```
CASE
WHEN GROUPING(Emp.Salary) = 1
THEN 'Отдел №'
ELSE Emp.First_Name
END AS "FIRST_NAME",
CASE
WHEN GROUPING(Emp.Salary) = 1
THEN TO_CHAR(Dep.Department_ID)
ELSE Emp.Last_Name
END AS "LAST_NAME" ,
CASE
WHEN GROUPING(Emp.Salary) = 1
THEN '«' || Dep.Department_Name || '»'
ELSE TO_CHAR(Emp.Salary)
END AS "SALARY"
FROM Employees Emp
JOIN Departments Dep
ON Emp.Department_ID = Dep.Department_ID
GROUP BY
GROUPING SETS (
(Dep.Department_ID, Dep.Department_Name),
(Dep.Department_ID, Dep.Department_Name,
Emp.Employee_ID,Emp.First_Name, Emp.Last_Name, Emp.Salary)
)
ORDER BY Dep.Department_ID, GROUPING(Emp.Salary) DESC;
```

5.2. Создать регулярное выражение для проверки сложности пароля. Пароль не должен содержать три последовательные буквы латинского алфавита.

Пример корректного пароля: O9gh$drW3

Пример некорректного пароля: O9gh$frW3 (f, g, h — три последовательные буквы латинского алфавита)

5.2

```
DEFINE PASSWORD1 = 'O9gh$drW3'; --корректен

DEFINE PASSWORD2 = 'O9gh$frW3'; --некорректен

DEFINE PASSWORD3 = 'Vuic0908098987987bjikjljkkljha'; --некорректен

DEFINE PASSWORD4 = 'ABC' --некорректен

DEFINE PASSWORD5 = 'AjhHJ- CJAkka 12b' --некорректен

DEFINE PASSWORD6 = 'BC0918209&%$#@a' --некорректен

DEFINE PASSWORD7 = 'avbababababbababa'--корректен

DEFINE PASSWORD8 =
'91827e981279e8217e987928e7982e79827e98729f98'--корректен

DEFINE alph='ABCDEFGHIJKLMNOPQRSTUVWXYZ'

WITH prog AS

        (SELECT CASE

        WHEN REGEXP_LIKE(UPPER('&PASSWORD2'),'(' ||
substr('&alph',level,1)||'.*'|| substr('&alph',level+1,1) ||'.*'||
substr('&alph',level+2,1)

        || ')|(' || substr('&alph',level,1)||'.*'|| substr('&alph',level+2,1) ||'.*'||
substr('&alph',level+1,1)

        || ')|(' || substr('&alph',level+1,1)||'.*'|| substr('&alph',level,1) ||'.*'||
substr('&alph',level+2,1)

        || ')|(' || substr('&alph',level+1,1)||'.*'|| substr('&alph',level+2,1) ||'.*'||
substr('&alph',level,1)

        || ')|(' || substr('&alph',level+2,1)||'.*'|| substr('&alph',level,1) ||'.*'||
substr('&alph',level+1,1)

        || ')|(' || substr('&alph',level+2,1)||'.*'|| substr('&alph',level+1,1) ||'.*'||
substr('&alph',level,1)||')' )  THEN 1
```

```
        ELSE 0

        END CORRECT

        FROM DUAL

        CONNECT BY LEVEL<=24)

SELECT '&PASSWORD2'
"Password",DECODE(SUM(CORRECT),0,'CORRECT','INCORRECT') FROM
prog;

UNDEFINE PASSWORD1;

UNDEFINE PASSWORD2;

UNDEFINE PASSWORD3;

UNDEFINE PASSWORD4;

UNDEFINE PASSWORD5;

UNDEFINE PASSWORD6;

UNDEFINE PASSWORD7;

UNDEFINE PASSWORD8;

UNDEFINE alph
```

5.4. Дана таблица из двух столбцов ACTION и CODE, в каждом из которых хранятся списки чисел, разделённых пробелами. Создать запрос для разделения данных.

Например, для таблицы:

ACTION	CODE
1000 1100 900	841000 841100 841111
700 500 400	923400 923411

Результат должен быть:

N_LIST	N_POS	ACTION	CODE
1	0	1000 1100 900	841000 841100 841111
	1	1000	841000
	2	1100	841100
	3	900	841111
2	0	700 500 400	923400 923411
	1	700	923400
	2	500	923411
	3	400	

В результате выборки приняты обозначения: N_LIST — порядковый номер списка в исходной таблице, N_POS — порядковый номер числа в списке.

5.4

CREATE TABLE ASSET AS (

SELECT '1000 1100 50' ACTION, '841000 841100' CODE FROM DUAL UNION ALL

SELECT '1000 1100' ACTION, '841000 841100 124' CODE FROM DUAL UNION ALL

SELECT '5000' ACTION, '435' CODE FROM DUAL UNION ALL

SELECT '1000 1100 68', '841000 841100 89' CODE FROM DUAL

);

WITH

```
-- Выборка данных из таблицы, включая ROWNUM
ASSET_TAB AS (
SELECT
ACTION, CODE, ROWNUM ROW_NUM
FROM ASSET
),
-- Применение регулярных выражений для парсинга каждого числа в строке
REGEXP_TAB AS (
SELECT
DISTINCT ACTION ACTION_A,
CODE CODE_C,
CASE
WHEN LEVEL=1
THEN ACTION
-- Парсинг числа в ACTION
ELSE REGEXP_SUBSTR(ACTION, '\d+', 1, LEVEL-1)
END ACTION,
CASE
WHEN LEVEL=1
THEN CODE
-- Парсинг числа в CODE
ELSE REGEXP_SUBSTR(CODE, '\d+', 1, LEVEL-1)
END CODE,
ROW_NUM,
-- Добавление вторичного индекса
```

```
LEVEL - 1 N_POS
FROM ASSET_TAB
-- Соединение по условию нахождения нового числа из ACTION или CODE
CONNECT BY REGEXP_SUBSTR(ACTION, '\d+', 1, LEVEL-1) IS NOT NULL
OR REGEXP_SUBSTR(CODE, '\d+', 1, LEVEL-1) IS NOT NULL
ORDER BY 5 ASC
),
-- Запрос сортировки исходных строк
SORT_TAB AS (
SELECT
ACTION,
CODE,
ROW_NUM,
ROWNUM,
CASE
-- Условие получения строк, содержащихся в исходной таблице
WHEN (REGEXP_COUNT(ACTION, '\d+')>1 OR REGEXP_COUNT(CODE,
'\d+')>1) THEN 1
ELSE 0
END FLEG,
N_POS
FROM REGEXP_TAB
ORDER BY 3, 5 DESC, 6
)
-- Запрос для добавления индексов первого порядка (N_LIST)
SELECT
```

```
CASE

WHEN N_POS = 0

THEN TO_CHAR(ROW_NUM)

ELSE' '

END N_LIST,

N_POS,

NVL(ACTION, ' ') ACTION,

NVL(CODE, ' ') CODE FROM SORT_TAB; DROP TABLE ASSET;
```

5.1. Для двух заданных сотрудников найти их ближайшего общего начальника.

Пример результата:

EMP_1	EMP_2	MAN_ID	MAN_NAME
174	178	149	Zlotkey

5.1

```
select '&&emp1' "EMP1",'&&emp2' "EMP2",MAN1.employee_id
"MAN_ID",Man1.Last_name "MAN_NAME"

from ((select last_name,employee_id,manager_id

from employees

start with employee_id='&emp1'

connect by prior to_number(manager_id)=to_number(employee_id)) MAN1

join

(select last_name,employee_id,manager_id

from employees

start with employee_id='&emp2'

connect by prior to_number(manager_id)=to_number(employee_id)) MAN2

on (MAN1.last_name=MAN2.last_name and
(MAN1.manager_id=MAN2.manager_id or

MAN1.manager_id is null and MAN2.manager_id is null)))

where rownum<2;
```

undefine emp1;

undefine emp2;

Строю для каждого из сотрудников дерево, далее соединяю их по общим начальникам и выбираю первую строку - она и будет первым общим начальником. Если же введен King, можно прописать лишний case, чтобы не выводило ничего ни с кем.

5.3

5.3. Определить список последовательностей подчиненности от сотрудников, не имеющих менеджера (менеджеров высшего уровня), до сотрудников, не имеющих подчиненных. Результат представить в виде:

MAN_LIST
King->Kochhar->Greenberg->Faviet (не имеет подчинённых)
…

select Path

from (select sys_connect_by_path(Last_name,'=>') Path,connect_by_isleaf C

from employees

start with manager_id is null

connect by prior employee_id=manager_id)

where C=1

Используем функцию для вывода всех возможных иерархических связей от менеджеров высшего порядка к низшему порядку. Далее используем псевдостолбец, который дает нам понять, что строка конечна, то есть, не может иметь продолжения по низшему уровню. Эти строки и выводим в ответ

5.6

5.6. Для каждого отдела из таблицы Departments отобразить в виде одной строки с запятой в качестве разделителя фамилии сотрудников, работающих в нем.

Пример результата:

DEPARTMENT_ID	EMP_LIST
10	Kent,Whalen
20	Fay,Hartstein,Smart
…	…
280	(null)

```
WITH Employees_Deps AS (
  SELECT Last_Name, Department_ID, ROWNUM AS TheOrder
  FROM (
      SELECT Last_Name, Department_ID
      FROM Employees
      ORDER BY Department_ID
    )
  WHERE Department_ID IS NOT NULL
),
Min_Employee AS (
  SELECT Department_ID, MIN(TheOrder) AS MinOrder
  FROM Employees_Deps
  GROUP BY Department_ID
)
SELECT
  Department_ID,
  SUBSTR(SYS_CONNECT_BY_PATH(Last_Name, ','), 2) AS Emp_List
FROM Employees_Deps JOIN Min_Employee USING (Department_ID)
  RIGHT JOIN Departments USING (Department_ID)
WHERE CONNECT_BY_ISLEAF = 1
START WITH TheOrder = MinOrder OR TheOrder IS NULL
CONNECT BY PRIOR Department_ID = Department_ID
  AND PRIOR TheOrder + 1 = TheOrder;
```

WITH Employees_Deps AS (

SELECT Last_Name, Department_ID, ROWNUM AS TheOrder

FROM (

SELECT Last_Name, Department_ID

FROM Employees

ORDER BY Department_ID

)

WHERE Department_ID IS NOT NULL

),

Min_Employee AS (

SELECT Department_ID, MIN(TheOrder) AS MinOrder

FROM Employees_Deps

GROUP BY Department_ID

)

```
SELECT
Department_ID,
SUBSTR(SYS_CONNECT_BY_PATH(Last_Name, ','), 2) AS Emp_List
FROM Employees_Deps JOIN Min_Employee USING (Department_ID)
RIGHT JOIN Departments USING (Department_ID)
WHERE CONNECT_BY_ISLEAF = 1
START WITH TheOrder = MinOrder OR TheOrder IS NULL
CONNECT BY PRIOR Department_ID = Department_ID
AND PRIOR TheOrder + 1 = TheOrder;
```

5.5. Дана таблица со следующей структурой:

Id	Linked_id	Part
0	-1	Оглавление
1	0	Глава 1
2	1	Часть 1
3	1	Часть 2
4	0	Глава 2
5	4	Часть 1
6	4	Часть 2

(количество глав и частей произвольное). Создать запрос для вывода оглавления в виде:

Оглавление
1 Глава 1
1.1 Часть 1
1.2 Часть 2
2 Глава 2
2.1 Часть 1
2.2 Часть 2

5.5

```
select "Text" from (select case
when "Linked_id1"=-1 then to_char("Part1")
when "Linked_id1"=0 then substr(to_char("Part1"),-1,1)||' '||to_char("Part1")
else substr(to_char("Part2"),-1,1)||'.'||substr(to_char("Part1"),-1,1)||'
'||to_char("Part1") end "Text",
"ID"
```

from

(select t1."Linked_id" "Linked_id1",t1."Part" "Part1",t2."Linked_id" "Linked_id2",t2."Part" "Part2",t1."Id" "ID"

from T t1

left join T t2

on (t1."Linked_id"=t2."Id"))

order by "ID")

Использую первый столбец для окончательной сортировки, второй для соединения нижних уровней с более верхним, а номер главы и части в главе беру как последний символ в поле с текстовой информацией.

5.7

5.7. В произвольной строке, состоящей из символьных элементов, разделенных запятыми, отсортировать элементы по алфавиту.

Пример результата:

Исходная строка	Результат
abc,cde,ef,gh,mn,test,ss, df,fw,ewe,wwe	abc,cde,df,ef,ewe, fw,gh,mn,ss,test,wwe

define str='&string';

with tab1 as (select substr(','||'&str'||',',instr(','||'&str'||',',',',1,level)+1,instr(','||'&str'||',',',',1,level+1)-instr(','||'&str'||',',',',1,level)-1) "Результат" from dual

connect by instr(','||'&str',',',1,level)!=0),

tab2 as (select * from tab1 order by 1)

select '&str' "Исходная строка",ltrim(sys_connect_by_path("Результат",','),',') "Результат"

from (select "Результат",rownum r from tab2)

where r=(select count(*) from tab2)

start with r=1

connect by prior r=r-1

order by r;

undefine str;

Полезный доп, учит, как конвертить из строки в столбец и наоборот.

5.8. Создать запрос для вывода всех дат, отсутствующих в некоторой последовательности дат. Граница последовательности дат должна определяться некоторым (в данном случае максимальным) значением даты.

Например, для таблицы:

DATE_LIST
16-11-2008
18-11-2008
19-11-2008
23-11-2008

Результат должен быть:

DATE_MISS
17-11-2008
20-11-2008
21-11-2008
22-11-2008

5.8

create table T2 as

(select to_date('16-11-2008','dd-mm-yyyy','nls_date_language=American') Date_list from dual union all

select to_date('18-11-2008','dd-mm-yyyy','nls_date_language=American') Date_list from dual union all

select to_date('19-11-2008','dd-mm-yyyy','nls_date_language=American') Date_list from dual union all

select to_date('23-11-2008','dd-mm-yyyy','nls_date_language=American') Date_list from dual);

with Tab1 as (select distinct (select min(to_date(date_list,'dd-mm-yyyy')) from T2)+level-1 "Date_list"

```
from (select date_list,rownum r from T2)
start with rownum=1
connect by level<=(select max(date_list)-min(date_list) from T2)+1
order by 1)
select * from Tab1
minus
select * from T2;
```

5.9

```
define numb='&num'
undefine numb;
with Tab1 as (select case
when id>to_number(&numb) then 0 else 1 end "Status",
id,rownum r from T3)
--select &numb "Заданное число",
--case when to_number(&numb)<(select min(id) from T3) or
to_number(&numb)>(select max(id) from T3)
--then 'Вне заданных диапазонов'
--else
--case
--when abs(&numb-(select id from T3 where rownum=(select min(r) from Tab1
where "Status"=0)-1))>
--abs(&numb-(select id from T3 where rownum=(select min(r) from Tab1 where
"Status"=0)-0)) then
--(select id from T3 where rownum=(select min(r) from Tab1 where "Status"=0))
--else (select id from T3 where rownum=(select min(r) from Tab1 where
"Status"=0)-1) end
--end
--from dual;
```

5.10. Предположим, что на билетах для проезда в городском транспорте указывают десятичный номер с заданным чётным количеством разрядов n и буквенно-цифровую серию билетной катушки. Нумерация билетов каждой катушки начинается с 000001.

Существует примета, что билет, у которого сумма $^{n}/_{2}$ первых цифр номера равна сумме $^{n}/_{2}$ последних цифр — это «Счастливый билет». Одной командой SELECT вывести количество «счастливых билетов» в каждой серии (одно число). Решение проверить для n = {2, 4, 6}.

Пример результата:

N	CNT
6	55251

5.10

```
select sum("Status")
from (select "Numb",
case
when
to_number(substr("Numb",1,1))+to_number(substr("Numb",2,1))+to_number(substr("Numb",3,1))=
to_number(substr("Numb",4,1))+to_number(substr("Numb",5,1))+to_number(substr("Numb",6,1)) then 1
else 0 end "Status" from (
select lpad(to_char(level),6,'0') "Numb"
from dual
connect by level<1000000))
```

Организую перебор за счет level, далее осуществляю проверку подстрокой и суммирую.

5.12. Из заданных наборов символов исключить те наборы символов, которые отличаются только порядком.

Например, для таблицы:

LIST
rtyew
trwye
rtgbfdsa
tgrbafsd

Результат должен быть:

RESULT
rtyew
rtgbfdsa

5.12

5.13. Создать запрос для определения сумм окладов сотрудников от сотрудников, не имеющих менеджера (менеджеров высшего уровня), до сотрудников, не имеющих подчиненных.

Пример результата:

MAN_LIST	SUM_SAL
King->Kochhar->Greenberg->Faviet	62008
…	…
King->Hartstein->Fay	43000

5.13

Взять за основу доп 5.3, и потом добавить учитывание суммы окладов.

5.14. Создать запрос для определения сумм окладов двух заданных сотрудников, находящихся в явном или неявном подчинении. Если заданы два сотрудника, не подчиняющиеся друг другу, то выводить фамилии и оклады каждого из них.

Например,

1) если заданы сотрудники с идентификаторами 105 и 121 (не подчиняются друг другу), то результат должен быть:

EMP_ID	SALARY
105, 121	4800, 8200

2) если заданы сотрудники с идентификаторами 105 и 102 (105-й сотрудник подчиняется 103-му, а 103-й — 102-му), то результат должен быть:

EMP_ID	SALARY
102, 105	21800

5.14

Построить для каждого из сотрудников дерево начальников, потом проверить для одного из них наличие второго в начальниках через exists, тогда вывести 2), иначе 1.