**Лабораторная работа № 5\_1 СХЕМА HR**

  
**Restricting and Sorting Data**  
  
Таблица Employees. Получить список с информацией обо всех сотрудниках

**Решение**

**SELECT** \* **FROM** employees

Таблица Employees. Получить список всех сотрудников с именем 'David'

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** first\_name = 'David';

Таблица Employees. Получить список всех сотрудников с job\_id равным 'IT\_PROG'

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** job\_id = 'IT\_PROG'

Таблица Employees. Получить список всех сотрудников из 50го отдела (department\_id) с зарплатой(salary), большей 4000

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** department\_id = 50 **AND** salary > 4000;

Таблица Employees. Получить список всех сотрудников из 20го и из 30го отдела (department\_id)

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** department\_id = 20 **OR** department\_id = 30;

Таблица Employees. Получить список всех сотрудников у которых последняя буква в имени равна 'a'

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** first\_name **LIKE** '%a';

Таблица Employees. Получить список всех сотрудников из 50го и из 80го отдела (department\_id) у которых есть бонус (значение в колонке commission\_pct не пустое)

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** (department\_id = 50 **OR** department\_id = 80)

**AND** commission\_pct **IS** **NOT** NULL;

Таблица Employees. Получить список всех сотрудников у которых в имени содержатся минимум 2 буквы 'n'

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** first\_name **LIKE** '%n%n%';

Таблица Employees. Получить список всех сотрудников у которых длина имени больше 4 букв

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** first\_name **LIKE** '%\_\_\_\_\_%';

Таблица Employees. Получить список всех сотрудников у которых зарплата находится в промежутке от 8000 до 9000 (включительно)

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** salary **BETWEEN** 8000 **AND** 9000;

Таблица Employees. Получить список всех сотрудников у которых в имени содержится символ '%'

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** first\_name **LIKE** '%\%%' ESCAPE '\';

Таблица Employees. Получить список всех ID менеджеров

**Решение**

**SELECT** **DISTINCT** manager\_id

**FROM** employees

**WHERE** manager\_id **IS** **NOT** NULL;

Таблица Employees. Получить список работников с их позициями в формате: Donald(sh\_clerk)

**Решение**

**SELECT** first\_name || '(' || **LOWER** (job\_id) || ')' employee **FROM** employees;

**Using Single-Row Functions to Customize Output**  
  
Таблица Employees. Получить список всех сотрудников у которых длина имени больше 10 букв

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** **LENGTH** (first\_name) > 10;

Таблица Employees. Получить список всех сотрудников у которых в имени есть буква 'b' (без учета регистра)

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** **INSTR** (**LOWER** (first\_name), 'b') > 0;

Таблица Employees. Получить список всех сотрудников у которых в имени содержатся минимум 2 буквы 'a'

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** **INSTR** (**LOWER** (first\_name),'a',1,2) > 0;

Таблица Employees. Получить список всех сотрудников зарплата которых кратна 1000

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** **MOD** (salary, 1000) = 0;

Таблица Employees. Получить первое 3х значное число телефонного номера сотрудника если его номер в формате ХХХ.ХХХ.ХХХХ

**Решение**

**SELECT** phone\_number, **SUBSTR** (phone\_number, 1, 3) new\_phone\_number

**FROM** employees

**WHERE** phone\_number **LIKE** '\_\_\_.\_\_\_.\_\_\_\_';

Таблица Departments. Получить первое слово из имени департамента для тех у кого в названии больше одного слова

**Решение**

**SELECT** department\_name,

**SUBSTR** (department\_name, 1, **INSTR** (department\_name, ' ')-1)

first\_word

**FROM** departments

**WHERE** **INSTR** (department\_name, ' ') > 0;

Таблица Employees. Получить имена сотрудников без первой и последней буквы в имени

**Решение**

**SELECT** first\_name, **SUBSTR** (first\_name, 2, **LENGTH** (first\_name) - 2) new\_name

**FROM** employees;

Таблица Employees. Получить список всех сотрудников у которых последняя буква в имени равна 'm' и длинной имени большей 5ти

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** **SUBSTR** (first\_name, -1) = 'm' **AND** **LENGTH**(first\_name)>5;

Таблица Dual. Получить дату следующей пятницы

**Решение**

**SELECT** NEXT\_DAY (**SYSDATE**, 'FRIDAY') next\_friday **FROM** DUAL;

Таблица Employees. Получить список всех сотрудников которые работают в компании больше 17 лет

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** MONTHS\_BETWEEN (**SYSDATE**, hire\_date) / 12 > 17;

Таблица Employees. Получить список всех сотрудников у которых последня цифра телефонного номера нечетная и состоит из 3ех чисел разделенных точкой

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** **MOD** (**SUBSTR** (phone\_number, -1), 2) != 0

**AND** **INSTR** (phone\_number,'.',1,3) = 0

**AND** **INSTR** (phone\_number,'.',1,2) > 0;

Таблица Employees. Получить список всех сотрудников у которых в значении job\_id после знака '\_' как минимум 3 символа но при этом это значение после '\_' не равно 'CLERK'

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** **LENGTH** (**SUBSTR** (job\_id, **INSTR** (job\_id, '\_') + 1)) > 3

**AND** **SUBSTR** (job\_id, **INSTR** (job\_id, '\_') + 1) != 'CLERK';

Таблица Employees. Получить список всех сотрудников заменив в значении PHONE\_NUMBER все '.' на '-'

**Решение**

**SELECT** phone\_number, **REPLACE** (phone\_number, '.', '-') new\_phone\_number

**FROM** employees;

**Using Conversion Functions and Conditional Expressions**  
  
Таблица Employees. Получить список всех сотрудников которые пришли на работу в первый день месяца (любого)

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** TO\_CHAR (hire\_date, 'DD') = '01';

Таблица Employees. Получить список всех сотрудников которые пришли на работу в 2008ом году

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** TO\_CHAR (hire\_date, 'YYYY') = '2008';

Таблица DUAL. Показать завтрашнюю дату в формате: Tomorrow is Second day of January

**Решение**

**SELECT** TO\_CHAR (**SYSDATE**, 'fm""Tomorrow is ""Ddspth ""day of"" Month') info

**FROM** DUAL;

Таблица Employees. Получить список всех сотрудников и дату прихода на работу каждого в формате: 21st of June, 2007

**Решение**

**SELECT** first\_name, TO\_CHAR (hire\_date, 'fmddth ""of"" Month, YYYY') hire\_date

**FROM** employees;

Таблица Employees. Получить список работников с увеличенными зарплатами на 20%. Зарплату показать со знаком доллара

**Решение**

**SELECT** first\_name, TO\_CHAR (salary + salary \* 0.20, 'fm$999,999.00') new\_salary

**FROM** employees;

Таблица Employees. Получить список всех сотрудников которые приши на работу в феврале 2007го года.

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** hire\_date **BETWEEN** **TO\_DATE** ('01.02.2007', 'DD.MM.YYYY')

**AND** **LAST\_DAY** (**TO\_DATE** ('01.02.2007', 'DD.MM.YYYY'));

**SELECT** \*

**FROM** employees

**WHERE** to\_char(hire\_date,'MM.YYYY') = '02.2007';

Таблица DUAL. Вывезти актуальную дату, + секунда, + минута, + час, + день, + месяц, + год

**Решение**

**SELECT** **SYSDATE** **now**,

**SYSDATE** + 1 / (24 \* 60 \* 60) plus\_second,

**SYSDATE** + 1 / (24 \* 60) plus\_minute,

**SYSDATE** + 1 / 24 plus\_hour,

**SYSDATE** + 1 plus\_day,

ADD\_MONTHS (**SYSDATE**, 1) plus\_month,

ADD\_MONTHS (**SYSDATE**, 12) plus\_year

**FROM** DUAL;

Таблица Employees. Получить список всех сотрудников с полными зарплатами (salary + commission\_pct(%)) в формате: $24,000.00

**Решение**

**SELECT** first\_name, salary, TO\_CHAR (salary + salary \* NVL (commission\_pct, 0), 'fm$99,999.00') full\_salary

**FROM** employees;

Таблица Employees. Получить список всех сотрудников и информацию о наличии бонусов к зарплате (Yes/No)

**Решение**

**SELECT** first\_name, commission\_pct, NVL2 (commission\_pct, 'Yes', 'No') has\_bonus

**FROM** employees;

Таблица Employees. Получить уровень зарплаты каждого сотрудника: Меньше 5000 считается Low level, Больше или равно 5000 и меньше 10000 считается Normal level, Больше иои равно 10000 считается High level

**Решение**

**SELECT** first\_name,

salary,

**CASE**

**WHEN** salary < 5000 **THEN** 'Low'

**WHEN** salary >= 5000 **AND** salary < 10000 **THEN** 'Normal'

**WHEN** salary >= 10000 **THEN** 'High'

**ELSE** 'Unknown'

**END**

salary\_level

**FROM** employees;

Таблица Countries. Для каждой страны показать регион в котором он находится: 1-Europe, 2-America, 3-Asia, 4-Africa (без Join)

**Решение**

**SELECT** country\_name country,

**DECODE** (region\_id,

1, 'Europe',

2, 'America',

3, 'Asia',

4, 'Africa',

'Unknown')

region

**FROM** countries;

**SELECT** country\_name

country,

**CASE** region\_id

**WHEN** 1 **THEN** 'Europe'

**WHEN** 2 **THEN** 'America'

**WHEN** 3 **THEN** 'Asia'

**WHEN** 4 **THEN** 'Africa'

**ELSE** 'Unknown'

**END**

region

**FROM** countries;

**Reporting Aggregated Data Using the Group Functions**  
  
Таблица Employees. Получить репорт по department\_id с минимальной и максимальной зарплатой, с ранней и поздней датой прихода на работу и с количествов сотрудников. Сорировать по количеству сотрудников (по убыванию)

**Решение**

**SELECT** department\_id,

**MIN** (salary) min\_salary,

**MAX** (salary) max\_salary,

**MIN** (hire\_date) min\_hire\_date,

**MAX** (hire\_date) max\_hire\_Date,

**COUNT** (\*) **count**

**FROM** employees

**GROUP** **BY** department\_id

**order** **by** **count**(\*) **desc**;

Таблица Employees. Сколько сотрудников имена которых начинается с одной и той же буквы? Сортировать по количеству. Показывать только те где количество больше 1

**Решение**

**SELECT** **SUBSTR** (first\_name, 1, 1) first\_char, **COUNT** (\*)

**FROM** employees

**GROUP** **BY** **SUBSTR** (first\_name, 1, 1)

**HAVING** **COUNT** (\*) > 1

**ORDER** **BY** 2 **DESC**;

Таблица Employees. Сколько сотрудников которые работают в одном и тоже отделе и получают одинаковую зарплату?

**Решение**

**SELECT** department\_id, salary, **COUNT** (\*)

**FROM** employees

**GROUP** **BY** department\_id, salary

**HAVING** **COUNT** (\*) > 1;

Таблица Employees. Получить репорт сколько сотрудников приняли на работу в каждый день недели. Сортировать по количеству

**Решение**

**SELECT** TO\_CHAR (hire\_Date, 'Day') **day**, **COUNT** (\*)

**FROM** employees

**GROUP** **BY** TO\_CHAR (hire\_Date, 'Day')

**ORDER** **BY** 2 **DESC**;

Таблица Employees. Получить репорт сколько сотрудников приняли на работу по годам. Сортировать по количеству

**Решение**

**SELECT** TO\_CHAR (hire\_date, 'YYYY') **year**, **COUNT** (\*)

**FROM** employees

**GROUP** **BY** TO\_CHAR (hire\_date, 'YYYY');

Таблица Employees. Получить количество департаментов в котором есть сотрудники

**Решение**

**SELECT** **COUNT** (**COUNT** (\*)) department\_count

**FROM** employees

**WHERE** department\_id **IS** **NOT** NULL

**GROUP** **BY** department\_id;

Таблица Employees. Получить список department\_id в котором работают больше 30 сотрудников

**Решение**

**SELECT** department\_id

**FROM** employees

**GROUP** **BY** department\_id

**HAVING** **COUNT** (\*) > 30;

Таблица Employees. Получить список department\_id и округленную среднюю зарплату работников в каждом департаменте.

**Решение**

**SELECT** department\_id, **ROUND** (**AVG** (salary)) avg\_salary

**FROM** employees

**GROUP** **BY** department\_id;

Таблица Countries. Получить список region\_id сумма всех букв всех country\_name в котором больше 60ти

**Решение**

**SELECT** region\_id

**FROM** countries

**GROUP** **BY** region\_id

**HAVING** **SUM** (**LENGTH** (country\_name)) > 60;

Таблица Employees. Получить список department\_id в котором работают работники нескольких (>1) job\_id

**Решение**

**SELECT** department\_id

**FROM** employees

**GROUP** **BY** department\_id

**HAVING** **COUNT** (**DISTINCT** job\_id) > 1;

Таблица Employees. Получить список manager\_id у которых количество подчиненных больше 5 и сумма всех зарплат его подчиненных больше 50000

**Решение**

**SELECT** manager\_id

**FROM** employees

**GROUP** **BY** manager\_id

**HAVING** **COUNT** (\*) > 5 **AND** **SUM** (salary) > 50000;

Таблица Employees. Получить список manager\_id у которых средняя зарплата всех его подчиненных находится в промежутке от 6000 до 9000 которые не получают бонусы (commission\_pct пустой)

**Решение**

**SELECT** manager\_id, **AVG** (salary) avg\_salary

**FROM** employees

**WHERE** commission\_pct **IS** NULL

**GROUP** **BY** manager\_id

**HAVING** **AVG** (salary) **BETWEEN** 6000 **AND** 9000;

Таблица Employees. Получить максимальную зарплату из всех сотрудников job\_id которыз заканчивается на слово 'CLERK'

**Решение**

**SELECT** **MAX** (salary) max\_salary

**FROM** employees

**WHERE** job\_id **LIKE** '%CLERK';

**SELECT** **MAX** (salary) max\_salary

**FROM** employees

**WHERE** **SUBSTR** (job\_id, -5) = 'CLERK';

Таблица Employees. Получить максимальную зарплату среди всех средних зарплат по департаменту

**Решение**

**SELECT** **MAX** (**AVG** (salary))

**FROM** employees

**GROUP** **BY** department\_id;

Таблица Employees. Получить количество сотрудников с одинаковым количеством букв в имени. При этом показать только тех у кого длина имени больше 5 и количество сотрудников с таким именем больше 20. Сортировать по длинне имени

**Решение**

**SELECT** **LENGTH** (first\_name), **COUNT** (\*)

**FROM** employees

**GROUP** **BY** **LENGTH** (first\_name)

**HAVING** **LENGTH** (first\_name) > 5 **AND** **COUNT** (\*) > 20

**ORDER** **BY** **LENGTH** (first\_name);

**SELECT** **LENGTH** (first\_name), **COUNT** (\*)

**FROM** employees

**WHERE** **LENGTH** (first\_name) > 5

**GROUP** **BY** **LENGTH** (first\_name)

**HAVING** **COUNT** (\*) > 20

**ORDER** **BY** **LENGTH** (first\_name);

**Displaying Data from Multiple Tables Using Joins**  
  
Таблица Employees, Departaments, Locations, Countries, Regions. Получить список регионов и количество сотрудников в каждом регионе

**Решение**

**SELECT** region\_name, **COUNT** (\*)

**FROM** employees e

**JOIN** departments d **ON** (e.department\_id = d.department\_id)

**JOIN** locations l **ON** (d.location\_id = l.location\_id)

**JOIN** countries c **ON** (l.country\_id = c.country\_id)

**JOIN** regions r **ON** (c.region\_id = r.region\_id)

**GROUP** **BY** region\_name;

Таблица Employees, Departaments, Locations, Countries, Regions. Получить детальную информацию о каждом сотруднике:  
First\_name, Last\_name, Departament, Job, Street, Country, Region

**Решение**

**SELECT** First\_name,

Last\_name,

Department\_name,

Job\_id,

street\_address,

Country\_name,

Region\_name

**FROM** employees e

**JOIN** departments d **ON** (e.department\_id = d.department\_id)

**JOIN** locations l **ON** (d.location\_id = l.location\_id)

**JOIN** countries c **ON** (l.country\_id = c.country\_id)

**JOIN** regions r **ON** (c.region\_id = r.region\_id);

Таблица Employees. Показать всех менеджеров которые имеют в подчинении больше 6ти сотрудников

**Решение**

**SELECT** man.first\_name, **COUNT** (\*)

**FROM** employees emp **JOIN** employees man **ON** (emp.manager\_id = man.employee\_id)

**GROUP** **BY** man.first\_name

**HAVING** **COUNT** (\*) > 6;

Таблица Employees. Показать всех сотрудников которые ни кому не подчиняются

**Решение**

**SELECT** emp.first\_name

**FROM** employees emp

**LEFT** **JOIN** employees man **ON** (emp.manager\_id = man.employee\_id)

**WHERE** man.FIRST\_NAME **IS** NULL;

**SELECT** first\_name

**FROM** employees

**WHERE** manager\_id **IS** NULL;

Таблица Employees, Job\_history. В таблице Employee хранятся все сотрудники. В таблице Job\_history хранятся сотрудники которые покинули компанию. Получить репорт о всех сотрудниках и о его статусе в компании (Работает или покинул компанию с датой ухода)  
Пример:  
first\_name | status  
Jennifer | Left the company at 31 of December, 2006  
Clara | Currently Working

**Решение**

**SELECT** first\_name,

NVL2 (

end\_date,

TO\_CHAR (end\_date, 'fm""Left the company at"" DD ""of"" Month, YYYY'),

'Currently Working')

**status**

**FROM** employees e **LEFT** **JOIN** job\_history j **ON** (e.employee\_id = j.employee\_id);

Таблица Employees, Departaments, Locations, Countries, Regions. Получить список сотрудников которые живут в Europe (region\_name)

**Решение**

**SELECT** first\_name

**FROM** employees

**JOIN** departments **USING** (department\_id)

**JOIN** locations **USING** (location\_id)

**JOIN** countries **USING** (country\_id)

**JOIN** regions **USING** (region\_id)

**WHERE** region\_name = 'Europe';

**SELECT** first\_name

**FROM** employees e

**JOIN** departments d **ON** (e.department\_id = d.department\_id)

**JOIN** locations l **ON** (d.location\_id = l.location\_id)

**JOIN** countries c **ON** (l.country\_id = c.country\_id)

**JOIN** regions r **ON** (c.region\_id = r.region\_id)

**WHERE** region\_name = 'Europe';

Таблица Employees, Departaments. Показать все департаменты в которых работают больше 30ти сотрудников

**Решение**

**SELECT** department\_name, **COUNT** (\*)

**FROM** employees e **JOIN** departments d **ON** (e.department\_id = d.department\_id)

**GROUP** **BY** department\_name

**HAVING** **COUNT** (\*) > 30;

Таблица Employees, Departaments. Показать всех сотрудников которые не состоят ни в одном департаменте

**Решение**

**SELECT** first\_name

**FROM** employees e

**LEFT** **JOIN** departments d **ON** (e.department\_id = d.department\_id)

**WHERE** d.department\_name **IS** NULL;

**SELECT** first\_name

**FROM** employees

**WHERE** department\_id **IS** NULL;

Таблица Employees, Departaments. Показать все департаменты в которых нет ни одного сотрудника

**Решение**

**SELECT** department\_name

**FROM** employees e

**RIGHT** **JOIN** departments d **ON** (e.department\_id = d.department\_id)

**WHERE** first\_name **IS** NULL;

Таблица Employees. Показать всех сотрудников у которых нет ни кого в подчинении

**Решение**

**SELECT** man.first\_name

**FROM** employees emp

**RIGHT** **JOIN** employees man **ON** (emp.manager\_id = man.employee\_id)

**WHERE** emp.FIRST\_NAME **IS** NULL;

Таблица Employees, Jobs, Departaments. Показать сотрудников в формате: First\_name, Job\_title, Department\_name.  
Пример:  
First\_name | Job\_title | Department\_name  
Donald | Shipping | Clerk Shipping

**Решение**

**SELECT** first\_name, job\_title, department\_name

**FROM** employees e

**JOIN** jobs j **ON** (e.job\_id = j.job\_id)

**JOIN** departments d **ON** (d.department\_id = e.department\_id);

Таблица Employees. Получить список сотрудников менеджеры которых устроились на работу в 2005ом году но при это сами эти работники устроились на работу до 2005 года

**Решение**

**SELECT** emp.\*

**FROM** employees emp **JOIN** employees man **ON** (emp.manager\_id = man.employee\_id)

**WHERE** TO\_CHAR (man.hire\_date, 'YYYY') = '2005'

**AND** emp.hire\_date < **TO\_DATE** ('01012005', 'DDMMYYYY');

Таблица Employees. Получить список сотрудников менеджеры которых устроились на работу в январе месяце любого года и длина job\_title этих сотрудников больше 15ти символов

**Решение**

**SELECT** emp.\*

**FROM** employees emp

**JOIN** employees man **ON** (emp.manager\_id = man.employee\_id)

**JOIN** jobs j **ON** (emp.job\_id = j.job\_id)

**WHERE** TO\_CHAR (man.hire\_date, 'MM') = '01' **AND** **LENGTH** (j.job\_title) > 15;

**Using Subqueries to Solve Queries**  
  
Таблица Employees. Получить список сотрудников с самым длинным именем.

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** **LENGTH** (first\_name) =

(**SELECT** **MAX** (**LENGTH** (first\_name)) **FROM** employees);

Таблица Employees. Получить список сотрудников с зарплатой большей средней зарплаты всех сотрудников.

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** salary > (**SELECT** **AVG** (salary) **FROM** employees);

Таблица Employees, Departments, Locations. Получить город в котором сотрудники в сумме зарабатывают меньше всех.

**Решение**

**SELECT** city

**FROM** employees e

**JOIN** departments d **ON** (e.department\_id = d.department\_id)

**JOIN** locations l **ON** (d.location\_id = l.location\_id)

**GROUP** **BY** city

**HAVING** **SUM** (salary) =

( **SELECT** **MIN** (**SUM** (salary))

**FROM** employees e

**JOIN** departments d **ON** (e.department\_id = d.department\_id)

**JOIN** locations l **ON** (d.location\_id = l.location\_id)

**GROUP** **BY** city);

Таблица Employees. Получить список сотрудников у которых менеджер получает зарплату больше 15000.

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** manager\_id **IN** (**SELECT** employee\_id

**FROM** employees

**WHERE** salary > 15000)

Таблица Employees, Departaments. Показать все департаменты в которых нет ни одного сотрудника

**Решение**

**SELECT** \*

**FROM** departments

**WHERE** department\_id **NOT** **IN** (**SELECT** department\_id

**FROM** employees

**WHERE** department\_id **IS** **NOT** NULL);

Таблица Employees. Показать всех сотрудников которые не являются менеджерами

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** employee\_id **NOT** **IN** (**SELECT** manager\_id

**FROM** employees

**WHERE** manager\_id **IS** **NOT** NULL)

Таблица Employees. Показать всех менеджеров которые имеют в подчинении больше 6ти сотрудников

**Решение**

**SELECT** \*

**FROM** employees e

**WHERE** (**SELECT** **COUNT** (\*)

**FROM** employees

**WHERE** manager\_id = e.employee\_id) > 6;

Таблица Employees, Departaments. Показать сотрудников которые работают в департаменте IT

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** department\_id = (**SELECT** department\_id

**FROM** departments

**WHERE** department\_name = 'IT');

Таблица Employees, Jobs, Departaments. Показать сотрудников в формате: First\_name, Job\_title, Department\_name.  
Пример:  
First\_name | Job\_title | Department\_name  
Donald | Shipping | Clerk Shipping

**Решение**

**SELECT** first\_name,

(**SELECT** job\_title

**FROM** jobs

**WHERE** job\_id = e.job\_id)

job\_title,

(**SELECT** department\_name

**FROM** departments

**WHERE** department\_id = e.department\_id)

department\_name

**FROM** employees e;

Таблица Employees. Получить список сотрудников менеджеры которых устроились на работу в 2005ом году но при это сами эти работники устроились на работу до 2005 года

**Решение**

**SELECT** \*

**FROM** employees

**WHERE** manager\_id **IN** (**SELECT** employee\_id

**FROM** employees

**WHERE** TO\_CHAR (hire\_date, 'YYYY') = '2005')

**AND** hire\_date < **TO\_DATE** ('01012005', 'DDMMYYYY');

Таблица Employees. Получить список сотрудников менеджеры которых устроились на работу в январе месяце любого года и длина job\_title этих сотрудников больше 15ти символов

**Решение**

**SELECT** \*

**FROM** employees e

**WHERE** manager\_id **IN** (**SELECT** employee\_id

**FROM** employees

**WHERE** TO\_CHAR (hire\_date, 'MM') = '01')

**AND** (**SELECT** **LENGTH** (job\_title)

**FROM** jobs

**WHERE** job\_id = e.job\_id) > 15;

Подзапросы (subquery) представляют такие запросы, которые могут быть встроены в другие запросы.

Например, определим таблицы для товаров, покупателей и заказов:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22 | CREATE TABLE Products  (      Id SERIAL PRIMARY KEY,      ProductName VARCHAR(30) NOT NULL,      Company VARCHAR(20) NOT NULL,      ProductCount INTEGER DEFAULT 0,      Price NUMERIC NOT NULL  );  CREATE TABLE Customers  (      Id SERIAL PRIMARY KEY,      FirstName VARCHAR(30) NOT NULL  );  CREATE TABLE Orders  (      Id SERIAL PRIMARY KEY,      ProductId INTEGER NOT NULL REFERENCES Products(Id) ON DELETE CASCADE,      CustomerId INTEGER NOT NULL REFERENCES Customers(Id) ON DELETE CASCADE,      CreatedAt DATE NOT NULL,      ProductCount INTEGER DEFAULT 1,      Price NUMERIC NOT NULL  ); |

Таблица Orders содержит ссылки на две другие таблицы через поля ProductId и CustomerId.

Добавим в эти таблицы некоторые данные:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35 | INSERT INTO Products(ProductName, Company, ProductCount, Price)  VALUES ('iPhone X', 'Apple', 2, 66000),  ('iPhone 8', 'Apple', 2, 51000),  ('iPhone 7', 'Apple', 5, 42000),  ('Galaxy S9', 'Samsung', 2, 56000),  ('Galaxy S8 Plus', 'Samsung', 1, 46000),  ('Nokia 9', 'HDM Global', 2, 26000),  ('Desire 12', 'HTC', 6, 38000);    INSERT INTO Customers(FirstName)  VALUES ('Tom'), ('Bob'),('Sam');    INSERT INTO Orders(ProductId, CustomerId, CreatedAt, ProductCount, Price)  VALUES  (      (SELECT Id FROM Products WHERE ProductName='Galaxy S9'),      (SELECT Id FROM Customers WHERE FirstName='Tom'),      '2017-07-11',      2,      (SELECT Price FROM Products WHERE ProductName='Galaxy S9')  ),  (      (SELECT Id FROM Products WHERE ProductName='iPhone 8'),      (SELECT Id FROM Customers WHERE FirstName='Tom'),      '2017-07-13',      1,      (SELECT Price FROM Products WHERE ProductName='iPhone 8')  ),  (      (SELECT Id FROM Products WHERE ProductName='iPhone 8'),      (SELECT Id FROM Customers WHERE FirstName='Bob'),      '2017-07-11',      1,      (SELECT Price FROM Products WHERE ProductName='iPhone 8')  ); |

При добавлении данных в таблицу Orders как раз используются подзапросы. Например, первый заказ был сделан покупателем Tom на товар Galaxy S9. Поэтому в таблицу Orders необходимо сохранить информацию о заказе, где поле ProductId указывает на Id товара Galaxy S9, поле Price - на его цену, а поле CustomerId - на Id покупателя Tom. Но на момент написания запроса нам может быть неизвестен ни Id покупателя, ни Id товара, ни цена товара. В этом случае можно выполнить подзапрос.

Подзапрос представляет команду SELECT и заключается в скобки. В данном же случае при добавлении одного товара выполняется три подзапроса. Каждый подзапрос возвращает одно скалярное значение, например, идентификатор товара или покупателя.

В данном случае подзапросы выполнялись к другой таблице, но могут выполняться и к той же, к которой вызывается основной запрос. Например, найдем товары из таблицы Products, которые имеют минимальную цену:

|  |  |
| --- | --- |
| 1  2  3 | SELECT \*  FROM Products  WHERE Price = (SELECT MIN(Price) FROM Products); |

Или найдем товары, цена которых выше средней:

|  |  |
| --- | --- |
| 1  2  3 | SELECT \*  FROM Products  WHERE Price > (SELECT AVG(Price) FROM Products); |

**Коррелирующие подзапросы**

Подзапросы бывают коррелирующими и некоррелирующими. В примерах выше команды SELECT выполняли фактически один подзапрос для всей команды, например, подзапрос возвращает минимальную или среднюю цену, которая не изменится, сколько бы мы строк не выбирали в основном запросе. Результат такого подзапроса не зависел от строк, которые выбираются в основном запросе. И такой подзапрос выполняется один раз для всего внешнего запроса.

Но кроме того есть **коррелирующие подзапросы** (correlated subquery), результаты которых зависят от строк, которые извлекаются в основном запросе.

Например, выберем все заказы из таблицы Orders, добавив к ним информацию о товаре:

|  |  |
| --- | --- |
| 1  2  3  4  5 | SELECT  CreatedAt,          Price,          (SELECT ProductName FROM Products          WHERE Products.Id = Orders.ProductId) AS Product  FROM Orders; |

Здесь для каждой строки из таблицы Orders будет выполняться подзапрос, результат которого зависит от столбца ProductId. И каждый подзапрос может возвращать различные данные.

Коррелирующий подзапрос может выполняться и для той же таблицы, к которой выполняется основной запрос. Например, выберем из таблицы Products те товары, стоимость которых выше средней цены товаров для данного производителя:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | SELECT ProductName,         Company,         Price,          (SELECT AVG(Price) FROM Products AS SubProds           WHERE SubProds.Company=Prods.Company)  AS AvgPrice  FROM Products AS Prods  WHERE Price >      (SELECT AVG(Price) FROM Products AS SubProds       WHERE SubProds.Company=Prods.Company) |

В данном случае определено два коррелирующих подзапроса. Первый подзапрос определяет спецификацию столбца AvgPrice. Он будет выполняться для каждой строки, извлекаемой из таблицы Products. В подзапрос передается производитель товара и на его основе выбирается средняя цена для товаров именно этого производителя. И так как производитель у товаров может отличаться, то и результат подзапроса в каждом случае также может отличаться.

Второй подзапрос аналогичен, только он используется для фильтрации извлекаемых из таблицы Products. И также он будет выполняться для каждой строки.

Чтобы избежать двойственности при фильтрации в подзапросе при сравнении производителей (SubProds.Company=Prods.Company) для внешней выборки установлен псевдоним Prods, а для выборки из подзапросов определен псевдоним SubProds.