SQL Задачи и решения

Учебник. Сергей Моисеенко.

Оператор SELECT

Получение итоговых значений

Комбинация детализированных и агрегированных данных

Пусть, помимо модели и цены принтера, требуется еще вывести максимальную и минимальную цену по всему множеству принтеров. Для новичка подобная задача зачастую представляет определенную сложность. Эта сложность состоит в дилемме: группировка или агрегация по всему множеству. Если использовать группировку, то максимум и минимум будут получены не по всей выборке, а для каждого подмножества, определяемого группировкой (в данном примере для каждой группы с одинаковой комбинацией значений {модель, цена}). Поскольку эта комбинация уникальна в таблице Printer учебной базы данных, то мы получим 3 совпадающих значения цены:

Выполнить Консоль

```
1.  SELECT model, price, MIN(price) min_price, MAX(price) max_price
2.  FROM printer
3.  GROUP BY model, price;
```

Если же не использовать группировку, то мы можем получить только минимальное и максимальное значение, поскольку стандартный синтаксис запрещает (ввиду неоднозначности трактовки результата) использовать наряду с агрегатами детализированные данные, по которым не выполняется группировка:

Выполнить Консоль

```
1.  SELECT MIN(price) min_price, MAX(price) max_price
2.  FROM printer;
```

Проблема разрешается довольно просто, причем не единственным способом. Так можно использовать подзапросы в предложении SELECT для каждого агрегатного значения. Это возможно, поскольку подзапрос вернет одно значение, а не набор:

Решение 1

Выполнить Консоль

```
1.  SELECT model, price,
2.  (SELECT MIN(price) FROM Printer) min_price,
3.  (SELECT MAX(price) FROM Printer) max_price
4.  FROM printer;
```

Более эффективным приемом будет использование подзапроса для вычисления агрегатов в предложении FROM, наряду с декартовым произведением. Бояться декартового произведения в этом случае не нужно, т.к. подзапрос вернет только одну строку, которая и будет соединяться с каждой строкой детализированных данных.

Решение 2

Выполнить Консоль

```
1.  SELECT model, price, min_price, max_price
2.  FROM printer CROSS JOIN
3.  (SELECT MIN(price) min_price, MAX(price) max_price
4.  FROM printer) X;
```

Почему мы утверждаем, что второй запрос будет эффективней? Дело в том, что в *решении 1* подзапрос будет вычисляться дважды, а не один раз, как это делается во втором решении. Кроме того, если оптимизатор недостаточно "умный", подзапросы в первом решении будут вычисляться для каждой строки детализированных данных. Проверьте планы выполнения для своей СУБД.

Рассмотрим теперь задачу, когда агрегат зависит от текущей строки детализированных данных, например, такую.

Вывести номер модели и цену принтера, а также максимальную и минимальную цену на принтеры того же типа.

Попытаемся адаптировать для этой задачи рассмотренные выше подходы. В *решении 1* подзапросы следует сделать коррелирующими :

Решение 1М

Выполнить Консоль

```
1.  SELECT model, price, type,
2.  (SELECT MIN(price) FROM Printer P1 WHERE P1.type=P.type) min_price,
3.  (SELECT MAX(price) FROM Printer P1 WHERE P1.type=P.type) max_price
4.  FROM printer P;
```

В *решении 2* мы можем воспользоваться нестандартным соединением **CROSS APPLY** (SQL Server), использующим коррелирующий подзапрос в предложении FROM.

Решение 2М

Выполнить Консоль

```
1.  SELECT model, price, P.type, min_price, max_price
2.  FROM Printer P CROSS APPLY
3.  (SELECT MIN(price) min_price, MAX(price) max_price
4.  FROM Printer P1
5.  WHERE P1.type=P.type) X;
```

Для наглядности в *решения 1М* и *2М* добавлен столбец *type*.

агрегатные функции группировка коррелирующие подзапросы CROSS APPLY

Предыдущая [Получение итоговых данных с помощью оператора ROLLUP]

[Сортировка и NULL-значения] Следующая

Последние изменения:

- Упражнение 151 (подсказки и решения)
- Предикат LIKE
- Приложение 2. Список задач стр. 2
- Приложение 2. Список задач
- Футбол
- Приложение 1. Описание учебных баз данных
- Функция STRING_AGG стр. 2
- Оператор UPDATE стр. 2
- Упражнение 151 стр. 4
- Упражнение 151

Тэги:

поиск по тэгам

ALL AND AUTO_INCREMENT AVG battles CASE CAST CHAR CHARINDEX CHECK classes COALESCE CONSTRAINT Convert COUNT CROSS APPLY CTE DATEADD DATEDIFF DATENAME DATEPART DATETIME DDL DEFAULT DELETE DISTINCT DML EXCEPT EXISTS EXTRACT FOREIGN KEY FROM FULL JOIN GROUP BY Guadalcanal HAVING IDENTITY IN INFORMATION_SCHEMA INNER JOIN insert INTERSECT IS NOT

NULL IS NULL ISNULL laptop LEFT
LEFT OUTER JOIN LEN maker

Больше тэгов

Учебник обновлялся
месяц назад
продать epayments .
Автоматический полив цветов и грядок