# מיניפ שלב ג

# מיכל שלומי

#### ביצעתי את העדכונים הבאים לטבלאות שלי:

- 1. הוספתי לטבלת ספר את העמודה DiscountPrice המציין את מחיר הספר לאחר ההנחה DiscountPrice במידה וללקוח מופיע בעמודה
- 2. הוספתי לטבלת הלקוח את העמודה DiscountPercentage המציין את אחוז ההנחה שיש לתת ללקוח בעת הרכישה. ברירת המחדל היא 0 ובמידה והלקוח זכאי להנחה כלשהי נעדכן את השדה הזה בהתאם.

```
ALTER TABLE Book
ADD DiscountPrice NUMBER;

ALTER TABLE Customer
ADD DiscountPercentage NUMBER DEFAULT 0;
```

'condition' להיות בשם 'status' החלפתי את העמודה 'status' המתאר את מצב בספר: חדש/משומש/משומש במצב חדש

```
ALTER TABLE bookcoppy
RENAME COLUMN Status TO Condition VARCHAR2(20);
```

או לא: 'status' המתארת אם הספר במלאי או לא: In stock/not in stock

```
ALTER TABLE bookcoppy
ADD Status VARCHAR2(15) NOT NULL;
```

עבור יצירת הזמנות חדשות "book\_orders" עבור יצירת הזמנות חדשות 5. לספרים.

```
CREATE TABLE book_orders
(
    orderID NUMBER GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
    bookID NUMBER NOT NULL,
    availableCopies NUMBER,
    orderQuantity NUMBER ,
    FOREIGN KEY (bookID) REFERENCES book(bookID)
);
```

6. הוספתי עמודה המציינת את הבונוס של עובד בטבלה worker.

```
ALTER TABLE Worker
ADD Bonus NUMBER DEFAULT 0;
```

## :1 פרוצ<del>ד</del>ורה

הפרוצדורה מקבלת מספר לקוח ובודקת אם מגיע לו הנחה וכמה, ומעדכנת בהתאם את ההנחה, הנחה ניתנת לפי מספר הספרים המקסימלי שרכש בקניה אחת.

עבור 100 פריטים ומעלה יקבל הנחה של 70אחוז
עבור 20 פריטים ומעלה יקבל הנחה של 40 אחוז
עבור 10 פריטים ומעלה יקבל הנחה של 20 אחוז
פרוצדורה מעדכנת את השדה DiscountPercentage בהתאם.

#### הקוד:

```
1 Greate or replace procedure update_customer_discount(customer_id IN NUMBER,
 2
                                                                        discount_info OUT SYS_REFCURSOR)
 3
    lis
         type purchase_record_type is record (
    purchase_id number,
 4 🖨
 5
 6
             total items number
 8
        purchase record purchase record type;
 9
10 🛱
         cursor purchase cursor is
            select p.purchaseid, count(bs.bookid) as total_items
11
12
             from purchase p
13
             join booksales bs on p.purchaseid = bs.purchaseid
14
             where p.customerid = customer_id
15
            group by p.purchaseid;
16
17
        max items number := 0;
18
         discount_percentage number := 0;
19
20
    begin
21
         -- open cursor
22
         open purchase cursor;
23
24
25
             fetch purchase_cursor into purchase_record;
26
             exit when purchase_cursor%notfound;
27
28
             -- check if the current nurchase has more items than the previous maximum
```

```
28
             -- check if the current purchase has more items than the previous maximum
             if purchase_record.total_items > max_items then
    max_items := purchase_record.total_items;
29 🖨
30
             end if;
31
32
         end loop;
33
         close purchase cursor;
34
35
36
         -- determine discount percentage based on max items
37
             when max_items >= 100 then
38 🖨
                 discount_percentage := 70;
39
40
             when max_items >= 20 then
                 discount_percentage := 40;
41
42
             when max_items >= 10 then
                discount_percentage := 20;
43
             else
44
                discount_percentage := 0;
45
46
         end case;
47
         -- update customer's discount percentage
48
         update customer
49 🖨
         set discountpercentage = discount_percentage
50
51
         where customerid = customer id;
52
         -- open ref cursor to return discount info
53
         open discount_info for
54
             select customerid, discountpercentage
```

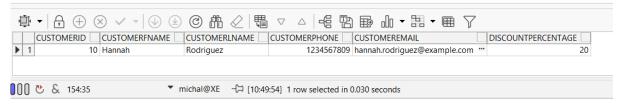
```
-- open ref cursor to return discount info
       open discount info for
55
           select customerid, discountpercentage
56
           from customer
           where customerid = customer_id;
57
58
59
        -- print message based on discount applied
       if discount percentage > 0 then
60 🛱
           dbms output.put line('customer' || customer id || ' is eligible for a ' || discount percentage ||
61
           '% discount. discount updated.'|| ' he purchased '|| max_items||' items');
62
63 🛱
           64
65
66
       end if;
67
68
   exception
69 E
       when no data found then
70
          dbms_output.put_line('no purchase records found for customer ' || customer_id);
71 🖨
       when others then
          dbms_output.put_line('error updating discount for customer ' || customer_id);
72
   end update_customer_discount;
73
```

#### ההרצה:

```
2
      -- Call the procedure
     update_customer_discount(customer_id => :customer id,
                          discount_info => :discount_info);
U Variable
                            Value
                Туре
                            * 10
customer id
                Float
discount_info
                            Cursor>
                Cursor
                Test script DBMS Output Statistics Profiler Trace
                              Enabled
            Buffer size 10000
 customer 10 is eligible for a 20% discount. discount updated. he purchased 10 items
```

# בדיקת פירטי הלקוח לאחר ההרצה:

select \* from customer where customer.customerid=10;



## פונקציה 1 חדשה:

(לפי נתוני הטבלאות שלי כל אחד מהעובדים מכר בין 1 ל 5 פריטים סה"כ ולכן בפונקציה נתייחס לנתון הזה.)

נכתוב פונקציה שרוצה לצ'פר עובדים שביצעו מספר מכירות גבוה ביחס לאחרים ולכן עובדים ש:

מכרו מעל 3 פריטים יזכו בבונוס של 50 ₪,

מכרו מעל 4 פריטים יזכו בבונוס של 100 ₪,

מכרו מעל 5 פריטים יזכו בבונוס של 200 ₪.

### :הקוד

```
1 pcreate or replace function update bonus for worker(p workerid in number)
 2 return number
 3
    is
v_items_sold number := 0;
v_bonus_amount number := 0;
begin
     -- Declare cursor for fetching worker details
     for r worker in (select workerid from worker where workerid = p workerid) loop
 8 占
        -- Get the count of items sold by the worker
 9
10
        select count(*)
        into v items sold
11
12
        from purchase p
        where p.workerid = r worker.workerid;
13
14
        -- Determine bonus amount based on items sold
15
16
17
          when v items sold > 5 then
            v_bonus_amount := 200;
18
19 🖯
         when v_items_sold > 4 then
20
            v bonus amount := 100;
20
21 🖹
          when v_items_sold > 3 then
22
            v_bonus_amount := 50;
23
          else
           v bonus amount := 0; -- No bonus if less than or equal to 3 items sold
24
25
        end case;
26
         -- Update the bonus for the worker if applicable
27
28 🛱
        if v bonus amount > 0 then
29 🛱
         update worker
          set bonus = bonus + v_bonus_amount
30
31
         where workerid = r worker.workerid;
32
33
34
        -- Return the bonus amount awarded
35
       return v bonus amount;
     end loop;
36
37
38
      -- If no data found, return 0
39
      return 0;
40
41
   exception
42
     when no_data_found then
        return 0; -- Return 0 if no data found for worker ID
43
     when others then
44
        return -1; -- Return -1 for any other error
46 end update_bonus_for_worker;
```

### לפני ההרצה עשיתי בדיקה של כמות הפריטים שמכרו העובדים:

```
ID: 123456789, Items Sold: 5
ID: 123568974, Items Sold: 1
ID: 323564871, Items Sold: 3
ID: 323135648, Items Sold: 5
ID: 315556965, Items Sold: 1
ID: 226755554, Items Sold: 1
ID: 743502274, Items Sold: 2
ID: 718109763, Items Sold: 2
ID: 335673276, Items Sold: 1
ID: 901683660, Items Sold: 1
ID: 397989285, Items Sold: 3
ID: 294357585, Items Sold: 4
ID: 822958626, Items Sold: 3
```

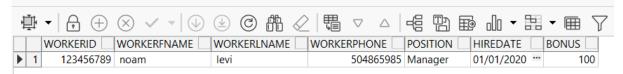
# פירטי לקוח 123456789 לפני הרצה של הפונקציה:

#### select \* from worker w where w.workerid=123456789;

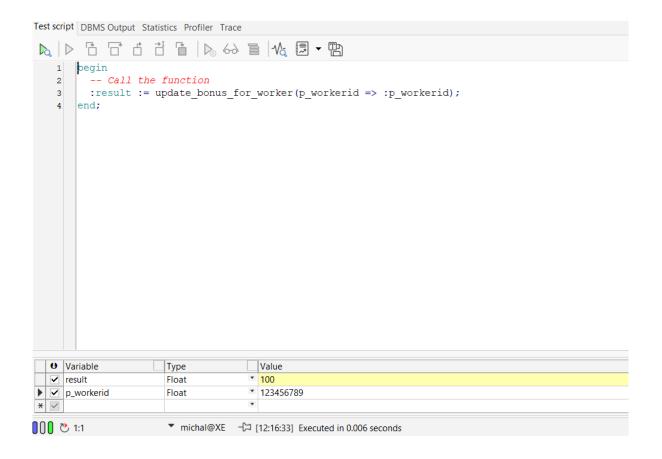
. —					om <u>→ m</u>	▲ 圃 人
WORKER	RID WORKERFNAME	WORKERLNAME	WORKERPHONE	POSITION	HIREDATE	BONUS
<b>1</b> 1234	56789 noam	levi	504865985	Manager (	01/01/2020 ***	0

# פירטי לקוח 123456789 לפני הרצה של הפונקציה:

select \* from worker w where w.workerid=123456789;



# הרצה של הפונקציה:



### <u>תוכנית ראשית 1:</u>

אנו מעוניינים לדאוג לשימור לקוחות רווחיים וכן לשימור עובדים שמכניסים יותר כסף לחברה ולכן ללקוחות הרווחיים אנו ניתן הנחות שאין לכולם, ולעובדים רווחיים ניתן מענקים לפי הרווח שהם מכניסים לחברה.

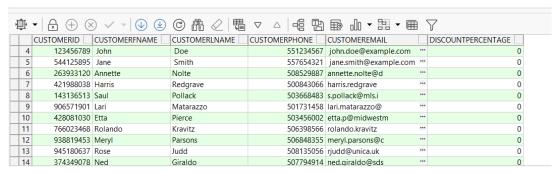
הפרוצדורה update customer discount יטפל בעדכוני ההנחות ללקוחות ואת העובדים נשלח לפונקציה update bonus for worker.

# הקוד:

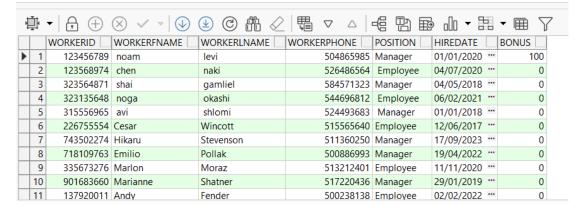
```
SQL
     Output Statistics
declare
  cursor c customers is
    select customerid
    from customer;
  cursor c workers is
   select workerid
    from worker;
  v customer id customer.customerid%type;
  v worker id worker.workerid%type;
  v bonus awarded number;
  discount info SYS REFCURSOR;
  -- Loop over customers and update discount
  open c customers;
  loop
    fetch c customers into v customer id;
    exit when c customers%notfound;
    -- Call procedure to update customer discount
    update customer discount (v customer id, discount info);
  end loop;
  close c_customers;
  -- Loop over workers and update bonus
  open c workers;
  loop
   fetch c_workers into v_worker_id;
   exit when c_workers%notfound;
   -- Call function to update worker bonus
   v_bonus_awarded := update_bonus_for_worker(v_worker_id);
   -- Print bonus information for the worker
      --dbms_output.put_line('Worker ID ' || v_worker_id || ': Bonus awarded: ' || v_bonus_awarded);
 end loop;
 close c workers;
end:
```

#### הטבלאות customer, worker לפני הרצת התוכנית:

select \* from customer;

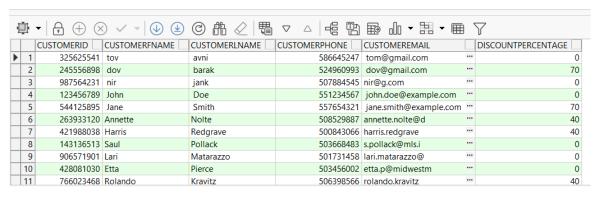


select \* from worker;

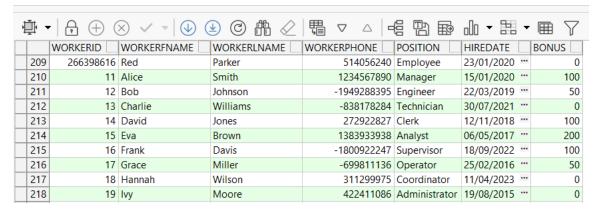


#### הטבלאות customer, worker אחרי הרצת התוכנית:

select \* from customer;



select \* from worker;



פרוצדורה 2: (מכאן מתחיל החלק השני. פרוצדורה 2, פונקציה 2, תוכנית ראשית 2.)

הפרוצדורה מקבלת מספר הזמנה וIDI של ספר ויוצרת הזמנה חדשה לטבלת ההזמנות לפי כמות העותקים הזמינים הקיימים במלאי. אנו מעוניינים שתמיד יישאר בסביבות ה30 עותקים זמינים ולכן נעדכן בהתאם:

אם נשאר פחות מ 5 עותקים זמינים ניצור הזמנה של 30 עותקים חדשים,

אם נשאר פחות מ 10 עותקים זמינים ניצור הזמנה של 20 עותקים חדשים,

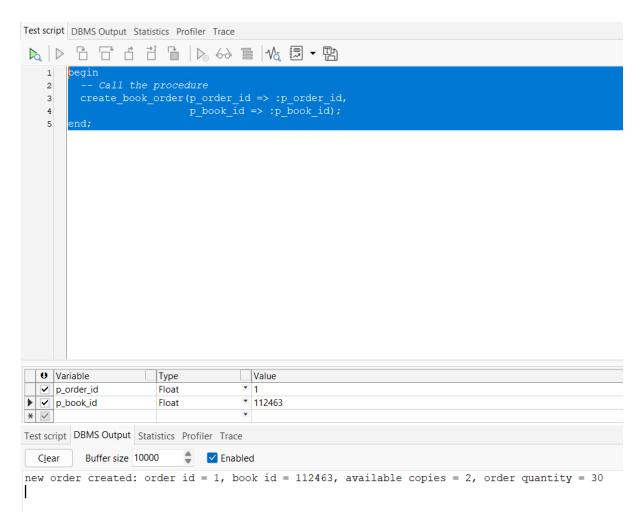
אם נשאר פחות מ 20 עותקים זמינים ניצור הזמנה של 10 עותקים חדשים,

אם נשאר פחות מ 30 עותקים זמינים ניצור הזמנה של 0 עותקים חדשים, ובעצם עודכן שלא צריך עוד ספרים.

#### הקוד:

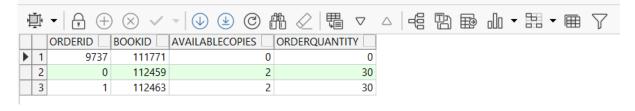
```
1 pcreate or replace procedure create book order(p order id in number, p book id in number)
     cursor book_cursor is
        select count(*) as total_available_copies
4 🛱
        from bookcoppy
5
        where bookid = p_book_id
and status = 'in stock';
 6
 7
 8
      v available_copies number;
10
      v order quantity number;
11
    begin
      -- Open the cursor
12
13
      open book cursor;
14
      fetch book_cursor into v_available_copies;
15
16
     if book cursor%notfound then
17
        -- If no available copies found
        dbms output.put line('no available copies found for book id ' || p book id);
18
        close book_cursor;
19
20
        return:
21
      end if;
22
      -- Use CASE to determine order quantity
23
      v order quantity := case
24
25
        when v available copies < 5 then 30
        when v_available_copies < 10 then 20
2.6
27
        when v_available_copies < 20 then 10
28
        when v available copies < 30 then 0
29
       else 0
30
      end;
31
       -- Create a new order
32
33 🛱
      if v order quantity>0 then
         insert into book orders (orderid, bookid, availablecopies, orderquantity)
34 ⊟
         values (p_order_id, p_book_id, v_available_copies, v_order_quantity);
35
36
      end if;
37
      -- Close the cursor
      close book_cursor;
38
39
       -- Print message about the new order created
40
     dbms_output.put_line('new order created: order id = ' || p_order_id || ', book id = ' ||
41
42
                                 p_book_id || ', available copies = ' || v_available_copies ||
43
                                  , order quantity = ' || v order quantity);
44
45
    exception
46 when no_data_found then
        dbms_output.put_line('no available copies found for book id ' || p book id);
47
48 🖨
     when others then
dbms_output.put_li
end create_book_order;
        dbms output.put line('error occurred: ' || sqlerrm);
```

# הרצה של הפרוצדורה עבור הזמנה מס 0 ומס' ספר 112459:



# ניראה את העדכון בטבלה:

### select \* from book\_orders;



#### פונקציה 2:

פונקציה שמקבלת בכל פעם IDBOOK ומספר סידורי ראשון שממנו תתחיל.

מה שהפונקציה תעשה זה ליצור מספר עותקים חדשים בטבלת העותקים של ספר ,לפי המספר המעודכן בטבלת ההזמנות בעמודה , order Quantity בפקודת הכנסה לטבלת העותקים תכניס את המעודכן בטבלת ההזמנות בעמודה ,'in stock' ובמצב הספר תעדכן 'new' ובכל עותק תגדיל את מספר העותק ב1.

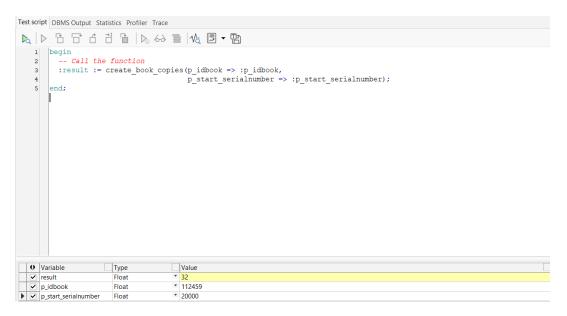
לאחר מכן תעדכן את ההזמנה של אותו הספר בהתאם כך שתאפס את שדה של כמות ההזמנות ותוסיף אותו למספר העותקים הקיים

ותחזיר את orderQuantity+0. ואח"כ availableCopies= availableCopies+orderQuantity , מספר העותקים הקיים במלאי. כתוב את כל הקוד באותיות קטנות

#### :קוד:

```
1 Dreate or replace function create book copies(p idbook in number, p start serialnumber in number)
    return number
3 is
4 🖨
      cursor c_order is
5 🖨
        select bookid, orderquantity, availablecopies
 6
        from book orders
        where bookid = p_idbook
        for update;
 9
10
      r order c order%rowtype;
      v available copies number;
11
     v_serialnumber number := p_start_serialnumber;
12
13
    begin
14
      -- Open the cursor
      open c order;
15
16
      fetch c order into r order;
17
18 p if c order%notfound then
        -- If no order found
19
        close c_order;
20
        raise_application_error(-20001, 'No order found for the given book ID');
21
22
      end if:
23
      -- Create new copies in the BookCoppy table
25 p for i in 1..r order.orderquantity loop
       insert into bookcoppy (serialnumber, status, condition, bookid)
values (v_serialnumber, 'in stock', 'new', r_order.bookid);
v_serialnumber := v_serialnumber + 1;
26 🖨
27
27
28
29
      end loop;
30
      -- Update availableCopies and reset orderQuantity in book orders table
31
32 update book orders
      set availablecopies = r order.availablecopies + r order.orderquantity,
33
         orderquantity = 0
34
35
      where bookid = r_order.bookid;
36
      -- Close the cursor
37
38
      close c order;
39
      -- Fetch the total available copies from the BookCoppy table
40
41 | select count(*)
42
      into v_available_copies
      from bookcoppy
43
      where bookid = p_idbook;
44
45
46
      -- Return the total available copies
     return v_available_copies;
47
48 exception
49 p when no data found then
        raise_application_error(-20001, 'No order found for the given book ID');
50
        raise application error(-20002, 'An error occurred: ' || sqlerrm);
end create book copies;
```

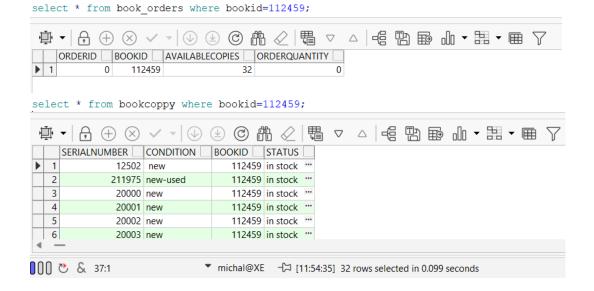
### ברצה עבור 112459=BOOKID ומס סידורי מתחיל מ20000:



#### הטבלאות לפני ההרצה:



#### הטבלאות לאחר ההרצה:



### תוכנית ראשית 2:

עבור התוכנית יצרנו טבלה חדשה להזמנות של ספרים. התוכנית עוברת על כל הספרים מעדכנת להם את ההזמנות שיש לבצע ואז יוצרת עותקים נוספים לכל ספר לפי ההזמנה שביצענו.

התוכנית עוברת על כל הספרים ושולחת בכל פעם ספר אחד לפרוצדורה להזמנה ומעדכנת את ההזמנה בפועל ההזמנה בהתאם לקריטריונים שפירטנו ולאחר מכן קוראת לפונקציה שתבצע את ההזמנה בפועל ומוסיפה עותקים חדשים ומחזירה את סכום העותקים שיש לכל ספר לאחר העדכון.

שם הקובץ הוא: declareNewOrders.sql

```
SQL Output Statistics
declare
  cursor c_books is
   select bookid
   from book;
  r_book c_books%rowtype;
  v_order_id number := 60000; -- starting order ID
  v_start_serialnumber number := 20000; -- starting serial number
  v total copies number;
  v_order_quantity number;
  type t order record is record (
    orderquantity number
  ):
  r order t order record;
  cursor c_orders(p_bookid number) is
    select orderquantity
    from book orders
    where bookid = p bookid
    for update;
begin
  -- Open the cursor for books
  open c books;
  loop
    fetch c books into r book;
    exit when c books%notfound;
                                 -- exit if no more books
    -- Call the procedure to create a new book order
    create_book_order(v_order_id, r_book.bookid);
    -- Initialize order quantity
    v_order_quantity := 0;
    -- Open the cursor for orders
    open c_orders(r_book.bookid);
    fetch c_orders into r_order;
    if c_orders%found then
     v order quantity := r order.orderquantity;
    end if;
    close c_orders;
    -- Call the function to create book copies
    if v order quantity > 0 then
      v_total_copies := create_book_copies(r_book.bookid, v_start_serialnumber);
      -- Increment order ID and starting serial number for the next iteration
      v order id := v order id + 1;
      {\tt v\_start\_serial number := v\_start\_serial number + v\_order\_quantity;}
     end if;
  end loop;
  -- Close the cursor
  close c_books;
exception -
  when others then
   dbms_output.put_line('Error: ' || sqlerrm);
▼ michal@XE -□ [09:37:10] Done in 0.076 seconds
```

# תוכנית נלוות בשם countBaookCopies

עשיתי בדיקה למצוא כמות העותקים יש לכל ספר לפני הרצת התוכנית:

:הקוד

```
declare
 cursor c books is
   select bookid, bookname
   from book;
 r book c books%rowtype;
 v copy count number;
begin
 open c_books;
  loop
    fetch c books into r book;
    exit when c_books%notfound;
    ספירת העותקים עבור הספר הנוכחי --
    select count(*)
    into v_copy_count
    from bookcoppy
    where bookid = r book.bookid;
    -- הדפסת כמות העותקים של הספר הנוכחי
    dbms_output.put_line('Book ID: ' || r_book.bookid || ' : ' || v_copy_count);
  end loop;
 close c books;
end;
```

ההרצה של הקוד:

```
Output Statistics
SOL
 Clear Buffer size 78000
                             Enabled
Book ID: 112459 : 29
Book ID: 112463 : 27
Book ID: 112467 : 16
Book ID: 112471 : 9
Book ID: 112475 : 5
Book ID: 112479
Book ID: 112483 : 26
Book ID: 112487
Book ID: 112491 : 9
Book ID: 112495 : 22
Book ID: 112499 : 29
Book ID: 112503 : 23
Book ID: 112507
               : 6
Book ID: 112511
Book ID: 112515
Book ID: 112519
Book ID: 112523
Book ID: 112527 : 14
Book ID: 112531
Book ID: 112535 : 29
Book ID: 112539 : 1
Book ID: 112543 : 2
Book ID: 112547
               : 21
Book ID: 112551
Book ID: 112555
Book ID: 112559
Book ID: 112563 : 13
Book ID: 112567 : 26
000 ₺ & 409:1
```

# הטבלה של הזמנות אחרי הרצה של התוכנית:

select \* from book orders;

Ę	<u> </u>		$\otimes$ $\checkmark$				₀00 ▾		abla
		ORDERID	BOOKID	AVAILABLECOPIES	ORDERQUANTITY				
١	1	60000	22	32	0				
	2	60001	23	34	0				
	3	60002	24	21	0				
	4	60003	27	29	0				
	5	60004	29	32	0				
	6	60005	111115	23	0				
	7	60006	111123	26	0				
	8	60007	111127	26	0				
T	9	60008	111135	24	0				
	10	60009	111139	22	0				
₫	_								