

# מערכת לניהול ספריה דו"ח פרויקט | 2024

רעיה פלדמר

חיה לקס



**The Library**  
Books & Games

## תוכן עניינים

2.....	Alter Table
3.....	Triggers
4.....	Functions
4.....	<i>.calculateDueDate_book</i>
6.....	<i>.calculateFine_book</i>
7.....	<i>blockClients</i>
10.....	<i>payFine</i>
13.....	Procedures
13.....	<i>lending_book</i>
17.....	<i>returning_book</i>
20.....	<i>extensionOfAllLendings_book</i>
23.....	Library_maintenance / תוכנית
27.....	Library_main / תוכנית



# Alter Table

ביצענו כמה שינויים בבסיס הנתונים (השינויים מצורפים בקובץ `AlterTable.sql`)

1. שינינו את העמודה של `lendingId` בטבלת `bookLending` כך שתיווצר באופן אוטומטי:

```
-- modify bookLending.lendingId to be auto generated
alter table bookLending drop column lendingId;

alter table bookLending
  add lendingId integer
  generated always as identity (START WITH 1 INCREMENT BY 1);

alter table bookLending
  add constraint bookLending_pk primary key (lendingId);

ALTER TABLE bookLending MODIFY (lendingdate INVISIBLE, duedate INVISIBLE, returndate INVISIBLE, clientId INVISIBLE,
librarianid INVISIBLE, copyid INVISIBLE);
ALTER TABLE bookLending MODIFY (lendingdate VISIBLE, duedate VISIBLE, returndate VISIBLE, clientId VISIBLE, librarianid VISIBLE,
copyid VISIBLE);
```

2. הוספנו טבלה `finest` לשמירת קנסות עבור לקוחות שאיחרו בהחזרת הספר:

```
-- create new table for saved culculated fines of clients.
CREATE TABLE finest
(
  clientId VARCHAR(9) NOT NULL,
  totalFine NUMBER(5) NOT NULL,
  PRIMARY KEY (clientId),
  FOREIGN KEY (clientId) REFERENCES client(clientId)
);
```

	Name	Type	Nullable	Default	Comments
1	CLIENTID	VARCHAR2(9)			
2	TOTALFINE	NUMBER(5)			

3. הוספנו טבלה `notifications` לשמירת הודעות לאנשים (לקוחות/ ספרנים)

```
-- create new table for notifications.
CREATE TABLE notifications
(
  notifyId INTEGER GENERATED ALWAYS AS IDENTITY (START WITH 1 INCREMENT BY 1),
  personId VARCHAR2(9) NOT NULL,
  msg VARCHAR2(500) NOT NULL,
  msgDate DATE NOT NULL,
  PRIMARY KEY (notifyId),
  FOREIGN KEY (personId) REFERENCES person(personId)
);
ALTER TABLE notifications
MODIFY msgDate DEFAULT SYSDATE;
```

	Name	Type	Nullable	Default	Comments
1	NOTIFYID	INTEGER		"LAX"."ISEQ\$\$_76962".nextval	
2	PERSONID	VARCHAR2(9)			
3	MSG	VARCHAR2(500)			
4	MSGDATE	DATE		SYSDATE	

# Triggers

הוספנו טריגרים שיעדכנו את העמודה *available* של ספר או משחק בהשאלה או בהחזרה:

(בחרנו לבצע זאת בטריגר ולא בפרוצדורת השאלה/ החזרה, כיוון שזה שינוי שנוגע לבסיס הנתונים עצמו, ולא שינוי לוגי).

כלומר, אנו רוצים שבכל עדכון של עמודת *returnDate* יעודכן בהתאם הערך *available* גם אם זה לא נעשה דרך פרוצדורה. שלא ייווצר מצב שזמינות הספר לא מעודכנת בהתאם למציאות...)

## 1. טריגר לטיפול בעמודה *available* של ספר

```
create or replace noneditionable trigger book_lending_returning
after insert or update of returndate
on booklending
for each row
begin
-- lending a book
if :new.returndate is null
then update bookcopy
set available = 0
where bookcopy.copyid = :new.copyid;
else
-- returning a book
IF :new.returndate > SYSDATE THEN
RAISE_APPLICATION_ERROR(-20001, 'Return date cannot be in the future');
END IF;
update bookcopy
set available = 1
where bookcopy.copyid = :new.copyid;
end if;
end bookLending_return;
```

## 2. טריגר לטיפול בעמודה *available* של משחק

```
-- This trigger updates availability of a play when lending or returning it
create or replace noneditionable trigger play_lending_returning
after insert or update of returndate
on playlending
for each row
begin
-- lending a play
if :new.returndate is null
then update playcopy
set available = 0
where playcopy.copyid = :new.copyid;
else
-- returning a play
IF :new.returndate > SYSDATE THEN
RAISE_APPLICATION_ERROR(-20001, 'Return date cannot be in the future');
END IF;
update playcopy
set available = 1
where playcopy.copyid = :new.copyid;
end if;
end playLending_return;
```



# Functions

1.

פונקציית **calculateDueDate\_book** היא פונקציית עזר לחישוב התאריך המיועד להחזרת ספר לפי מספר העותקים הזמינים מאותו הספר.

## פרמטר

**Copy\_id** המספר המזהה של העותק שרוצים לשאול.

## פעולת הפונקציה

הפונקציה בודקת את מספר העותקים הזמינים מספר זה.

אם יש רק עותק אחד זמין, ההשאלה היא לשבוע.

אם יש 2 עותקים זמינים, ההשאלה היא לשבועיים.

אם יש יותר מ-2 עותקים זמינים, ההשאלה היא לחודש.

## ערך מוחזר

תאריך אחרון להחזרת הספר.


## קוד הפונקציה

```
create or replace noneditionable function calculateDueDate_book(copy_id in number) return date is
/*-----
This function calculates the due date for returning the book
parameter: copy_id=> the copyId of the book.
return the due date.
-----*/

    due_date date;
    tmp_num integer;
begin
    -- Checking the amount of copies available of this book excluding the requiered book
    select count(*) into tmp_num
    from Bookcopy
    where bookId = (select bookId from bookCopy where copyId = copy_id) and
           copyId <> copy_id and
           available = 1;
    if tmp_num = 0 then
        due_date:= TRUNC(SYSDATE + 7);
    elsif tmp_num < 3 then
        due_date:= TRUNC(SYSDATE + 14);
    else due_date:= TRUNC(ADD_MONTHS((SYSDATE),1));
    end if;
    return(due_date);
end calculateDueDate_book;
```



Test scriptDBMS OutputStatisticsProfilerTrace



```
1 begin
2   -- Call the function
3   :result := calculateDueDate_book(copy_id => :copy_id);
4 end;
```

<input type="checkbox"/>	Variable	Type	Value
<input checked="" type="checkbox"/>	result	Date	21/07/2024
<input checked="" type="checkbox"/>	copy_id	Float	658

פונקציית **calculateFine\_book** היא פונקציית עזר לחישוב הקנס עבור איחור של ספר.

### פרמטר

**lending\_id** המספר המזהה של ההשאלה.

### פעולת הפונקציה

הפונקציה מחשבת את מספר הימים שעברו מהתאריך המיועד להחזרה **dueDate** עד לתאריך הנוכחי, ומכפילה את מספר הימים בקבוע **FINE\_PER\_DAY**.

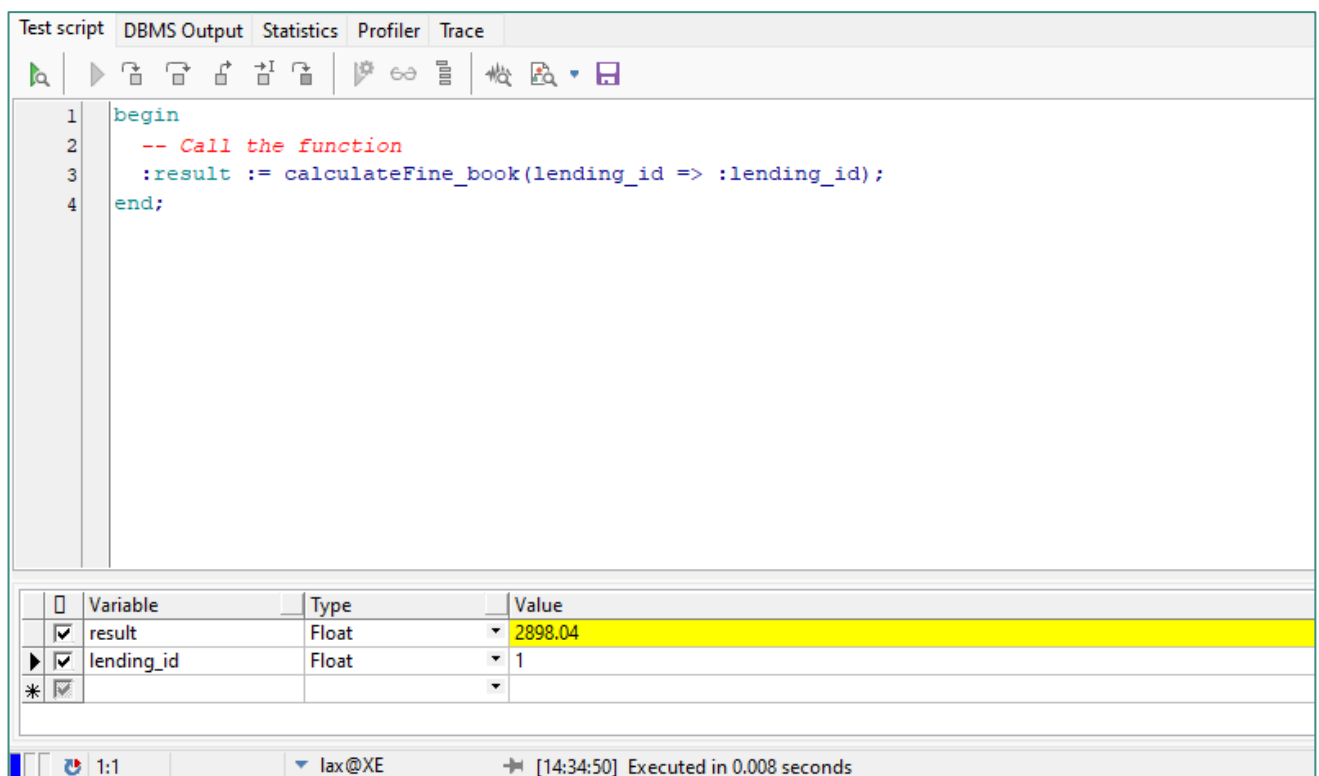
### ערך מוחזר

סכום הקנס עבור האיחור.

### קוד הפונקציה

```
create or replace noneditionable function calculateFine_book(lending_id in integer) return number is
/*-----
This function calculates the fine for returning a book late.
parameter: lending_id.
return the the amount of the fine.
-----*/
    calculatedFine number:=0;
    FINE_PER_DAY constant number(2) := 5;
begin
    select (sysdate-dueDate)*FINE_PER_DAY into calculatedFine
    from bookLending
    where lendingId = lending_id;
    return(calculatedFine);
end calculateFine_book;
```

### דוגמת הרצה



The screenshot shows the SQL Developer interface with a test script and its execution results. The test script is as follows:

```
1 begin
2     -- Call the function
3     :result := calculateFine_book(lending_id => :lending_id);
4 end;
```

The execution results are shown in the table below:

Variable	Type	Value
result	Float	2898.04
lending_id	Float	1

The status bar at the bottom indicates that the script was executed in 0.008 seconds.

פונקציית **blockClients** היא פונקציה שמטרתה לחסום לקוחות שמתעכבים בהחזרת ספרים למעלה מזמן נתון.

## פרמטר

*allowedLate* את הזמן המקסימלי שניתן לאחר בהחזרת הספרים ללא חסימת הלקוח.

## פעולת הפונקציה

הפונקציה עוברת על כל ההשאלות, וכאשר מגיעה להשאלה של ספר שעדיין לא הוחזר, והזמן המיועד להחזרה חלף לפני הזמן שניתן כפרמטר, הפונקציה חוסמת את הלקוח ששאל ספר זה.

בנוסף, הפונקציה מכניסה לטבלת notifications הודעה ללקוח שנחסם ובו פרטי הספרים שמאחר בהחזרתם.

## ערך מוחזר

מספר הלקוחות שנחסמו בפעולה זו.

## קוד הפונקציה

```
create or replace noneditionable function blockClients(allowedLate in number default 60) return number is
/*-----
This functions blocks all clients who had not returned books although their due date passed.
parameter: allowedLate=> the maximum days allowed to late.
return the ammount of clients blocked in this transaction.
-----*/
counter number:= 0;
header constant varchar2(100):='You are blocked due to books you had not returned:'||chr(13)||chr(10);
notifyMessage varchar2(500);

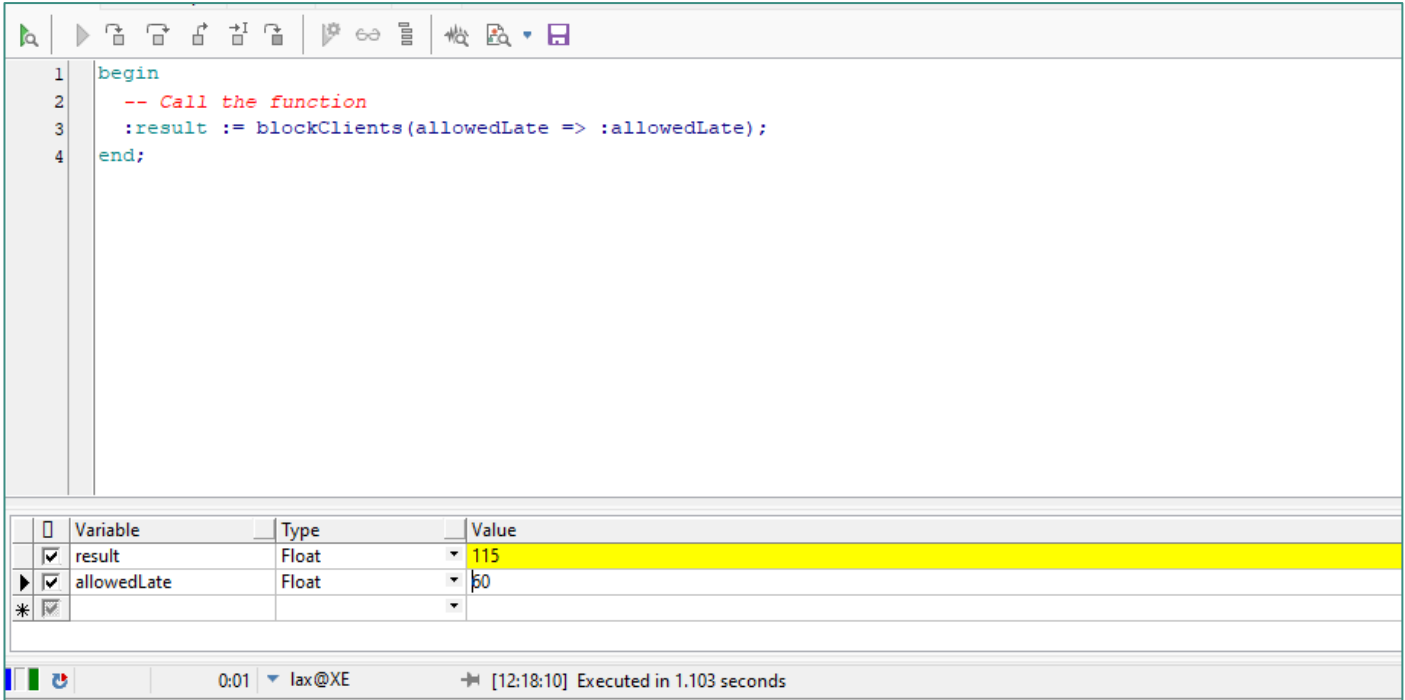
cursor clientsToBlock is
select clientId, count(*)
from booklending natural join client
where active = 1 and sysdate-duedate > allowedLate
group by clientId;
clientsRec clientsToBlock%rowtype;

cursor lateBooksDetails is
select clientId, dueDate, title
from booklending natural join bookcopy natural join book
where trunc(sysdate)>duedate;
booksDetailsRec lateBooksDetails%rowtype;

begin
for clientsRec in clientsToBlock
loop
-- Block the client
update client
set active = 0
where clientId = clientsRec.Clientid;
-- Create notify message
notifyMessage:= header;
for booksDetailsRec in lateBooksDetails
loop
if booksDetailsRec.Clientid=clientsRec.Clientid then
notifyMessage:= notifyMessage||'The book '||booksDetailsRec.title||' had to be returned in '
||booksDetailsRec.dueDate||chr(13)||chr(10);
end if;
end loop;
--insert to notification table
insert into notifications(personId, msg)
values(clientsRec.Clientid, notifyMessage);
-- Increase the counter
counter:= counter+1;
end loop;
return(counter);
end blockClients;
```



## דוגמת הרצה



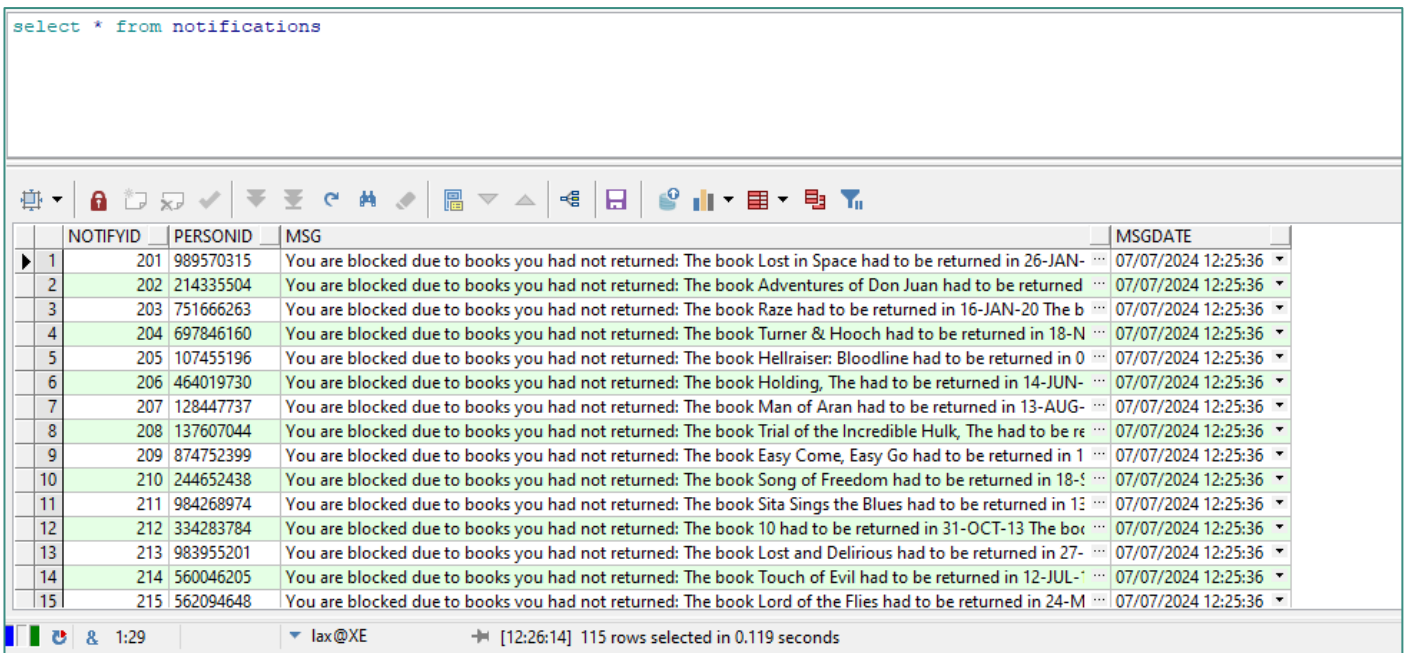
```

1 begin
2     -- Call the function
3     :result := blockClients(allowedLate => :allowedLate);
4 end;
```

Variable	Type	Value
result	Float	115
allowedLate	Float	60

0:01 lax@XE [12:18:10] Executed in 1.103 seconds


טבלת *notifications* לאחר הפעולה:



```
select * from notifications
```

	NOTIFYID	PERSONID	MSG	MSGDATE
1	201	989570315	You are blocked due to books you had not returned: The book Lost in Space had to be returned in 26-JAN-	07/07/2024 12:25:36
2	202	214335504	You are blocked due to books you had not returned: The book Adventures of Don Juan had to be returned	07/07/2024 12:25:36
3	203	751666263	You are blocked due to books you had not returned: The book Raze had to be returned in 16-JAN-20 The b	07/07/2024 12:25:36
4	204	697846160	You are blocked due to books you had not returned: The book Turner & Hooch had to be returned in 18-N	07/07/2024 12:25:36
5	205	107455196	You are blocked due to books you had not returned: The book Hellraiser: Bloodline had to be returned in 0	07/07/2024 12:25:36
6	206	464019730	You are blocked due to books you had not returned: The book Holding, The had to be returned in 14-JUN-	07/07/2024 12:25:36
7	207	128447737	You are blocked due to books you had not returned: The book Man of Aran had to be returned in 13-AUG-	07/07/2024 12:25:36
8	208	137607044	You are blocked due to books you had not returned: The book Trial of the Incredible Hulk, The had to be re	07/07/2024 12:25:36
9	209	874752399	You are blocked due to books you had not returned: The book Easy Come, Easy Go had to be returned in 1	07/07/2024 12:25:36
10	210	244652438	You are blocked due to books you had not returned: The book Song of Freedom had to be returned in 18-S	07/07/2024 12:25:36
11	211	984268974	You are blocked due to books you had not returned: The book Sita Sings the Blues had to be returned in 13	07/07/2024 12:25:36
12	212	334283784	You are blocked due to books you had not returned: The book 10 had to be returned in 31-OCT-13 The bo	07/07/2024 12:25:36
13	213	983955201	You are blocked due to books you had not returned: The book Lost and Delirious had to be returned in 27-	07/07/2024 12:25:36
14	214	560046205	You are blocked due to books you had not returned: The book Touch of Evil had to be returned in 12-JUL-	07/07/2024 12:25:36
15	215	562094648	You are blocked due to books you had not returned: The book Lord of the Flies had to be returned in 24-M	07/07/2024 12:25:36

1:29 lax@XE [12:26:14] 115 rows selected in 0.119 seconds

SQL Output Statistics				
<pre>select * from client where clientId in (select personId from notifications)</pre>				
				
	CLIENTID	ACTIVE	MAXBOOKS	
1	029124580	0	11	
2	029466664	0	2	
3	029695329	0	2	
4	033035228	0	11	
5	035510904	0	8	
6	039153067	0	4	
7	043939492	0	1	
8	046657439	0	11	
9	063528944	0	1	
10	066613546	0	12	
11	069849743	0	6	
12	070289159	0	9	
13	087859207	0	10	

-ניתן לראות שכל הלקוחות שהפונקציה היתה אמורה לחסום, אכן לא פעילים!

פונקציה **payFine** היא פונקציה שמטרתה לבצע תשלום של קנס

### פרמטרים

**Client\_id** המספר המזהה של הלקוח.

**Payment** סכום לתשלום.

### פעולת הפונקציה

הפונקציה בודקת את החוב של הלקוח ומפחיתה ממנו את התשלום שהועבר לפונקציה. אם החוב שולם במלואו ואם המשתמש לא פעיל, הפונקציה בודקת אם ניתן להפוך אותו לפעיל (אם אין ללקוח ספרים שתאריך החזרתם חלף ועדיין לא החזיר אותם). אם כן, הופכת אותו לפעיל. אם עדיין הוא מחזיק בספרים שחלף מועד החזרתם, הפונקציה מדפיסה הודעה מתאימה (פרטי הספר שצריך להחזיר, וסכום הקנס המשוער נכון לתאריך הנוכחי).

### ערך מוחזר

הפונקציה מחזירה מספר 1 אם הלקוח פעיל, 0 אם הלקוח עדיין חסום. במשתנה שהוכנס – **payment**, מוחזר חישוב יתרת החוב (אם הסכום שלילי) או העודף (אם הסכום חיובי).

### חריגות

הפונקציה מטפלת בחריגות מסוג **NO\_DATA\_FOUND** – מדפיסה הודעה מתאימה.

### קוד הפונקציה

```
create or replace noneditionable function payFine(client_id in varchar2, payment in out number) return number is
/*-----
This function performs a fine payment.
Parameters: client_id - ID of the client who pays the fine.
            payment    - The amount of money he pays.
            In the function, this parameter is updated to the change of the payment: if the payment>0, the
            if the payment<0, the
-----*/
isActive number;
total_fine fines.totalfine%type;
lastInsertFrom varchar2(10); -- for raising a specific exception when not data found

cursor overDueDate is
select lendingId, copyId, title, dueDate
from (select lendingId, copyId, dueDate
      from bookLending
      where clientId = client_id and dueDate<sysdate and returnDate is null)l
natural join (select copyId, bookId
              from Bookcopy)c
              natural join (select bookId, title
                           from book)b;

overDueDateRec overDueDate%rowtype;
begin
lastInsertFrom:='client';
select active into isActive
from client
where clientId=client_id;
lastInsertFrom:='fines';
select totalFine into total_fine
from fines
where clientId=client_id;
--update fine
if total_fine>payment then
update fines
set totalFine = total_fine-payment
where clientId = client_id;
else
```



```
-- There is no remaining fine
-- 1. Remove the client from fines
delete from fines
where clientId = client_id;
-- 2. The client should be active...
if isActive = 0 then
    -- Checking if he has books whose due date has passed, If so, print their details
    isActive:= 1;
    for overDueDateRec in overDueDate
        loop
            if isActive=1 then
                isActive:=0;
            end if;
            dbms_output.put_line('-----');
            dbms_output.put_line('Book copy #'||overDueDateRec.Copyid||' ('||overDueDateRec.Title||')
            had to be returned in: '||overDueDateRec.Duedate);
            dbms_output.put_line('The amount of the estimated fine for this book is: '||calculateFine_book
            (overDueDateRec.Lendingid)||' shekel.');
```

---

```
            dbms_output.put_line('-----');
        end loop;
    -- If client can be active, update it
    if isActive=1 then
        update client
        set active= isActive
        where clientId=client_id;
        dbms_output.put_line('Total fine was paid, client: '||client_id||' is active now.');
```

---

```
    end if;
end if;
--calculate change or the remaining fine (positive payment=> change. negative payment=> remaining fine).
payment:= payment-total_fine;

return (isActive);

EXCEPTION
WHEN NO_DATA_FOUND THEN
    if lastInsertFrom = 'client' then
        RAISE_APPLICATION_ERROR(-20001, 'Client ID: '||client_id||' does not exist in the system.');
```

---

```
    else
        dbms_output.put_line('There is not a fine for client: '||client_id);
        return (isActive);
    end if;
end payFine;
```

## דוגמת הרצה

The screenshot shows the SQL Developer interface. The top pane displays a PL/SQL block with the following code:

```
1 begin
2     -- Call the function
3     :result := payFine(client_id => :client_id,
4                       payment => :payment);
5 end;
```

The bottom pane shows the Variable Window with the following data:

Variable	Type	Value
result	Float	0
client_id	String	013427448
payment	Float	50

The status bar at the bottom indicates the execution was successful and completed in 0.023 seconds.

Test script DBMS Output Statistics Profiler Trace

Clear Buffer size 10000 ☒ Enabled

```

Book copy #35 (Happy Vacation) had to be returned in: 24-SEP-18
The amount of the estimated fine for this book is: 10568.03 shekel.
  
```

5:1 lax@XE [14:32:34] Executed in 0.021 seconds

לפני פעולת הפונקציה היתה ללקוח רשומת קנס

	CLIENTID	TOTALFINE
▶ 1	013427448	250

לאחר פעולת הפונקציה אין לו קנס (אין רשומה על שמו):

```

select *
from fines
where clientId = '013427448'
  
```

CLIENTID TOTALFINE

# Procedures

1.

הפרוצדורה ***lending\_book*** מבצעת השאלת ספר ללקוח.

## פרמטרים

*Librarian\_id* מספר מזהה הספרן/ית.

*Client\_id* מספר מזהה הלקוח.

*Copy\_id* מספר העותק שברצונו לשאול.

## פעולת הפרוצדורה

הפרוצדורה בודקת אם הלקוח יכול לשאול ספר (הלקוח פעיל, וכמות ההשאלות שלו בפועל (של ספרים שעדיין לא החזיר).

לאחר מכן, הפרוצדורה בודקת אם ניתן לשאול את העותק הנוכחי.

אם כן, הפרוצדורה מחשבת את הזמן המיועד להחזרת הספר, ומוסיפה רשומה מתאימה לטבלת ההשאלות. (הטריגר שהוספנו, מעדכן מיד גם את זמניות הספר בטבלת העותקים)

לסיום, הפרוצדורה מדפיסה הודעה ללקוח עם פרטי ההשאלה והזמן המיועד להחזרת הספר.

## חריגות

להלן טבלה של חריגות שעלולות להיזרק מהפרוצדורה (אין טיפול בחריגות אלו בפרוצדורה כי אנו מעוניינים שמי שזימן את הפרוצדורה יהיה מודע לבעיה ויטפל בה כראות עיניו).

בנוסף, יש טיפול בחריגות של NO\_DATA\_FOUND (במידה ומס' העותק לא קיים במערכת), במצב זה תיזרק חריגה מתאימה.

שם החריגה	SQLCODE	תאור
client_not_exists	-20001	זרקת כאשר הלקוח לא קיים במערכת.
book_not_exists	-20002	זרקת כאשר העותק לא קיים במערכת.
client_not_active	-20003	זרקת כאשר הלקוח לא פעיל (ומנסה לבצע השאלה).
over_max_books	-20004	זרקת כאשר הלקוח מנסה לבצע השאלה, בעוד שמחזיק אצלו את כמות הספרים המקסימלית עבורו.
book_not_available	-20005	זרקת כאשר העותק לא זמין.

## קוד הפרוצדורה

```
create or replace noneditionable procedure lending_book(librarian_id in varchar, client_id in varchar, copy_id
in number) is
/*-----
This procedure performs a lending to a requested book.
parameters: librarian_id, client_id, copy_id.
-----*/

tmp_bool number(1);
tmp_num integer;
max_books number(2);
due_date date;
lastInsertFrom varchar2(10); -- for raising a specofic exceptin when not data found
client_not_exists EXCEPTION;
book_not_exists EXCEPTION;
client_not_active EXCEPTION;
over_max_books EXCEPTION;
book_not_avaiableble EXCEPTION;
PRAGMA EXCEPTION_INIT( client_not_exists, -20001 );
PRAGMA EXCEPTION_INIT( book_not_exists, -20002 );
PRAGMA EXCEPTION_INIT( client_not_active, -20003 );
PRAGMA EXCEPTION_INIT( over_max_books, -20004 );
PRAGMA EXCEPTION_INIT( book_not_avaiableble, -20005 );
```

```

begin
-- Checking if the client is allowed to lend a book
-- Checking if the client is active
lastInsertFrom:='client';
select active, maxBooks into tmp_bool, max_books
from client
where clientId = client_id;
if tmp_bool = 0 then
    RAISE_APPLICATION_ERROR(-20003, 'Client ID: '||client_id||' is not active.');
```

end if;

```

-- Checking if the client has not exceeded the max books
select count(*) into tmp_num
from bookLending
where clientId = client_id and returnDate is null;
if tmp_num >= max_books then
    RAISE_APPLICATION_ERROR(-20004, 'Client ID: '||client_id||' has already lent the maximum number of books
    he is allowed.');
```

end if;

```

-- Checking if the book is available
lastInsertFrom:='bookcopy';
select available into tmp_bool
from Bookcopy
where copyId = copy_id;
if tmp_bool=0 then
    RAISE_APPLICATION_ERROR(-20005, 'Book Copy ID: '||copy_id||' is not available.');
```

end if;

```

-- Calculation of dueDate
-- Checking the amount of copies available of this book
select count(*) into tmp_num
from Bookcopy
where bookId = (select bookId from bookCopy where copyId = copy_id) and
    available = 1;
if tmp_num = 1 then
    due_date:= TRUNC(SYSDATE + 7);
elseif tmp_num < 4 then
    due_date:= TRUNC(SYSDATE + 14);
else due_date:= TRUNC(ADD_MONTHS((SYSDATE),1));
end if;
```

```

-- Inserting new tuple into bookLending table
insert into bookLending(DUEDATE, CLIENTID, LIBRARIANID, COPYID)
values (due_date, client_id, librarian_id, copy_id)
returning lendingId into tmp_num;
```

```

-- Printing a success message with the due date
if SQL%FOUND then
    dbms_output.put_line('The lending was done successfully.');
```

dbms\_output.put\_line('LendingId: '||tmp\_num);

```

    dbms_output.put_line('The book must be returned by: '||to_char(due_date, 'dd-mon-yyyy'));
end if;
```

```

exception
when NO_DATA_FOUND then
    if lastInsertFrom = 'client' then
        RAISE_APPLICATION_ERROR(-20001, 'Client ID: '||client_id||' does not exist in the system.');
```

elseif lastInsertFrom = 'bookcopy' then

```

        RAISE_APPLICATION_ERROR(-20002, 'Book Copy ID: '||copy_id||' does not exist in the system.');
```

end if;

```

end lending book;
```



## דוגמת הרצה

Test script
DBMS Output
Statistics
Profiler
Trace

```

1 begin
2     -- Call the procedure
3     lending_book(librarian_id => :librarian_id,
4                 client_id => :client_id,
5                 copy_id => :copy_id);
6 end;

```

	Variable	Type	Value
<input checked="" type="checkbox"/>	librarian_id	String	087859207
<input checked="" type="checkbox"/>	client_id	String	418542865
<input checked="" type="checkbox"/>	copy_id	Float	658
* <input checked="" type="checkbox"/>			

1:1
lax@XE
[13:37:42] Executed in 0.01 seconds

נבדוק את ההדפסה:

The screenshot shows the SQL Developer application window. At the top, there are tabs for 'Test script', 'DBMS Output', 'Statistics', 'Profiler', and 'Trace'. The 'DBMS Output' tab is active, showing a list of messages. The messages are: 'The lending was done successfully.', 'LendingId: 402', and 'The book must be returned by: 21-jul-2024'. Above the messages, there is a 'Clear' button, a 'Buffer size' dropdown set to '10000', and a checked checkbox labeled 'Enabled'. The status bar at the bottom of the window shows the session name 'lax@XE' and the execution time '[13:37:42] Executed in 0.01 seconds'.



	LENDINGID	LENDINGDATE	DUEDATE	RETURNDATE	CLIENTID	LIBRARIANID	COPYID
1	402	07/07/2024 13:37:42	21/07/2024		418542865	087859207	658
2	346	22/12/2023	05/01/2024		494302245	691006719	154
3	343	07/12/2023	21/12/2023		782812737	146137062	138
4	38	02/11/2023	16/11/2023		963083303	694117241	467
5	299	11/10/2023	25/10/2023		310404144	268281389	962
6	126	07/10/2023	21/10/2023		376247399	771616343	580
7	274	01/10/2023	15/10/2023		803660894	241206486	169
8	85	21/09/2023	05/10/2023		846104984	934564949	91
9	231	17/09/2023	01/10/2023		898194084	309732749	587
10	347	05/07/2023	19/07/2023		396904603	066163526	647
11	27	28/06/2023	12/07/2023		874752399	319029281	292
12	236	02/05/2023	16/05/2023		446332100	659345030	603
13	225	27/04/2023	11/05/2023		274220045	461510573	603

הטריגר עדכן את זמינות הספר:

	COPYID	AVAILABLE	YEARPUBLISHED	EDITION	BOOKID
1	658	0	1999	1	492

## הפרוצדורה *returning\_book* מבצעת החזרת ספר.

### פרמטר

*Copy\_id* הפרוצדורה מקבלת את המספר המזהה של העותק המושאל.

### פעולת הפרוצדורה

הפרוצדורה בודקת שהספר אכן מושאל בצורה תקינה, אם כן:  
 א. בודקת אם הספר מוחזר באיחור, אם כן, מחשבת את הקנס (בעזרת פונקציית העזר *calculatefine\_book*) ומוסיפה או מעדכנת רשומת קנס ללקוח.  
 ב. מעדכנת את העמודה *returnDate* לתאריך הנוכחי ברשומת ההשאלה המתאימה.  
 לסיום, הפרוצדורה מדפיסה הודעה על הצלחת ההחזרה, ובמידה ויש קנס, מדפיסה אותו.

### חריגות

להלן טבלה של חריגות שעלולות להיזרק מהפרוצדורה (אין טיפול בחריגות אלו בפרוצדורה כי אנו מעונינים שמי שזימן את הפרוצדורה יהיה מודע לבעיה ויטפל בה כראות עיניו).  
 בנוסף, יש טיפול בחריגות של *NO\_DATA\_FOUND* (במידה ומס' העותק לא קיים במערכת), במצב זה תודפס הודעה מתאימה / תיזרק חריגה משלנו.

שם החריגה	SQLCODE	תאור
<i>book_not_borrowed</i>	-20011	נזרקת כאשר העותק אינו רשום כמושאל כרגע ( <i>returnDate is null</i> ).
<i>book_multiple_borrowed</i>	-20012	נזרקת כאשר העותק רשום כמושאל בפועל ליותר מלקוח אחד.

### קוד הפרוצדורה

```
create or replace noneditionable procedure returning_book(copy_id in number) is
/*-----
This procedure performs a returning a specific book.
parameter: copy_id.
-----*/

fineStr varchar2(100);
finePerDay constant number(2) := 5;
calculatedFine fines.totalfine%type;
book_title book.title%type;

book_not_borrowed EXCEPTION;
PRAGMA EXCEPTION_INIT( book_not_borrowed, -20011 );
book_multiple_borrowed EXCEPTION;
PRAGMA EXCEPTION_INIT( book_multiple_borrowed, -20012 );

cursor c is
  select * from Booklending
  where copyId = copy_id and returnDate is null
  for update of returnDate;
lending_rec c%rowtype;

begin
  select title into book_title
  from bookcopy natural join book
  where copyid = copy_id;
```



```
open c;
fetch c into lending_rec;
if c%notfound then
    RAISE_APPLICATION_ERROR(-20011, 'Book copy: '||copy_id||' is not borrowed.');
```

```
elseif c%rowcount > 1 then
    RAISE_APPLICATION_ERROR(-20012, 'Book copy: '||copy_id||' is borrowed for multiple clients...');
```

```
elseif sysdate > lending_rec.duedate then -- The book was returned late. A fine must be calculated for it.
    calculatedFine:= (sysdate - lending_rec.dueDate)*finePerDay;
    MERGE INTO fines f
    USING (SELECT lending_rec.clientid clientId, calculatedFine totalFine from dual) tmpTable
    ON (f.clientId = tmpTable.clientId)
    WHEN MATCHED THEN UPDATE SET f.totalFine = f.totalfine+tmpTable.Totalfine
    WHEN NOT MATCHED THEN INSERT (clientId, totalFine) VALUES (tmpTable.clientId, tmpTable.totalFine);
    fineStr:= chr(13)||chr(10)||'Client #'||lending_rec.clientid||' owes a fine of '||calculatedFine||'
    NIS for this book.';
end if;
update Booklending -- Update the returningDate (Current date).
    set returnDate = sysdate
    where current of c;
close c;

dbms_output.put_line('The book '||book_title||' has been successfully returned'||fineStr);
exception
    when NO_DATA_FOUND then
        RAISE_APPLICATION_ERROR(-20002, 'Book Copy ID: '||copy_id||' does not exist in the system.');
```

```
end returning_book;
```

דוגמת הרצה

Test script ☒ DBMS Output ☐ Statistics ☐ Profiler ☐ Trace

```
1 begin
2     -- Call the procedure
3     returning_book(copy_id => :copy_id);
4 end;
```

	Variable	Type	Value
▶	copy_id	Float	658
*			

1:1 lax@XE [17:13:21] Executed in 0.043 seconds

Test script DBMS Output Statistics Profiler Trace

Clear Buffer size 10000 ☒ Enabled


The book House has been successfully returned

טבלת ההשאלות לאחר פעולת הפרוצדורה

LENDINGID	LENDINGDATE	DUEDATE	RETURNDATE	CLIENTID	LIBRARIANID	COPYID
402	07/07/2024 13:37:42	21/07/2024	07/07/2024 17:13:21	418542865	087859207	658

הטריגר עדכן את העמודה *active*:

select \*  
from bookcopy  
where copyId = 658



	COPYID	AVAILABLE	YEARPUBLISHED	EDITION	BOOKID
▶ 1	658	1	1999	1	492



הפרוצדורה `extensionOfAllLendings_book` היא פונקציה שמטרתה לבצע הארכת כל ההשאלות הפעילות של לקוח נתון

### פרמטר

`Client_id` המספר המזהה של הלקוח

### פעולת הפרוצדורה

הפרוצדורה בודקת שהלקוח אכן קיים.

אם כן, עוברת על כל אחת מההשאלות הפעילות שלו ומחשבת את התאריך המאוחר ביותר שניתן להאריך אליו את ההשאלה (בעזרת פונקציית העזר `calculateDueDate_book`).

אם התאריך המחושב שונה מהתאריך המיועד להחזרה השמור כרגע ברשומה, הפרוצדורה מעדכנת את הרשומה.

הפרוצדורה מדפיסה הודעה מתאימה אם ההארכה הצליחה או לא.

בסיום הפעולה, הפרוצדורה מדפיסה את כמות ההארכות שבוצעו במהלכה.

### חריגות

הפרוצדורה זורקת חריגה אם הלקוח לא קיים במערכת.

### קוד הפרוצדורה

```
create or replace noneditionable procedure extensionOfAllLendings_book(client_id in varchar2) is
/*-----
This procedure extends all book lendings of specific client.
parameter: client_id
-----*/
i number:=0;
due_date date;
book_title book.title%type;
flag integer;

cursor lendingsBooks is
select *
from booklending
where clientId=client_id and returnDate is null
for update of returnDate;
lendingBooksRec lendingsBooks%rowtype;

begin
    --Checking if client_id exists.
    select count(*) into flag
    from client
    where clientId = client_id;
    if flag = 0 then
        RAISE_APPLICATION_ERROR(-20001, 'Client ID: '||client_id||' does not exist in the system.');
```

```
    end if;

    for lendingBooksRec in lendingsBooks
    loop
        flag:= 0;
        -- Calculating the possible due date by calculateDueDate_book function.
        due_date:= calculateDueDate_book(lendingBooksRec.copyId);
        -- Selecting title for printing lending details
        select title into book_title
        from bookcopy natural join book
        where copyid = lendingBooksRec.Copyid;

        if due_date>lendingBooksRec.Duedate then
```













```

update bookLending
set dueDate = due_date
where current of lendingsBooks;
if SQL%rowcount>0 then
    i:= i+1;
    -- Printing details of the updating lending
    dbms_output.put_line('Book copy #'||lendingBooksRec.Copyid||' ('||book_title||') due date has
    been successfully extended. ');
    dbms_output.put_line('The previous due date: '||lendingBooksRec.Duedate);
    dbms_output.put_line('The updated due date: '||due_date);
    dbms_output.new_line();
end if;
else
    dbms_output.put_line('Book copy #'||lendingBooksRec.Copyid||' ('||book_title||') due date could
    not been extended. ');
    dbms_output.put_line('The due date: '||lendingBooksRec.Duedate);
    dbms_output.new_line();
end if;
end loop;
if flag=1 then
    dbms_output.put_line('The client has no books that he did not return');
else
    dbms_output.put_line('The due date of '||i||' books has been successfully extended. ');
end if;
end extensionOfAllLendings_book;

```

## דוגמת הרצה

Test script
DBMS Output
Statistics
Profiler
Trace















```

1 begin
2     -- Call the procedure
3     extensionOfAllLendings_book(client_id => :client_id);
4 end;

```

	Variable	Type	Value
▶	client_id	String	898194084
*			


1:1
lax@XE
[19:40:01] Executed in 0.055 seconds

Test script
DBMS Output
Statistics
Profiler
Trace

Clear
Buffer size 10000
☒ Enabled

Book copy #587 (Sunshine) due date has been successfully extended.  
The previous due date: 01-OCT-23  
The updated due date: 14-JUL-24

Book copy #954 (A Blank on the Map) due date has been successfully extended.  
The previous due date: 20-DEC-15  
The updated due date: 21-JUL-24

Book copy #447 (Fritz the Cat) due date has been successfully extended.  
The previous due date: 25-APR-19  
The updated due date: 21-JUL-24

Book copy #407 (Campfire Tales) due date has been successfully extended.  
The previous due date: 20-MAY-16  
The updated due date: 07-AUG-24

The due date of 4 books has been successfully extended.

18:1
lax@XE
[19:40:01] Executed in 0.055 seconds

לפני פעולת הפרוצדורה

```
select *
from client natural join bookLending
where clientId = '898194084'
```

	CLIENTID	ACTIVE	MAXBOOKS	LENDINGID	LENDINGDATE	DUEDATE	RETURNDATE	LIBRARIANID	COPYID
1	898194084	1	10	231	17/09/2023	01/10/2023		309732749	587
2	898194084	1	10	255	15/01/2000	20/12/2015		461518573	954
3	898194084	1	10	278	25/07/2015	25/04/2019		367775460	447
4	898194084	1	10	372	22/09/2014	20/05/2016		279175065	407

לאחר פעולת הפרוצדורה

```
select *
from client natural join bookLending
where clientId = '898194084'
```

	CLIENTID	ACTIVE	MAXBOOKS	LENDINGID	LENDINGDATE	DUEDATE	RETURNDATE	LIBRARIANID	COPYID
1	898194084	1	10	231	17/09/2023	14/07/2024		309732749	587
2	898194084	1	10	255	15/01/2000	21/07/2024		461518573	954
3	898194084	1	10	278	25/07/2015	21/07/2024		367775460	447
4	898194084	1	10	372	22/09/2014	07/08/2024		279175065	407

# תוכנית / Library\_maintenance

תוכנית זו מדמה תוכנית לתחזוקת בסיס הנתונים של הספרייה

## פעולת התוכנית

- א. התוכנית מגרילה מספר לוחות שאמורים להיחסם, ומאריכה להם את כל ההשאלות הפעילות בעזרת הפרוצדורה `extensionofalllendings_book`.
- ב. בעזרת הפונקציה `blockclients` התוכנית חוסמת את כל הלקוחות שמחזיקים בספרים שמועד החזרתם חלף לפני מספר שהתוכנית הגרילה.
- ג. התוכנית מדפיסה למסך את פרטי יצירת הקשר של כל הלקוחות שנחסמו, יחד עם ההודעה המפרטת להם את סיבת החסימה (כפי ששמורה בטבלת `notifications`).

## קוד התוכנית

```

/*-----
This program simulates a library maintenance program.
The program performs:
1. Extending book lendings to random clients
2. Blocking clients
3. Notification of blocked clients
-----*/

declare
    max_late integer;
    random_clients_counter integer;
    blocked_ammount number;

    cursor random_clients is
    select distinct clientId
    from booklending natural join client
    where active = 1 and returnDate is null and sysdate-duedate > max_late
    order by dbms_random.value;
    client_rec random_clients%rowtype;
begin
    max_late:= dbms_random.value(30, 120);

    dbms_output.put_line('*****
    dbms_output.put_line('                               Welcome to the the library maintenance program');
    dbms_output.put_line('*****
    dbms_output.new_line();

    -- Extention of all book lendings of random clients
    random_clients_counter:= dbms_random.value(1, 15);
    dbms_output.put_line('The system has chosen randomly '||random_clients_counter||' customers and is extending all
    the lendings of the books they hold...');
    for client_rec in random_clients
    loop
        dbms_output.put_line('-----
        dbms_output.put_line('Client #'||client_rec.clientId||', you have been saved from a blocking!!!');
        dbms_output.put_line('The system is extending all the lendings of the books you hold...'||chr(13)||chr(10));
        extensionofalllendings_book(client_rec.Clientid);
        random_clients_counter:= random_clients_counter-1;
        dbms_output.put_line('-----
        exit when random_clients_counter=0;
    end loop;

    --Blocking all clients who have books whose due date has passed before more than max_late days.
    blocked_ammount:= blockclients(max_late);
    dbms_output.put_line(blocked_ammount||' clients had blocked!');

    --Print notifications details
    dbms_output.put_line(chr(13)||chr(10)||'The system is printing the blocked clients contact information and the messages...'
    ||chr(13)||chr(10));
    for item in (select phone, firstName, lastName, msg
        from person natural join notifications
        where trunc(msgDate)=trunc(sysdate))
    loop
        dbms_output.put_line('Phone: '||item.phone);
        dbms output.put line('Name: '||item.firstname||' '||item.lastname);
        dbms_output.put_line('Message: '||chr(13)||chr(10)||item.msg);
    end loop;

end;

```



```

*****
Welcome to the the library maintenance program
*****

The system has chosen randomly 6 customers and is extending all the lendings of the books they hold...

-----

Client #915923130, you have been saved from a blocking!!!
The system is extending all the lendings of the books you hold...

Book copy #207 (Krabat) due date has been successfully extended.
The previous due date: 10-APR-20
The updated due date: 21-JUL-24

Book copy #952 (Chase, The) due date has been successfully extended.
The previous due date: 29-JUL-11
The updated due date: 21-JUL-24

The due date of 2 books has been successfully extended.

-----

Client #669562483, you have been saved from a blocking!!!
The system is extending all the lendings of the books you hold...

Book copy #196 (Debt, The) due date has been successfully extended.
The previous due date: 05-MAY-22
The updated due date: 14-JUL-24

The due date of 2 books has been successfully extended.

-----

Client #725276037, you have been saved from a blocking!!!
The system is extending all the lendings of the books you hold...

Book copy #5 (Last Ride) due date has been successfully extended.
The previous due date: 25-APR-13
The updated due date: 07-AUG-24

Book copy #354 (Quiet Ones, The) due date has been successfully extended.
The previous due date: 25-AUG-22
The updated due date: 21-JUL-24

The due date of 2 books has been successfully extended.

-----

Client #149245218, you have been saved from a blocking!!!
The system is extending all the lendings of the books you hold...

Book copy #728 (Man of Aran) due date has been successfully extended.
The previous due date: 19-APR-10
The updated due date: 21-JUL-24

Book copy #553 (Walk All Over Me) due date has been successfully extended.
The previous due date: 27-OCT-11
The updated due date: 07-AUG-24

The due date of 2 books has been successfully extended.

-----

Client #751666263, you have been saved from a blocking!!!
The system is extending all the lendings of the books you hold...

Book copy #726 (Love and Death on Long Island) due date has been successfully extended.
The previous due date: 21-SEP-19
The updated due date: 14-JUL-24

Book copy #454 (Raze) due date has been successfully extended.
The previous due date: 16-JAN-20
The updated due date: 14-JUL-24

Book copy #822 (Dr. Jekyll and Mr. Hyde) due date has been successfully extended.
The previous due date: 20-MAR-20
The updated due date: 21-JUL-24

The due date of 3 books has been successfully extended.

-----

108 clients had blocked!

The system is printing the blocked clients contact information and the messages...

```



Phone: 051-3333906  
Name: Ani Cotton  
Message:  
You are blocked due to books you had not returned:  
The book Easy Come, Easy Go had to be returned in 12-JUL-23  
The book Miracle at Oxford (True Blue) had to be returned in 29-NOV-20  
The book Two Weeks in September had to be returned in 24-NOV-20

Phone: 055-6609282  
Name: Nickel Davidtz  
Message:  
You are blocked due to books you had not returned:  
The book Song of Freedom had to be returned in 18-SEP-13  
The book Arizona Dream had to be returned in 19-OCT-12

Phone: 054-3838503  
Name: Giovanni McConaughy  
Message:  
You are blocked due to books you had not returned:  
The book Sita Sings the Blues had to be returned in 13-OCT-19

Phone: 055-3737119  
Name: Annie Goodall  
Message:  
You are blocked due to books you had not returned:  
The book 10 had to be returned in 31-OCT-13  
The book Invisible Target (Naam yi boon sik) had to be returned in 16-JAN-20  
The book A Lesson Before Dying had to be returned in 05-MAY-23  
The book Nightmare on Elm Street 5: The Dream Child, A had to be returned in 24-DEC-19

Phone: 056-1193184  
Name: Randall Saxon  
Message:  
You are blocked due to books you had not returned:  
The book Lost and Delirious had to be returned in 27-MAR-22

Phone: 053-6877496  
Name: Mika Botti  
Message:  
You are blocked due to books you had not returned:  
The book Touch of Evil had to be returned in 12-JUL-14

Phone: 052-5872116  
Name: Bradley Reid  
Message:  
You are blocked due to books you had not returned:  
The book Lord of the Flies had to be returned in 24-MAR-24  
The book Miss Representation had to be returned in 31-MAR-16

Phone: 057-6843669  
Name: Ernie Broadbent  
Message:  
You are blocked due to books you had not returned:  
The book Far From Home: The Adventures of Yellow Dog had to be returned in 12-NOV-11  
The book White Fang (Zanna Bianca) had to be returned in 02-FEB-15

Phone: 056-5163114  
Name: Cevin Kirkwood  
Message:  
You are blocked due to books you had not returned:  
The book 2012: Supernova had to be returned in 11-DEC-21

Phone: 057-5257322  
Name: Kathleen Botti  
Message:  
You are blocked due to books you had not returned:  
The book Impy's Island had to be returned in 02-NOV-20  
The book Devil's Carnival, The had to be returned in 18-MAY-22  
The book Romeo and Juliet had to be returned in 12-JAN-10

Phone: 050-9544354  
Name: Loren Curfman  
Message:  
You are blocked due to books you had not returned:  
The book Ars?ne Lupin had to be returned in 06-JAN-12

התוכנית ממשיכה בהדפסות באותו האופן לכל החסומים...

```
select * from client
where clientId in (select personId from notifications)
```



	CLIENTID	ACTIVE	MAXBOOKS
1	029124580	0	11
2	029466664	0	2
3	029695329	0	2
4	033035228	0	11
5	035510904	0	8
6	039153067	0	4
7	043939492	0	1
8	046657439	0	11
9	063528944	0	1
10	066613546	0	12
11	069849743	0	6
12	070289159	0	9
13	087859207	0	10
14	093818015	0	12
15	107455196	0	1
16	126770208	0	9
17	127097395	0	9
18	128447737	0	4
19	136074808	0	2
20	137607044	0	12

2:55 lax@XE [20:53:00] 108 rows selected in 0.527 seconds

טבלת notifications לאחר ריצת התוכנית

```
select *
from notifications
```



	NOTIFYID	PERSONID	MSG	MSGDATE
1	782	874752399	You are blocked due to books you had not returned: The book Easy Come, Easy Go had to be returned in 12-JUL-23 The book Miracle at C	07/07/2024 20:29:04
2	783	244652438	You are blocked due to books you had not returned: The book Song of Freedom had to be returned in 18-SEP-13 The book Arizona Drear	07/07/2024 20:29:04
3	784	984268974	You are blocked due to books you had not returned: The book Sita Sings the Blues had to be returned in 13-OCT-19	07/07/2024 20:29:04
4	785	334283784	You are blocked due to books you had not returned: The book 10 had to be returned in 31-OCT-13 The book Invisible Target (Naam yi bo	07/07/2024 20:29:04
5	786	983955201	You are blocked due to books you had not returned: The book Lost and Delirious had to be returned in 27-MAR-22	07/07/2024 20:29:04
6	787	560046205	You are blocked due to books you had not returned: The book Touch of Evil had to be returned in 12-JUL-14	07/07/2024 20:29:04
7	788	562094648	You are blocked due to books you had not returned: The book Lord of the Flies had to be returned in 24-MAR-24 The book Miss Represer	07/07/2024 20:29:04
8	789	029124580	You are blocked due to books you had not returned: The book Far From Home: The Adventures of Yellow Dog had to be returned in 12-M	07/07/2024 20:29:04
9	790	029466664	You are blocked due to books you had not returned: The book 2012: Supernova had to be returned in 11-DEC-21	07/07/2024 20:29:04
10	791	350227847	You are blocked due to books you had not returned: The book Impy's Island had to be returned in 02-NOV-20 The book Devil's Carnival,	07/07/2024 20:29:04
11	792	696456011	You are blocked due to books you had not returned: The book Ars?ne Lupin had to be returned in 06-JAN-12	07/07/2024 20:29:04
12	793	591957858	You are blocked due to books you had not returned: The book Spring Subway had to be returned in 26-APR-21	07/07/2024 20:29:04
13	794	033035228	You are blocked due to books you had not returned: The book Viva had to be returned in 25-APR-14	07/07/2024 20:29:04
14	795	856216050	You are blocked due to books you had not returned: The book Crude Oasis, The had to be returned in 31-MAY-19	07/07/2024 20:29:04
15	796	888279161	You are blocked due to books you had not returned: The book Miracle at Oxford (True Blue) had to be returned in 14-AUG-20	07/07/2024 20:29:04
16	797	520422787	You are blocked due to books you had not returned: The book Mutants had to be returned in 22-JAN-20	07/07/2024 20:29:04
17	798	448707100	You are blocked due to books you had not returned: The book Hellzapoppin' had to be returned in 06-APR-23	07/07/2024 20:29:04
18	799	376247399	You are blocked due to books you had not returned: The book Primal had to be returned in 21-OCT-23	07/07/2024 20:29:04
19	800	039153067	You are blocked due to books you had not returned: The book Man of Aran had to be returned in 11-FEB-17 The book Spidenwick Chroni	07/07/2024 20:29:04
20	801	185553927	You are blocked due to books you had not returned: The book Not Without My Daughter had to be returned in 13-JUN-16	07/07/2024 20:29:04

1 of 108 lax@XE [20:55:18] 108 rows selected in 0.113 seconds



## תוכנית / Library\_main

תוכנית זו מדמה את מערכת השאלות הספרים בספרייה.

### פעולת התוכנית

התוכנית מנסה להשאל ספר ללקוח אקראי בעזרת הפרוצדורה *lending\_book*.

אם הלקוח נחסם עקב איחור בספרים או עקב קנס שנרשם על שמו, התוכנית מחזירה את כל הספרים שברשותו וחלף מועד החזרתם בעזרת הפרוצדורה *returning\_book* וכן מבצעת תשלום קנס והחזרת עודף בעזרת הפונקציה *payFine*.

לאחר מכן, התוכנית מבצעת שוב את השאלת הספר.

אם ההשאלה נכשלה בגלל שהלקוח שאל את כמות הספרים המקסימלית המותרת, התוכנית מדפיסה הודעה מתאימה.

### קוד התוכנית

```
/*-----  
This program simulates the system of book lendings in the library.  
The program tries to lend a book to a random customer.  
If the customer is blocked due to books he holds late or due to a fine registered on his name,  
the program returns all the books in his possession and the date for their return has passed, as well as payment of a  
fine and the return of the excess.  
After that, the program performs the book loan again.  
If the loan failed because the client borrowed the maximum amount of books allowed,  
the program prints an appropriate message.  
-----*/  
  
declare  
  librarian_ID librarian.librarianid%type;  
  client_ID client.clientid%type;  
  copy_ID bookcopy.copyid%type;  
  book_ID book.bookid%type;  
  client_name varchar2(100);  
  book_title book.title%type;  
  fine_payment fines.totalfine%type;  
  isActive number:= 0;  
  flag boolean:= true;  
  
  cursor over_due_date_books is  
    select copyId  
    from bookLending  
    where clientId = client_ID and returnDate is null and dueDate<TRUNC(sysdate);  
  book_rec over_due_date_books%rowtype;  
  
begin  
  dbms_output.put_line('*****  
dbms_output.put_line('                               Welcome to the the library program');  
dbms_output.put_line('*****  
  
  --select randomly librarian  
  select librarianId into librarian_ID  
  from librarian  
  order by dbms_random.value  
  fetch first row only;  
  
  --select randomly client  
  select clientid into client_ID  
  from client  
  order by dbms_random.value  
  fetch first row only;  
  
  --select randomly book  
  select copyid, bookId into copy_ID, book_ID  
  from bookcopy  
  where available=1  
  order by dbms_random.value  
  fetch first row only;  
  
  --select client and book details for user friendly message...  
  select firstName||' '||lastName into client_name  
  from person  
  where personId = client_ID;
```



```
select title into book_title
from book
where bookId = book_ID;

dbms_output.put_line('Client #'||client_ID||' ('||client_name||') wants to lend the book copy #'||copy_ID||' ('||book_title||').');

lending_book(librarian_id => librarian_ID, client_id => client_ID, copy_id => copy_ID);

EXCEPTION
when OTHERS then
case SQLCODE
when -20003 then
dbms_output.new_line();
dbms_output.put_line('The client is not active!!!');
for book_rec in over_due_date_books
loop
if flag then
dbms_output.new_line();
dbms_output.put_line('The client returns the books that are overdue...');
flag:= false;
end if;
dbms_output.new_line();
returning_book(book_rec.copyid);
end loop;
if flag = false then
dbms_output.new_line();
end if;
--check for a fine
for fine in (select totalfine from fines where clientId = client_ID)
loop
dbms_output.put_line('The client have a fine of '||fine.totalfine||' NIS. ');
fine_payment:= ROUND(fine.totalfine+5, -1);
dbms_output.put_line('The client pays '||fine_payment||' NIS');
isActive:= payfine(client_id => client_ID, payment => fine_payment);
dbms_output.put_line('The client receives an excess of '||fine_payment||' NIS. ');
end loop;
if isActive = 1 then
-- The client is active now, so he tries to lend the book again
dbms_output.new_line();
dbms_output.put_line('The client tries to lend the book again...');
lending_book(librarian_id => librarian_ID, client_id => client_ID, copy_id => copy_ID);
dbms_output.new_line();
end if;
when -20004 then
dbms_output.put_line('Too much books');
end case;
END;
```

## דוגמת הרצה

```
*****
Welcome to the the library program
*****
Client #480612577 (Naomi Mathis) wants to lend the book copy #576 (Sweet Nothing).

The client is not active!!!

The client returns the books that are overdue...

The book The Burglar has been successfully returned
Client #480612577 owes a fine of 4779 NIS for this book.

The client have a fine of 4779 NIS.
The client pays 4780 NIS
Total fine was paid, client: 480612577 is active now.
The client receives an excess of 1 NIS.

The client tries to lend the book again...
The lending was done successfully.
LendingId: 421
The book must be returned by: 14-jul-2024
```

## טבלת ההשאלות של הלקוח המוגרל לאחר ריצת התוכנית

```
select *
from Booklending
where clientId = '480612577'
```

	LENDINGID	LENDINGDATE	DUE DATE	RETURNDATE	CLIENTID	LIBRARIANID	COPYID
1	421	07/07/2024 20:56:17	14/07/2024		480612577	485100893	576
2	91	10/11/2019	25/11/2021	07/07/2024 20:56:17	480612577	376926882	948

## רשומת הלקוח המוגרל בטבלת client לאחר ריצת התוכנית

```
select *
from client
where clientId = '480612577'
```

	CLIENTID	ACTIVE	MAXBOOKS
1	480612577	1	11

## ניתן לראות שאין רשומת קנס ללקוח

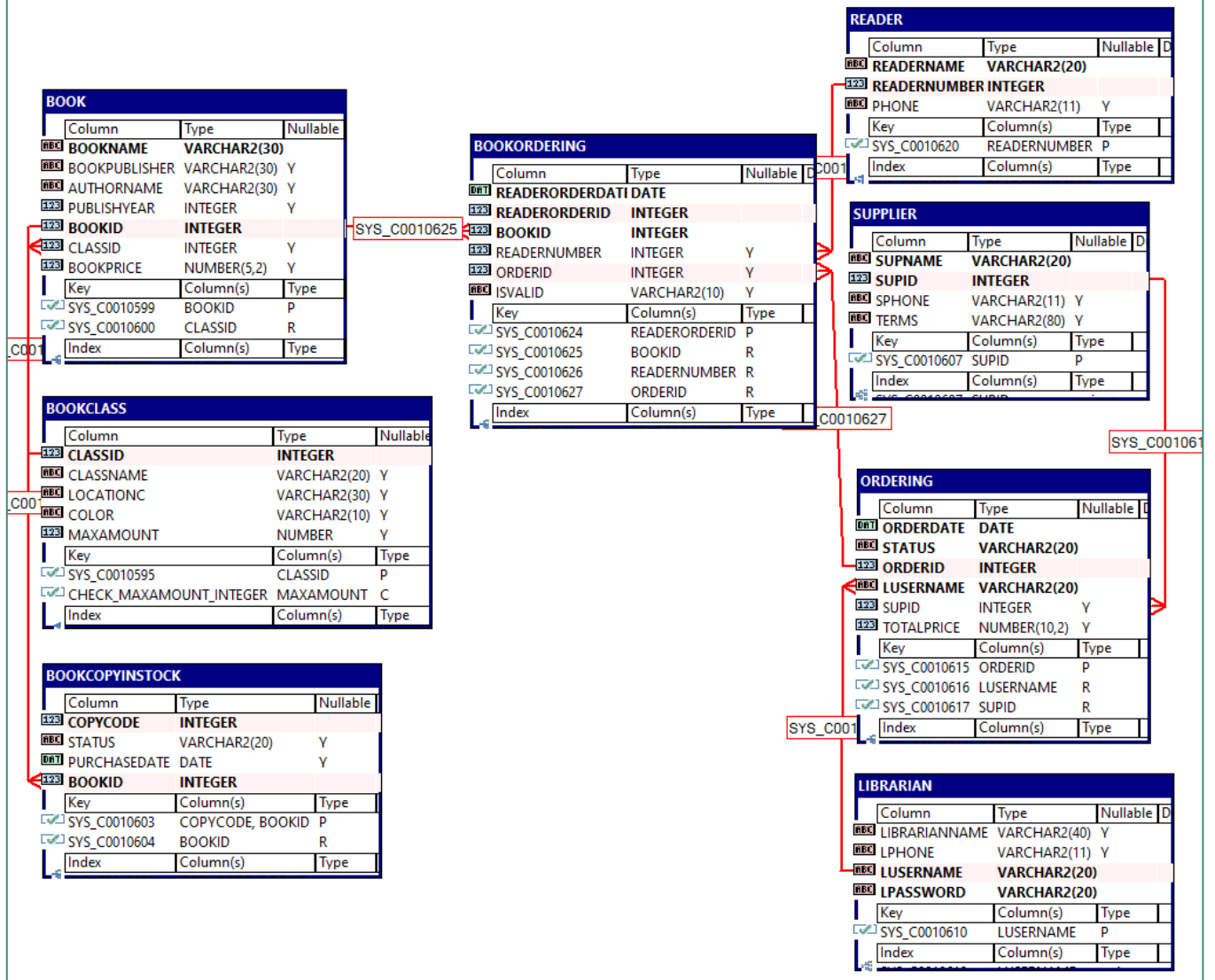
```
select *
from fines
where clientId = '480612577'
```

	CLIENTID	TOTALFINE
--	----------	-----------

היינו צריכות להתממשק למחלקת ניהול מלאי הספרים בספריה.

קיבלנו את הגיבוי שלהן ובתהליך reversing יצרנו את ה-DSD שלהן וממנו את ה-ERD.

### Book Inventory Management Section - DSD



מכאן יצרנו את ה-ERD:

יצרנו ישויות לפי הטבלאות שבתרשים הנ"ל יחד עם כל המאפיינים שמופיעים בתרשים (מלבד המפתחות הזרים) (Book, BookClass, BookCopyInStock, BookOrdering, Reader Supplier, Ordering, Librarian).

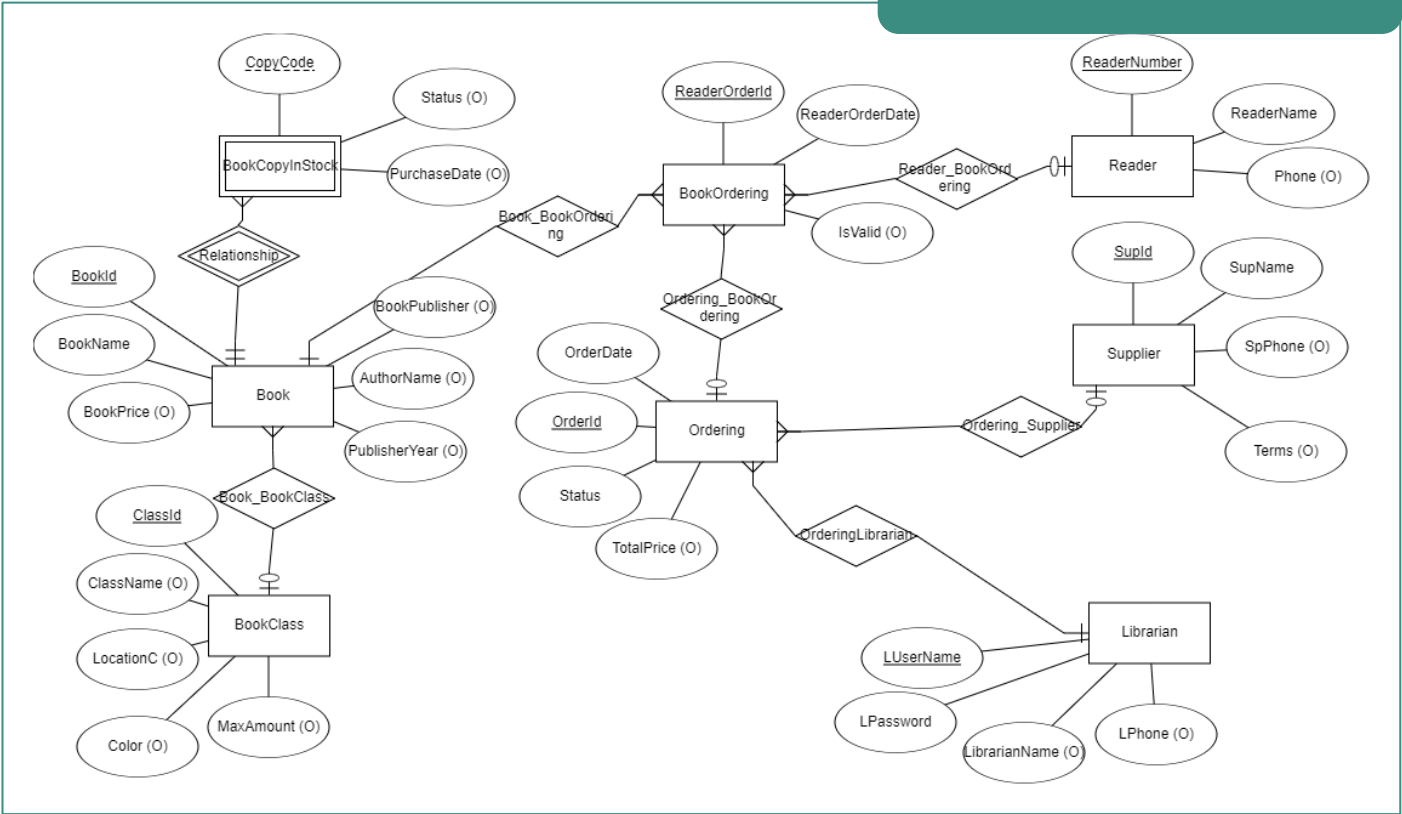
בכל טבלה שיש בה מפתח זר של טבלה אחרת שלא מרכיב את המפתח הראשי, הסקנו שיש קשר יחיד לרבים. (למשל: קשר יחיד לרבים מ-Book ל-BookClass).

בטבלת BookCopyInStock המפתח הראשי מורכב מ-2 עמודות (copyCode + bookID), כלומר, מורכב ממפתח של הישות (copyCode) יחד עם מפתח זר (bookID) המפנה למחלקת Book. מכאן שישות זו היא ישות חלשה. ולכן אפיינו אותה כך ב-ERD.





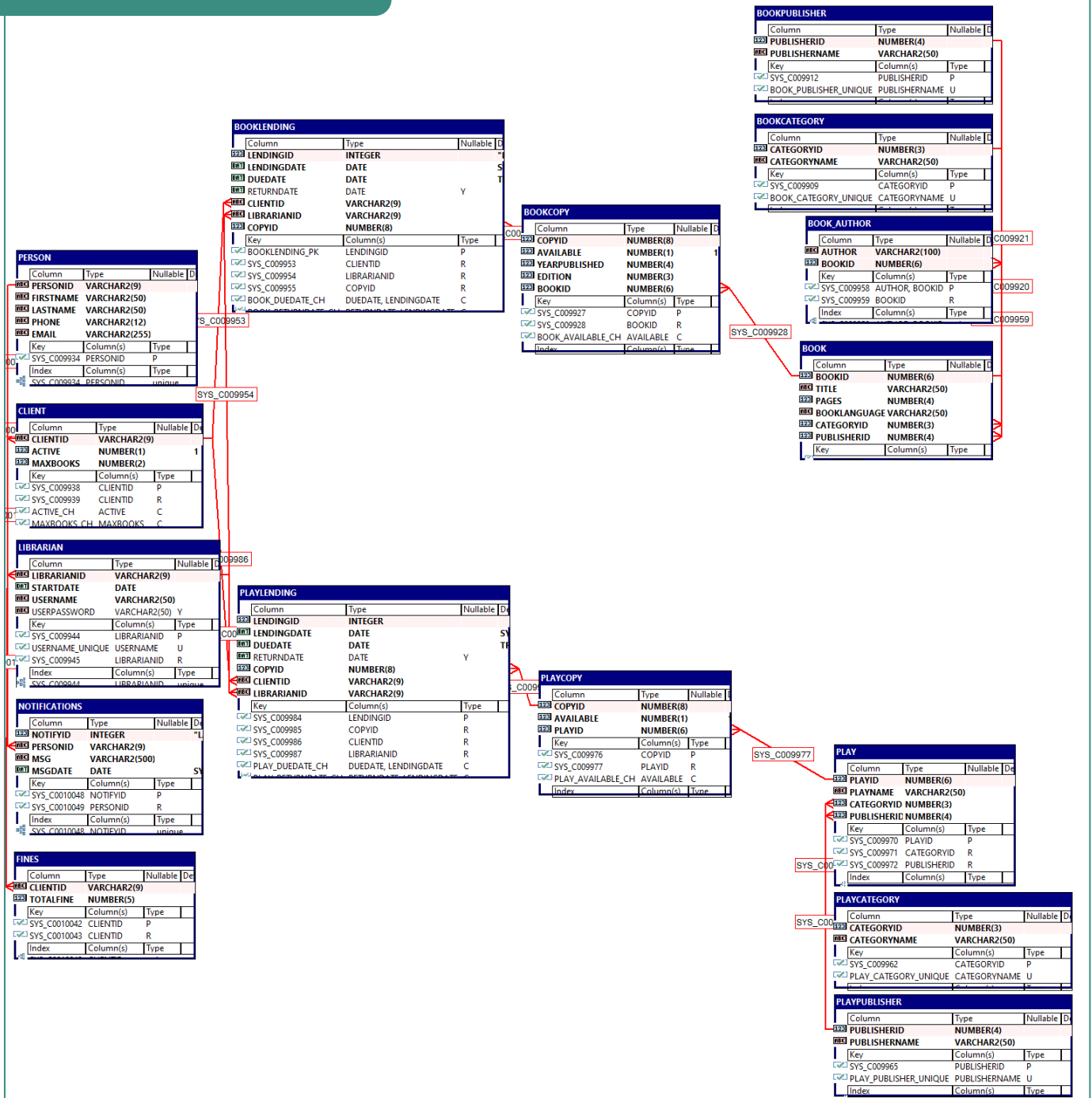
Book Inventory Management  
Section - ERD





כעת, בנינו את ה-DSD של האגף שלנו:

## Book & Plays Lending Section - DSD



ראינו שב-DSD הנוכחי שלנו יש כמה שינויים מה-DSD המקורי, בשל תוספות שהוכנסו במהלך פיתוח בסיס הנתונים, ולכן, יש לעדכן, בהתאם, את ה-ERD המקורי שלנו:

יש להוסיף ישויות Notifications-ו Fines.

כיון שב-Notifications יש מפתח זר המפנה ל-**Person**, הוספנו קשר יחיד לרבים.



## Book & Plays Lending Section - ERD



טבלאות שקיימות ב-2 האגפים במשמעות זהה:

Book & Plays Lending Section	Book Inventory Management Section
Book	Book
BookCategory	BookClass
BookCopy	BookCopyInStock
Client	Reader
Librarian	Librarian

החלטנו להשאיר את הטבלאות של האגף שלנו, אך לשנות אותן קצת כדי שיתאימו לטבלאות המקבילות שקיימות באגף השני:

1. לטבלת Book יש להוסיף עמודה (אופציונאלית) price.
2. לטבלת BookCategory יש להוסיף עמודות (אופציונאליות) המתייחסות לקטלוג ולמיקום בספריה: locationC, color, maxAmount.
3. לטבלת BookCopy יש להוסיף עמודה אופציונאלית בשם purchaseDate.
4. יש לשנות את כל המאפיינים שלא קיימים באגף השני, או שמכילים ערכי null לאופציונאליים.
5. יש להפוך את bookCopy לישות חלשה כדי לתמוך במזהים של העותקים שנייבא מהאגף השני.
6. יש להוסיף לטבלת BookCategory קטגוריה בשם "UNDEFINED", אליה נפנה את כל הספרים שנייבא מהאגף השני שלא מכילים ערך ב-classId.

טבלאות שלא קיימות באגף שלנו, ויש לייבא מהאגף השני:

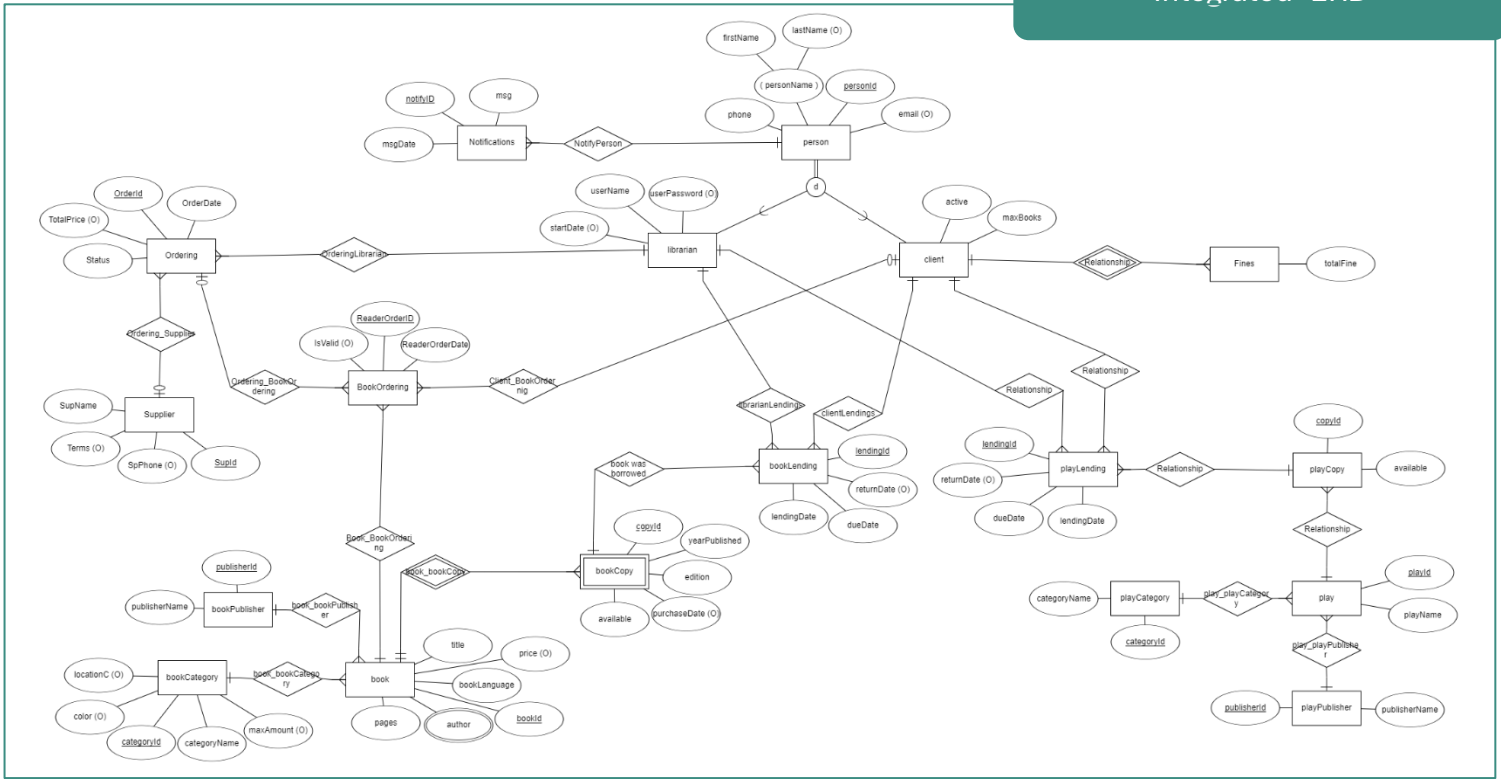
1. את הטבלאות BookOrdering ו-Ordering של האגף השני, נשאיר בסכמה המשולבת (אך כמובן נחבר אותן לטבלאות שבאגף שלנו).
2. את טבלת Supplier של האגף השני, בחרנו להשאיר כמו שהיא בסכמה המשולבת ולא לשנות אותה שתירש מ-Person, כיוון שספק מייצג ארגון ולא אדם פרטי...

### הערה

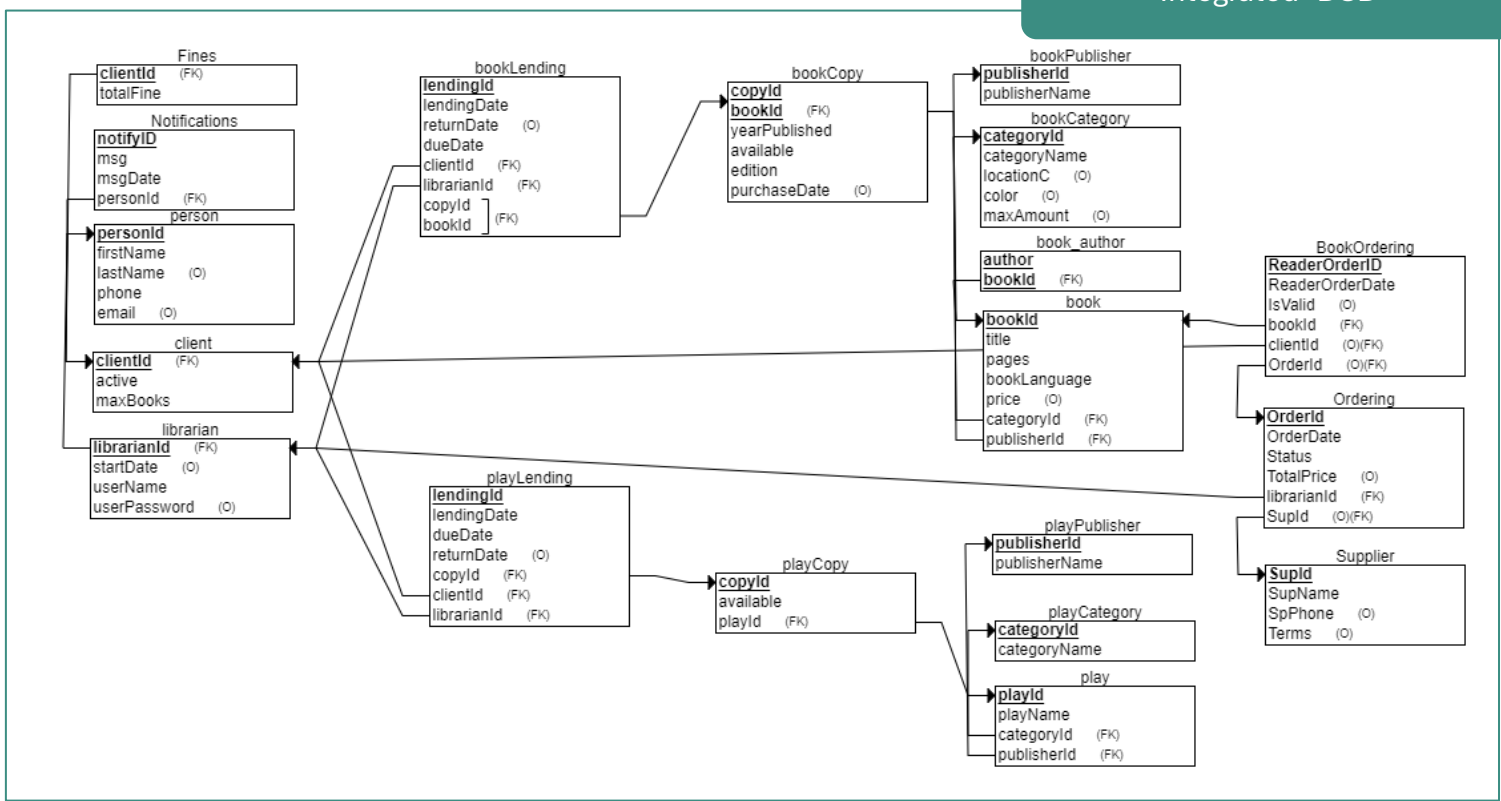
כיוון שבאגף השני הטבלאות Reader ו-Librarian לא יורשות מישות משותפת (Person), בייבוא הנתונים נצטרך לפצל את המידע בין טבלת האב לטבלת הבן המתאימה.



## Integrated- ERD



## Integrated- DSD



שינוי שמות:

לפני הוספת הגיבוי שקיבלנו לגיבוי שלנו שינינו את השמות של הטבלאות book וlibrariani כיוון שגם בגיבוי שקיבלנו יש טבלאות עם שמות זהים.

```
--rename tables
RENAME Book TO Book1;
RENAME Librarian TO Librarian1;
COMMIT;
```

פקודות alter table:

הוספת העמודות החסרות בטבלאות שלנו

שינוי העמודות בטבלאות שלנו שלא קיימות בטבלאות של הפרוייקט שקיבלנו כך שיוכלו להיות ריקות.

```
--alter tables
ALTER TABLE Book1
ADD price NUMBER(5,2) NULL;

ALTER TABLE BookCategory
ADD ( locationC VARCHAR2(30) NULL,
    color VARCHAR2(10) NULL,
    maxAmount NUMBER NULL);

ALTER TABLE Bookcopy
ADD purchaseDate DATE NULL;

ALTER TABLE Person
MODIFY email NULL;

ALTER TABLE Librarian1
MODIFY startDate NULL;

ALTER TABLE person
MODIFY lastName NULL;

ALTER TABLE client
MODIFY maxBooks DEFAULT 4;

ALTER TABLE Book1
MODIFY bookLanguage DEFAULT 'Hebrew';

ALTER TABLE Book1
MODIFY pages NULL;

ALTER TABLE Bookcopy
MODIFY edition NULL;
```

שינוי טבלת bookCopy לישות חלשה-

```
--change book copy to weak entity
ALTER TABLE Booklending
DROP CONSTRAINT SYS_C0010682

ALTER TABLE Bookcopy
DROP PRIMARY KEY;
ALTER TABLE Bookcopy
ADD PRIMARY KEY(copyId, bookId);

ALTER TABLE Booklending
ADD bookId NUMBER(6) NOT NULL;
ALTER TABLE Booklending
ADD FOREIGN KEY (copyId, bookId) REFERENCES BookCopy(copyId, bookId);
```

שינוי מספר הקטגוריה להיות מספר רץ החל מ1.

בטבלה אצלנו המספר קטגוריה הוא לפי הסדר מ1- כלומר לא נעשה שינוי בערכים אלא רק שינוי שהמספרים יוכנסו בצורה אוטומטית לפי הסדר כאשר נכניס את שמות הקטגוריות מהפרוייקט השני.

הכנסת קטגוריה בשם "UNDEFINED" בשביל הספרים בפרוייקט השני שאין להם קטגוריה.

```
-- modify BookCategory.categoryId to be auto generated
ALTER TABLE Book1
DROP CONSTRAINT SYS_C0010643;

alter table BookCategory drop column categoryId;

alter table bookCategory
  add categoryId integer
  generated always as identity (START WITH 1 INCREMENT BY 1);

alter table bookCategory
add constraint bookCategory_pk primary key (categoryId);

ALTER TABLE bookCategory MODIFY (categoryName INVISIBLE, locationC INVISIBLE, color INVISIBLE, maxAmount INVISIBLE);
ALTER TABLE bookCategory MODIFY (categoryName VISIBLE, locationC VISIBLE, color VISIBLE, maxAmount VISIBLE);

ALTER TABLE Book1
ADD FOREIGN KEY (categoryId) REFERENCES BookCategory(categoryId);

INSERT INTO Bookcategory(Categoryname)
VALUES ('UNDEFINED');
```

שינוי מספר ההוצאה להיות מספר רץ החל מ1.

בטבלה אצלנו המספר ההוצאה הוא לפי הסדר מ1- כלומר לא נעשה שינוי בערכים אלא רק שינוי שהמספרים יוכנסו בצורה אוטומטית לפי הסדר כאשר נכניס את שמות ההוצאות מהפרוייקט השני.

```
-- modify BookPublisher.publisherId to be auto generated
ALTER TABLE Book1
DROP CONSTRAINT SYS_C0010644;

alter table BookPublisher drop column publisherId;

alter table BookPublisher
  add publisherId integer
  generated always as identity (START WITH 1 INCREMENT BY 1);

alter table BookPublisher
add constraint bookPublisher_pk primary key (publisherId);

ALTER TABLE BookPublisher MODIFY (publisherName INVISIBLE);
ALTER TABLE BookPublisher MODIFY (publisherName VISIBLE);

ALTER TABLE Book1
ADD FOREIGN KEY (publisherId) REFERENCES BookPublisher(publisherId);
```

## יבוא שמות הוצאות.

```
-- import data
-- import data to BookPublisher
INSERT INTO BookPublisher(Publishername)
SELECT DISTINCT bookpublisher
from Book
where bookpublisher not in (select publishername from bookpublisher);
```

## יבוא הקטגוריות שאינן נמצאות אצלנו בטבלה.

## מיזוג העמודות החסרות אצלנו בקטגוריות משותפות.

```
-- import data to BookCategory
INSERT INTO Bookcategory(Categoryname, Locationc, Color, Maxamount)
SELECT className, Locationc, Color, Maxamount
FROM Bookclass
WHERE className NOT IN (SELECT categoryName FROM Bookcategory);

MERGE INTO Bookcategory BT
USING (
    SELECT B.Classname, B.LocationC, B.Color, B.maxAmount
    FROM Bookclass B) TMP
ON (BT.categoryName = TMP.className)
WHEN MATCHED THEN
UPDATE SET
    BT.LocationC = TMP.LocationC,
    BT.Color = TMP.Color,
    BT.maxAmount = TMP.maxAmount;
```

## יבוא נתונים מreaders לperson והוספת id לclient.

```
-- import data to Client (and to Person)
SELECT COUNT(*)
FROM Reader
WHERE readerNumber IN (SELECT TO_NUMBER(personID) FROM Person);

INSERT INTO Person(Personid, Firstname, Lastname, Phone)
SELECT (TO_CHAR(readerNumber)), regexp_substr(readerName, '\S+', 1, 1), regexp_substr(readerName, '\S+', 1, 2), phone
FROM Reader;

INSERT INTO CLIENT(CLIENTID)
SELECT (TO_CHAR(readerNumber))
FROM Reader;
```



כיוון שבפרויקט השני הזיהוי של ספרנית הוא ע"י שם משתמש ואצלנו הוא ע"י מספר זהות הוספנו מספרי זהות רצים שאינם בשימוש.

```
CREATE SEQUENCE fakePersonId INCREMENT BY 1 START WITH 374;

-- import data to Librarian (and to Person)
SELECT COUNT(*)
FROM Librarian
WHERE lUserName IN (SELECT userName FROM Librarian1);

CREATE GLOBAL TEMPORARY TABLE
    tmp_librarian(personId VARCHAR2(9),
    firstName VARCHAR2(50),
    lastName VARCHAR2(50),
    phone VARCHAR2(12),
    userName VARCHAR2(50),
    userPassword VARCHAR2(50))
ON COMMIT DELETE ROWS;

INSERT INTO tmp_librarian
SELECT | (TO_CHAR(fakePersonId.Nextval)),
    regexp_substr(librarianName, '\S+', 1, 1),
    regexp_substr(librarianName, '\S+', 1, 2),
    lPhone,
    lUserName,
    lPassword
    FROM Librarian;

INSERT INTO Person(Personid, Firstname, Lastname, Phone)
SELECT Personid, Firstname, Lastname, Phone
FROM tmp_librarian;

INSERT INTO Librarian1(librarianId, Username, Userpassword)
SELECT personId, Username, Userpassword
FROM tmp_librarian;

DROP TABLE tmp_librarian;
```





יבוא נתונים מbook לbook1.

שינוי id של הספרים המיובאים כדי שלא יהיו התנגשויות.

בהתחלה הכנסנו לכל הספרים המיובאים את הקטגוריה "UNDEFINED" ואח"כ מיזגנו את מי שלא null לקטגוריה הנכונה.

```
-- import data to Book
SELECT COUNT(*)
FROM book JOIN book1 NATURAL JOIN book_author
ON bookname = title
WHERE book.authorname = author

SELECT COUNT(*)
FROM book
WHERE bookpublisher IS NULL
OR authorName IS NULL
OR publishYear IS NULL
ORDER BY bookid

SELECT MAX(bookId) -- returned 500, so the sequence should start at 501
FROM Book1;

CREATE GLOBAL TEMPORARY TABLE
    tmp_book(bookId NUMBER(6) GENERATED ALWAYS AS IDENTITY (START WITH 501 INCREMENT BY 1),
    oldBookId INTEGER)
ON COMMIT PRESERVE ROWS;

INSERT INTO tmp_book(oldbookid)
SELECT bookId
FROM Book;

INSERT INTO Book1(Bookid, Title, Categoryid, Publisherid, Price)
SELECT tmp_book.bookid, bookName, 81, publisherId, bookPrice
FROM Book JOIN tmp_book ON Book.Bookid = tmp_book.oldbookid
    JOIN Bookpublisher ON Book.Bookpublisher = Bookpublisher.Publishername;

MERGE INTO Book1 BT
USING (
    SELECT C.categoryId, B1.Bookid FROM
        Book1 B1 JOIN tmp_book ON B1.BOOKID = tmp_book.bookid
        JOIN (SELECT * FROM Book WHERE classId IS NOT NULL)B ON B.Bookid = tmp_book.oldbookid
        NATURAL JOIN Bookclass
        JOIN Bookcategory C ON Bookclass.Classname = C.Categoryname) TMP
ON (BT.bookId = TMP.bookId)
WHEN MATCHED THEN
UPDATE SET
    BT.categoryId = TMP.categoryId;
```

יבוא שמות הסופרים, עם מספר הספר החדש.

```
-- import data to Book_Author table
INSERT INTO Book_Author
SELECT authorName, tmp_book.bookId
FROM Book JOIN tmp_book ON Book.Bookid = tmp_book.oldbookid;
```

יבוא נתונים לbookCopy.

כיוון שבפרויקט השני במקום התכונה של זמינות הספר - 0/1, יש תכונה של status עם כל מיני ערכים, מלכתחילה לא הוספנו את התכונה וכך בזמינות כל הספרים שלהם נוסף 1 לפי ברירת המחדל. לאחר מכן שינינו את כל מי ש 'Withdrawn', 'borrowed', 'Borrowed' להיות 0.

```
-- import data to BookCopy
INSERT INTO bookCopy(Copyid, Yearpublished, Bookid, Purchasedate)
SELECT copyCode, publishYear, tmp_book.bookId, purchasedate
FROM Bookcopyinstock JOIN Book ON Bookcopyinstock.Bookid = Book.Bookid
    JOIN tmp_book ON Bookcopyinstock.Bookid = tmp_book.oldbookid

MERGE INTO Bookcopy BT
USING (
    SELECT tmp_book.bookId, copyCode FROM
        (SELECT * FROM Bookcopyinstock WHERE status IN ('Withdrawn', 'borrowed', 'Borrowed')) BC1
        JOIN tmp_book ON BC1.bookId = tmp_book.oldbookid) TMP
ON (BT.bookId = TMP.bookId AND BT.copyId = TMP.copyCode)
WHEN MATCHED THEN
UPDATE SET
    available = 0;
```

שינוי המפתח הזר מהספרנית בordering מהשם משתמש של הספרנית למספר זהות.

```
-- change librarian identifier in Ordering table
ALTER TABLE Ordering
ADD librarianId VARCHAR2(9) REFERENCES Librarian1(Librarianid);

UPDATE Ordering
SET librarianId = (SELECT librarianId
                  FROM Librarian1
                  WHERE Ordering.Lusername = Librarian1.userName);

ALTER TABLE Ordering DROP COLUMN lUserName;
```

שינוי השם וסוג העמודה של המפתח הזר מclient בbookOrdering.

```
-- change column name and data type of client identifier in BookOrdering table
ALTER TABLE Bookordering
ADD clientId VARCHAR2(9) REFERENCES Client(clientId);

UPDATE Bookordering
SET clientId = TO_CHAR(readerNumber);

ALTER TABLE Bookordering DROP COLUMN readerNumber;
```

עדכון הbookId החדש בbookOrdering.

```
-- update bookId in BookOrdering table
ALTER TABLE Bookordering RENAME COLUMN bookId TO bookId_old;

ALTER TABLE Bookordering
ADD bookId NUMBER(6) REFERENCES Book1(Bookid);

UPDATE Bookordering
SET bookId = (SELECT bookId
              FROM tmp_book
              WHERE Bookordering.Bookid = tmp_book.oldbookid);

ALTER TABLE Bookordering DROP COLUMN bookId_old;
```

מחיקת הטבלאות הלא נחוצות מהפרויקט השני.

```
-- drop unnecessary tables
DROP TABLE Bookcopyinstock;
DROP TABLE Book;
DROP TABLE Bookclass;
DROP TABLE Reader;
DROP TABLE Librarian;
DROP TABLE tmp_book;
```



:Views

:View 1

כל המידע עבור הספרים.

כולל: מספר ספר, כותרת, שפה, עמודים, קטגוריה, סופר, הוצאה, מיקום, כמות עותקים, כמות עותקים זמינים.

```
-- create a view of all details of books in the library
CREATE OR REPLACE VIEW all_existings_books_details AS
SELECT bookId, title, bookLanguage, pages, categoryName,
       author, publisherName, locationC,
       count(*) AS AMOUNT_OF_COPIES,
       C.AVAILABLES AS AMOUNT_OF_AVAILABLE_COPIES
FROM Book NATURAL JOIN
     Book_Author NATURAL JOIN
     Bookcategory NATURAL JOIN
     Bookpublisher NATURAL JOIN
     Bookcopy NATURAL JOIN
     (SELECT bookId, SUM(available) AS availables
      FROM Bookcopy
      GROUP BY bookId) C
GROUP BY bookId, title, bookLanguage, pages, categoryName, author, publisherName, locationC, C.AVAILABLES;
```

תוצאות הרצה:

SELECT * FROM all_existings_books_details										
	BOOKID	TITLE	BOOKLANGUAGE	PAGES	CATEGORYNAME	AUTHOR	PUBLISHERNAME	LOCATIONC	AMOUNT_OF_COPIES	AMOUNT_OF_AVAILABLE_COPIES
1	46	Brother Rat	French	506	Autobiography	Melba-Houston	Penguin Random House		2	2
2	80	Nightmare on Elm Street 5: The Dream Child, A	Hebrew	111	Science fiction	Melba-Wiedlin	Jewish Publication Society	Floor 2 Column 4	4	3
3	200	Everyone's Hero	Spanish	251	Science	Miguel-Jay	DELTA publishing		5	4
4	449	Stuart: A Life Backward	French	29	Encyclopedia	Miki-Yorn	HarperCollins		1	1
5	796	The Winds of Winter	Hebrew		UNDEFINED	Millie Thompson	Oxford University Press		1	0
6	870	The Naval Treaty	Hebrew		UNDEFINED	Mint Northam	Ballantine Books		1	0
7	652	Clockwork Prince	Hebrew		UNDEFINED	Nanci Cube	Bloomsbury		1	0
8	856	The Kingdom of God is Within Y	Hebrew		UNDEFINED	Naomi Red	MIT Press		1	0
9	92	Stars	Russian	384	Poetry	Naomi-Gore	Jewish Publication Society	Floor 3 Column 3	3	3
10	580	The Blithedale Romance	Hebrew		UNDEFINED	Natascha Ifans	Pearson Education		3	2
11	812	The Sound and the Fury	Hebrew		UNDEFINED	Nathan Hauser	Algonquin Books		1	1
12	370	The Burglar	Russian	566	Suspense	Neve-Sandoval	HarperCollins Publishers		3	2



## שאלתה 1:

מביא את הקטגוריה, כותרת, שפה, כמות עותקים זמינה ומיקום, של כל הספרים ששייכים לקטגוריות והשפות הנבחרות.

```
--select books by category and language
SELECT categoryName, title, bookLanguage, AMOUNT_OF_AVAILABLE_COPIES, locationC
FROM all_existings_books_details
WHERE categoryName IN
    (<NAME="category"
      LIST="select distinct categoryName from all_existings_books_details"
      MULTISELECT="yes"
      TYPE="string"
      RESTRICTED="yes">)
    <NAME="language"
      MULTISELECT="yes"
      TYPE="string"
      LIST="select distinct bookLanguage from all_existings_books_details"
      RESTRICTED="yes"
      TYPE="string"
      PREFIX="and bookLanguage in (" SUFFIX=")">
GROUP BY bookId, title, bookLanguage, locationC, categoryName, AMOUNT_OF_AVAILABLE_COPIES
ORDER BY categoryName;
```

### תוצאה:

	CATEGORYNAME	TITLE	BOOKLANGUAGE	AMOUNT_OF_AVAILABLE_COPIES	LOCATIONC
1	Children's	Firestarter	Yiddish	1	...
2	Hobbies	Escape to Athena	Hebrew	3	...
3	Hobbies	Fraternity Demon	Yiddish	2	...
4	Hobbies	Glory Daze	Yiddish	0	...
5	Hobbies	Jesse Stone: Benefit of the Doubt	Spanish	1	...
6	Hobbies	Upstairs and Downstairs	Spanish	1	...
7	Math	Fountain, The	Spanish	2	...
8	Math	Khumba	Spanish	1	...
9	Math	Prince of Pennsylvania, The	Hebrew	0	...
10	Math	Simply Irresistible	Hebrew	1	...

## שאלתה 2:

מביא את מספר ספר, כותרת, שפה, קטגוריה, כמות עותקים לא זמינים של כל הספרים שמספר העותקים הזמינים שלהם הוא 0.

```
--select books that have no copies available
SELECT bookId, title, bookLanguage, categoryName, AMOUNT_OF_COPIES AS AMOUNT_OF_NOT_AVAILABLE_COPIES
FROM all_existings_books_details
WHERE AMOUNT_OF_AVAILABLE_COPIES = 0
GROUP BY bookId, title, bookLanguage, categoryName, AMOUNT_OF_COPIES
ORDER BY bookId;
```

### תוצאה:

	BOOKID	TITLE	BOOKLANGUAGE	CATEGORYNAME	AMOUNT_OF_NOT_AVAILABLE_COPIES
1	14	Ladies in Lavender	Russian	Poetry	1
2	20	Day of the Dead 2: Contagium	Arabic	Business	3
3	25	Salaam Bombay!	English	Art	3
4	26	The Hire: Powder Keg	English	Business	3
5	30	Malice	Hebrew	Business	2
6	66	Raging Bull	Spanish	Action	1
7	71	Salut cousin!	Russian	Humor	1
8	72	Glory Daze	Yiddish	Hobbies	3
9	90	Call, The	Arabic	Cookbook	1
10	91	Honeymoon	Russian	Hobbies	1

:View 2

מידע על הזמנות ספרים.

כולל: מספר ספר, כותרת, שפה, כמות עותקים זמינים, תאריך הזמנה, מספר הקורא שהזמין, כמות הזמנות פעילות.

```
--create view
CREATE OR REPLACE VIEW books_details_and_related_orders AS
SELECT Book.Bookid, title, bookLanguage, availables AS AMOUNT_OF_AVAILABLE_COPIES , readerOrderDate, clientId, AMOUNT_OF_ACTIVE_ORDERS
FROM Book JOIN
  (SELECT bookId, SUM(available) AS availables
   FROM Bookcopy
   GROUP BY bookId) C ON Book.Bookid = C.BOOKID
LEFT JOIN Bookordering ON Book.bookId = Bookordering.Bookid
LEFT JOIN
  (SELECT bookId, COUNT(*) AS AMOUNT_OF_ACTIVE_ORDERS
   FROM Ordering NATURAL JOIN Bookordering
   WHERE status IN('Confirmed', 'Delivered', 'Shipped')
   GROUP BY bookId) O ON Bookordering.Bookid = O.BOOKID;
```

תוצאות הרצה:

	BOOKID	TITLE	BOOKLANGUAGE	AMOUNT_OF_AVAILABLE_COPIES	READERORDERDATE	CLIENTID	AMOUNT_OF_ACTIVE_ORDERS
1	672	A Nasty Story	Hebrew	0			
2	781	Go Set a Watchman	Hebrew	0			
3	749	Moby Dick	Hebrew	1	21/12/2003	4746823	1
4	647	A Study in Scarlet	Hebrew	0			
5	902	The Dune Series	Hebrew	1	31/01/2018	6172351	1
6	902	The Dune Series	Hebrew	1	13/12/1990	7720421	1
7	684	Madame Bovary	Hebrew	0	11/11/2009	7653878	1
8	649	The Buried Giant	Hebrew	1	10/06/1999	8271171	1
9	588	The Book Thief	Hebrew	1			
10	677	Angels & Demons	Hebrew	0	22/04/2013	8020506	1

שאלתה 1:

מביא את כל ההזמנות שביצע קורא ספציפי.

```
--select all orders of specific client
SELECT *
FROM books_details_and_related_orders
WHERE clientId = &<NAME= "clientId" REQUIRED=TRUE>;
```

תוצאה:

	BOOKID	TITLE	BOOKLANGUAGE	AMOUNT_OF_AVAILABLE_COPIES	READERORDERDATE	CLIENTID	AMOUNT_OF_ACTIVE_ORDERS
▶ 1	837	Babylon's Ashes	Hebrew	1	16/04/1987	8835171	1
2	901	The Trials of Apollo	Hebrew	1	03/05/2001	8835171	1

שאלתה 2:

מביא עבור כל ספר שאין לו עותקים זמינים: מספר ספר, כותרת, כמות הזמנות פעילות.

```
--select all books that have no copies available and display active orders of them
SELECT bookId, title, NVL(AMOUNT_OF_ACTIVE_ORDERS, 0) AS AMOUNT_OF_ACTIVE_ORDERS
FROM books_details_and_related_orders
WHERE AMOUNT_OF_AVAILABLE_COPIES = 0
GROUP BY bookId, title, AMOUNT_OF_ACTIVE_ORDERS
ORDER BY bookId;
```

תוצאה:

	BOOKID	TITLE	AMOUNT_OF_ACTIVE_ORDERS
▶ 1	14	Ladies in Lavender	0
2	20	Day of the Dead 2: Contagium	0
3	25	Salaam Bombay!	0
4	26	The Hire: Powder Keg	0
5	28	Saving Otter 501	0
6	30	Malice	0
7	66	Raging Bull	0
8	71	Salut cousin!	0
9	72	Glory Daze	0
10	76	Fire and Ice	0