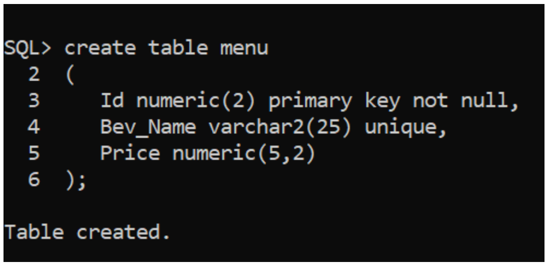
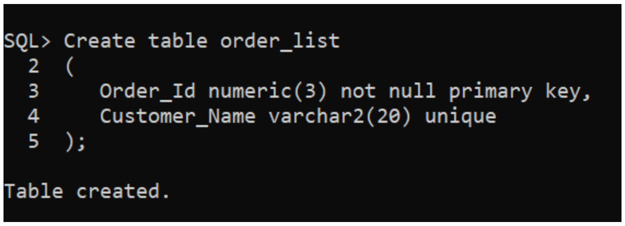
SQL Project:

```
create table menu
 (
   Id numeric(2) primary key not null,
   Bev_Name varchar2(25) unique,
   Price numeric(5,2)
 );
```



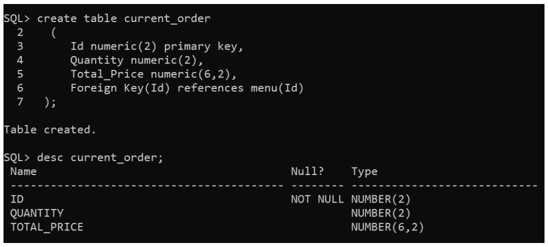
```
SQL> create table menu
     (
        Id numeric(2) primary key not null,
        Bev_Name varchar2(25) unique,
        Price numeric(5,2)
     );

Table created.
```

```
Create table order_list
 (
   Order_Id numeric(3) not null primary key,
   Customer_Name varchar2(20) unique
 );
```



```
SQL> Create table order_list
     (
        Order_Id numeric(3) not null primary key,
        Customer_Name varchar2(20) unique
     );

Table created.
```

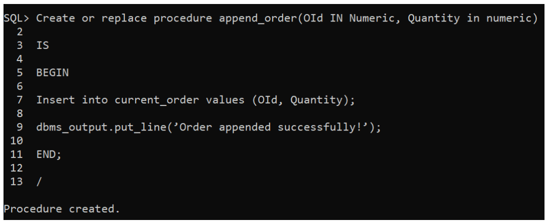
```
create table current_order
 (
   Id numeric(2) primary key,
   Quantity numeric(2),
   Total_price numeric(6)
      Foreign Key(Id) references menu(Id)
    );
```



```
SQL> create table current_order
       (
          Id numeric(2) primary key,
          Quantity numeric(2),
          Total_Price numeric(6,2),
          Foreign Key(Id) references menu(Id)
      );

Table created.

SQL> desc current_order;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ID                                        NOT NULL NUMBER(2)
 QUANTITY                                           NUMBER(2)
 TOTAL_PRICE                                        NUMBER(6,2)
```

```
Create or replace procedure Append_Menu (Id IN Numeric, Bev_Name IN Varchar2, Price IN Numeric)

  IS
  BEGIN
    INSERT into Menu values (Id, Bev_Name, Price);
  END;
  /
```

```
Create or replace procedure append_order(OId IN Numeric, Quantity in numeric)
   IS
   BEGIN
       Insert into current_order values (OId, Quantity);
       dbms_output.put_line('Order appended successfully!');
   END;
   /
```



```
SQL> Create or replace procedure append_order(OId IN Numeric, Quantity in numeric)

     IS

     BEGIN

     Insert into current_order values (OId, Quantity);

     dbms_output.put_line('Order appended successfully!');

     END;

     /

Procedure created.
```

```
Begin
    Append_order(1,2);
End;
/
```

```
create or replace trigger Bill
After Insert On current_order
for each row
Begin
  select Current_order.Id, Bev_name, Price, Quantity,
  (menu.(Price)*current_order.(Quantity)) as Total_price
  from menu m, current_order o
  where o.Id = m.Id
End;
/

drop trigger bill;
create or replace trigger Bill
After Insert On current_order
for each row

Begin
    dbms_output.put_line('Quantity triggered fired. ');
End;
/

begin
    append_order(1,3);
    append_order(0,0);
end;
/

Select Id, bev_name, price, Quantity, (Price*Quantity) as Total_Price from menu natural join current_order;

create or replace trigger Bill
After Insert On current_order
for each row
when(new.Id=0)
Begin
  total_cost:=0;
  open biller;
  LOOP
    Fetch Biller into c_id, c_bev_name, c_price, c_quantity, c_total_price;
    Exit when biller%notfound;
    Total_cost := Total_cost + c_total_price;
    dbms_output.put_line(c_id || ' ' || c_bev_name || ' ' || c_price || ' ' || c_quantity || ' ' || c_total_price);
  END LOOP;
  dbms_output.put_line('Total cost is ' || total_cost);
  close biller;
End;
/
```

select Current_order.Id, Bev_name, Price, Quantity   from menu m, current_order o
where o.Id = m.Id

select Bev_name, Price, Quantity,
'Total_price' as (menu.(Price)*current_order.(Quantity))
from menu m, current_order o
where o.Id = m.Id

```
DECLARE
  c_id menu.id%type;
  c_bev_name menu.bev_name%type;
  c_price menu.price%type;
  c_quantity current_order.quantity%type;
  c_total_price number(7,2);
  total_cost number(7,2);
  CURSOR biller is
    Select Id, Bev_name, price, quantity, (Price*Quantity) as Total_price from menu natural join current_order;

BEGIN
  total_cost:=0;
  open biller;
  LOOP
    Fetch Biller into c_id, c_bev_name, c_price, c_quantity, c_total_price;
    Exit when biller%notfound;
    Total_cost := Total_cost + c_total_price;
    dbms_output.put_line(c_id || ' ' || c_bev_name || ' ' || c_price || ' ' || c_quantity || ' ' || c_total_price);
  END LOOP;
  dbms_output.put_line('Total cost is ' || total_cost);
  close biller;
END;
/
```

```
create or replace trigger Bill
After Insert On current_order
for each row
when(new.Id=0)
DECLARE
  c_id menu.id%type;
  c_bev_name menu.bev_name%type;
  c_price menu.price%type;
  c_quantity current_order.quantity%type;
  c_total_price number(7,2);
  total_cost number(7,2);
  CURSOR biller is
    Select Id, Bev_name, price, quantity, (Price*Quantity) as Total_price from menu natural join current_order;
Begin
  total_cost:=0;
  open biller;
  LOOP
    Fetch Biller into c_id, c_bev_name, c_price, c_quantity, c_total_price;
    Exit when biller%notfound;
    Total_cost := Total_cost + c_total_price;
    dbms_output.put_line(c_id || ' ' || c_bev_name || ' ' || c_price || ' ' || c_quantity || ' ' || c_total_price);
  END LOOP;
  dbms_output.put_line('Total cost is ' || total_cost);
  close biller;
End;
/
```