



# **Отчетность по агрегированным данным с использованием групповых функций**

# Цели

Изучив материал этого занятия, вы сможете:

- Определять доступные групповые функции
- Описывать использование групповых функций
- Группировать данные при помощи предложения `GROUP BY`
- Включать или исключать сгруппированные строки при помощи предложения `HAVING`

# План занятия

- Групповые функции:
  - Типы и синтаксис
  - Использование AVG, SUM, MIN, MAX, COUNT
  - Использование в групповых функциях ключевого слова DISTINCT
  - Значения NULL в групповых функциях
- Группирование строк:
  - Предложение GROUP BY
  - Предложение HAVING
- Вложенные групповые функции

# Что такое групповые функции?

Групповые функции оперируют с наборами строк и выдают по одному результату на группу.

## EMPLOYEES

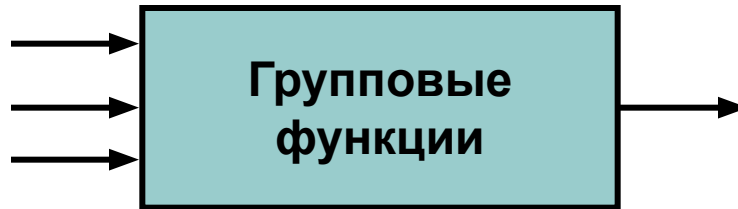
	DEPARTMENT_ID	SALARY
1	90	24000
2	90	17000
3	90	17000
4	60	9000
5	60	6000
6	60	4200
7	50	5800
8	50	3500
9	50	3100
10	50	2600
...		
18	20	6000
19	110	12000
20	110	8300

Максимальный  
оклад в таблице  
EMPLOYEES

MAX(SALARY)
24000

# Типы групповых функций

- AVG
- COUNT
- MAX
- MIN
- STDDEV
- SUM
- VARIANCE



# Групповые функции: синтаксис

```
SELECT      групповая_функция (столбец) , ...  
FROM        таблица  
[WHERE      условие]  
[ORDER BY   столбец] ;
```

# Использование функций AVG и SUM

Функции AVG и SUM можно использовать для числовых данных.

```
SELECT AVG(salary), MAX(salary),  
       MIN(salary), SUM(salary)  
FROM   employees  
WHERE  job_id LIKE '%REP%';
```

	AVG(SALARY)	MAX(SALARY)	MIN(SALARY)	SUM(SALARY)
1	8150	11000	6000	32600

# Использование функций MIN и MAX

Функции MIN и MAX можно использовать с числовыми и символьными типами данных, а также с датами.

```
SELECT MIN(hire date), MAX(hire date)  
FROM employees
```

	MIN(HIRE_DATE)	MAX(HIRE_DATE)
1	17-JUN-87	29-JAN-00



# Использование функции COUNT

COUNT (\*) возвращает число строк в таблице:

1

```
SELECT COUNT (*)  
FROM employees  
WHERE department_id = 50;
```

	COUNT(*)
1	5

COUNT (expr) возвращает число строк, для которых значение *expr* не равно NULL:

2

```
SELECT COUNT (commission_pct)  
FROM employees  
WHERE department_id = 80;
```

	COUNT(COMMISSION_PCT)
1	3

# Использование ключевого слова DISTINCT

- COUNT (DISTINCT *expr*) возвращает число уникальных и не равных NULL значений *expr*.
- Чтобы показать число различных отделов, представленных в таблице EMPLOYEES:

```
SELECT COUNT(DISTINCT department id)  
FROM employees
```

	COUNT(DISTINCTDEPARTMENT_ID)
1	7

# Групповые функции и значения Null

Групповые функции игнорируют значения NULL в

столбцах:

1

```
SELECT AVG(commission_pct)
FROM employees
```

AVG(COMMISSION_PCT)	
1	0.2125

Функция NVL заставляет групповые функции учитывать значения NULL.

2

```
SELECT AVG(NVL(commission_pct, 0))
FROM employees
```

AVG(NVL(COMMISSION_PCT,0))	
1	0.0425

# План занятия

- Групповые функции:
  - Типы и синтаксис
  - Использование AVG, SUM, MIN, MAX, COUNT
  - Использование в групповых функциях ключевого слова DISTINCT
  - Значения NULL в групповых функциях
- Группирование строк:
  - Предложение GROUP BY
  - Предложение HAVING
- Вложенные групповые функции

# Создание групп данных

## EMPLOYEES

	DEPARTMENT_ID	SALARY	
1	10	4400	4400
2	20	13000	9500
3	20	6000	
4	50	5800	3500
5	50	2500	
6	50	2600	
7	50	3100	
8	50	3500	
9	60	4200	6400
10	60	6000	
11	60	9000	
12	80	11000	10033
13	80	10500	
14	80	8600	
...			
19	110	12000	
20	(null)	7000	

Средний оклад в таблице  
EMPLOYEES для  
каждого из отделов

	DEPARTMENT_ID	AVG(SALARY)
1	10	4400
2	20	9500
3	50	3500
4	60	6400
5	80	10033.333333333333...
6	90	19333.333333333333...
7	110	10150
8	(null)	7000

# Создание групп данных: синтаксис предложения GROUP BY

```
SELECT      столбец, групповая_функция (столбец)  
FROM        таблица  
[WHERE      условие]  
[GROUP BY   выражение_группировки]  
[ORDER BY   столбец] ;
```

Строки таблицы можно объединить в более компактные группы при помощи предложения GROUP.

# Использование предложения GROUP BY

Все столбцы из списка SELECT, не используемые групповыми функциями, должны быть перечислены в предложении GROUP BY.

```
SELECT department id, AVG(salary)
FROM employees
GROUP BY department id ;
```

	DEPARTMENT_ID	AVG(SALARY)
1	(null)	7000
2	90	19333.3333333333...
3	20	9500
4	110	10150
5	50	3500
6	80	10033.3333333333...
7	60	6400
8	10	4400

# Использование предложения GROUP BY

Столбец, перечисленный в предложении GROUP BY, не обязательно должен присутствовать в списке SELECT.

```
SELECT    AVG(salary)
FROM      employees
GROUP BY  department_id ;
```

	AVG(SALARY)
1	7000
2	19333.333333333333333333333333...
3	9500
4	10150
5	3500
6	10033.333333333333333333333333...
7	6400
8	4400



# Группирование по нескольким столбцам

## EMPLOYEES

	DEPARTMENT_ID	JOB_ID	SALARY
1	10	AD_ASST	4400
2	20	MK_MAN	13000
3	20	MK_REP	6000
4	50	ST_MAN	5800
5	50	ST_CLERK	2500
6	50	ST_CLERK	2600
7	50	ST_CLERK	3100
8	50	ST_CLERK	3500
9	60	IT_PROG	4200
10	60	IT_PROG	6000
11	60	IT_PROG	9000
12	80	SA_REP	11000
13	80	SA_MAN	10500
14	80	SA_REP	8600
...			
19	110	AC_MGR	12000
20	(null)	SA_REP	7000

Суммирование окладов  
в таблице EMPLOYEES  
для всех должностей,  
с группировкой по отделах.

	DEPARTMENT_ID	JOB_ID	SUM(SALARY)
1	10	AD_ASST	4400
2	20	MK_MAN	13000
3	20	MK_REP	6000
4	50	ST_CLERK	11700
5	50	ST_MAN	5800
6	60	IT_PROG	19200
7	80	SA_MAN	10500
8	80	SA_REP	19600
9	90	AD_PRES	24000
10	90	AD_VP	34000
11	110	AC_ACCOUNT	8300
12	110	AC_MGR	12000
13	(null)	SA_REP	7000

# Использование предложения GROUP BY с несколькими столбцами


```
SELECT    department_id dept_id, job_id, SUM(salary)
FROM      employees
GROUP BY  department_id, job_id
ORDER BY  department_id;
```

	DEPARTMENT_ID	JOB_ID	SUM(SALARY)
1	10	AD_ASST	4400
2	20	MK_MAN	13000
3	20	MK_REP	6000
4	50	ST_CLERK	11700
5	50	ST_MAN	5800
6	60	IT_PROG	19200
7	80	SA_MAN	10500
8	80	SA_REP	19600
9	90	AD_PRES	24000
10	90	AD_VP	34000
11	110	AC_ACCOUNT	8300
12	110	AC_MGR	12000
13	(null)	SA_REP	7000

# Недопустимые запросы при использовании групповых функций

Любой столбец или выражение из списка `SELECT`, которые не являются агрегатной функцией, должны быть перечислены в предложении `GROUP BY`:


```
SELECT department_id, COUNT(last_name)
FROM employees
```



ORA-00937: not a single-group group function  
00937. 00000 - "not a single-group group function"

**Чтобы для каждого идентификатора отдела `department_id` подсчитать число фамилий, необходимо добавить предложение `GROUP BY`.**

```
SELECT department_id, job_id, COUNT(last_name)
FROM employees
GROUP BY department_id ;
```



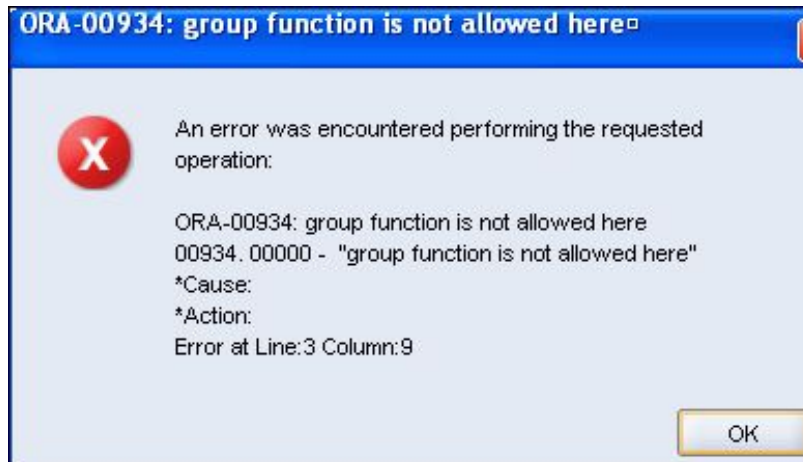
ORA-00979: not a GROUP BY expression  
00979. 00000 - "not a GROUP BY expression"

**Либо добавьте в `GROUP BY` столбец `job_id`, либо удалите столбец `job_id` из списка `SELECT`.**

# Недопустимые запросы при использовании групповых функций

- Для ограничения групп нельзя использовать предложение `WHERE`.
- Для ограничения групп следует использовать предложение `HAVING`.
- Групповые функции в предложении `WHERE` использовать нельзя.

```
SELECT    department_id, AVG(salary)
FROM      employees
WHERE     AVG(salary) > 8000
GROUP BY department_id ;
```



**Для ограничения  
групп нельзя  
использовать  
предложение  
`WHERE`**

# Ограничение групповых результатов

## EMPLOYEES

R2	DEPARTMENT_ID	R2	SALARY
1	10		4400
2	20		13000
3	20		6000
4	50		5800
5	50		2500
6	50		2600
7	50		3100
8	50		3500
9	60		4200
10	60		6000
11	60		9000
12	80		11000
13	80		10500
14	80		8600

...

18	110		8300
19	110		12000
20	(null)		7000

**Максимальный оклад  
по отделам, у которых  
он больше \$10 000**

R2	DEPARTMENT_ID	R2	MAX(SALARY)
1	20		13000
2	80		11000
3	90		24000
4	110		12000

# Ограничение групповых результатов при помощи предложения HAVING

Когда используется предложение HAVING, сервер Oracle ограничивает группы следующим образом:

1. Строки объединяются в группы.
2. Применяется групповая функция.
3. Отображаются группы, соответствующие предложению HAVING.

```
SELECT      столбец, групповая_функция
FROM        таблица
[WHERE      условие]
[GROUP BY   выражение group by]
[HAVING     условие группирования]
[ORDER BY   столбец];
```

# Использование предложения HAVING

```
SELECT    department_id, MAX(salary)
FROM      employees
GROUP BY  department_id
HAVING    MAX(salary)>10000 ;
```

	DEPARTMENT_ID	MAX(SALARY)
1	90	24000
2	20	13000
3	110	12000
4	80	11000

# Использование предложения HAVING

```
SELECT    job_id, SUM(salary) PAYROLL
FROM      employees
WHERE     job_id NOT LIKE '%REP%'
GROUP BY  job_id
HAVING    SUM(salary) > 13000
ORDER BY  SUM(salary);
```

	 JOB_ID	 PAYROLL
1	IT_PROG	19200
2	AD_PRES	24000
3	AD_VP	34000



# План занятия

- Групповые функции:
  - Типы и синтаксис
  - Использование AVG, SUM, MIN, MAX, COUNT
  - Использование в групповых функциях ключевого слова DISTINCT
  - Значения NULL в групповых функциях
- Группирование строк:
  - Предложение GROUP BY
  - Предложение HAVING
- Вложенные групповые функции

# Вложенные групповые функции

Отображение максимального среднего оклада:

```
SELECT MAX (AVG (salary))
FROM employees
GROUP BY department id ;
```

[illegible]

# Заключение

На этом занятии были изучены следующие темы:

- Использование групповых функций COUNT, MAX, MIN, SUM и AVG
- Создание запросов, использующих предложение GROUP BY
- Создание запросов, использующих предложение HAVING

```
SELECT      столбец, групповая_функция
FROM        таблица
[WHERE      условие]
[GROUP BY   выражение_группирования]
[HAVING     условие_группирования]
[ORDER BY   столбец];
```

# Упражнение 5: обзор

Упражнение охватывает следующие темы:

- Создание запросов, использующих групповые функции
- Группирование строк для получения нескольких результатов
- Ограничение групп при помощи предложения `HAVING`







