COFFEE SHOP DATABASE MANAGEMENT

This system provides client service, order, and employee performance at the coffee shop.Also helps to find a quick access to any information such as clients,employees,menu,etc.It shows the step from the client's visit to the client's payment of the bill.

Ingkar Koilybai,Nurai Abugalieva,
Aknur Satybay,Nuray Zhumabay,Ali Maksotov

ERD

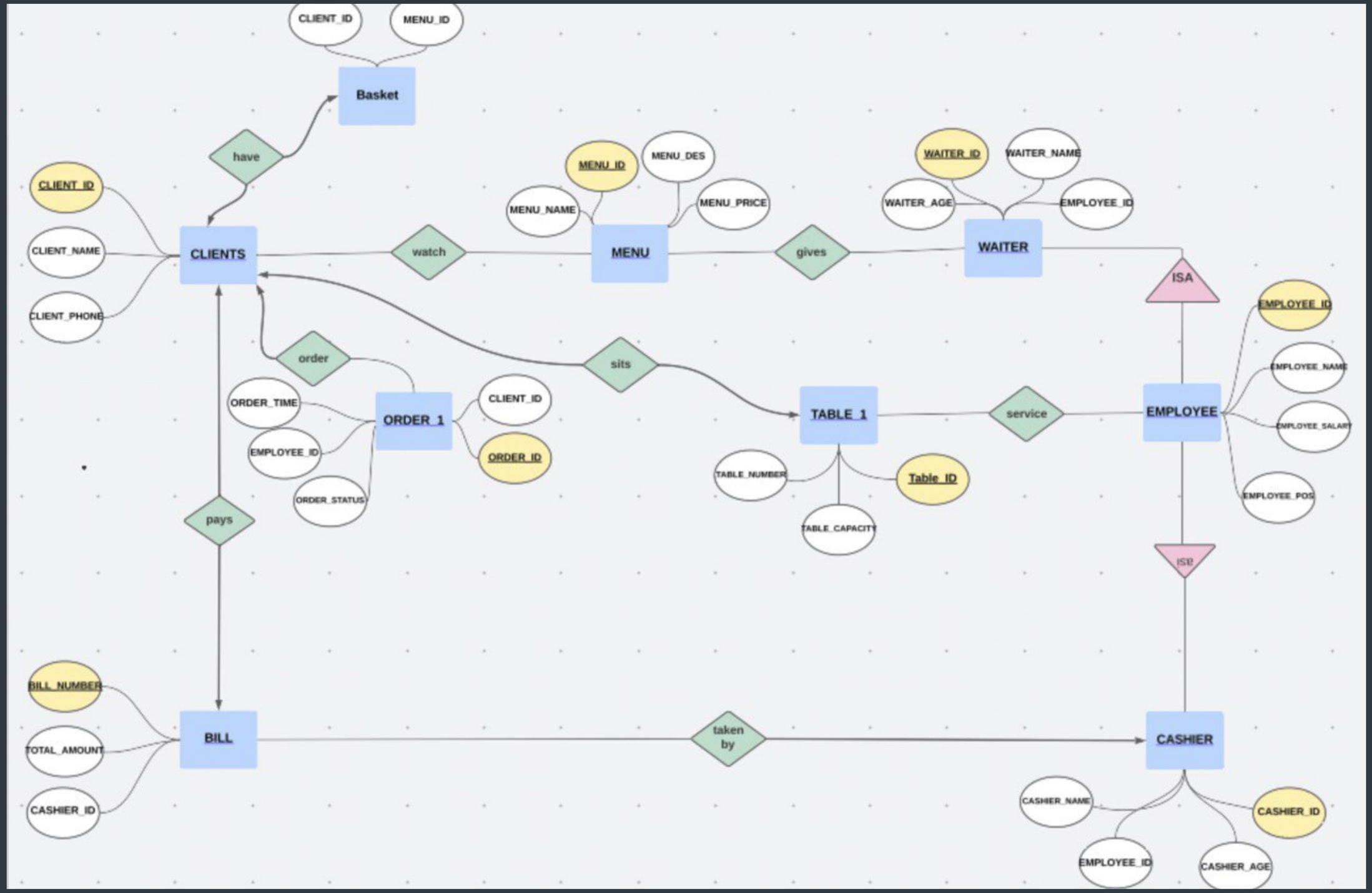

CLIENT_ID
MENU_ID
Basket
have
CLIENT_ID
CLIENT_NAME
CLIENT_PHONE
CLIENTS
watch
MENU_ID
MENU_DES
MENU_NAME
MENU_PRICE
MENU
gives
WAITER_ID
WAITER_NAME
WAITER_AGE
EMPLOYEE_ID
WAITER
ISA
EMPLOYEE_ID
EMPLOYEE_NAME
EMPLOYEE_SALARY
EMPLOYEE_POS
EMPLOYEE
order
sits
ORDER_TIME
EMPLOYEE_ID
ORDER_STATUS
ORDER_1
CLIENT_ID
ORDER_ID
TABLE_1
service
TABLE_NUMBER
Table_ID
TABLE_CAPACITY
pays
ISA
BILL_NUMBER
TOTAL_AMOUNT
CASHIER_ID
BILL
taken by
CASHIER
CASHIER_NAME
CASHIER_ID
EMPLOYEE_ID
CASHIER_AGE

Table Client
{Client_id}->{Client_name},{Client_phone}
Client_id->p key

CLIENT_ID	CLIENT_NAME	CLIENT_PHONE
2	Remy	248-647-8968
4	Debby	412-139-2609
5	Kelcy	304-774-0196

Table Employee
{Employee_id}->{Employee_name},{Employee_pos},
{Employee_salary}
Employee_id->p key

EMPLOYEE_ID	EMPLOYEE_NAME	EMPLOYEE_POS	EMPLOYEE_SALARY
1	Rogers	chef	110000
25	Glynn	chef	110000
26	Damita	chef	110000
15	Aisha	waiter	110000

Table Waiter(subclass of Employee)
{Waiter_id},{Emp_id}->{Waiter_name},
{Waiter_age}
Waiter_id->p key
Emp_id->foreign key

WAITER_ID	WAITER_NAME	WAITER_AGE	EMPLOYEE_ID
20	Aisha	30	15
21	Aisha	18	16
22	Akhan	22	17
23	Alisher	19	18

koilibai04@bk.ru sdu en Copyright © 1999, 2023, Oracle and/or its affiliates.

Table Cashier(subclass of Employee)
{Cashier_id},{Emp_id}->{Cashier_name},{Cashier_age}
Cashier_id->p key
Emp_id->foreign key

CASHIER_ID	CASHIER_NAME	CASHIER_AGE	EMPLOYEE_ID
200	Nana	18	30
2	Boysey	24	2
3	Karel	31	3
202	Sabrina	20	32

koilibai04@bk.ru sdu en Copyright © 1999, 2023, Oracle and/or its affiliates.

Table Bill

- **Bill_number, Cashier_id, Client_id -> Total_amount**
- **Bill_number, Cashier_id, Client_id -> Payment_method**

BILL_NUMBER	TOTAL_AMOUNT	CASHIER_ID	CLIENT_ID	PAYMENT_METHOD
70	1300	200	1	CARD
71	11000	201	2	CASH
72	20000	202	3	CARD

MENU_ID	MENU_DESCRIPTION	MENU_NAME	MENU_PRICE
2	boloneze	pasta	1790
3	ice limonade	moxito	1790
4	classic	black tea	690
5	with meat	pelmeni	1490

Table Menu
{Menu_id}->{Menu_name},
{Menu_description,Menu_price}
Menu_id->p key

Table Order_1(subclass of Employee,Clients)
{Order_id},{Emp_id},{Client_id}->
{Order_time},{Order_status}
Waiter_id->p key
Emp_id,Client_id->foreign keys

Results | Explain | Describe | Saved SQL | History

ORDER_ID	ORDER_STATUS	ORDER_TIME	CLIENT_ID	TABLE_ID	EMPLOYEE_ID
51	reservation	0:16	99	60	8
59	runout	12:00	7	68	1
40	ready	12:00	1	60	1

Table Table_1
{Table_id}->{Table_number},
{Table_capacity}
Table_id->p key

Results | Explain | Describe | Saved SQL | History

TABLE_ID	TABLE_NUMBER	TABLE_CAPACITY
60	1	15
61	2	10
62	3	15
63	4	20
64	5	5

Table Basket
Client_id,Menu_ID->foreign key

Results | Explain | Describe | Saved SQL | History

CLIENT_ID	MENU_ID
1	2
1	5
1	2

FUNCTION

creating a function which calculates total amount of food that client ordered

```
create or replace function f3 return VARCHAR2 is
TOTAL integer := 0;
t varchar(32767);
tt integer := 0;
total_clients integer;
i integer := 1;
b boolean := FALSE;
cursor c is
select CLIENT_ID, MENU_ID
from basket;
begin
    select count(*) into total_clients
    from clients;
    while i <= total_clients loop
        b := FALSE;
        for row in c loop
            if row.CLIENT_ID = i then
                b := TRUE;
            end if;
        end loop;
        if b = TRUE then
            for row in c loop
                if row.CLIENT_ID = i then
                    select MENU_PRICE into tt from menu where menu_id = row.MENU_ID;
                    TOTAL := TOTAL + tt;
                end if;
            end loop;
            t := t  CHR(10)  ' Client ID is '  i   ' and his total is ' || TOTAL;
            TOTAL := 0;
            tt := 0;
        end if;
        i := i + 1;
    end loop;
    return t;
end;

DECLARE
 n varchar2(32767);
BEGIN
 n:=f3() ;
 dbms_output.put_line(n);
END;
```

Results | Explain | Describe | Saved SQL | History

```
Client ID is 1 and his total is 5070
Client ID is 3 and his total is 1790
```

FUNCTION 2

```
1   CREATE OR REPLACE FUNCTION cashierAge
2   RETURN NUMBER IS
3    counts number(20):=0;
4
5   begin
6     SELECT count(*) into counts FROM CASHIER WHERE CASHIER_AGE<18 ;
7     RETURN counts;
8
9    EXCEPTION
10    WHEN NO_DATA_FOUND
11    THEN
12     RETURN 0;
13    WHEN OTHERS
14    THEN
15     RETURN -1;
16   END;
17
18  DECLARE
19     n number(20);
20  BEGIN
21     n := cashierAge();
22     dbms_output.put_line('Total number of cashiers who are under 18 is ' || n);
23  END;
24
```

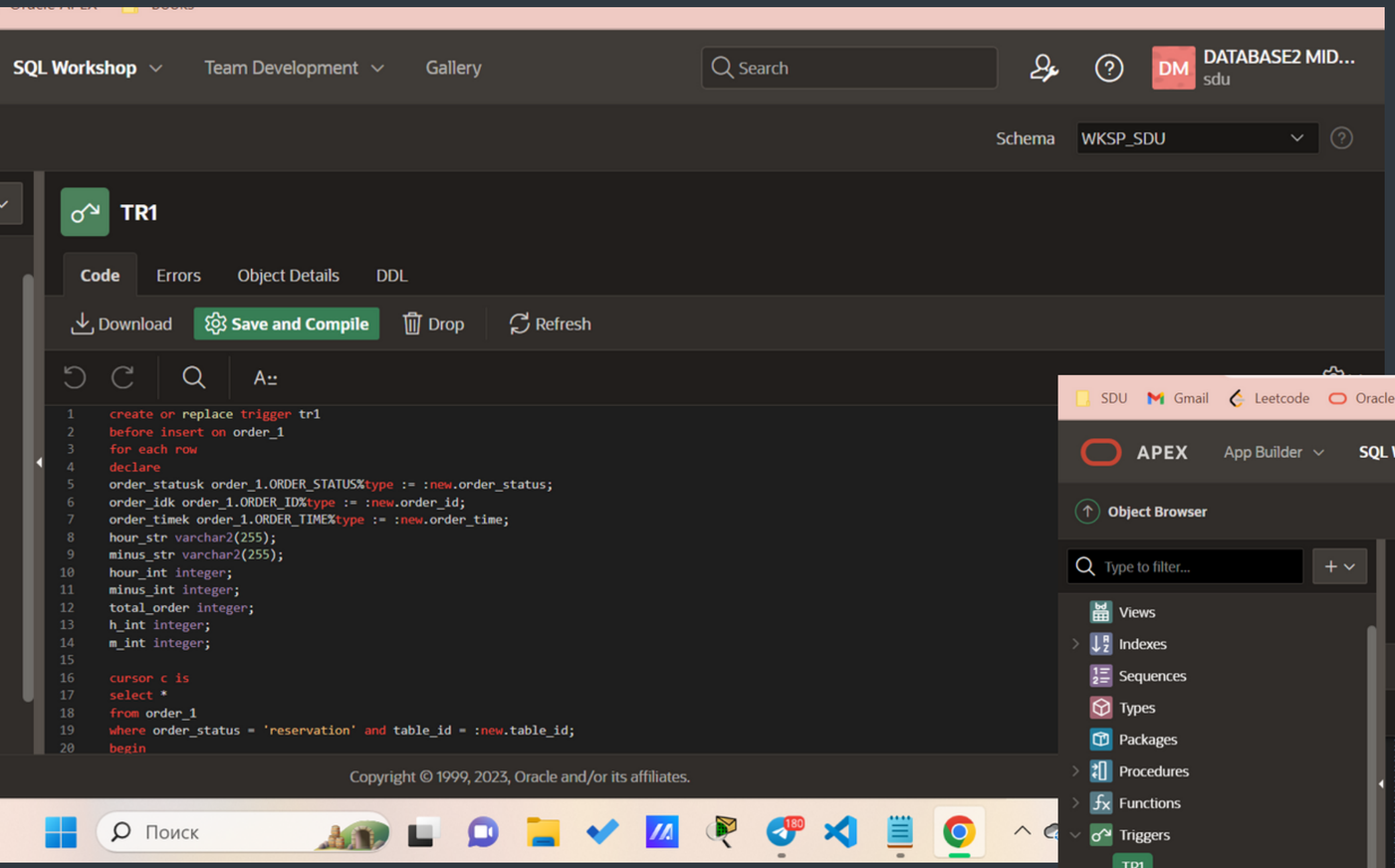
Results | Explain | Describe | Saved SQL | History

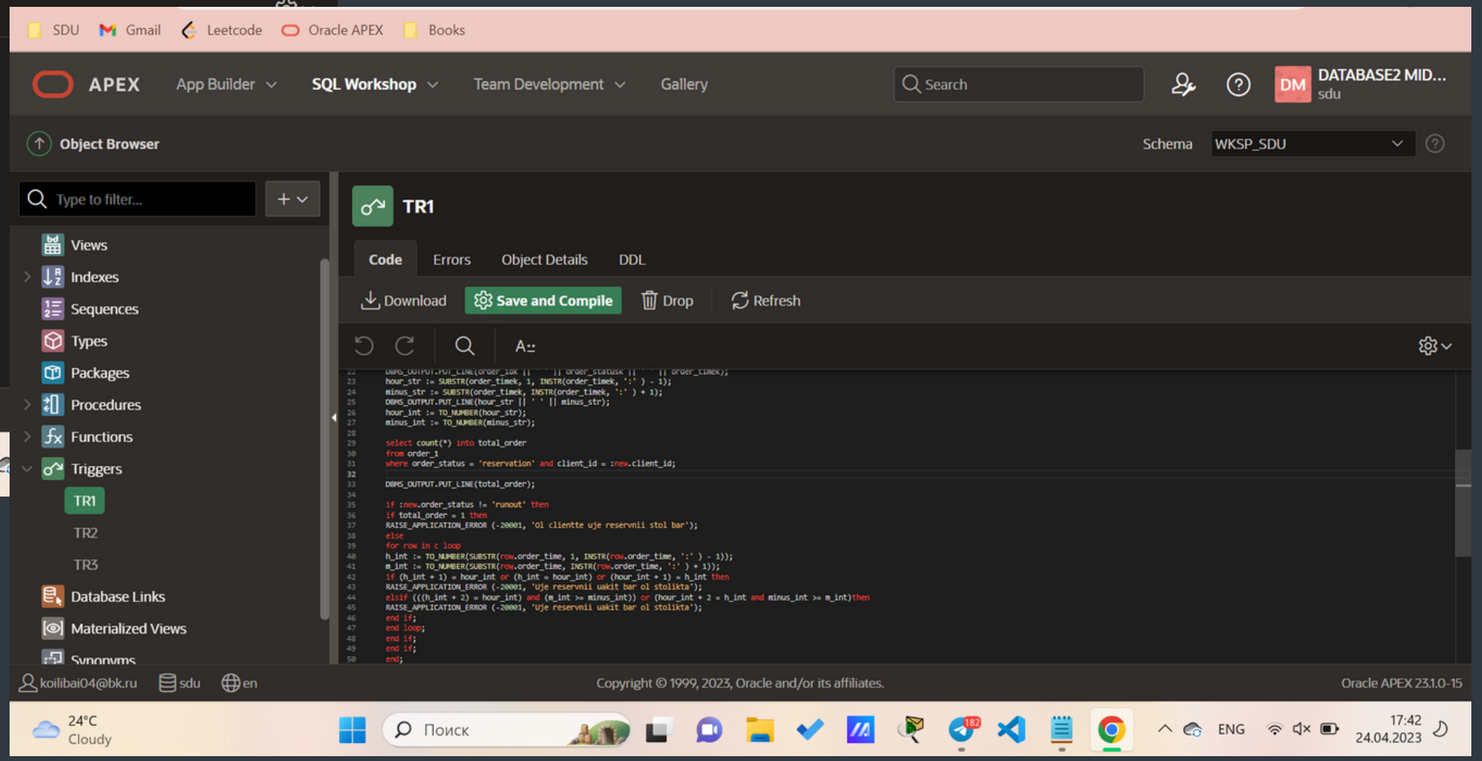
```
Total number of cashiers who are under 18 is 2

Statement processed.
```

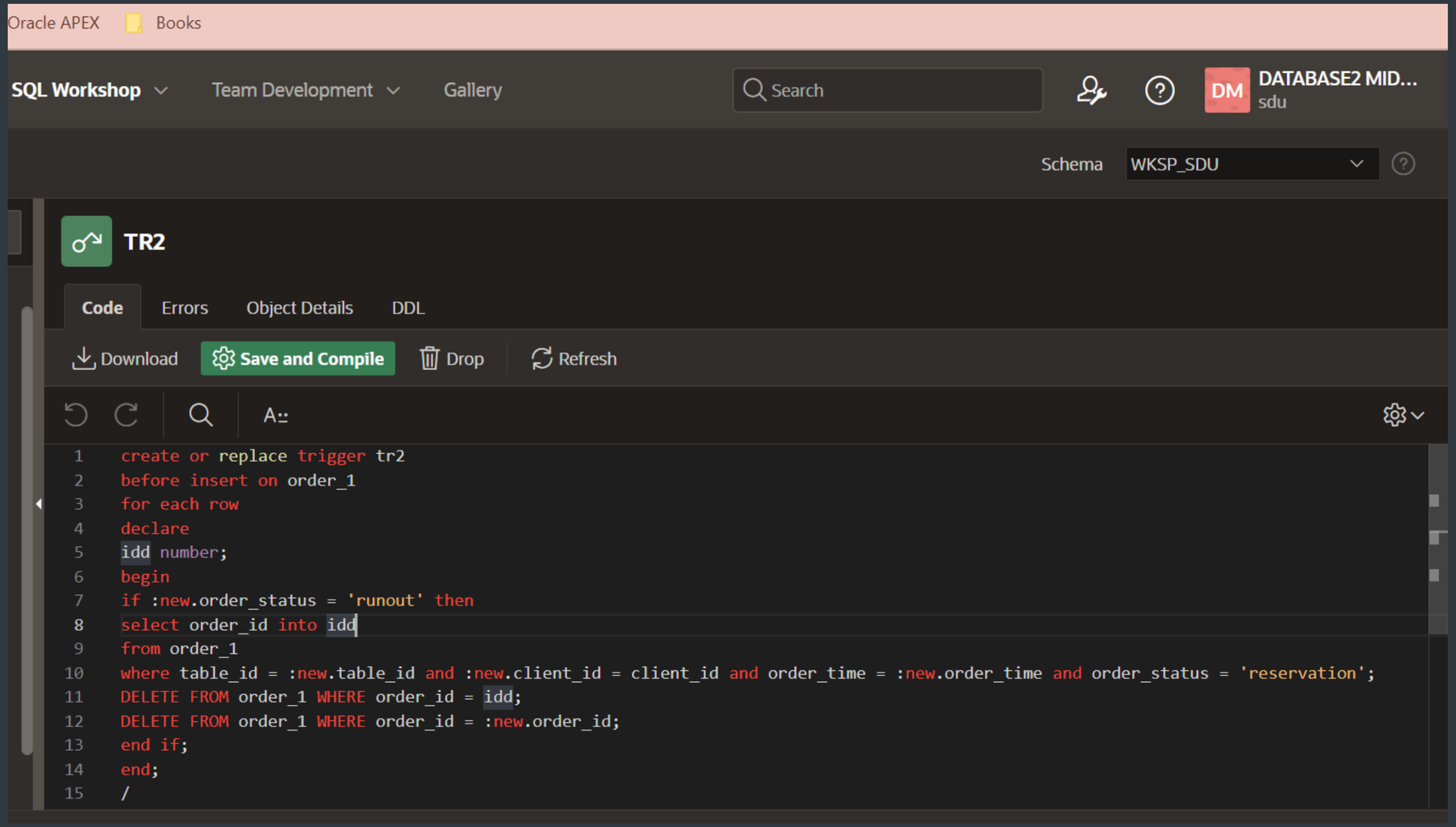
CREATING A FUNCTION WHICH SHOWS NUMBER OF CASHIERS WHO ARE UNDER 18

Trigger 1


SQL Workshop
Team Development
Gallery
Search
DATABASE2 MID...
sdu
Schema
WKSP_SDU
TR1
Code
Errors
Object Details
DDL
Download
Save and Compile
Drop
Refresh
create or replace trigger tr1
before insert on order_1
for each row
declare
order_statusk order_1.ORDER_STATUS%type := :new.order_status;
order_idk order_1.ORDER_ID%type := :new.order_id;
order_timek order_1.ORDER_TIME%type := :new.order_time;
hour_str varchar2(255);
minus_str varchar2(255);
hour_int integer;
minus_int integer;
total_order integer;
h_int integer;
m_int integer;
cursor c is
select *
from order_1
where order_status = 'reservation' and table_id = :new.table_id;
begin
Copyright © 1999, 2023, Oracle and/or its affiliates.
Поиск


SDU
Gmail
Leetcode
Oracle APEX
Books
APEX
App Builder
SQL Workshop
Team Development
Gallery
Search
DATABASE2 MID...
sdu
Object Browser
Schema
WKSP_SDU
Type to filter...
Views
Indexes
Sequences
Types
Packages
Procedures
Functions
Triggers
TR1
TR2
TR3
Database Links
Materialized Views
Synonyms
TR1
Code
Errors
Object Details
DDL
Download
Save and Compile
Drop
Refresh
hour_str := SUBSTR(order_timek, 1, INSTR(order_timek, ':') - 1);
minus_str := SUBSTR(order_timek, INSTR(order_timek, ':') + 1);
DBMS_OUTPUT.PUT_LINE(hour_str || ' ' || minus_str);
hour_int := TO_NUMBER(hour_str);
minus_int := TO_NUMBER(minus_str);
select count(*) into total_order
from order_1
where order_status = 'reservation' and client_id = :new.client_id;
DBMS_OUTPUT.PUT_LINE(total_order);
if :new.order_status != 'runout' then
if total_order = 1 then
RAISE_APPLICATION_ERROR (-20001, 'Ol clientte uje reservnii stol bar');
else
for row in c loop
h_int := TO_NUMBER(SUBSTR(row.order_time, 1, INSTR(row.order_time, ':') - 1));
m_int := TO_NUMBER(SUBSTR(row.order_time, INSTR(row.order_time, ':') + 1));
if (h_int + 1) = hour_int or (h_int = hour_int) or (hour_int + 1) = h_int then
RAISE_APPLICATION_ERROR (-20001, 'Uje reservnii uakit bar ol stolikta');
elsif (((h_int + 2) = hour_int) and (m_int >= minus_int)) or (hour_int + 2 = h_int and minus_int >= m_int)then
RAISE_APPLICATION_ERROR (-20001, 'Uje reservnii uakit bar ol stolikta');
end if;
end loop;
end if;
end if;
end;
koilibai04@bk.ru
sdu
en
Copyright © 1999, 2023, Oracle and/or its affiliates.
Oracle APEX 23.1.0-15
24°C
Cloudy
Поиск
ENG
17:42
24.04.2023

Trigger 2


Oracle APEX
Books
SQL Workshop
Team Development
Gallery
Search
DM
DATABASE2 MID...
sdu
Schema
WKSP_SDU
TR2
Code
Errors
Object Details
DDL
Download
Save and Compile
Drop
Refresh
create or replace trigger tr2
before insert on order_1
for each row
declare
idd number;
begin
if :new.order_status = 'runout' then
select order_id into idd
from order_1
where table_id = :new.table_id and :new.client_id = client_id and order_time = :new.order_time and order_status = 'reservation';
DELETE FROM order_1 WHERE order_id = idd;
DELETE FROM order_1 WHERE order_id = :new.order_id;
end if;
end;
/

Trigger 3

SQL Workshop | Team Development | Gallery | Search

TR3

Code | Errors | Object Details | DDL

Download | Save and Compile | Drop | Refi

```
1   create or replace trigger tr3
2   before insert on order_1
3   for each row
4   declare
5   total_bill integer;
6   tt integer := 0;
7
8   function total_amountko(c_id number) return number is
9   cursor c_c is
10  select *
11  from basket;
```

TR3

Code | Errors | Object Details | DDL

Download | Save and Compile | Drop | Refresh

```
12  dish_price integer := 0;
13  total integer := 0;
14  begin
15  for row in c_c loop
16  if row.CLIENT_ID = c_id then
17      select menu_price into dish_price
18      from menu
19      where menu_id = row.menu_id;
20      total := total + dish_price;
21      dish_price := 0;
22  end if;
23  end loop;
24  return total;
25  end;
26
27  begin
28  if :new.order_status = 'runout' then
29  select count(*) into total_bill
30  from bill;
31  tt := total_amountko(:new.client_id);
32  insert into bill values(total_bill, tt, 200, :new.client_id, 'CARD');
33  delete from basket where client_id = :new.client_id;
34  end if;
35  end;
```

1 procedure

Language SQL Rows 10 Clear Co

```
CREATE OR REPLACE PROCEDURE update_employee_salary(
    n_employee_id IN employee.employee_id%TYPE,
    i_perc IN NUMBER
)
IS
no_found exception;
BEGIN
    UPDATE employee
    SET employee_salary = employee_salary * (1 + i_perc/100)
    WHERE employee_id = n_employee_id;

    IF SQL%ROWCOUNT = 0 THEN
        RAISE_APPLICATION_ERROR(-20001, 'No employee found with ID ' || n_employee_id);
    ELSE
        DBMS_OUTPUT.PUT_LINE('Salary updated for employee ' || n_employee_id || '.');
    END IF;
EXCEPTION
   WHEN no_found THEN
      dbms_output.put_line('No such employee!');
   WHEN OTHERS THEN
      DBMS_OUTPUT.PUT_LINE('Error: '|| SQLERRM);
END;
```

Results Explain Describe Saved SQL History

rocedure created.

2

```
1 BEGIN
2     update_employee_salary(5, 10);
3 END;
4
5
```

Results Explain Describe Saved SQL History

Salary updated for employee 5.

Statement processed.

procedura 2

```
1   CREATE OR REPLACE PROCEDURE find_sname
2   (o_employee_id IN NUMBER,
3   o_employee_name OUT VARCHAR2,
4   o_employee_pos OUT varchar2
5   )
6   AS
7   BEGIN
8   SELECT employee_name, employee_pos
9   INTO o_employee_name, o_employee_pos
10  FROM employee
11  WHERE employee_id = o_employee_id;
12  EXCEPTION
13  WHEN OTHERS
14  THEN
15  DBMS_OUTPUT.PUT_LINE('Error in finding employee_id: '||o_employee_id);
16  END find_sname;
17
18
```

Results | Explain | Describe | Saved SQL | History

```
Salary updated for employee 5.

Statement processed.
```

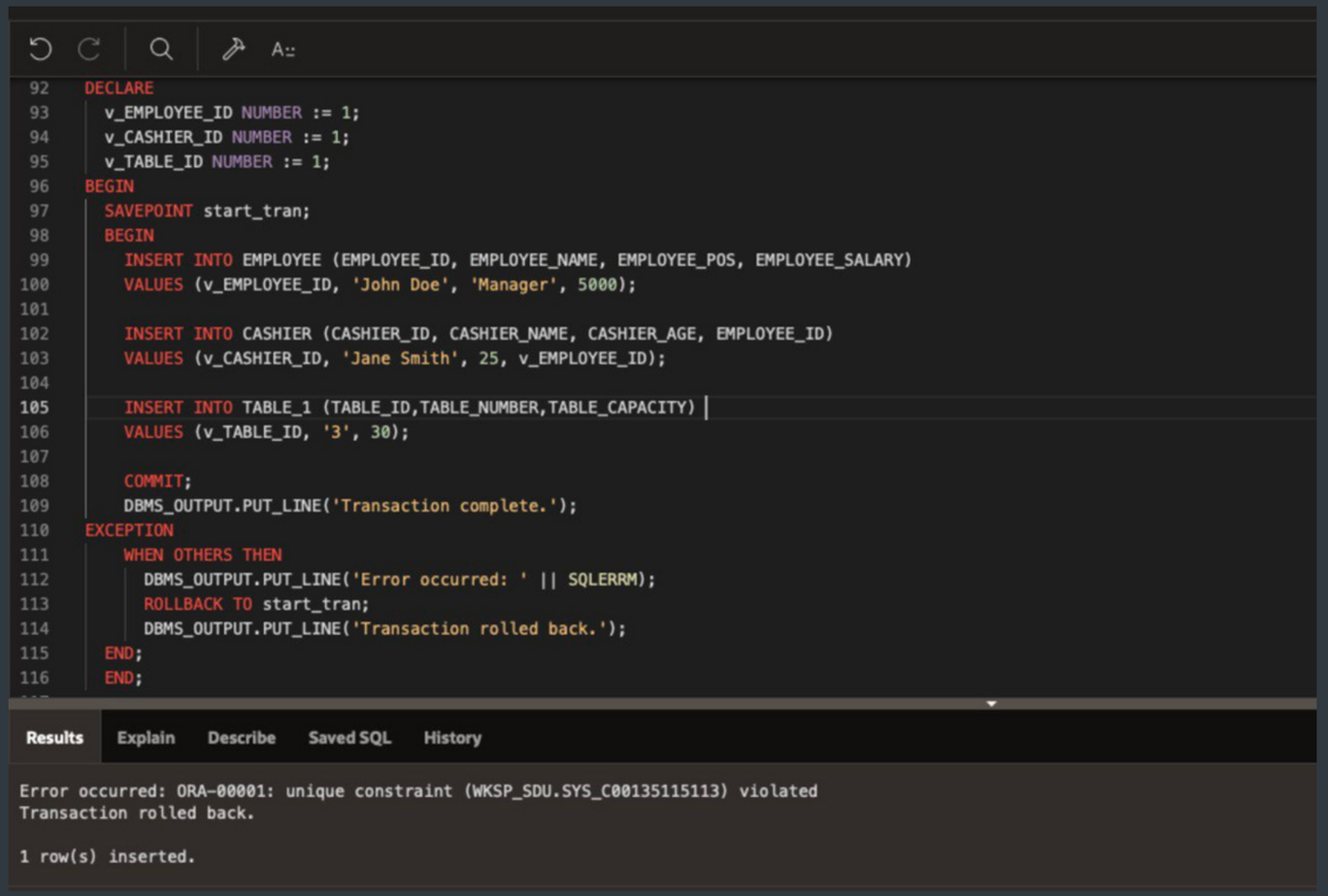
```
1  DECLARE
2    v_local_employee_name employee.employee_name%TYPE;
3    v_local_employee_pos employee.employee_pos%TYPE;
4  BEGIN
5    find_employee(1, v_local_employee_name, v_local_employee_pos);
6    DBMS_OUTPUT.PUT_LINE('Employee is: ' || v_local_employee_name || ' ' || v_local_employee_pos || '.');
7  END;
8  /
```

Results | Explain | Describe | Saved SQL | History

```
Employee is: Rogers chef.

Statement processed.
```

Transaction 1



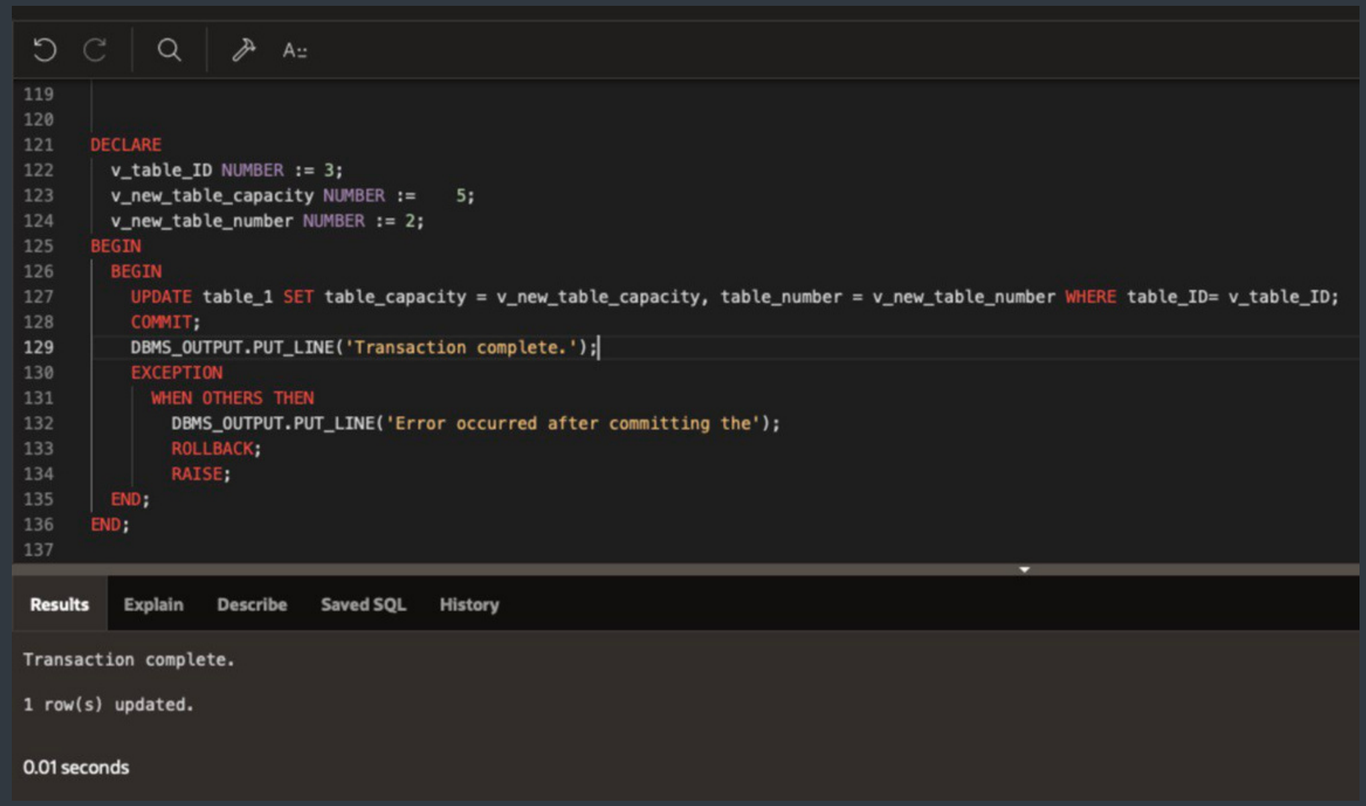
```
92  DECLARE
93    v_EMPLOYEE_ID NUMBER := 1;
94    v_CASHIER_ID NUMBER := 1;
95    v_TABLE_ID NUMBER := 1;
96  BEGIN
97    SAVEPOINT start_tran;
98    BEGIN
99      INSERT INTO EMPLOYEE (EMPLOYEE_ID, EMPLOYEE_NAME, EMPLOYEE_POS, EMPLOYEE_SALARY)
100     VALUES (v_EMPLOYEE_ID, 'John Doe', 'Manager', 5000);
101
102     INSERT INTO CASHIER (CASHIER_ID, CASHIER_NAME, CASHIER_AGE, EMPLOYEE_ID)
103     VALUES (v_CASHIER_ID, 'Jane Smith', 25, v_EMPLOYEE_ID);
104
105     INSERT INTO TABLE_1 (TABLE_ID,TABLE_NUMBER,TABLE_CAPACITY)
106     VALUES (v_TABLE_ID, '3', 30);
107
108     COMMIT;
109     DBMS_OUTPUT.PUT_LINE('Transaction complete.');
110 EXCEPTION
111     WHEN OTHERS THEN
112       DBMS_OUTPUT.PUT_LINE('Error occurred: ' || SQLERRM);
113       ROLLBACK TO start_tran;
114       DBMS_OUTPUT.PUT_LINE('Transaction rolled back.');
115   END;
116   END;
```

Results | Explain | Describe | Saved SQL | History

```
Error occurred: ORA-00001: unique constraint (WKSP_SDU.SYS_C00135115113) violated
Transaction rolled back.

1 row(s) inserted.
```

Transaction 2



```
119
120
121 DECLARE
122   v_table_ID NUMBER := 3;
123   v_new_table_capacity NUMBER :=    5;
124   v_new_table_number NUMBER := 2;
125 BEGIN
126   BEGIN
127     UPDATE table_1 SET table_capacity = v_new_table_capacity, table_number = v_new_table_number WHERE table_ID= v_table_ID;
128     COMMIT;
129     DBMS_OUTPUT.PUT_LINE('Transaction complete.');
130     EXCEPTION
131       WHEN OTHERS THEN
132         DBMS_OUTPUT.PUT_LINE('Error occurred after committing the');
133         ROLLBACK;
134         RAISE;
135   END;
136 END;
137
```

Results | Explain | Describe | Saved SQL | History

```
Transaction complete.

1 row(s) updated.

0.01 seconds
```

```
declare
w_id waiter.waiter_id%type:=204;
w_age waiter.waiter_age%type;
w_name waiter.waiter_name%type;
   wboutage EXCEPTION;
BEGIN
SELECT  waiter_name, waiter_age INTO  w_name, w_age
      FROM waiter
      WHERE waiter_id = w_id;
   IF w_age<18 then
      RAISE wboutage;
   ELSE
      DBMS_OUTPUT.PUT_LINE ('Name: '||  w_name);
      DBMS_OUTPUT.PUT_LINE ('Age: ' || w_age);
   END IF;

   EXCEPTION
      WHEN wboutage THEN
         dbms_output.put_line('Will be fired!');
      WHEN no_data_found THEN
         dbms_output.put_line('No such waiter!');
      WHEN others THEN
         dbms_output.put_line('Error!');
   END;
```

Exception 1

```
No such waiter!

Statement processed.


0.06 seconds
```

Exception 2

```
declare
c_id cashier.cashier_id%type:=204;
c_age cashier.cashier_age%type;
c_name cashier.cashier_name%type;
   cboutage EXCEPTION;
BEGIN
SELECT  cashier_name, cashier_age INTO  c_name, c_age
      FROM cashier
      WHERE cashier_id = c_id;
   IF c_age<18 then
      RAISE cboutage;
   ELSE
      DBMS_OUTPUT.PUT_LINE ('Name: '||  c_name);
      DBMS_OUTPUT.PUT_LINE ('Age: ' || c_age);
   END IF;

EXCEPTION
   WHEN cboutage THEN
      dbms_output.put_line('Will be fired!');
   WHEN no_data_found THEN
      dbms_output.put_line('No such cashier!');
   WHEN others THEN
      dbms_output.put_line('Error!');
END;
```

```
Will be fired!

Statement processed.


0.00 seconds
```

Transaction 3

```
declare
e_id order_1.order_id%type:=58;
e_status order_1.order_status%type;
e_amount bill.total_amount%type;
invalidorder exception;
begin
select order_id, order_status, total_amount into e_id, e_status,e_amount
from order_1, bill
where
order_1.client_id = bill.client_id
   AND order_1.order_id = e_id;
if e_amount<=0 then
raise invalidorder;
else
dbms_output.put_line('Order status: '||e_status);
end if;
```

```
exception
when invalidorder then
dbms_output.put_line('Error!');
when no_data_found then
dbms_output.put_line('No such order');
when others then
dbms_output.put_line('Error');
end;
```

```
Error!

Statement processed.

0.00 seconds
```

```
Order status: runout

Statement processed.

0.07 seconds
```

Everything is fully explained in the video.