Тема 10. Агрегатные функции

Агрегатные функции позволяют вычислять обобщенные групповые значения. При этом предполагается предварительная группировка множества – разбиение его на группы кортежей с одинаковыми значениями атрибутов группирования.

Например, группирование таблицы Поставщики по атрибуту Товар выделяет в ней группы поставщиков конкретных товаров. Для каждой такой группы можно вычислить мощность, min/max цену товара, среднее значение цены и т.д. Агрегатные функции вычисляют одно значение для всей группы.

Агрегатные функции:

- COUNT – количество строк или непустых значений столбца;
- SUM – сумма всех элементов столбца;
- AVG – среднее по столбцу;
- MIN – минимальное в столбце;
- MAX – максимальное в столбце.

Пример.

Указать число поставщиков каждого поставляемого товара.

```
SELECT Товар, COUNT (*)
FROM Поставщики
GROUP BY Товар
```

Таблица Поставщики группируется по атрибуту Товар, в результате чего определяются группы кортежей, каждая из которых связана с одним поставляемым товаром. Для каждой группы в результат включается один кортеж, состоящий из названия товара и числа поставщиков товара.

Пример.

Определить суммарные заказы на спички, мыло и сигареты.

```
SELECT Товар, SUM(Количество)
FROM Заказы
WHERE Товар=спички OR Товар=мыло OR Товар=сигареты
GROUP BY Товар
```

Определяются группы заказанных товаров, и в результат выдаются названия товаров и суммарное число заказов этих товаров.

Пример.

Для каждого города (адреса) и каждого заказанного товара определить число поставщиков и заказчиков.

```
SELECT Покупатели.Адрес, Заказы.Товар, COUNT(DISTINCT Название),
                COUNT(DISTINCT Заказы.Фамилия)
FROM Покупатели, Поставщики, Заказы
WHERE Покупатели.Фамилия=Заказы.Фамилия
            AND Покупатели.Адрес=Поставщики.Адрес
            AND Поставщики.Товар=Заказы.Товар
GROUP BY Покупатели.Адрес, Заказы.Товар
```

Декартово произведение отношений Покупатели, Поставщики, Заказчики селектируется в соответствии с условием селекции и группируется по атрибутам Покупатели.Адрес и Заказы.Товар. Для каждой найденной группы подсчитывается число разных поставщиков и покупателей.

Пример.

Определить товары, суммарные заказы на которые превышают 100 единиц.

```
SELECT Товар, SUM(Количество)
FROM Заказы
GROUP BY Товар
HAVING SUM(Количество)>100
```

Предложение HAVING задает селекцию групп. В результат включаются только такие группы товаров, в которых суммарные заказы превышают 100.

В предложении HAVING можно использовать только агрегатные функции и поля группировки.

Вложенные запросы

Вложенный запрос порождает некоторое значение, которое проверяется в предложении WHERE или HAVING внешнего запроса. При этом используются предикаты IN, EXISTS и многократные сравнения ALL, ANY.

Пример.

Найти покупателей, заказы которых не могут быть выполнены.

```
SELECT DISTINCT Фамилия, Товар
FROM Заказы
WHERE NOT EXISTS
        (  SELECT Название
           FROM Поставщики
          WHERE Заказы.Товар=Товар)
```

Внешний запрос просматривает кортежи отношения Заказы и проверяет их на соответствие условию. Для этого для каждого заказа выполняется вложенный запрос, отбирающий всех поставщиков, способных удовлетворить заказ. Условие внешнего запроса выполняется, если таких поставщиков нет.

Пример.

Найти поставщиков заказанных товаров.

```
SELECT Название, Товар
FROM Поставщики
WHERE EXISTS
          ( SELECT Товар
            FROM Заказы
            WHERE Товар=Поставщики.Товар)
```

Пример.

Определить адреса покупателей, заказавших не менее 20 единиц какого-нибудь товара.

```
SELECT Фамилия, Адрес
FROM Покупатели
WHERE 20 <=  ANY
```

```
( SELECT Количество
  FROM Заказы
  WHERE Фамилия=Покупатели.Фамилия)
```

Условие внешнего запроса выполняется для покупателя, если какой-нибудь (ANY) его заказ превышает 20 единиц.

Определить адреса покупателей, все заказы которых >= 20 ед.

```
SELECT Фамилия, Адрес
FROM Покупатели
WHERE 20 <= ALL
        ( SELECT Количество
          FROM Заказы
          WHERE Фамилия=Покупатели.Фамилия)
```

Внутренние и внешние соединения, объединение

Результатом оператора

```
SELECT A, …
FROM R1, R2
WHERE условие
```

является множество кортежей, формируемых из кортежей отношений R1 и R2 и удовлетворяющих условию. Эта операция называется внутренним (INNER) условным соединением. На практике часто требуется включать в результат все кортежи отношения R1, а вместо кортежей R2, не удовлетворяющих условию, включать неопределенные значения. Или наоборот. Такая операция называется внешним (OUTER) соединением.

Внутреннее естественное соединение задается предложением:

```
FROM R1 NATURAL INNER JOIN R2.
```

Внешнее естественное соединение:

```
FROM R1 NATURAL { FULL | LEFT | RIGHT } [OUTER] JOIN R2.
```

Здесь вариант LEFT означает, что результат содержит все кортежи из R1, вариант RIGHT – все кортежи из R2, вариант FULL - все кортежи из R1 и R2.

Перекрестное соединение:

FROM R1 CROSS JOIN R2.

Вычисляется декартово произведение R1 и R2.

Объединение запросов производится операцией UNION:

Запрос1 UNION Запрос2.

Практические задания для самостоятельного выполнения

Задание 1. Используя оператор CREATE TABLE создать следующие таблицы:

а) Преподаватели (Фамилия, Адрес, Оклад),

б) Факультеты (Название, Специальность),

в) Нагрузка (Фамилия, Предмет, Количество часов).

Задание 2. Произвести реструктуризацию таблиц:

а) Добавить в таблицу **Преподаватели** новое поле – Должность преподавателя

б) Добавить в таблицу **Факультеты** новое поле – Код специальности

Задание 3. Построить SQL-запросы, эквивалентные следующим предложениям естественного языка:

а) найти преподавателей, проживающих в заданном городе

б) найти всех преподавателей, оклад которых превышает 15000