**10. Агрегирование данных с помощью групповых функций.**

Есть несколько **групповых функций**, доступных в SQL, таких как AVG, COUNT, MAX, MIN, SUM, STDDEV и VARIANCE.

Можно создавать подгруппы при использовании предложения GROUP BY. Далее, группы могут быть ограничены, используя предложение HAVING.

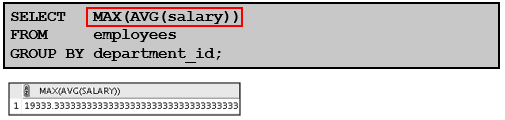
Поместите предложения HAVING и GROUP BY после предложения WHERE в операторе. Порядок предложений GROUP BY и HAVING после предложения WHERE не важен. В конце можно добавить предложение ORDER BY.

Сервер Oracle обрабатывает эти предложения в следующем порядке:

1. Если оператор содержит предложение WHERE, сервер устанавливает строки-кандидаты.
2. Сервер идентифицирует группы, которые определяются в предложении GROUP BY.
3. Предложение HAVING затем ограничивает группы результата, отбрасывая группы, которые не соответствуют групповому критерию в предложении HAVING.

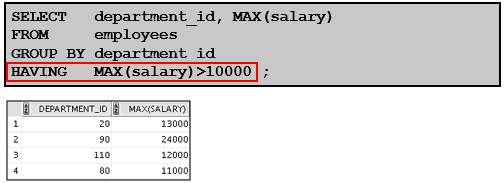
ВЛОЖЕНИЕ ГРУППОВЫХ ФУНКЦИЙ

Вывод на экран максимальной средней зарплаты:



Групповые функции могут быть вложены до глубины двух функций. Пример на рисунке вычисляет среднюю зарплату для каждого department\_id и затем выводит на экран максимальную среднюю зарплату. Отметьте, что предложение GROUP BY обязательно при вложении групповых функций.

Пример на рисунке выводит на экран номера отделов и максимальные зарплаты для тех отделов, у которых максимальная зарплата больше чем 10 000$.

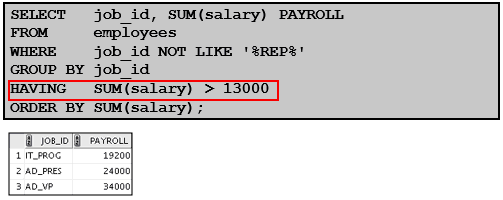


Можно использовать предложение GROUP BY, не используя групповую функцию в списке SELECT. Если Вы ограничиваете строки, основываясь на результате групповой функции, у Вас должно быть предложение GROUP BY, так же как и **предложение HAVING**.

Следующий пример выводит на экран номера отделов и средние зарплаты для тех отделов, у которых максимальная зарплата больше чем 10 000$:

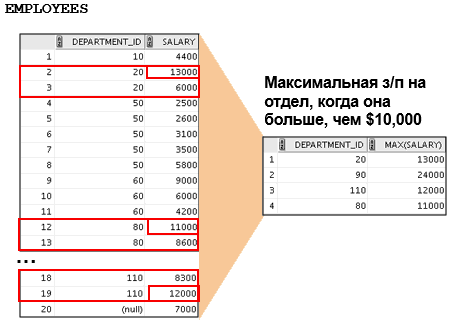
|  |  |
| --- | --- |
|  | SELECT department\_id, AVG(salary) FROM employees GROUP BY department\_id HAVING max(salary)>10000; |

ПРЕДЛОЖЕНИЕ HAVING - ПРИМЕР



Пример на рисунке выводит на экран ID работы и общую месячную зарплату для каждой работы, у которой полная платежная ведомость превышает 13 000$. Пример исключает торговых представителей и сортирует список по общей месячной зарплате.

Вы используете предложение HAVING, чтобы **ограничить группы** таким же образом, как Вы используете предложение WHERE, чтобы ограничить строки, которые Вы выбираете.



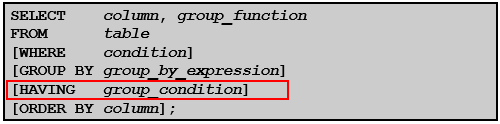
Чтобы найти максимальную зарплату в каждом из отделов, у которых максимальная зарплата больше чем 10 000$, Вы должны сделать следующее:

1. Найдите максимальную зарплату для каждого отдела, группируя по номеру отдела.
2. Ограничьте группы теми отделами, в которых максимальная зарплата больше чем 10 000$.

РЕЗУЛЬТАТЫ ОГРАНИЧЕНИЯ ГРУПП С ПРЕДЛОЖЕНИЕМ HAVING

Когда Вы используете предложение HAVING, сервер Oracle ограничивает группы следующим образом:

1. Строки группируются.
2. Групповая функция применяется.
3. Группы, соответствующие предложению HAVING, выводится на экран.



Вы используете предложение HAVING, чтобы определить группы, которые должны быть выведены на экран, таким образом далее ограничивая группы на основе совокупной информации.

В этом синтаксисе, group\_condition ограничивает группы строк теми группами, для которых указанное условие является истиной.

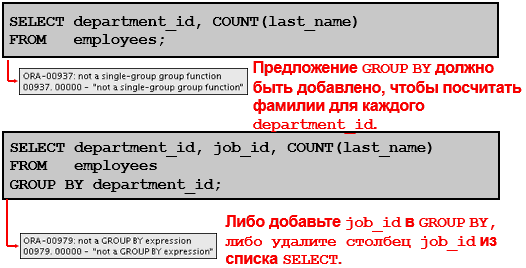
Сервер Oracle выполняет следующие шаги, когда Вы используете предложение HAVING:

1. Строки группируются.
2. Групповая функция применяется к группе.
3. Группы, которые соответствуют критерию в предложении HAVING, выводится на экран.

Предложение HAVING может предшествовать предложению GROUP BY, но рекомендуется, чтобы Вы помещали предложение GROUP BY сначала, потому что это более логично. Группы формируются и групповые функции вычисляются перед тем, как предложение HAVING применяется к группам в списке SELECT.

**Отметьте**: предложение WHERE ограничивает строки, тогда как предложение HAVING ограничивает группы.

Любой столбец или выражение в списке SELECT, который не является агрегатной функцией, должен быть в предложении GROUP BY:



Всякий раз, когда Вы используете смесь отдельных элементов (DEPARTMENT\_ID) и групповые функции (COUNT) в том же самом операторе SELECT, следует включать предложение GROUP BY, которое будет указывать отдельные элементы (в этом случае, DEPARTMENT\_ID). Если предложение GROUP BY отсутствует, появляется сообщение об ошибке “not a single-group group function” и звездочка (\*) указывает незаконный столбец. Можно исправить ошибку в первом примере на рисунке, добавляя предложение GROUP BY:

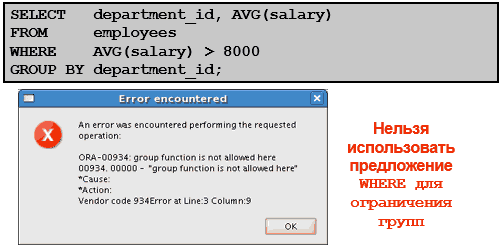
|  |  |
| --- | --- |
|  | SELECT department\_id, count(last\_name) FROM employees GROUP BY department\_id; |

Любой столбец или выражение в списке SELECT, который не находится в агрегатной функции, должен быть включен в предложение GROUP BY. Во втором примере на рисунке job\_id не состоит в предложении GROUP BY и не используется групповой функцией, таким образом возникает ошибка “not a GROUP BY expression”. Можно исправить ошибку во втором примере рисунка, включая job\_id в предложение GROUP BY.

|  |  |
| --- | --- |
|  | SELECT department\_id, job\_id, COUNT(last\_name) FROM employees GROUP BY department\_id, job\_id; |

НЕКОРРЕКТНЫЕ ЗАПРОСЫ С ИСПОЛЬЗОВАНИЕМ ГРУППОВЫХ ФУНКЦИЙ

* Нельзя использовать предложение WHERE, чтобы ограничить группы.
* Вы используете предложение HAVING для ограничения групп.
* Нельзя использовать групповые функции в предложении WHERE.



Предложение WHERE не может использоваться, чтобы ограничить группы. Оператор SELECT в примере на рисунке приводит к ошибке, потому что он использует предложение WHERE, чтобы ограничить вывод средних зарплат теми отделами, у которых средняя зарплата больше чем $8,000.

Однако, можно исправить ошибку в примере при использовании предложения HAVING, чтобы ограничить группы:

|  |  |
| --- | --- |
|  | SELECT department\_id, AVG(salary) FROM employees GROUP BY department\_id HAVING AVG(salary) > 8000; |

Иногда, Вы должны увидеть результаты для групп внутри групп.

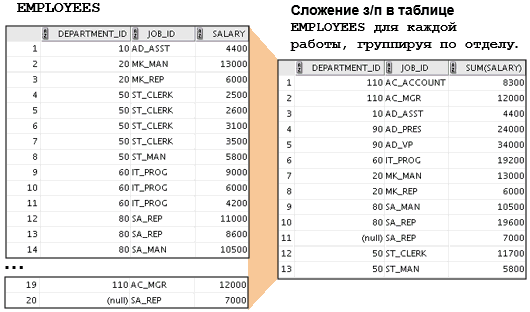


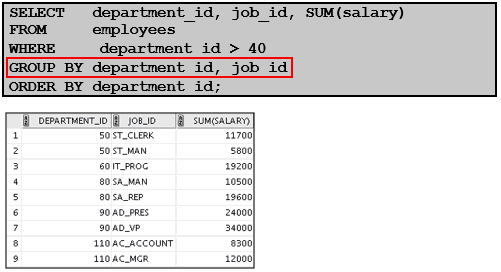
Рисунок показывает отчет, который выводит на экран полную зарплату, которая платится для каждой работы в каждом отделе.

Таблица EMPLOYEES группируется сначала по номеру отдела, а затем по названию работы в пределах этой группировки. Например, четыре клерка в отделе 50 группируются вместе, и единственный результат (полная зарплата) вычисляется для всех клерков в группе.

Следующий оператор SELECT возвращает результат, показанный на рисунке:

|  |  |
| --- | --- |
|  | SELECT department\_id, job\_id, sum(salary) FROM employees GROUP BY department\_id, job\_id ORDER BY job\_id; |

ИСПОЛЬЗОВАНИЕ ПРЕДЛОЖЕНИЯ GROUP BY ДЛЯ НЕСКОЛЬКИХ СТОЛБЦОВ



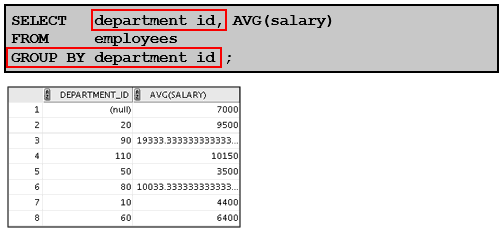
Можно возвратить сводные результаты для групп и подгрупп, перечисляя **несколько столбцов в GROUP BY**. Предложение GROUP BY группирует строки, но не гарантирует порядка в наборе результатов. Чтобы упорядочить группировки, используйте предложение ORDER BY.

В примере на рисунке, оператор SELECT, который содержит предложение GROUP BY, обрабатывается следующим образом:

* Предложение SELECT определяет столбец, который будет получен:
  + ID отдела в таблице EMPLOYEES
  + ID работы в таблице EMPLOYEES
  + Сумма всех зарплат в группе, которую Вы определили в предложении GROUP BY
* Предложение FROM определяет таблицы, к которым должна обратиться база данных: таблица EMPLOYEES.
* Предложение WHERE сужает набор результатов до тех строк, где ID отдела больше чем 40.
* Предложение GROUP BY определяет, как следует сгруппировать получающиеся строки:
  + Во-первых, строки группируются по ID отдела.
  + Во-вторых, строки группируются по ID работы в группах с одинаковым ID отдела.
* Предложение ORDER BY сортирует результаты по ID отдела.

**Отметьте**: функция SUM применяется к столбцу зарплаты для всех ID работы в наборе результатов для каждой группы с одним и тем же ID отдела. Кроме того, отметьте что строка SA\_REP не возвращается. ID отдела для этой строки равен NULL и, поэтому, она не удовлетворяет условию WHERE.

Все столбцы в списке SELECT, которые не находятся в групповых функциях, должны быть в предложении GROUP BY.



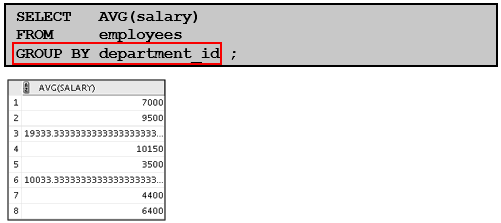
При **использовании предложения GROUP BY**, удостоверьтесь что все столбцы в списке SELECT, которые не являются включенными в групповые функции, находятся в предложении GROUP BY. Пример на рисунке выводит на экран номер отдела и среднюю зарплату для каждого отдела. Вот, как этот оператор SELECT, содержащий предложение GROUP BY, вычисляется:

* Предложение SELECT определяет столбцы, которые будут получены, следующим образом:
  + Столбец номера отдела в таблице EMPLOYEES
  + Среднее значение всех зарплат в группе, которую Вы определили в предложении GROUP BY
* Предложение FROM определяет таблицы, к которым должна обратиться база данных: таблица EMPLOYEES.
* Предложение WHERE определяет строки, которые будут получены. Поскольку нет предложения WHERE, все строки извлекаются по умолчанию.
* Предложение GROUP BY определяет, как строки должны быть сгруппированы. Строки группируются по номеру отдела, таким образом, функция AVG, которая применяется к столбцу зарплаты, вычисляет среднюю зарплату для каждого отдела.

**Отметьте**: Чтобы упорядочить результаты запроса в порядке по возрастанию или в порядке по убыванию, включите предложение ORDER BY в запрос.

УКАЗАНИЯ ПО ПРЕДЛОЖЕНИЮ GROUP BY

Столбец GROUP BY не обязан быть в списке SELECT.

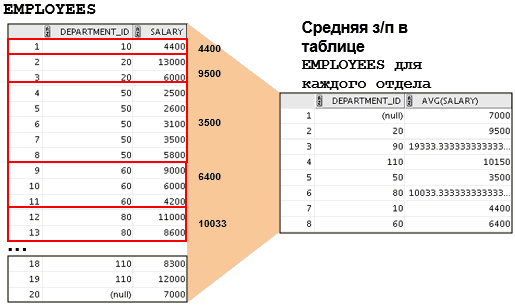


Столбец GROUP BY не обязан быть в предложении SELECT. Например, оператор SELECT на рисунке выводит на экран средние зарплаты для каждого отдела, не выводя на экран соответствующие номера отделов. Без номера отдела, однако, результаты не выглядят значимыми.

Можно также использовать групповую функцию в предложение ORDER BY:

|  |  |
| --- | --- |
|  | SELECT department\_id, AVG(salary) FROM employees GROUP BY department\_id ORDER BY AVG(salary); |

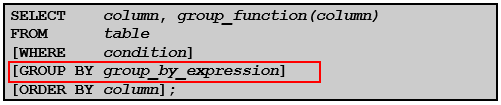
До сих пор в обсуждении все групповые функции обрабатывали таблицу как одну большую группу информации.



Иной раз, однако, требуется разделять таблицу информации в меньшие группы. Это может быть сделано при использовании предложения GROUP BY.

СОЗДАНИЕ ГРУПП ДАННЫХ: СИНТАКСИС ПРЕДЛОЖЕНИЯ GROUP BY

Можно разделить строки таблицы в меньшие группы при использовании предложения GROUP BY.



Можно использовать предложение GROUP BY, чтобы разделить строки **таблицы в группы**. Можно затем использовать групповые функции, чтобы возвратить итоговую информацию для каждой группы.

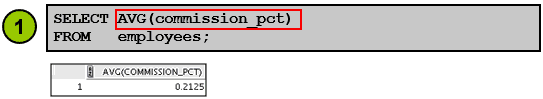
В синтаксисе:

|  |  |
| --- | --- |
|  | group\_by\_expression Указывает столбцы, значения которых определяют критерий для того, чтобы сгруппировать строки |

ИНСТРУКЦИИ

* Если Вы включаете групповую функцию в предложение SELECT, невозможно выбрать также и отдельные результаты, если отдельный столбец не появляется в предложении GROUP BY. Вы получаете сообщение об ошибке, если Вы не включаете список столбцов в предложение GROUP BY.
* Используя предложение WHERE, можно исключить строки прежде, чем разделить их на группы.
* Следует включать *столбцы* в предложение GROUP BY.
* Невозможно использовать псевдоним столбца в предложении GROUP BY.

Групповые функции игнорируют null-значения в столбце:



Функция NVL принуждает групповые функции включать null-значения:



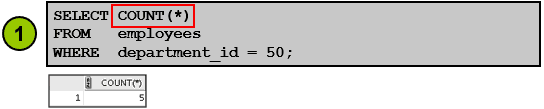
Все **групповые функции игнорируют значения null** в столбце.

Однако, функция NVL заставляет групповые функции включать значения null.

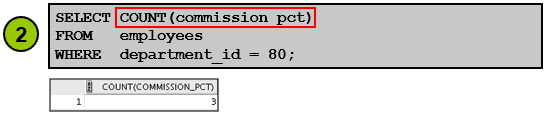
Примеры:

1. Среднее значение вычисляется на основе *только* тех строк в таблице, в которых хранится допустимое значение в столбце COMMISSION\_PCT. Среднее значение вычисляется как итоговая комиссия, которая платится *всем* сотрудникам, разделенная на численность персонала, получающего комиссионные (четыре).
2. Среднее значение вычисляется на основе всех строк в таблице, независимо от того, сохранены ли null-значения в столбце COMMISSION\_PCT. Среднее значение вычисляется как итоговая комиссия, которая платится всем сотрудникам, разделенная на общее количество сотрудников в компании (20).

COUNT(\*) возвращает число строк в таблице:



COUNT(*expr*) возвращает число строк с не-null значениями для *expr*:



У **функции COUNT** есть три формата:

* COUNT(\*)
* COUNT(expr)
* COUNT(DISTINCT expr)

COUNT(\*) возвращает число строк в таблице, которые удовлетворяют критерию оператора SELECT, включая дублирующиеся строки и строки, содержащие null-значения в любом из столбцов. Если предложение WHERE включается в оператор SELECT, COUNT(\*) возвращает число строк, которые удовлетворяют условию в предложении WHERE.

Напротив, COUNT(expr) возвращает число не-null значений, которые находятся в столбце, идентифицированном expr.

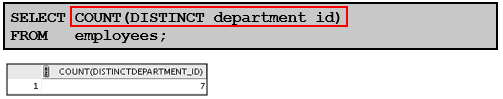
COUNT(DISTINCT expr) возвращает число уникальных, не-null значений, которые находятся в столбце, определяемом expr.

Примеры:

1. Пример на рисунке выводит на экран численность персонала в отделе 50.
2. Пример на рисунке выводит на экран численность сотрудников в отделе 80, которые могут заработать комиссионные.

ИСПОЛЬЗОВАНИЕ КЛЮЧЕВОГО СЛОВА DISTINCT

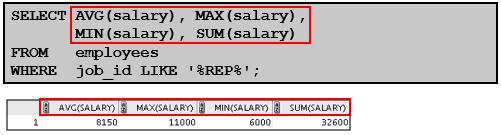
* COUNT(DISTINCT expr) возвращает число различных не-null значений *expr*.
* Пример вывода на экран числа различных отделов в таблице EMPLOYEES:



Используйте ключевое слово DISTINCT, чтобы не считать дублирующиеся значения в столбце.

Пример на рисунке выводит на экран число различных значений отделов, которые находятся в таблице EMPLOYEES.

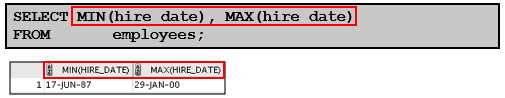
Можно использовать AVG и SUM для числовых данных.



Можно использовать **функции AVG, SUM, MIN и MAX** для столбцов, которые хранят числовые данные. Пример на рисунке выводит на экран среднее, самое высокое, самое низкое и сумму значений ежемесячных зарплат для всех торговых представителей.

ИСПОЛЬЗОВАНИЕ ФУНКЦИЙ MIN И MAX

Можно использовать MIN и MAX для числовых, символьных типов данных и для даты.



Можно использовать функции MAX и MIN для числовых и символьных типов данных, а также для дат. Пример на рисунке выводит на экран самых младших и самых старших сотрудников.

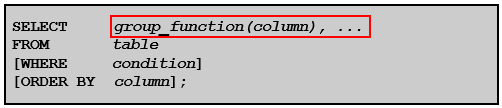
Следующий пример выводит на экран фамилию сотрудника, которая является первой и фамилию сотрудника, которая является последней в алфавитном списке всех сотрудников:

|  |  |
| --- | --- |
|  | SELECT MIN(last\_name), MAX(last\_name) FROM employees; |



**Отметьте**: Функции AVG, SUM, VARIANCE и STDDEV могут использоваться только с числовыми типами данных. MAX и MIN не могут использоваться с LOB или типами данных LONG.

Групповая функция помещается после ключевого слова SELECT.



Можно использовать несколько **групповых функций**, разделенных запятыми.

Инструкции для того, чтобы использовать групповые функции:

* DISTINCT заставляет функцию рассматривать только уникальные (недублирующиеся) значения; ALL заставляет ее рассматривать каждое значение, включая дубликаты. Значение по умолчанию ALL и, поэтому, его можно не указывать.
* Типы данных для функций с параметром expr могут быть CHAR, VARCHAR2, NUMBER или DATE.
* Все групповые функции игнорируют null-значения. Чтобы заменить значением null значения, используйте NVL, NVL2, COALESCE, CASE или функцию DECODE.