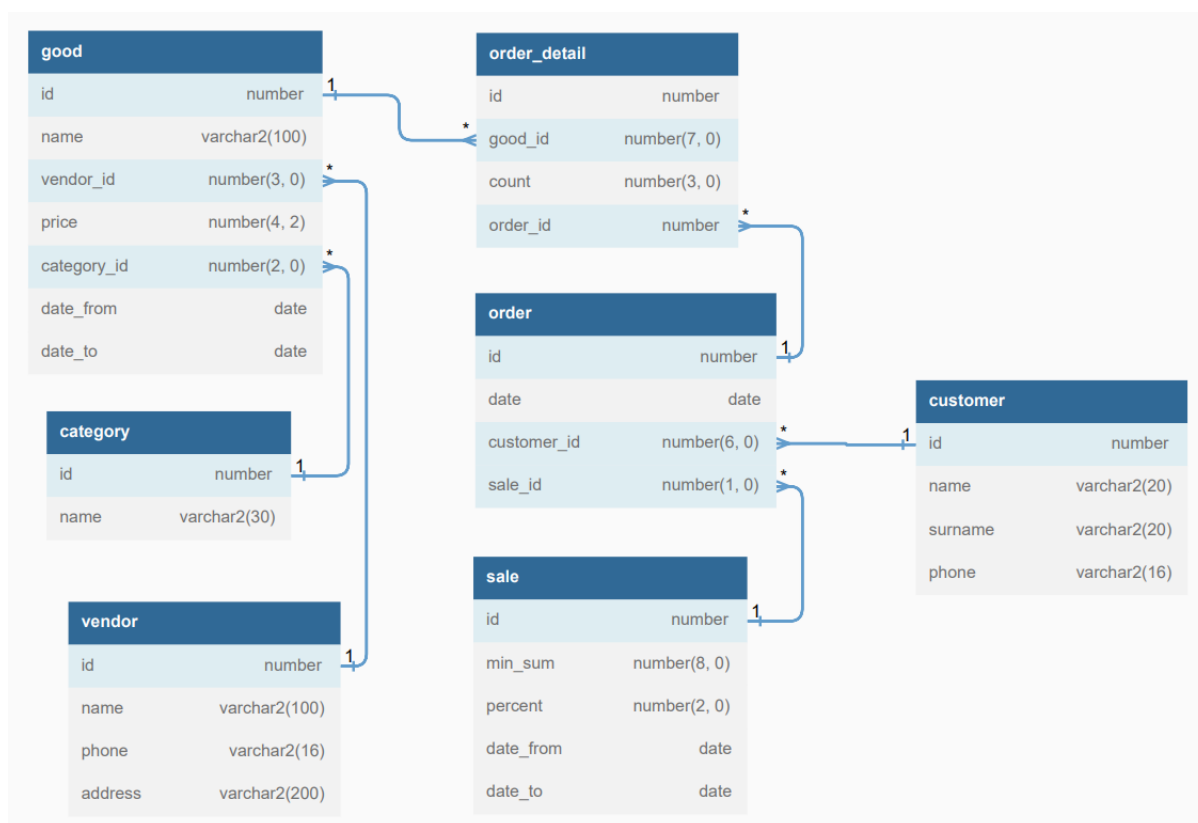


МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение
высшего образования
«Национальный исследовательский ядерный университет «МИФИ»»

ЛАБОРАТОРНАЯ РАБОТА №2-2:
«Пользователи. Роли. Привилегии. Профили.»



Выполнила студентка группы Б19-515
Щербакова Александра

Москва, 2022 г.

1. Создание PDB.

Создана новая PDB shop_pdb с администратором shop_main_admin, которому предоставлены все возможные привилегии внутри shop_pdb (код см. в Приложении А).

```
SQL> create pluggable database SHOP_PDB admin user shop_main_admin identified by pas
2 default tablespace SHOP_PDB_USERS
3 datafile 'C:\APP\ALEXM\PRODUCT\18.0.0\ORADATA\XE\SHOP_PDB\SHOP_PDB_USERS01.dbf' size 250m autoextend on
4 file_name_convert=('C:\APP\ALEXM\PRODUCT\18.0.0\ORADATA\XE\pdbseed', 'C:\APP\ALEXM\PRODUCT\18.0.0\ORADATA\XE\SHOP_PD
B');

Pluggable database created.

SQL> show pdbs;

  CON_ID CON_NAME                                OPEN MODE  RESTRICTED
-----
2 PDB$SEED                                     READ ONLY  NO
3 XEPDB1                                       READ WRITE NO
4 SHOP_PDB                                     MOUNTED

SQL> alter pluggable database SHOP_PDB open;

Pluggable database altered.

SQL> show pdbs;

  CON_ID CON_NAME                                OPEN MODE  RESTRICTED
-----
2 PDB$SEED                                     READ ONLY  NO
3 XEPDB1                                       READ WRITE NO
4 SHOP_PDB                                     READ WRITE NO

SQL> ALTER SESSION SET CONTAINER=shop_pdb;

Session altered.

SQL> GRANT
2 all privileges
3 to shop_main_admin
4 CONTAINER=CURRENT;

Grant succeeded.
```

2. Создание схем и таблиц.

В данной лабораторной вся работа будет производиться в одной схеме, потому что это было бы существенным усложнением для базы данных текущего размера.

Лирическое отступление – фантазии об увеличении базы данных.

Тем не менее, если пофантазировать насчет расширения базы данных, то встанет вопрос о разделении схем. Выдавать каждому приложению привилегии на каждую таблицу по отдельности достаточно трудоемко. Поэтому (не в данной лабе, а в теории) в качестве метода создания разделяемой среды базы данных был бы выбран метод «несколько схем». Предположим, что на данный момент в инфраструктуре не предполагается наличие приложений, которые будут пользоваться объектами из одной и

той же схемы. Таким образом, при необходимости можно легко переделать базу данных на тип «разделяемая схема».

// Источник: <https://www.interface.ru/fset.asp?Url=/oracle/0003.htm>

<i>Название схемы</i>	<i>Таблицы, входящие в схему</i>	<i>Назначение</i>	<i>Дополнительные таблицы, если предположить, что модель БД будет расширена</i>
prchdpt	vendor	(purchasing_department) Отдел закупок. Вся информация о поставщиках и поставках на склад	Количество, наименования товаров в поставках; время поставок
cstsvc	customer	(customer_service) Отдел по работе с клиентами. Информация о покупателях	Адрес для доставки на дом; выбранные покупателем категории скидки/кешбека
warehouse	good, category	Информация о товарах	Наличие товаров на складе
orders	Order, order_detail, sale	Информация о совершенных покупках	

Пользователи prchdpt, cstsvc, warehouse, orders обладают всеми привилегиями внутри своей схемы. Это уникальные пользователи, для них роли не будут создаваться. Однако этим пользователям реальный человек не соответствует – ими всеми управляет shop_main_admin.

Также для каждой схемы нужно создать роль администратора (обязательно с правом на создание ролей и новых пользователей в этой схеме). И дальше уже можно создавать пользователей, описанных с следующей части лабораторной работы.

Лирическое отступление закончено – вернемся к примеру с одной схемой.

- 1) От имени пользователя shop_main_admin созданы все таблицы, согласно erd-диаграмме (приложение В).
- 2) Созданы и заполнены тестовыми данными все таблицы (код в приложении С).

- 3) Для некоторых таблиц созданы представления, чтобы ограничить доступ пользователей к полям, которые им не требуются. Для всех таблиц и представлений созданы публичные синонимы для удобства использования (приложение D).

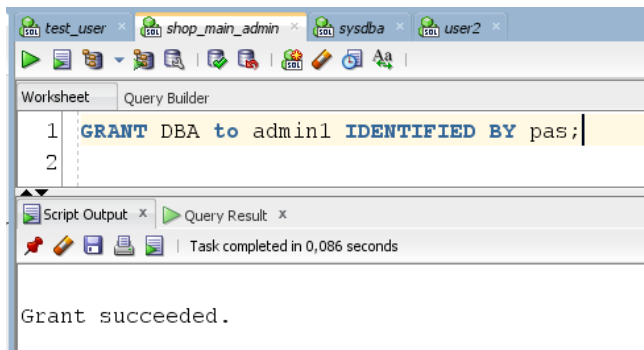
3. Создание ролей для новых пользователей.

Разработана ролевая модель в соответствии с задачами пользователей.

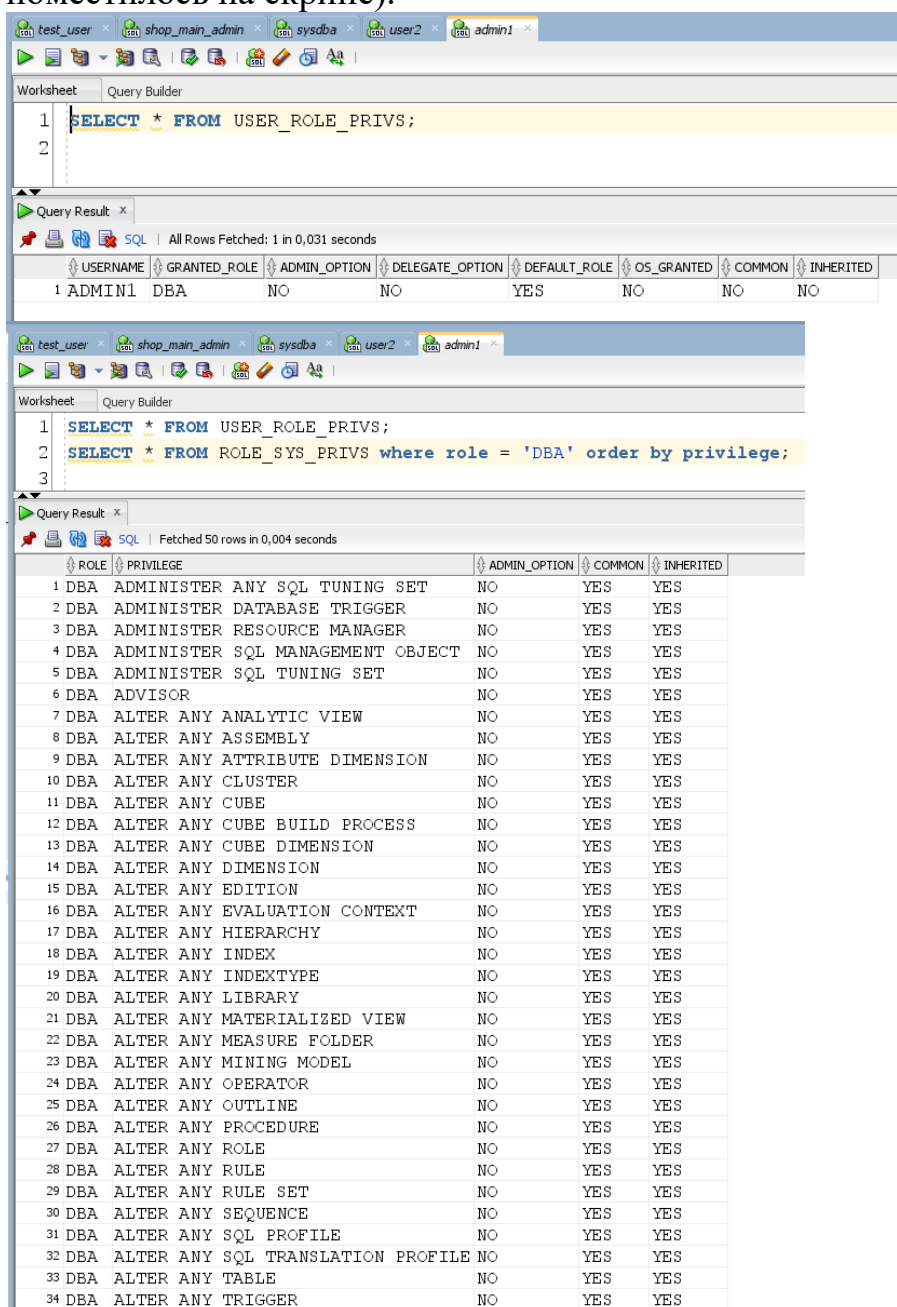
Код для создания ролей, а также одного тестового пользователя для каждой роли в приложении E.

	Роль	Задачи	Привилегии
1	admin	Любая администраторская деятельность. Создание новых ролей и пользователей. Любая работа с объектами.	Встроенная в Oracle роль DBA. Все привилегии: SELECT * FROM ROLE_SYS_PRIVS where role = 'DBA';
2	accounting	Бухгалтерия. Вычисляет суммарные доходы и расходы (такой таблицы пока нет). Read only.	Select from orders, order_detail, goodV, saleV
3	client_manager	Менеджер по работе с клиентами. Добавляет и изменяет информацию о клиентах. Может вычислить сумму покупок клиента	Select, insert, update on customer Select on orders, order_detail, goodV, saleV
4	seller	Продавец. Добавляет новые заказы. Нужно знать сумму покупок клиента, чтобы применить правильную скидку.	Select, insert, update on order, order_detail Select on customer, good, sale
5	prch_manager	Менеджер по работе с поставщиками. Добавляет, изменяет информацию о поставщиках. Добавляет новые товары и категории.	Select, insert, update on vendor, category, good
6	analytic	Анализ продаж: какие товары/категории товаров лучше продаются. В какое время дня больше покупок. Read only.	Select on good, category, order_detail, ordersV, saleV

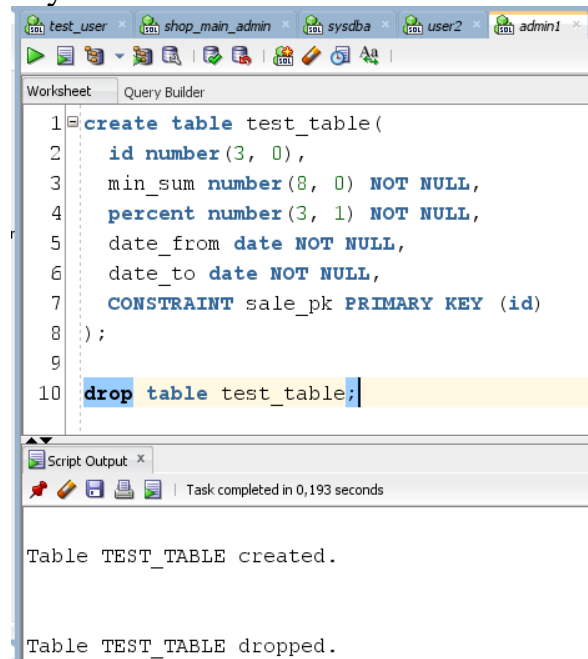
- 1) Создан один из админов базы данных admin1 (от лица пользователя shop_main_admin):



Войдем в бд от лица admin1 и посмотрим, какие привилегии ему предоставлены. Роли – только DBA, привилегий всего 235 штук (не поместилось на скрине).



Протестируем работоспособность привилегий create any table, drop any table:



The screenshot shows the SQL Developer interface with the 'Query Builder' tab active. The SQL script in the editor consists of two lines: `create table test_table(` followed by column definitions (`id number(3, 0),` `min_sum number(8, 0) NOT NULL,` `percent number(3, 1) NOT NULL,` `date_from date NOT NULL,` `date_to date NOT NULL,` `CONSTRAINT sale_pk PRIMARY KEY (id)`) and `);` on line 9, and `drop table test_table;` on line 10. The 'Script Output' window at the bottom shows the results of the execution: 'Table TEST_TABLE created.' on the first line and 'Table TEST_TABLE dropped.' on the second line. The status bar indicates 'Task completed in 0,193 seconds'.

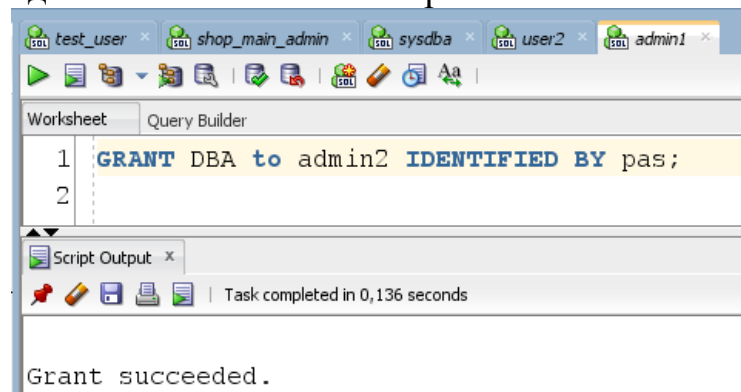
```
1 create table test_table(  
2   id number(3, 0),  
3   min_sum number(8, 0) NOT NULL,  
4   percent number(3, 1) NOT NULL,  
5   date_from date NOT NULL,  
6   date_to date NOT NULL,  
7   CONSTRAINT sale_pk PRIMARY KEY (id)  
8 );  
9  
10 drop table test_table;
```

Script Output: x
Task completed in 0,193 seconds

Table TEST_TABLE created.

Table TEST_TABLE dropped.

Протестируем работу привилегии grant any role, создав второго админа с аналогичными правами:



The screenshot shows the SQL Developer interface with the 'Query Builder' tab active. The SQL script in the editor consists of two lines: `GRANT DBA to admin2 IDENTIFIED BY pas;` on line 1 and an empty line on line 2. The 'Script Output' window at the bottom shows the result: 'Grant succeeded.' The status bar indicates 'Task completed in 0,136 seconds'.

```
1 GRANT DBA to admin2 IDENTIFIED BY pas;  
2
```

Script Output: x
Task completed in 0,136 seconds

Grant succeeded.

- 2) От лица admin1 создадим роль для бухгалтера, а также одного пользователя с этой ролью.

The screenshot shows the SQL Developer interface with a script in the Query Builder. The script contains SQL commands to create a role named 'accounting' and grant it various privileges, then create a user named 'accountant1' and grant the 'accounting' role to it. The script is as follows:

```
1 -- create roles
2
3 create role accounting;
4 grant create session      to accounting;
5 grant select on orders    to accounting;
6 grant select on order_detail to accounting;
7 grant select on good      to accounting;
8 grant select on saleV     to accounting;
9
10 -- create users
11 CREATE USER accountant1
12 IDENTIFIED BY pas
13 DEFAULT TABLESPACE SHOP_PDB_USERS
14 TEMPORARY TABLESPACE temp
15 QUOTA 500M ON SHOP_PDB_USERS;
16 grant accounting to accountant1;
17
```

Below the script, the 'Script Output' pane shows the execution results:

```
Grant succeeded.

Grant succeeded.

Grant succeeded.

User ACCOUNTANT1 created.

Grant succeeded.
```

// далее скриншоты создания пользователей приводиться не будут, чтобы не перегружать отчет. Код для создания всех ролей и пользователей в приложении Е.

Подключимся к базе как бухгалтер. Можно посмотреть все таблицы, к которым есть доступ. SaleV здесь не отображается, так как это представление, а не таблица.

The screenshot shows the SQL Developer interface with a query in the Query Builder. The query is designed to list all tables owned by 'SHOP_MAIN_ADMIN'. The query is as follows:

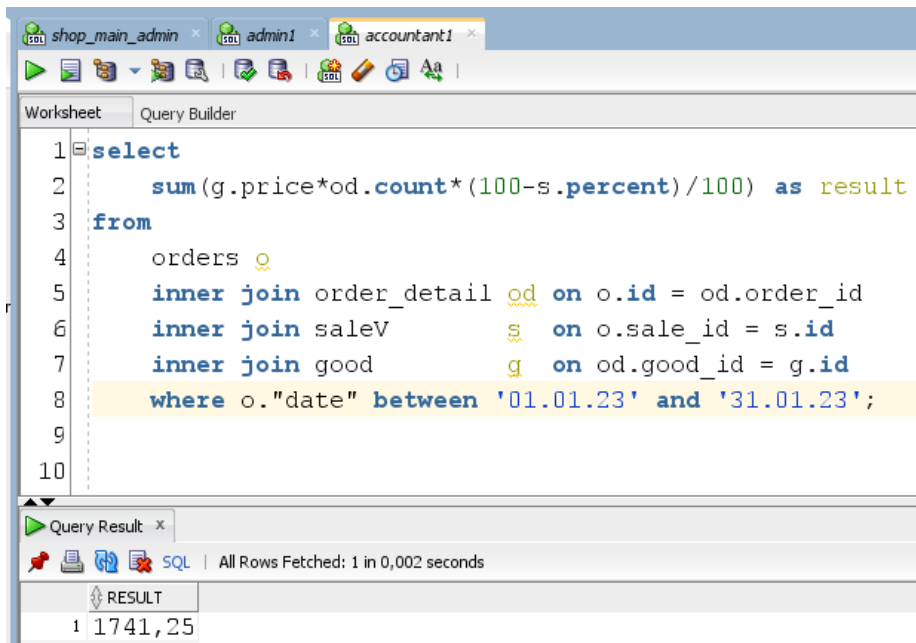
```
1 SELECT owner, table_name
2 FROM all_tables where owner = 'SHOP_MAIN_ADMIN';
```

Below the query, the 'Query Result' pane shows the execution results:

```
Query Result
SQL | All Rows Fetched: 3 in 0,052 seconds
```

OWNER	TABLE_NAME
SHOP MAIN ADMIN	GOOD
SHOP MAIN ADMIN	ORDERS
SHOP MAIN ADMIN	ORDER DETAIL

Протестируем работу бухгалтера. Стандартный запрос для этой роли – вычислить сумму продаж (за определенное время, например, январь).



The screenshot shows a database query editor with three tabs: 'shop_main_admin', 'admin1', and 'accountant1'. The 'Query Builder' tab is active, displaying a SQL query. The query calculates the sum of sales for January 2023 by joining the 'orders', 'order_detail', 'saleV', and 'good' tables. The result is shown in the 'Query Result' tab below the editor.

```

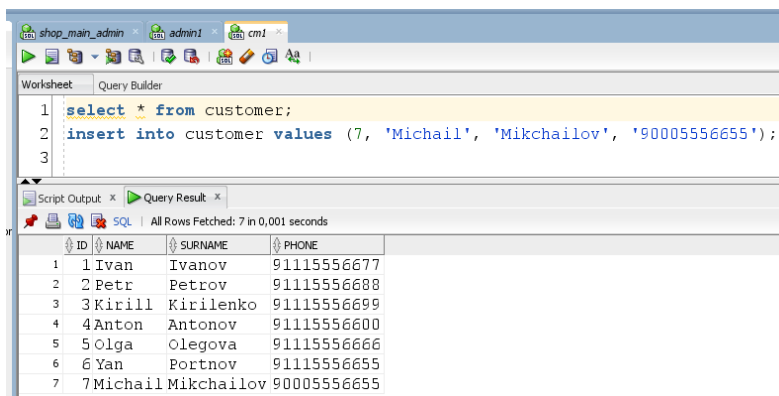
1 select
2     sum(g.price*od.count*(100-s.percent)/100) as result
3 from
4     orders o
5     inner join order_detail od on o.id = od.order_id
6     inner join saleV s on o.sale_id = s.id
7     inner join good g on od.good_id = g.id
8     where o."date" between '01.01.23' and '31.01.23';
9
10

```

Query Result: All Rows Fetched: 1 in 0,002 seconds

RESULT
1 1741,25

- 3) Роль client_manager и пользователь cm1
Стандартный запрос – добавить пользователя.



The screenshot shows a database query editor with three tabs: 'shop_main_admin', 'admin1', and 'cm1'. The 'Query Builder' tab is active, displaying a SQL query. The query consists of a 'select' statement followed by an 'insert' statement. The result is shown in the 'Query Result' tab below the editor.

```

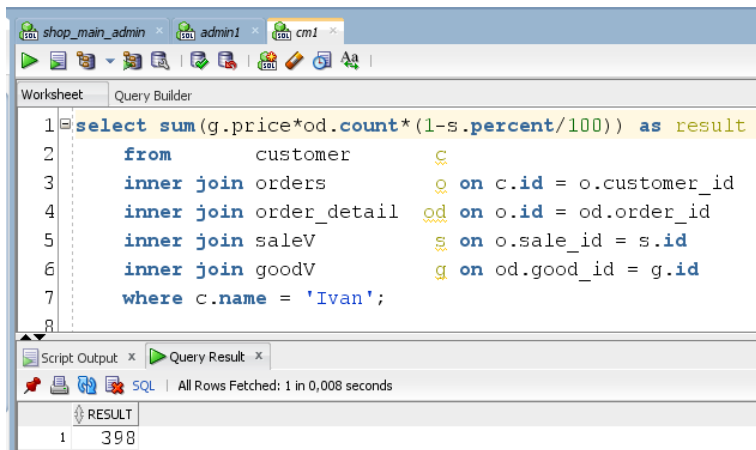
1 select * from customer;
2 insert into customer values (7, 'Michail', 'Mikchailov', '90005556655');
3

```

Script Output: All Rows Fetched: 7 in 0,001 seconds

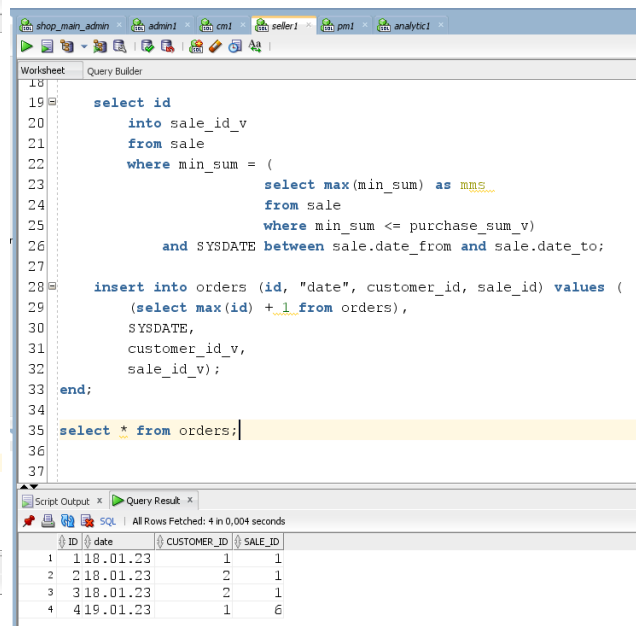
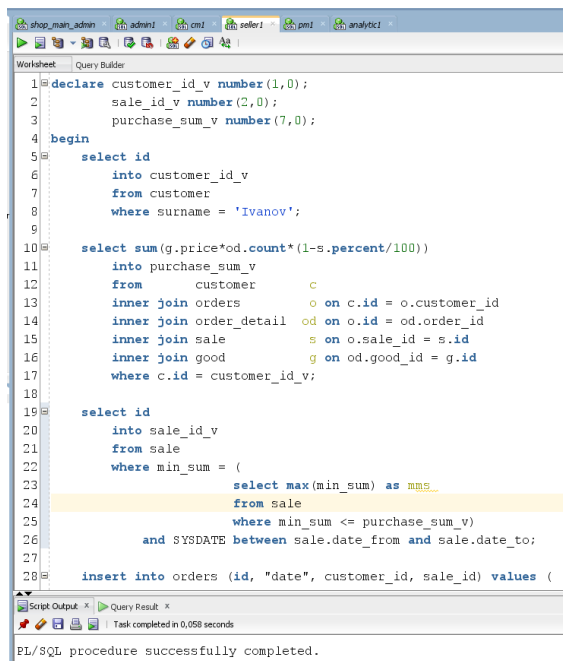
ID	NAME	SURNAME	PHONE
1	Ivan	Ivanov	91115556677
2	Petr	Petrov	91115556688
3	Kirill	Kirilenko	91115556699
4	Anton	Antonov	91115556600
5	Olga	Olegova	91115556666
6	Yan	Portnov	91115556655
7	Michail	Mikchailov	90005556655

Еще запрос: вычислить сумму покупок клиента.



4) Роль seller и пользователь seller1.

Добавить новый заказ. Для этого сначала вычисляется сумма предыдущих покупок клиента, выбирается соответствующая скидка, и только потом добавляется заказ.



Теперь добавляются детали (сколько и каких товаров куплено) в order_details.

Worksheet Query Builder

```

1 insert into order_detail values (9, 4, 2, 4);
2 insert into order_detail values (10, 5, 5, 4);
3 insert into order_detail values (11, 6, 1, 4);
4
5 select * from order_detail;
6

```

Script Output x Query Result x

SQL All Rows Fetched: 11 in 0,001 seconds

ID	GOOD_ID	COUNT	ORDER_ID
1	1	1	5
2	2	2	10
3	3	3	2
4	4	4	7
5	5	5	1
6	6	6	2
7	7	1	3
8	8	1	20
9	9	4	2
10	10	5	5
11	11	6	1

5) Роль prch_manager и пользователь pm1.
Изменить информацию о поставщике.

Worksheet Query Builder

```

1 update vendor
2   set address = 'Kirovsk'
3   where name = 'chocolate vendor 1';
4 select * from vendor;

```

Script Output x Query Result x

SQL All Rows Fetched: 5 in 0,001 seconds

ID	NAME	PHONE	ADDRESS
1	fruit vendor	81234567788	Moscow
2	fruit and vegetables vendor	81234568899	Krasnodar
3	chocolate vendor 1	81112223344	Kirovsk
4	water vendor 1	89998887766	Yaroslavl
5	chocolate vendor 2	83334445566	Minsk

Добавить новый товар из существующей категории.

Worksheet Query Builder

```

1 insert into good values (7, 'sparkling water', 5, 50, 4, SYSDATE, SYSDATE + 30);
2 select * from good;
3

```

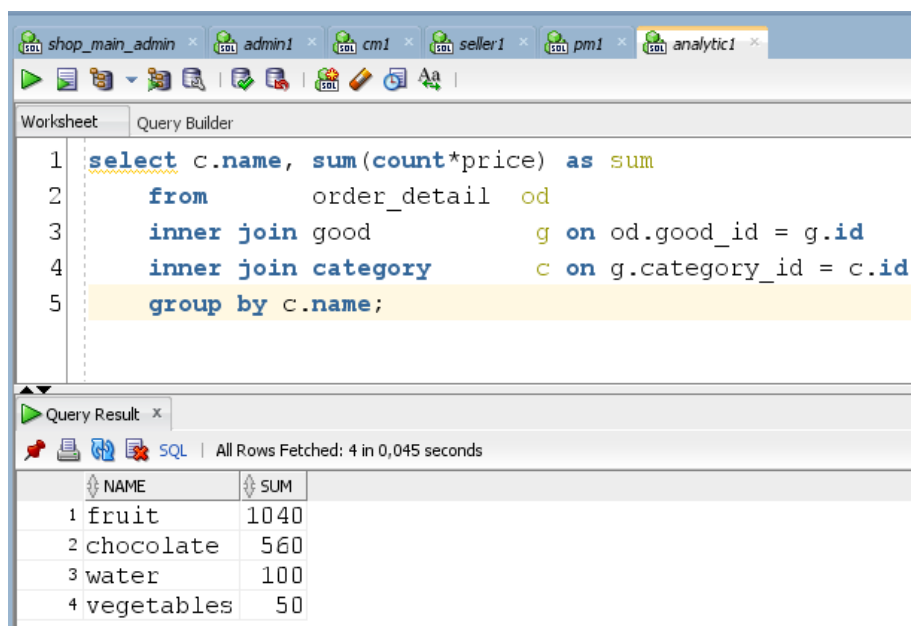
Script Output x Query Result x

SQL All Rows Fetched: 7 in 0,001 seconds

ID	NAME	VENDOR_ID	PRICE	CATEGORY_ID	DATE_FROM	DATE_TO
1	apple	1	30	101.01.22	01.01.23	
2	banana	1	20	101.01.22	01.01.23	
3	cucumber	2	25	201.01.22	01.01.23	
4	alionka	3	70	301.01.22	01.01.23	
5	alpen gold	4	70	301.01.22	01.01.23	
6	sviatoy istochnik	5	50	401.01.22	01.01.23	
7	sparkling water	5	50	419.01.23	18.02.23	

6) Роль analytic и пользователь analytic1.

Вычислить сумму проданных товаров из всех категорий.



The screenshot shows a database query builder interface with multiple tabs at the top: shop_main_admin, admin1, cm1, seller1, pm1, and analytic1. The 'Query Builder' tab is active, displaying a SQL query in a text area. The query is as follows:

```
1 select c.name, sum(count*price) as sum
2   from      order_detail  od
3   inner join good         g on od.good_id = g.id
4   inner join category     c on g.category_id = c.id
5   group by c.name;
```

Below the query editor, the 'Query Result' tab is active, showing the results of the query. It indicates 'All Rows Fetched: 4 in 0,045 seconds'. The results are displayed in a table with two columns: NAME and SUM.

	NAME	SUM
1	fruit	1040
2	chocolate	560
3	water	100
4	vegetables	50

4. Заключение

В данной работе реализована сегментация работы пользователей с базой данных магазина. Созданы базовые роли для работы с данными, созданы пользователи для каждой роли, для них протестирована работа привилегий.

Если БД содержит на порядки больше таблиц, выдавать объектные привилегии, как сделано в данной лабораторной, слишком трудозатратно. Поэтому в теории следовало бы расширить БД и, продумав, какие приложения будут использовать какие таблицы, разбить БД на схемы.

Также можно было бы сделать представления из соединения нескольких таблиц, так как соединение, например, order-order_detail-good используется для похожих целей пользователями различных ролей.

Приложение А. Создание базы данных.

```
/ as sysdba
create pluggable database SHOP_PDB admin user shop_main_admin identified
by pas
default tablespace SHOP_PDB_USERS
datafile
'C:\APP\ALEXM\PRODUCT\18.0.0\ORADATA\XE\SHOP_PDB\SHOP_PDB
_USERS01.dbf' size 250m autoextend on
file_name_convert=('C:\APP\ALEXM\PRODUCT\18.0.0\ORADATA\XE\pdbse
ed','C:\APP\ALEXM\PRODUCT\18.0.0\ORADATA\XE\SHOP_PDB');

show pdbs;
alter pluggable database SHOP_PDB open;
show pdbs;

ALTER SESSION SET CONTAINER=shop_pdb;
GRANT
    all privileges
    to shop_main_admin
    CONTAINER=CURRENT;
```

Приложение В. Создание таблиц.

```
-- shop_main_amdin/pas@"localhost:1521/shop_pdb"
```

```
CREATE TABLE sale (  
  id number(3, 0),  
  min_sum number(8, 0) NOT NULL,  
  percent number(3, 1) NOT NULL,  
  date_from date NOT NULL,  
  date_to date NOT NULL,  
  CONSTRAINT sale_pk PRIMARY KEY (id)  
);
```

```
CREATE TABLE vendor (  
  id number(4, 0),  
  name varchar2(100) NOT NULL,  
  phone varchar2(16) NOT NULL,  
  address varchar2(200),  
  CONSTRAINT vendor_pk PRIMARY KEY (id)  
);
```

```
CREATE TABLE category (  
  id number(3, 0),  
  name varchar2(30) NOT NULL,  
  CONSTRAINT category_pk PRIMARY KEY (id)  
);
```

```
CREATE TABLE customer (  
  id number(6, 0),  
  name varchar2(20) NOT NULL,  
  surname varchar2(20) NOT NULL,  
  phone varchar2(16),  
  CONSTRAINT customer_pk PRIMARY KEY (id)  
);
```

```
CREATE TABLE good (  
  id number(9, 0),  
  name varchar2(100) NOT NULL,  
  vendor_id number(4, 0) NOT NULL,  
  price number(6, 2) NOT NULL,  
  category_id number(3, 0) NOT NULL,  
  date_from date NOT NULL,
```

```
date_to date NOT NULL,  
CONSTRAINT good_pk PRIMARY KEY (id),  
CONSTRAINT good_fk_category  
  FOREIGN KEY (category_id)  
  REFERENCES category (id),  
CONSTRAINT good_fk_vendor  
  FOREIGN KEY (vendor_id)  
  REFERENCES vendor (id)  
);
```

```
CREATE TABLE orders (  
  id number(7, 0),  
  "date" date NOT NULL,  
  customer_id number(6, 0) NOT NULL,  
  sale_id number(3, 0),  
  CONSTRAINT orders_pk PRIMARY KEY (id),  
  CONSTRAINT orders_fk_cistomer  
    FOREIGN KEY (customer_id)  
    REFERENCES customer (id),  
  CONSTRAINT orders_fk_sale  
    FOREIGN KEY (sale_id)  
    REFERENCES sale (id)  
);
```

```
CREATE TABLE order_detail (  
  id number(8, 0),  
  good_id number(9, 0) NOT NULL,  
  count number(3, 0) NOT NULL,  
  order_id number(7, 0) NOT NULL,  
  CONSTRAINT order_detail_pk PRIMARY KEY (id),  
  CONSTRAINT order_detail_fk_good  
    FOREIGN KEY (good_id)  
    REFERENCES good (id),  
  CONSTRAINT order_detail_fk_order  
    FOREIGN KEY (order_id)  
    REFERENCES orders (id)  
);
```

Приложение С. Заполнение таблиц.

```
-- category
insert into category values (1, 'fruit');
insert into category values (2, 'vegetables');
insert into category values (3, 'chocolate');
insert into category values (4, 'water');

-- vendor
insert into vendor values (1, 'fruit vendor', '81234567788', 'Moscow');
insert into vendor values (2, 'fruit and vegetables vendor', '81234568899',
'Krasnodar');
insert into vendor values (3, 'chocolate vendor 1', '81112223344', 'Podolsk');
insert into vendor values (4, 'chocolate vendor 2', '83334445566', 'Minsk');
insert into vendor values (5, 'water vendor 1', '89998887766', 'Yaroslavl');

-- customer
insert into customer values (1, 'Ivan', 'Ivanov', '91115556677');
insert into customer values (2, 'Petr', 'Petrov', '91115556688');
insert into customer values (3, 'Kirill', 'Kirilenko', '91115556699');
insert into customer values (4, 'Anton', 'Antonov', '91115556600');
insert into customer values (5, 'Olga', 'Olegova', '91115556666');
insert into customer values (6, 'Yan', 'Portnov', '91115556655');

declare
    c_date_to   date := to_date('01-01-2023', 'dd-mm-yyyy');
    c_date_from date := to_date('01-01-2022', 'dd-mm-yyyy');
begin
    -- good
    insert into good values (1, 'apple', 1, 30, 1, c_date_from, c_date_to);
    insert into good values (2, 'banana', 1, 20, 1, c_date_from, c_date_to);
    insert into good values (3, 'cucumber', 2, 25, 2, c_date_from, c_date_to);
    insert into good values (4, 'alionka', 3, 70, 3, c_date_from, c_date_to);
    insert into good values (5, 'alpen gold', 4, 70, 3, c_date_from, c_date_to);
    insert into good values (6, 'sviatoy istochnik', 5, 50, 4, c_date_from,
c_date_to);

    -- sale
    insert into sale values (1, 0, 0.5, c_date_from, c_date_to);
    insert into sale values (2, 30000, 1, c_date_from, c_date_to);
    insert into sale values (3, 50000, 2, c_date_from, c_date_to);
```

```
        insert into sale values (4, 100000, 5, c_date_from, c_date_to);
        insert into sale values (5, 2000000, 7, c_date_from, c_date_to);
end;
```

```
-- orders
insert into orders values (1, SYSDATE, 1, 1);
insert into orders values (2, SYSDATE, 2, 1);
insert into orders values (3, SYSDATE, 2, 1);
```

```
-- order_detail
insert into order_detail values (1, 1, 5, 1);
insert into order_detail values (2, 2, 10, 1);
insert into order_detail values (3, 3, 2, 1);
insert into order_detail values (4, 4, 7, 2);
insert into order_detail values (5, 5, 1, 2);
insert into order_detail values (6, 6, 2, 2);
insert into order_detail values (7, 1, 3, 2);
insert into order_detail values (8, 1, 20, 3);
```


Приложение D. Создание представлений и публичных синонимов.

-- create views

create view saleV as

select id, percent

from sale;

create view ordersV as

select id, "date", sale_id

from orders;

create view goodV as

select id, name, price

from good;

-- create public synonyms for all tables

create public synonym vendor

for shop_main_admin.vendor;

create public synonym customer

for shop_main_admin.customer;

create public synonym good

for shop_main_admin.good;

create public synonym category

for shop_main_admin.category;

create public synonym orders

for shop_main_admin.orders;

create public synonym order_detail

for shop_main_admin.order_detail;

create public synonym sale

for shop_main_admin.sale;

-- create public synonyms for all views

create public synonym saleV

for shop_main_admin.saleV;

create public synonym ordersV

for shop_main_admin.ordersV;

create public synonym goodV

for shop_main_admin.goodV;

Приложение Е. Создание ролей и пользователей.

-- create roles

```
create role accounting;  
grant create session      to accounting;  
grant select on orders    to accounting;  
grant select on order_detail to accounting;  
grant select on good      to accounting;  
grant select on saleV     to accounting;
```

```
create role client_manager;  
grant create session      to client_manager;  
grant  
    select,  
    insert,  
    update on customer    to client_manager;  
grant select on orders    to client_manager;  
grant select on order_detail to client_manager;  
grant select on goodV     to client_manager;  
grant select on saleV     to client_manager;
```

```
create role seller;  
grant create session      to seller;  
grant  
    select,  
    insert,  
    update on orders      to seller;  
grant  
    select,  
    insert,  
    update on order_detail to seller;  
grant select on customer  to seller;  
grant select on good      to seller;  
grant select on sale      to seller;
```

```
create role prch_manager;  
grant create session      to prch_manager;  
grant  
    select,  
    insert,
```

```
        update on vendor to prch_manager;
grant
    select,
    insert,
    update on category to prch_manager;
grant
    select,
    insert,
    update on good to prch_manager;
```

```
create role analytic;
grant create session to analytic;
grant select on ordersV to analytic;
grant select on order_detail to analytic;
grant select on good to analytic;
grant select on saleV to analytic;
grant select on category to analytic;
```

```
-- create users
CREATE USER accountant1
    IDENTIFIED BY pas
    DEFAULT TABLESPACE SHOP_PDB_USERS
    TEMPORARY TABLESPACE temp
    QUOTA 500M ON SHOP_PDB_USERS;
grant accounting to accountant1;
```

```
CREATE USER cm1
    IDENTIFIED BY pas
    DEFAULT TABLESPACE SHOP_PDB_USERS
    TEMPORARY TABLESPACE temp
    QUOTA 500M ON SHOP_PDB_USERS;
grant client_manager to cm1;
```

```
CREATE USER seller1
    IDENTIFIED BY pas
    DEFAULT TABLESPACE SHOP_PDB_USERS
    TEMPORARY TABLESPACE temp
    QUOTA 500M ON SHOP_PDB_USERS;
```

grant seller to seller1;

```
CREATE USER pm1
  IDENTIFIED BY pas
  DEFAULT TABLESPACE SHOP_PDB_USERS
  TEMPORARY TABLESPACE temp
  QUOTA 500M ON SHOP_PDB_USERS;
grant prch_manager to pm1;
```

```
CREATE USER analytic1
  IDENTIFIED BY pas
  DEFAULT TABLESPACE SHOP_PDB_USERS
  TEMPORARY TABLESPACE temp
  QUOTA 500M ON SHOP_PDB_USERS;
grant analytic to analytic1;
```