - это алгебраическое представление запроса.

Пример

```
SELECT Ri.Ck FROM Ri
   WHERE Ri.Ch IS IN
                          ( SELECT Rj.Cm FROM Rj WHERE Ri.Cn = Rj.Cp)
```

эквивалентно

```
SELECT Ri.Ck FROM Ri, Rj
   WHERE  Ri.Ch = Rj.Cm AND Ri.Cn = Rj.Cp
```

Разумность таких преобразований обосновывается тем, что оптимизатор получает возможность выбора большего числа способов выполнения запросов

Вопрос 3. Семантическая оптимизация запросов

Рассмотренные преобразования запросов основывались на семантике языка запросов, но в них не использовалась семантика базы данных. На самом же деле, при каждой истинно реляционной базе данных хранится и некоторая семантическая информация, определяющая, например, целостность базы данных.

3.1. Преобразования запросов на основе семантической информации

Использование представлений.
Представления хранятся в каталогах базы данных во внутренней форме, получаемой после выполнения грамматического разбора (а также, возможно, логической оптимизации) соответствующего запроса. При обработке запроса над представлением до выполнения фазы логической оптимизации производится слияние внутренних форм запроса и представления.

Пример.

Представление DEFINE VIEW V (C2) AS SELECT C1 FROM R WHERE C1 > 6.
Запрос SELECT C2 FROM V WHERE C2 < 6.

При слиянии внутренних форм представления и запроса
SELECT C1 FROM R WHERE C1 < 6 AND C1 >6.
На фазе логической оптимизаци устанавливается эквивалентность запросу
SELECT C1 FROM R WHERE FALSE,
из чего следовало бы, что результат запроса пуст.

Использование информации об ограничениях целостности

Пусть база данных состоит из отношений EMP и DEPT.
Схема отношения *Служащие* - EMP (EMP#, EMPNAME, DEPT#);
Схема отношения *Отделы* - DEPT (DEPT#, EMPCOUNT),
Ограничение целостности *ASSERT B IMMEDIATE ON EMP: EMP.DEPT# = 6* означает запрещение занесения, удаления и модификации кортежей в отношении EMP со значением поля DEPT#, отличным от 6.

Запрос
DELETE EMP WHERE EMPNAME = 'Brown'.
При компиляции запроса
- не учитываются наличие ограничения целостности. Тогда алгоритм выполнения запроса будет следующий: последовательно выбирать кортежи, у которых значением поля EMPNAME является 'Brown'; проверять удовлетворение очередного кортежа ограничению целостности, и если проверка удовлетворительна, удалить кортеж.

- ограничения целостности учитываются. Тогда можно слить внутренние формы запроса и соответствующих ограничений целостности, а потом произвести совместную оптимизацию. Получим внутреннюю форму, эквивалентная запросу
DELETE EMP WHERE EMPNAME = 'Brown' AND DEPT# = 6.

Д: -при выполнении такого запроса не понадобятся дополнительные вызовы проверок ограничений целостности второго типа, поскольку они все уже включены в логическое условие выполнения операции удаления.
- оптимизатор получает большую свободу в выборе способа выполнения запроса.

3.2. Использование семантической информации при оптимизации запросов

Если семантическая оптимизация имеет дело только со знаниями, представленными в виде ограничений целостности базы данных, то при семантической оптимизации производится множество преобразованных внутренних представлений запроса; при каждом преобразовании используется некоторый поднабор ограничений целостности.

Пусть есть два ограничения целостности A и B с логическими условиями F1 и F2 и обрабатывается запрос с условием выборки F, то в ходе семантической оптимизации будут получены внутренние представления, эквивалентные запросам с условиями F, F AND F1, F AND F2 и F AND F1 AND F2.

Пусть, например, отношение EPM расширено полями STATUS и SALARY. Наложено ограничение целостности: ASSERT A ON EMP: IF SALARY > 40.000 THEN STATUS = 'Manager'.

Предположим, что обрабатывается запрос
SELECT EMPNAME, STATUS FROM EMP WHERE SALARY = 20.000.

Если не учитывать импликативного характера ограничения целостности, то по общим правилам будет произведено слияние внутренних представлений запроса и ограничения целостности, которое заведомо ничего не даст. Если же рассматривать ограничение целостности как правило преобразования, а левую часть импликации как условие применения правила, то слияние производиться и не будет, что в общем случае сэкономит время обработки запроса.

Для запроса
SELECT EMPNAME, STATUS FROM EMP WHERE SALARY > 40.000 правило преобразования применимо и приводит к семантически эквивалентному запросу
SELECT EMPNAME, STATUS FROM EMP WHERE STATUS = 'Manager', выполнение которого может быть более эффективным.

Вопрос 4. Выбор и оценка альтернативных планов выполнения запросов

Оптимизирующие преобразования, рассмотренные выше, оставляют внутреннее представление запроса непроцедурным.
Процедурным планом выполнения запроса называется такое его представление, в котором детализирован порядок выполнения операций доступа к базе данных физического уровня

Процедурное представление, вообще говоря, выбирается неоднозначно, поэтому необходимо выбрать среди альтернативных планов запроса один в соответствии с некоторыми критериями. Как правило, критерием выбора плана выполнения запроса является минимизация стоимости выполнения.
Тем самым, при обработке запроса на стадии, следующей за логической оптимизацией, решаются две задачи.

Первая задача: исходя из внутреннего представления запроса и информации, характеризующе управляющие структуры базы данных (например, индексы), выбрать набор потенциально возможных планов выполнения данного запроса.

Вторая задача: оценить стоимость выполнения запроса в соответствии с каждым альтернативным планом и выбрать план с наименьшей стоимостью.

4.1. Генерация планов

Обе задачи решаются на основе фиксированных встроенных в оптимизатор алгоритмов. Оптимизатор может быть рассчитан на то, что ограничение любого отношения в соответствии с заданным предикатом может быть выполнено путем некоторого последовательного просмотра отношения.

Например, запрос

SELECT EMPNAME FROM EMP WHERE DEPT# = 6 AND SALARY > 15.000

может выполняться:

- последовательным сканированием отношения EMP с выбором кортежей с DEPT# = 6 и SALARY > 15.000;
- сканированием отношения через индекс I1, определенный на поле DEPT#, с условием доступа к индексу DEPT# = 6 и условием выборки кортежа SALARY > 15.000;
- сканированием отношения через индекс I2, определенный на поле SALARY, с условием доступа к индексу SALARY > 15.000 и условием выборки кортежа DEPT# = 6.

Аналогично, фиксированы и стратегии выполнения более сложных операций - реляционных соединений отношений, вычисления агрегатных функций на группах кортежей отношения и т. д.

Например, для (экви)соединения двух отношений используются две основные стратегии: метод вложенных циклов и метод сортировок со слияниями.

Генерация плана выполнения сложного запроса - это многоэтапный процесс, в ходе которого учитываются свойства создаваемых при выполнении запроса по данному плану временных объектов базы данных.
Например, пусть запрос задан над тремя отношениями и в нем имеются два предиката соединения:
SELECT R1.C1, R2.C2, R3.C3 FROM R1, R2, R3 WHERE R1.C4 = R2.C5 AND R2.C5 = R3.C6.

Тогда, если в плане запроса выбирается порядок выполнения соединений сначала R1 с R2, а затем полученного временного отношения - с R3, и при этом для выполнения первого соединения выбирается метод сортировок со слиянием, то временное отношение будет заведомо отсортировано по С5, и одна сортировка не потребуется, если и второе соединение будет выполняться тем же методом.

Компонент оптимизатора, ведающий порождением множества альтернативных планов выполнения запроса, базируется на стратегиях декомпозиции запроса на элементарные составляющие и стратегиях выполнения элементарных составляющих. Первая группа стратегий определяет пространство поиска оптимального плана выполнения запроса, вторая направлена на то, чтобы в этом пространстве действительно находились эффективные планы выполнения запроса. Ключом к обеспечению эффективного выполнения сложного запроса является наличие эффективных стратегий выполнения элементарных составляющих.

4.2. Оценка стоимости плана запроса

После генерации множества планов выполнения запроса нужно выбрать один, наиболее эффективный план, в соответствии с которым будет происходить реальное выполнение запроса.
Необходимо определить, что понимается под эффективностью выполнения запроса.
Мерой эффективности запроса можно считать количество системных ресурсов, требуемых для его выполнения и т.д.
Будем говорить о стоимости плана выполнения запроса, определяемой ресурсами процессора и устройств внешней памяти, которые расходуются при выполнении запроса.

Компонент стоимости выполнения запроса, связанный с ресурсами устройств внешней памяти, монотонно зависит от числа блоков внешней памяти, доступ к которым потребуется при выполнении запроса.
С другой стороны, число блоков внешней памяти, доступ к которым требуется при выполнении запроса, монотонно зависит от числа кортежей, затрагиваемых запросом.

Из этого следует важность показателя предиката ограничения, характеризующего долю кортежей отношения, которые удовлетворяют данному предикату, и называемого ***степенью селективности предиката***.
Степени селективности простых предикатов вида R.C op const, где

R.C - имя поля отношения базы данных,
op - операция сравнения (=, !=, >, <, >=, <=),
a const - константа,

являются основой для оценок стоимости планов запроса.

Рассмотрим предикат R.C op const.
Степень селективности предиката зависит:
- от вида операции сравнения;
- значения константы;
- распределения значений поля C отношения R в базе данных.

Первые два параметра селективности могут быть известны при проведении оценок, но истинные распределения значений полей отношений, как правило, неизвестны. Существует ряд подходов к приближенным определениям распределений на основе статистической информации.

Если известна степень селективности предиката ограничения отношения, то тем самым известна и мощность результирующего отношения (обычно мощности хранимых отношений известны при обработке запроса). В оценочных формулах учитывается необходимое для выполнения запроса число обменов с внешней памятью. Конечно, число кортежей является верхней оценкой требуемого числа обменов, но эта оценка может быть очень завышенной. Все зависит от распределения кортежей по блокам внешней памяти.

Для сложных запросов, например, соединения более двух отношений, требуется оценивать размеры возникающих в процессе выполнения запроса промежуточных отношений. Чтобы оценить мощность результата соединения двух отношений, вообще говоря, необходимо знать степень селективности предиката соединения по отношению к прямому произведению отношений-операндов. В общем случае степень селективности такого предиката невозможно определить на основе простой статистической информации.

Характеристики отношения R:
T - число кортежей в данном отношении
N - число блоков внешней памяти, в которых располагаются кортежи отношения;
CD - число различных значений для каждого поля C
CMin - минимальное хранимое значение этого поля
CMax - максимальное значение
При наличии такой информации степени селективности простых предикатов вычисляется просто.
Пусть SEL (P) - степень селективности предиката P.

Тогда
SEL (R.C = const) = CD / (CMax - CMin)
(при равномерном распределении степень селективности такого предиката не зависит от значения константы).

SEL (R.C > const) = (CMax - const) / (CMax - CMin)

При оценках числа блоков, в которых могут располагаться кортежи, удовлетворяющие предикату R.C op const, различаются случаи кластеризованности или некластеризованности отношения по полю C.
Эти оценки имеют смысл только при рассмотрении варианта сканирования отношения с использованием индекса на поле C. При просмотре отношения без использования индекса понадобится всегда обратиться ровно к N блокам внешней памяти.

4.3. Более точные оценки

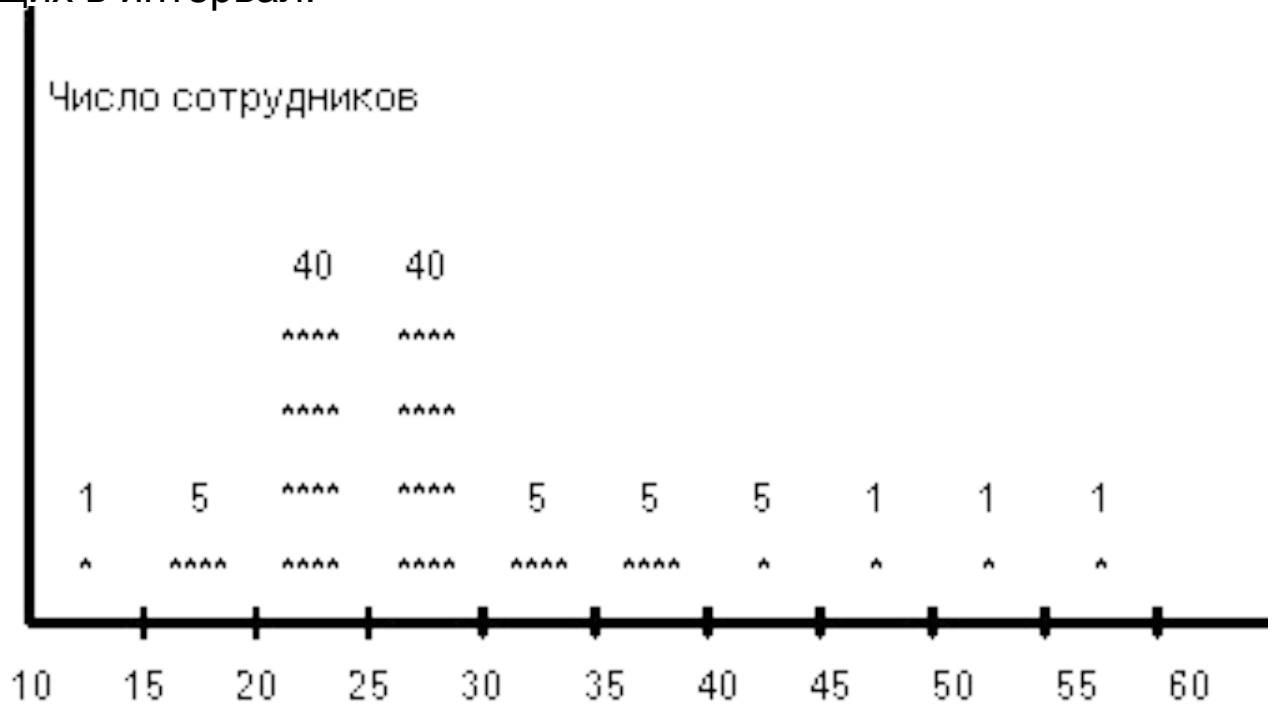
При отказе от предположения о равномерности распределения значений поля отношения необходимо уметь установить реальное распределение значений. Существует два базовых подхода к оценкам распределения значений поля отношения: параметрический и основанный на методе сигнатур. В первом случае распределение выбирается из серии путем вычисления нескольких параметров на основе выборок реально встречающихся значений. Примеры практического применения этого подхода нам неизвестны.

Метод оценки распределения на основе сигнатур заключается в том, что:

- область значений поля разбивается на несколько интервалов;
- для каждого интервала некоторым образом устанавливается число значений поля, попадающих в этот интервал;
- внутри интервала значения считаются распределенными по некоторому фиксированному закону (как правило, принимается равномерное приближение).

Рассмотрим два альтернативных подхода, связанных с сигнатурным описанием распределений.

Область значений поля разбивается на N интервалов равного размера, и для каждого интервала подсчитывается число значений полей из кортежей данного отношения, попадающих в интервал.



Пусть в интервал значений AGE Si попадает Ki значений.
Тогда SEL (EMP.AGE = const), если значение константы попадает в интервал значений Si, можно оценить следующим образом: 0 <= SEL (EMP.AGE) <= Ki/T (T - общее число кортежей в отношении EMP). Отсюда средняя оценка степени селективности предиката - Ki / (2 (T).

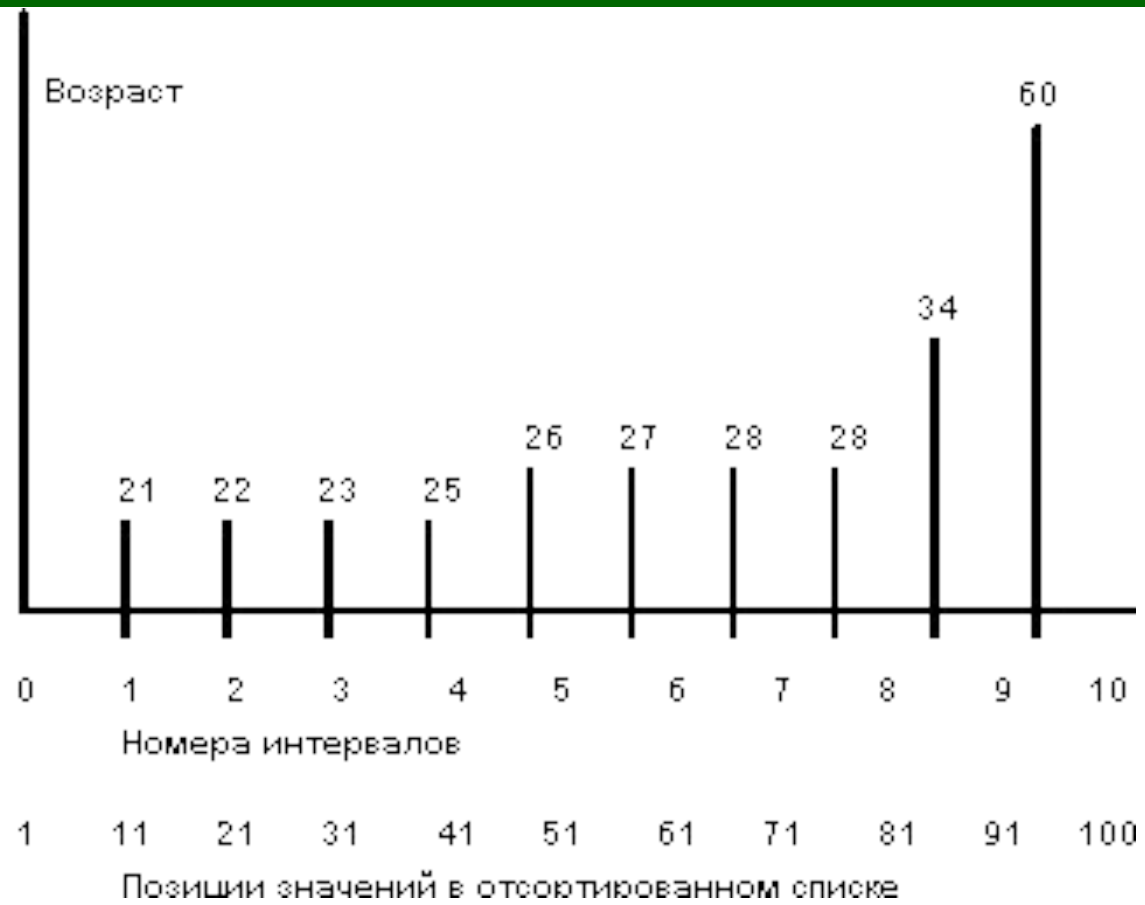
Например, SEL (AGE = 29) оценивается в 40/200 = 0.2, а SEL (AGE = 16) оценивается в 5/200 = 0.025. Это, конечно, существенно более точные оценки, чем те, которые можно получить, исходя из предположений о равномерности распределений.

Например, пусть требуется оценить степень селективности предиката EMP.AGE < const. Если значение константы попадает в интервал Si, и SUMi - суммарное количество значений AGE, попадающих в интервалы S1, S2, ..., Si,
то SUMi-1 / T <= SEL (AGE < const) <= SUMi / T.

Тогда средняя оценка степени селективности (SUMi-1 + SUMi) / (2 (T), и ошибка оценки может достигать половины веса подобласти, в которую попадает значение константы предиката.
Самое неприятное, что ошибка оценки зависит от значения константы и тем больше, чем больше значений поля содержится в соответствующем интервале гистограммы.

Например, SEL (AGE < 29) оценивается как 46/100 <= SEL (AGE < 29) <= 86/100, откуда оценка степени селективности (46 + 86) / 200 = 0.66; при этом ошибка оценки может достигать 0.2. В то же время SEL (AGE < 49) оценивается существенно более точно.

Для устранения этого дефекта был предложен другой подход к описанию распределений значений поля отношения. Идея состоит в том, что множество значений поля разбивается на интервалы разного размера так, чтобы в каждый интервал попадало одинаковое число значений поля.
Количество интервалов выбирается исходя из ограничений по памяти, и чем оно больше, тем точнее получаемые оценки.
При разбиении области значений на десять интервалов получаемая псевдогистограммная картина распределений значений поля AGE показана на рисунке.



Область значений поля AGE отношения EMP разбита на 10 интервалов. В каждый интервал попадает ровно по 10 значений поля AGE. Интервалы имеют разные размеры. В псевдогистограмме допустимы интервалы, правая и левая граница которых совпадают, например, интервал (28,28).

При использовании "псевдогистограммы" ошибки в оценках степеней селективности предикатов с операцией, отличной от равенства, уменьшаются.