**Получить имя базы данных**

select name from v$database;

**Получить список профилей безопасности**

select \* from dba\_profiles;

**Получить список сегментов пользователя (ЕСЛИ МЫ ЗАШЛИ ПОД ЕГО ИМЕНЕМ)**

select \* from user\_segments;

**Получить список сегментов пользователя**

select segment\_name from dba\_extents where owner= "user\_name";

**Получить список pdb**

select name from dba\_pdbs;

7)

select \* from v$psfile\_users;

9)

select component, granule\_size from v$sga\_dynamic\_components where current\_size > 0;

**Получить список сессий**

select \* from v$session;

**--1. Определите размеры областей памяти SGA.**

select name, value from v$sga;

**--2. Получите список всех параметров экземпляра.**

select \* from v$instance;

**--3. Получите список управляющих файлов.**

select \* from v$controlfile;

**--4. Сформируйте PFILE.**

create pfile = 'KAS\_PFILE.ora' from memory;

**--5. Создайте таблицу из двух столбцов, один из которых первичный ключ.**

**--Получите перечень всех сегментов. Вставьте данные в таблицу.**

**--Определите, сколько в сегменте таблицы экстентов, их размер в блоках и байтах.**

CREATE TABLE TEST (

ID NUMBER(10) PRIMARY KEY,

NAME VARCHAR2(20)

);

SELECT SEGMENT\_NAME, SEGMENT\_TYPE FROM USER\_SEGMENTS;

INSERT INTO TEST VALUES (1, 'TEST');

SELECT SEGMENT\_NAME, SEGMENT\_TYPE, EXTENTS, BLOCKS, BYTES FROM USER\_SEGMENTS;

**--6. Получите перечень всех процессов СУБД Oracle.**

select \* from v$process;

?????????????

**--Для серверных процессов укажите режим подключения.**

select \* from v$services;

**--Для фоновых укажите работающие в настоящий момент.**

select \* from v$bgprocess where paddr != '00';

**--7. Получите перечень всех табличных пространств и их файлов.**

select tablespace\_name, block\_size, status from dba\_tablespaces;

select FILE\_NAME,TABLESPACE\_NAME,STATUS, MAXBYTES,USER\_BYTES from DBA\_DATA\_FILES

UNION

SELECT FILE\_NAME,TABLESPACE\_NAME,STATUS, MAXBYTES,USER\_BYTES from DBA\_TEMP\_FILES;

**--8. Получите перечень всех ролей.**

select \* from dba\_roles;

**--9. Получите перечень привилегий для определенной роли.**

select \* from DBA\_SYS\_PRIVS where grantee = 'C##RL\_KAS';

**--10. Получите перечень всех пользователей.**

select \* from dba\_users;

**--11. Создайте роль.**

create role C##RL\_KAS;

grant create session,

create table,

create view,

create procedure,

drop any table,

drop any view,

drop any procedure to C##RL\_KAS;

**--12. Создайте пользователя.**

create user C##KAS identified by 1111

default tablespace KAS\_QDATA

profile C##PF\_KAS

account unlock

**--13. Получите перечень всех профилей безопасности.**

select \* from dba\_profiles;

**--14. Получите перечень всех параметров профиля безопасности.**

select \* from DBA\_PROFILES WHERE PROFILE = "PROFILE\_NAME";

**--15. Создайте профиль безопасности.**

create profile C##PF\_KAS limit

password\_life\_time 180

sessions\_per\_user 3

failed\_login\_attempts 7

password\_lock\_time 1

password\_reuse\_time 10

password\_grace\_time default

connect\_time 180

idle\_time 30;

--16. Создайте последовательность S1, со следующими характеристиками:

--начальное значение 1000; приращение 10; минимальное значение 0;

--максимальное значение 10000; циклическую; кэширующую 30 значений в памяти;

--гарантирующую хронологию значений.

--Создайте таблицу T1 с тремя столбцами и введите (INSERT) 10 строк, со значениями из S1.

create sequence S1

start with 1000

increment by 10

minvalue 0

maxvalue 10000

cycle

cache 30

order;

commit;

select S1.nextval from dual; --увеличивает

select S1.currval from dual;

create table T1

(

N1 number(20),

N2 number(20),

N3 number(20)

) storage(buffer\_pool KEEP);

commit;

declare x number:=0;

begin

loop

insert into T1(N1,N2,N3)

values(S1.nextval,S2.nextval,S3.nextval);

x:= x + 1;

exit when x = 10;

end loop;

end;

**--17. Создайте частный и публичный синоним для одной из таблиц и**

**--продемонстрируйте его область видимости.**

**--Найдите созданные синонимы в представлениях словаря Oracle.**

create synonym SYN1 for T1;

insert into SYN1 values (900);

select \* from SYN1;

drop synonym SYN1;

create public synonym SYN\_G for T1;

insert into SYN\_G values (910);

select \* from SYN\_G;

drop synonym SYN\_G;

**--18. Разработайте анонимный блок, демонстрирующий возникновение и обработку исключений**

--WHEN TO\_MANY\_ROWS и NO\_DATA\_FOUND.

declare

num1 number(5):= 10;

num2 number(5):= 0;

rez number(5);

begin

rez := num1/num2;

exception

when no\_data\_found

then dbms\_output.put\_line('Данные не найдены');

when too\_many\_rows

then dbms\_output.put\_line('В результате несколько строк');

when others

then dbms\_output.put\_line('error' || sqlcode || ': ' || sqlerrm);

end;

**--19. Получите перечень всех файлов групп журналов повтора.**

select \* from V$LOGFILE;

**--20. Определите текущую группу журналов повтора.**

select \* from V$LOG where status = 'CURRENT';

**--21. Получите перечень контрольных файлов.**

select \* from v$controlfile;

**22. Создайте таблицу и вставьте в нее 100 записей. Найдите таблицу и ее свойства в представлениях словаря.**

CREATE TABLE TESTT (

ID NUMBER(10) PRIMARY KEY,

NAME VARCHAR2(20)

);

begin for i in 1..100 loop insert into TESTT values (i, 'TESTT'); end loop; end;

select \* from dba\_tables where table\_name = 'TESTT';

**--23. Получите список сегментов табличного пространства.**

select segment\_name, segment\_type

from DBA\_SEGMENTS where tablespace\_name='KAS';

24. Выведите список всех объектов, доступных пользователю.

**25. Вычислите количество блоков, занятых таблицей.**

select sum(blocks) from dba\_segments where tablespace\_name = 'HTACORE';

**--26. Выведите список текущих сессий.**

select \* from v$session;

**--27. Выведите, производится ли архивирование журналов повтора.**

select dest\_name, status from v$archive\_dest\_status;

--28. Создайте представление с определенными параметрами.

29. Создайте database link с определенными параметрами.

30. Продемонстрируйте эскалацию исключения.

**Процедуры и функции**

**--9. Создайте процедуру, которая увеличивает на 10% стоимость определенного товара.**

**--Параметр – наименование товара. Обработайте возможные ошибки**

select product\_id, price from products where product\_id = '4100Y';

create or replace procedure prod\_5(

N number,

product products.product\_id%type

)

as

cursor prod is

select product\_id, price from products for update;

prc prod%rowtype;

begin

open prod;

loop

update product\_id set price = price \* N where current of prod;

dbms\_output.put\_line(prc.product\_id || ' ' || prc.price);

end loop;

close prod;

exception

when no\_data\_found

then dbms\_output.put\_line('Данные не найдены');

when too\_many\_rows

then dbms\_output.put\_line('В результате несколько строк');

when others

then dbms\_output.put\_line('error' || sqlcode || ': ' || sqlerrm);

end;

**--7. Создайте процедуру, которая выводит список покупателей, у которых есть заказы в этом временном периоде.**

**--Параметры – дата начала периода, дата окончания периода. Обработайте возможные ошибки**

select customers.company, orders.order\_date

from orders inner join customers on orders.cust = customers.cust\_num

where orders.order\_date between '01.12.07' and '12.02.08';

create or replace procedure prod\_4(

date\_from orders.order\_date%type,

date\_to orders.order\_date%type

)

as

cursor c\_order is

select customers.company, orders.order\_date from orders inner join customers on orders.cust = customers.cust\_num

where orders.order\_date between date\_from and date\_to;

cust c\_order%rowtype;

begin

open c\_order;

loop

fetch c\_order into cust;

exit when c\_order%notfound;

dbms\_output.put\_line(cust.order\_date || ' ' || cust.company);

end loop;

close c\_order;

exception

when no\_data\_found

then dbms\_output.put\_line('Данные не найдены');

when too\_many\_rows

then dbms\_output.put\_line('В результате несколько строк');

when others

then dbms\_output.put\_line('error' || sqlcode || ': ' || sqlerrm);

end;

begin

prod\_4(to\_date('01/12/2007', 'dd/mm/yyyy'), to\_date('12/02/2008', 'dd/mm/yyyy'));

end;

**--5. Создайте процедуру, которая выводит список покупателей, в порядке убывания общей стоимости заказов.**

**--Параметры – дата начала периода, дата окончания периода. Обработайте возможные ошибки.**

select customers.company, orders.amount, orders.order\_date

from orders inner join customers on orders.cust = customers.cust\_num

where orders.order\_date between '01.12.07' and '12.02.08'

order by orders.amount;

select sum(orders.amount) from orders inner join customers on orders.cust = customers.cust\_num

where customers.company = 'Zetacorp';

drop procedure prod\_3

--просто выводит заказы за определенный период в порядке убывания стоимости

create or replace procedure prod\_3(

date\_from orders.order\_date%type,

date\_to orders.order\_date%type

)

as

cursor c\_order is

select customers.company, orders.amount, orders.order\_date from orders inner join customers on orders.cust = customers.cust\_num

where orders.order\_date between date\_from and date\_to order by orders.amount;

cust c\_order%rowtype;

begin

open c\_order;

loop

fetch c\_order into cust;

exit when c\_order%notfound;

dbms\_output.put\_line(cust.order\_date || ' ' || cust.company || ' ' || cust.amount);

end loop;

close c\_order;

exception

when no\_data\_found

then dbms\_output.put\_line('Данные не найдены');

when too\_many\_rows

then dbms\_output.put\_line('В результате несколько строк');

when others

then dbms\_output.put\_line('error' || sqlcode || ': ' || sqlerrm);

end;

begin

prod\_3(to\_date('01/12/2007', 'dd/mm/yyyy'), to\_date('12/02/2008', 'dd/mm/yyyy'));

end;

**--3. Создайте процедуру, которая выводит список всех товаров, приобретенных покупателем, с указанием суммы продаж по возрастанию.**

**--Параметр – наименование покупателя. Обработайте возможные ошибки.**

select orders.product, orders.amount, customers.company

from orders inner join customers on orders.cust = customers.cust\_num

where customers.company = 'Zetacorp'

order by orders.amount;

create or replace procedure prod\_2(name\_cus customers.company%type)

as

cursor c\_order is

select orders.product, orders.amount, customers.company from orders inner join customers on orders.cust = customers.cust\_num

where customers.company = name\_cus order by orders.amount;

cust c\_order%rowtype;

begin

open c\_order;

loop

fetch c\_order into cust;

exit when c\_order%notfound;

dbms\_output.put\_line(cust.product || ' ' || cust.company || ' ' || cust.amount);

end loop;

close c\_order;

exception

when no\_data\_found

then dbms\_output.put\_line('Данные не найдены');

when too\_many\_rows

then dbms\_output.put\_line('В результате несколько строк');

when others

then dbms\_output.put\_line('error' || sqlcode || ': ' || sqlerrm);

end;

begin

prod\_2('Zetacorp');

end;

**Создайте процедуру, которая выводит список заказов и их итоговую стоимость для определенного покупателя.**

**--Параметр – наименование покупателя. Обработайте возможные ошибки.**

select orders.order\_num, orders.product, orders.amount, customers.company

from orders inner join customers on orders.cust = customers.cust\_num

where customers.company = 'JCP Inc.';

select sum(orders.amount) from orders inner join customers on orders.cust = customers.cust\_num

where customers.company = 'JCP Inc.';

drop procedure prod\_1;

create or replace procedure prod\_1(name\_cus customers.company%type)

as

cursor c\_order is

select orders.order\_num, orders.product, orders.amount, customers.company from orders inner join customers on orders.cust = customers.cust\_num

where customers.company = name\_cus;

cust c\_order%rowtype;

begin

open c\_order;

loop

fetch c\_order into cust;

exit when c\_order%notfound;

dbms\_output.put\_line(cust.order\_num || ' ' || cust.product || ' ' || cust.amount || ' ' || cust.company);

end loop;

close c\_order;

exception

when no\_data\_found

then dbms\_output.put\_line('Данные не найдены');

when too\_many\_rows

then dbms\_output.put\_line('В результате несколько строк');

when others

then dbms\_output.put\_line('error' || sqlcode || ': ' || sqlerrm);

end;

begin

prod\_1('JCP Inc.');

end;